applied sciences

MDPI

Article

DC-SoC: Optimizing a Blockchain Data Dissemination Model Based on Density Clustering and Social Mechanisms

Xinhua Dong, Xiaohong Dang, Zhigang Xu *, Kangze Ye, Hongmu Han and Enda Zheng

School of Computer Science, Hubei University of Technology, 28 Nanli Road, Wuhan 430068, China; xhdong@hbut.edu.cn (X.D.); 102211122@hbut.edu.cn (X.D.); 102010967@hbut.edu.cn (K.Y.); hanhongmu@hbut.edu.cn (H.H.); 102211200@hbut.edu.cn (E.Z.)

* Correspondence: xzg@hbut.edu.cn

Abstract: Due to its partially decentralized and highly scalable features, the consortium blockchain has currently overtaken other blockchain technologies as the one most frequently used and studied across various industries. However, performance issues such as low transaction efficiency and redundant communication processes continue to hinder the development of consortium blockchains. In the Hyperledger Fabric consortium blockchain system, transaction efficiency is largely influenced by the consensus protocol and broadcast protocol. This paper introduces a novel consortium blockchain network model, DC-SoC, focused on optimizing broadcast protocols. By incorporating the concept of density clustering, a stable propagation structure is established for the blockchain network, thus optimizing data dissemination in the Gossip protocol. Additionally, the concept of social networks is integrated, using trustworthiness scores and economic incentives to evaluate node security. The experimental results demonstrate that when DC-SoC is applied in a large-scale consortium blockchain environment, it significantly improves communication performance between nodes and ensures transmission reliability.

Keywords: hyperledger fabric; broadcast protocol; network clustering; social network; malicious detection

check for updates

Citation: Dong, X.; Dang, X.; Xu, Z.; Ye, K.; Han, H.; Zheng, E. DC-SoC: Optimizing a Blockchain Data Dissemination Model Based on Density Clustering and Social Mechanisms. *Appl. Sci.* **2024**, *14*, 10058. https://doi.org/10.3390/app142110058

Academic Editors: Gianluigi Ferrari and Eui-Nam Huh

Received: 1 September 2024
Revised: 7 October 2024
Accepted: 25 October 2024
Published: 4 November 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

In recent years, blockchain technology has garnered significant attention from both academia and society due to its inherent characteristics, such as data immutability and decentralization. Depending on the specific application requirements, blockchain technologies can be classified into public chains, private chains, and consortium chains, each having varying levels of data operation permissions. Among these, the consortium chain can be regarded as an alliance comprising multiple authorized institutions, adopting a "partially decentralized" structure and exclusively permitting participation in transactions by alliance members with CA certification [1]. This combination of public and private characteristics enables consortium chains to offer a broad spectrum of applications. Hyperledger Fabric [2] has emerged as the leading consortium chain project, owing to features such as the MSP security certification mechanism, free channel creation, and high modularity. Nonetheless, similar to other blockchain variants, the low efficiency of communication between peers continues to be a significant constraint on the advancement of consortium chains, despite ensuring transaction security and privacy with a decentralized structure. Primarily, this issue can be attributed to the consensus layer and network layer within the blockchain.

The architecture of blockchain can be broadly summarized as consisting of the data layer, network layer, consensus layer, smart contract layer, and application layer [3]. Among these, the network layer serves as the source of decentralization in blockchain. It primarily defines the networking process between peers in the blockchain and controls the propagation of data in the network through broadcasting protocols. Each user peer's block data generated through the consensus service is redundantly stored in ledger form on

different peers through multiple peer-to-peer (P2P) transactions, effectively addressing single-point-of-failure issues and ensuring data security. Therefore, optimizing the data propagation mechanism [4] at the network layer is a highly valuable and necessary measure for improving the overall communication performance of blockchain. Additionally, as the practical applications mature, the demand for the number of peers in blockchain systems is increasing. The impact of the broadcasting protocol on communication efficiency becomes more significant [5]. However, compared to research on optimizing consensus algorithms, data storage structures, and other aspects, research on the network layer has received relatively little attention.

In response to the above issues, this paper proposes a novel model for a consortium blockchain network named density clustering-social network (DC-SoC), designed to address the communication inefficiencies commonly found in consortium blockchains. The model enhances propagation speed and ensures network-layer reliability, effectively improving overall communication performance.

Generally, our contribution includes the following:

1. Building on the concept of high-density connectivity enabled by density clustering, we have enhanced the infection algorithm in traditional Gossip broadcasting, effectively reducing network latency and node load during data dissemination.
2. A structured approach to constructing network topology is implemented to mitigate the issue of isolated nodes arising from unstructured random propagation in consortium blockchain networks, ensuring comprehensive coverage of block propagation.
3. By introducing social trust and economic incentive mechanisms, we merge the social network model with the consortium blockchain network, providing an effective solution for improving the reliability of node transactions and the security of the network within consortium blockchains.

The remaining sections of this paper are as follows. Section 2 provides a comprehensive review of key research areas and significant breakthroughs in the field of communication performance optimization for permissioned blockchains, particularly focusing on the latest advancements in broadcasting protocols in blockchain technology, along with the progress in related simulation tools. Section 3 outlines the overall framework and design approach of this study. Section 4 then explains the social mechanism based on node trustworthiness and economic incentive values. Section 5 reviews the performance evaluation results, analyzing the significant advantages of DC-SoC in terms of propagation efficiency and reliability as demonstrated through experimental comparison. Finally, Section 6 concludes with a summary of our research findings and provides insights into future research directions.

2. Related Works

2.1. Communication Performance

Currently, the primary focus of research in enhancing the communication performance of permissioned blockchains is on optimizing the local blockchain architecture and improving the performance and scalability of Byzantine fault-tolerant consensus. For instance, Xu et al. [6] proposed a new PBFT consensus algorithm, designing a reputation evaluation model to select more trustworthy nodes for participation in system consensus, thereby achieving role differentiation. By constructing a dual consensus layer structure, communication complexity is reduced, fault tolerance is improved, and scalability is enhanced. This development is particularly relevant to Hyperledger Fabric, which relies on consensus mechanisms such as PBFT for secure and efficient block validation. Assiri [7] introduced a fair proof-of-stake consensus protocol for block creation, which proposes a simplified block verification procedure to reduce the verification process of trusted blocks and transactions, lowering computational costs, accelerating processing time, and enhancing security and privacy. Spengler et al. [8] utilized CouchDB to improve the storage performance of heterogeneous medical data within the Fabric consortium blockchain. This enhancement allows the blockchain to support more complex and efficient query operations, ultimately optimizing the consortium blockchain system's overall throughput and latency. Mar-

son et al. [9] introduced the MITOSIS method, achieving low latency in permissioned blockchains through a sharding technique similar to cell mitosis, and concurrently ensuring high scalability via parallel processing. Yi et al. [10] suggested the use of a novel threshold digital signature technique, based on NP-Hard problems, to enhance the consensus algorithm. They developed a new, highly efficient, and secure blockchain system based on this approach, representing a classic research methodology. Lu et al. [11] introduced the LEAP method, which integrates edge data distribution with coalition formation games to dynamically adjust the correlations between clients and ESs, while rationally allocating communication resources to mitigate the straggler issue. Seid et al. [12] introduced a blockchain and MADRL integrated framework, enabling IoT devices and UAVs to continuously adjust resource demands and pricing strategies, thereby reducing the computation costs of IoTDs while maximizing the utility of UAVs and enhancing overall network performance.

However, compared to research on the data layer and consensus layer, the network layer, situated between the two, has received less attention. In fact, optimizing the network topology to enhance the transmission rate and reliability can significantly contribute to positive effects on the overall performance and stability of the blockchain.

2.2. Broadcasting Protocol

The broadcasting protocol, a critical distributed protocol in blockchain systems, is responsible for constructing the network topology and facilitating block transmission. Within Hyperledger Fabric, the Gossip protocol serves as the mechanism for disseminating data across channels. The Gossip protocol originated in 1987 as part of the Epidemic algorithm, used for maintaining replicated databases [13]. The operation of the Gossip protocol resembles epidemic propagation: the source peer initiates the message and randomly selects multiple peers for dissemination. Upon receiving it, the "infected" peers further propagate the message to another set of peers. This iterative process continues until the message reaches the network's periphery [14]. In case of network issues or when a peer first joins the network, retrieval, and recovery mechanisms activate to obtain the missing block content either from within the organization or from others. Nonetheless, this propagation process ensures message dissemination reliability but unavoidably leads to information redundancy and a higher processing load on peers. Reducing the number of recipients to alleviate the load during message dissemination could result in isolated peers and consequent data dissemination gaps. To mitigate these network instability scenarios, frequently activating the retrieval mechanism would inevitably increase propagation delays substantially.

In recent years, significant advancements have been made in the field of broadcasting. Hao et al. [15] developed the BlockP2P propagation structure, which uses the K-Means algorithm for peer clustering and a parallel spanning tree broadcasting algorithm for rapid data dissemination across clusters. Subsequently, they introduced a method to efficiently detect peer failures and reduce network load [16]. However, this study is limited by its exclusive focus on public blockchain environments, overlooking potential issues in consortium and private blockchains. Furthermore, the absence of a comparative analysis of clustering algorithms impedes the evaluation of their performance and applicability. Berendea et al. [17] concentrated on Hyperledger Fabric, proposing the infect-upon-contagion algorithm to replace the Gossip protocol's push mechanism. They optimized the remaining propagation mechanism, enhancing both efficiency and fairness in the Fabric broadcast layer. However, this study focuses solely on block propagation coverage and reliability, omitting an analysis of peer security. Additionally, the discussion on network topology construction lacks detail. To further improve the security of blockchain networks, some studies have introduced peer review mechanisms into blockchain systems. In distributed networks, peer review mechanisms are used to assess node reliability, with nodes reviewing each other's transactions to ensure that data are validated by multiple nodes before or after being added to the blockchain. This mechanism helps prevent the dissemination of false information by malicious nodes and guarantees the accuracy and integrity of broadcasted data. Oktian et al. [18] designed a peer review model to construct a mature blockchain-based federated

learning protocol, incentivizing participating nodes to act honestly and optimize their training efforts for higher rewards, while also punishing malicious behavior. This approach enhanced data privacy and system reliability. However, the study's primary application scenario was limited to federated learning, failing to adequately address block propagation efficiency issues within blockchain systems. Furthermore, the protocol's punishment mechanism for malicious nodes is relatively simplistic and fails to fully leverage the network topology to optimize data dissemination paths. Yang et al. [19] integrated a reputation mechanism into the blockchain data-sharing process, allowing users to receive reputation feedback based on their partners' behavior, allowing honest users to benefit from more data-sharing opportunities, while malicious users are held accountable and removed from the system. However, the reputation feedback mechanism mainly relies on cooperative relationships between users, overlooking the effects of node changes in dynamic networks on propagation efficiency. These two studies have made valuable contributions to improving system security, but further advancements are needed in optimizing network topology and dynamic trust mechanisms. In addition, recent advances in blockchain verification and model checking have significantly enhanced the security and reliability of blockchain systems. Among these advancements, two notable studies by AlSobeh introduced innovative frameworks that enhance the verification of dynamic behaviors in blockchain systems. For example, the BlockASP [20] framework employs aspect-oriented programming (AOP) to analyze and monitor dynamic behaviors within blockchain systems, integrating the strengths of aspect-oriented programming and model checking into the blockchain architecture to ensure the security of decentralized networks. Similarly, the OSM [21] framework extends model-checking techniques to analyze and verify dynamic behaviors in software systems, and this approach can also be applied to blockchain systems to ensure the secure and dynamic operation of blockchain protocols. Both frameworks provide valuable insights into the evolving field of blockchain consensus algorithms, particularly in enhancing the security and dynamic verification of blockchain networks. However, these frameworks are limited to runtime verification in software environments and do not explicitly address the challenges of maintaining efficient communication and scalability in decentralized, large-scale blockchain environments.

2.3. Network Simulation with NS-3

The majority of commonly used simulation software focuses primarily on testing upper-layer application protocols, such as consensus protocols, without giving sufficient consideration to the construction of the P2P network layer. In contrast, simulation platforms with built-in P2P components, such as NS-2 and PeerSim, are less frequently considered because of their outdated nature. The communities that once extended and maintained these platforms have gradually diminished, resulting in limited available reference materials and a lack of updates.

In contrast to older platforms, the NS-3 [22] simulation platform has gained increasing prominence in recent years, offering numerous built-in network components, including P2P networking, along with high programmability and scalability. This makes NS-3 suitable for simulating various large-scale topologies. NS-3 is a widely adopted open-source network simulator designed to study the behavior of communication networks, including blockchain-based systems. Its modular architecture enables researchers to model and simulate various network protocols and environments, enabling accurate performance evaluations of blockchain protocols. Recent research has employed NS-3 to simulate and validate the effectiveness of models in addressing security and performance challenges in real-world environments. For instance, Li et al. [23] used NS-3 to simulate a blockchain-based trust evaluation scheme, which evaluates trust values considering factors such as initial trust, historical experience, recommendation trust, and observation trust. This approach improved the detection of malicious nodes, reduced transmission costs, and enhanced overall network reliability. Similarly, Razaque et al. [24] demonstrated through NS-3 simulations that multi-label classification could successfully detect malicious vehicles,

improving accuracy, and reducing loss rates. Therefore, in this paper, we utilize NS-3 to simulate the performance of DC-SoC. By modeling the communication between nodes in a dynamic network environment, we can accurately assess the block propagation efficiency and reliability of the model. NS-3 enables us to explore a variety of network scenarios and evaluate the enhancements of our model in terms of transmission delay and network scalability. This simulation framework is essential for validating the effectiveness of the model under real-world conditions.

To address the current research status, this paper extends the framework of "DC-Gossip" [25], providing a more secure and robust network model for consortium blockchain systems. Specifically, while DC-Gossip only offers a stable data dissemination mechanism for Fabric, DC-SoC not only enhances broadcasting efficiency but also defines information reception rules for transactions between nodes through the concept of social networks, effectively preventing malicious behavior. This allows DC-SoC to resist complex attacks and ensures reliable data transmission within the network. These improvements address the limitations identified in previous literature, significantly enhancing network reliability and performance in dynamic environments. Based on the above discussion, some comparisons of schemes related to the work of this paper are listed in Table 1.

Table 1. Comparison with other data dissemination models.

Model	High Transmission Efficiency	Comprehensive Propagation Coverage	Transmission Reliability
Ref. [16]	✓	×	×
Ref. [17]	✓	✓	×
Refs. [18,19]	×	×	✓
Refs. [20,21]	×	✓	✓
Ours	✓	✓	✓

3. DC-SoC Consortium Chain Network Model

3.1. Overview

In traditional centralized networks, server functions are concentrated in a limited number of peers, leading to a single-point-of-failure issue, while server peers face constraints in their capacity to concurrently process services due to hardware limitations. Consequently, blockchain systems that prioritize fairness and security adopt a peer-to-peer (P2P) network architecture. Within a P2P network, each peer possesses the capability to both receive and provide services to other peers. This approach effectively leverages substantial terminal resources and simultaneously addresses the primary shortcomings of centralized networks [14].

Blockchain performance is characterized by two dimensions: transmission rate and transmission reliability [16]. Within the network layer, these dimensions are typically manifested as peer propagation delay and data coverage, respectively. A reduced propagation delay among peers correlates with a higher transmission rate for the entire blockchain network. An increased coverage rate of block data propagation within the network leads to a more robust network state. Conversely, incomplete block propagation to numerous peers diminishes network transmission reliability, often due to instability in the propagation structure and inadequacies in peer deactivation detection mechanisms.

To ensure message delivery to every peer, Fabric's P2P network employs the Gossip protocol for data dissemination. However, in the context of the contagion algorithm used in traditional Gossip protocols, achieving high data coverage necessitates enhanced network connectivity, as indicated by the number of neighbors for each peer. Given the limited processing power of each peer, this approach not only heightens the peer workload but also results in a reduction of the transmission rate when network connectivity becomes excessive. To address this challenge, we propose integrating the clustering concept to establish a rational network topology by partitioning cluster relationships among peers. This strategy not only ensures network connectivity but also minimizes the network diameter during communication among interconnected peers.

Furthermore, ensuring transmission reliability is crucial, not only through optimizing the propagation structure to enhance data coverage but also by focusing on the peers' reliability themselves. In this work, we propose adopting the concept of community detection to view the entire blockchain network as a social network. Additionally, we employ both trustworthiness and economic incentive mechanisms to assign appropriate values to each peer and identify suitable propagation targets, thereby boosting the network's robustness [26].

Building on the previously mentioned concepts, we introduce a network transaction processing model specifically designed for the federation chain environment, named DC-SoC. The operational logic of the DC-SoC model is also depicted in Figure 1, this framework operates through four phases: initial clustering, social mechanism, block dissemination, and network updates, which are triggered by timing mechanisms. Initially, at network startup, peers are clustered and segmented, with the transmission delay between each peer serving as the distance metric. Subsequently, formal block broadcasting begins, encompassing three stages: push, dynamic maintenance, and restoration. Throughout the broadcasting phase, transactional judgment rules between peers are established through social mechanism operations. After a specified operational period, re-clustering takes place to reflect changes in the current network state, ensuring structural updates and the maintenance of network availability.

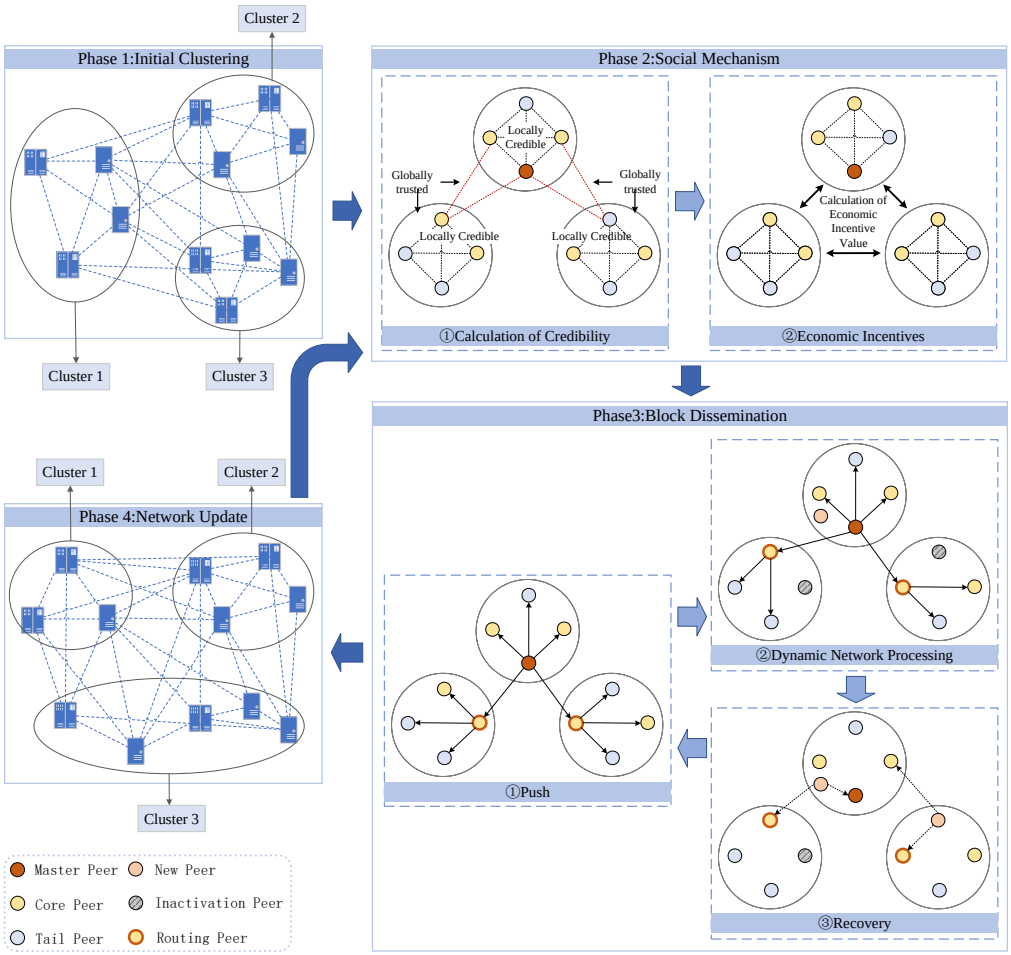


Figure 1. Overview of DC-SoC.

The DC-SoC model optimizes traditional blockchain systems by enhancing efficiency, security, and scalability. In terms of efficiency, block propagation is optimized by clustering nodes based on proximity and trust metrics, thereby reducing communication overhead and minimizing network latency. Security is bolstered through a dynamic trust-based evaluation

system that continuously assesses node trustworthiness. This mechanism mitigates risks such as Sybil and Byzantine attacks, preventing malicious nodes from compromising the network's integrity. Scalability is addressed through the density clustering method, which divides the network into manageable segments, ensuring stable performance and reliable propagation as the network expands. These advancements offer a comprehensive solution to the limitations of traditional blockchain models, making DC-SoC highly suitable for large-scale decentralized networks. The subsequent sections of this chapter offer a detailed exploration of the design intricacies of DC-SoC.

3.2. Initial Clustering

The conventional Gossip protocol employs unstructured random propagation for data transmission, potentially resulting in high latency and incomplete propagation. To address these issues, we opt to regard peers within each organization within the channel as data objects in the dataset. We then map the network latency between them as Euclidean distances in the vector space and employ clustering based on this to construct a rational network topology as a deterministic propagation structure within the organization, ensuring that each peer can complete block reception in the shortest time possible. It is important to note that clustering is conducted within each organization within the channel. Additionally, the clustering process for each organization is independent and does not impact the others.

Clustering algorithms are widely used in research on contemporary blockchain networks, including their applications in social networks. For instance, Wu et al. [27] described the Bitcoin transaction network as an undirected graph and employed spectral clustering to segment this network for mining interests. After conducting numerous comparisons, the paper concludes that density clustering is the most suitable method for practical applications within coalition chains.

In real-world network environments, each blockchain peer operates on a physical machine, located in a distinct network location. Communication delay between peers is influenced by factors such as physical distance between machines, hardware performance, bandwidth, and more. Thus, peers can be abstractly visualized in a multidimensional vector space, with propagation delay as the Euclidean distance metric. However, the limited impact of propagation delay maintains this vector space as low-dimensional, making it unsuitable for high-dimensional clustering algorithms like spectral clustering. Alternatively, algorithms like K-Means segment peers into clusters through iterative processes. Yet, in this vector space, peer distribution often appears uniform yet irregular. As a result, creating effective propagation trees for clusters identified by algorithms like K-Means is challenging in this environment.

Density clustering excels at identifying clusters of arbitrary shapes and efficiently pinpointing outliers. It facilitates identifying high-density connectivity regions, enabling efficient block distribution from the master peer to all channel peers, including leaf peers. Crucially, identifying outliers with density clustering does not imply their immediate removal. Instead, these outliers are included in analyses to mitigate their impact on network stability, due to factors like high latency or unstable connections. This approach is adopted because each peer in the blockchain typically represents a user. Therefore, removing users with poor network conditions is inappropriate, as it would compromise blockchain fairness and disrupt application usage.

DBSCAN is a classic density clustering algorithm that utilizes two parameters: the radius eps and the minimum number of neighboring peers $MinPts$. Starting from the initial input peers and their neighborhoods, it iterates continuously to identify core objects and their neighborhoods until the iterative process concludes. Subsequently, the first cluster is identified. If any peers remain in the peer set N, the process continues with a second iteration until N is empty.

However, the recursive nature of DBSCAN necessitates multiple iterations, leading to a worst-case time complexity of $O(N^2)$. Consequently, in large blockchain networks, DBSCAN incurs significant computational costs in creating the propagation structure, ren-

dering any reduction in propagation latency insignificant. To address the aforementioned issue, we introduced DBSCAN++ within the framework of DC-SoC [28]. In the original DBSCAN algorithm, core objects are identified by traversing the entire peer set. However, this approach often results in a high sampling rate, unnecessary as the identification of cluster classes and construction of propagation structures typically do not require many core objects. Therefore, DBSCAN++ approximates the K-center algorithm through the use of a greedy initialization method. In this method, the peer with the highest latency from the current input peer is consistently selected during the core peer search and added to the set S until the number of peers in S meets the specified parameter m. The value of the parameter m is determined based on specific circumstances. Subsequently, peers in the set S are traversed to identify the core peer and its neighborhood, ensuring the computation process for high overhead density stays within m iterations. Lastly, the remaining outliers are incorporated into the neighborhood of the nearest core object via a KD tree. In conclusion, DBSCAN++ leverages the idea of K-center to reduce the sampling rate of core objects while maintaining effective neighborhood coverage in the space. Its time complexity is $O(mn)$, where n represents the total number of peers. Figure 2 illustrates the schematic diagram of the peer clustering process in DC-SoC.

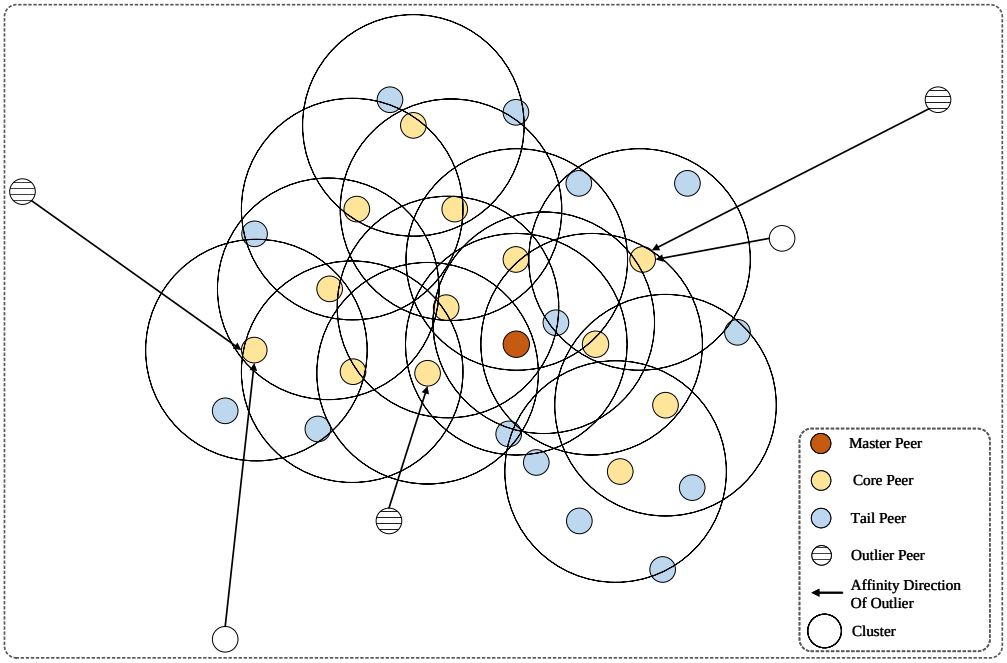


Figure 2. The process of density clustering.

The DBSCAN++ algorithm enhances the clustering process within DC-SoC by optimizing the sampling of core objects while ensuring comprehensive neighborhood coverage. In traditional decentralized systems, establishing an optimal network structure is essential for efficient block propagation. DBSCAN++ reduces the computational overhead typically associated with high-dimensional clustering, while also improving scalability and minimizing latency in the network. Specifically, the algorithm employs a greedy initialization technique that approximates the K-center method, dynamically selecting core nodes based on their network latency to neighboring peers. This dynamic selection process facilitates faster clustering, particularly in environments with large and constantly changing sets of peers.

In a decentralized context, where nodes frequently leave or join the network, DC-SoC incorporates a real-time dynamic clustering maintenance mechanism to ensure continuous network stability. When nodes become inactive or new nodes request to join, the system detects these changes and adjusts cluster configurations in real-time. For newly joining nodes, the algorithm assigns them to the nearest cluster based on a proximity-based metric, such as the Euclidean distance to the cluster's core node. If a core node becomes

inactive, the algorithm triggers an election process to select a new core node from the remaining cluster members, evaluated by their trustworthiness and performance within the network. Additionally, the system periodically checks cluster integrity to prevent cluster fragmentation or underutilization. If a cluster becomes too small or loses cohesion, it is integrated with a nearby cluster to maintain optimal propagation efficiency and network stability. This dynamic process ensures that DC-SoC sustains a resilient cluster architecture, even in highly dynamic environments with frequent changes in node availability. The specific process is detailed in Algorithm 1.

Algorithm 1 Dynamic Clustering Maintenance

Input: Set of ActiveNodes, Set of InactiveNodes, eps, $minPts$
Output: Updated Clusters

for each node $i \in$ ActiveNodes **do**
  if i is a CoreNode **then**
    Maintain current status as ClusterHead
  else if i has newly joined the network **then**
    Assign i to the nearest cluster C such that:
    $C = \arg\min(\text{distance}(i, \text{CoreNode_C}))$
    subject to: distance $\leq eps$, neighborhood size $\geq minPts$
  end if
end for
for each node $j \in$ InactiveNodes **do**
  if j is a CoreNode **then**
    Elect a new CoreNode from cluster C:
    $\text{CoreNode_C} = \arg\max(\text{Trustworthiness}(j))$
  else if j has rejoined the network **then**
    Reassign j to its previous cluster C_{prev} or to a new cluster C_{new} based on proximity and trust:
    $C_{\text{new}} = \arg\min(\text{distance}(j, \text{CoreNode_C}))$
    subject to: distance $\leq eps$, neighborhood size $\geq minPts$
  end if
end for
for each cluster C **do**
  if cluster C has insufficient members or is fragmented **then**
    Merge C with a neighboring cluster C_{neighbor}:
    $C_{\text{neighbor}} = \arg\min(\text{distance}(C, C_{\text{neighbor}}))$
  end if
end for

The DBSCAN++ algorithm is employed to identify clusters in the network. The primary variables that govern this clustering process are eps, which defines the radius for neighborhood detection, and $minPts$, the minimum number of nodes required to form a core object. These core objects form the basis of clusters, which are created around nodes that meet these density criteria. The number of clusters is determined dynamically based on the density of nodes and their network latency values. The clustering process adapts in real-time, adjusting to changes in node availability and network topology. As the number of peers or the network's geographic distribution changes, DBSCAN++ dynamically recalculates the number of clusters. Clusters are formed when a node's local neighborhood, defined by eps, contains at least $minPts$ nodes. Nodes that do not meet this criterion are treated as outliers but are still included in the network's propagation structure to ensure fairness and robustness.

3.3. Data Dissemination

Once the construction of the propagation structure is completed, DC-SoC initiates the broadcasting process. Broadcasting, which refers to the propagation of blocks, ensures the

synchronous update of ledger contents across all peers in the channel. It is important to note that the social mechanism operates continuously during the propagation process, and each propagation activity triggers the calculation of the corresponding social value. This value is essential for ensuring the reliability of block propagation. The detailed working principle of the social mechanism will be discussed in Section 3 and will not be further elaborated here. The process of block propagation is typically divided into the following three parts.

3.3.1. Push Mechanism

Once clustering and topology are established, all nodes within each cluster are organized in an adjacency list, where the core object serves as the header, collectively forming a directed acyclic graph. In the block propagation phase, the network can be conceptualized as a broadcast multi-way tree. Newly generated blocks are initiated at the master node, and then propagated layer by layer until all leaf nodes receive them. Leaf nodes refer to the tail-end nodes in the network structure. The following operational steps outline the block-push mechanism within this propagation structure:

- When the master node receives a new block from the ordering service, it records the block in its ledger and stores it in a buffer.
- Once the buffer is full or the push interval expires, the block is propagated to nodes in the current neighborhood and routing nodes in other clusters.
- Upon receipt, the nodes compute the sender's social credibility via a social mechanism, deciding whether to accept the block based on the sender's reputation level.
- The nodes then decrypt the hash value stored in the block header, using it to check their ledger for any prior receipt of the same block. If no corresponding record is found, the block content is recorded, effectively appending the block to the chain.
- If the current node is a core object, it continues to propagate the block to other nodes in the neighborhood until all core objects have completed their propagation tasks. Non-core nodes are only required to perform the block-chaining operation.
- Once all core nodes have completed propagation, they must send a confirmation message to their superior core object. If the master node confirms that it has received a sufficient number of confirmation messages, the push process is considered complete, and the block content is fully synchronized. Figure 3 presents a graphical illustration of the aforementioned operational example.

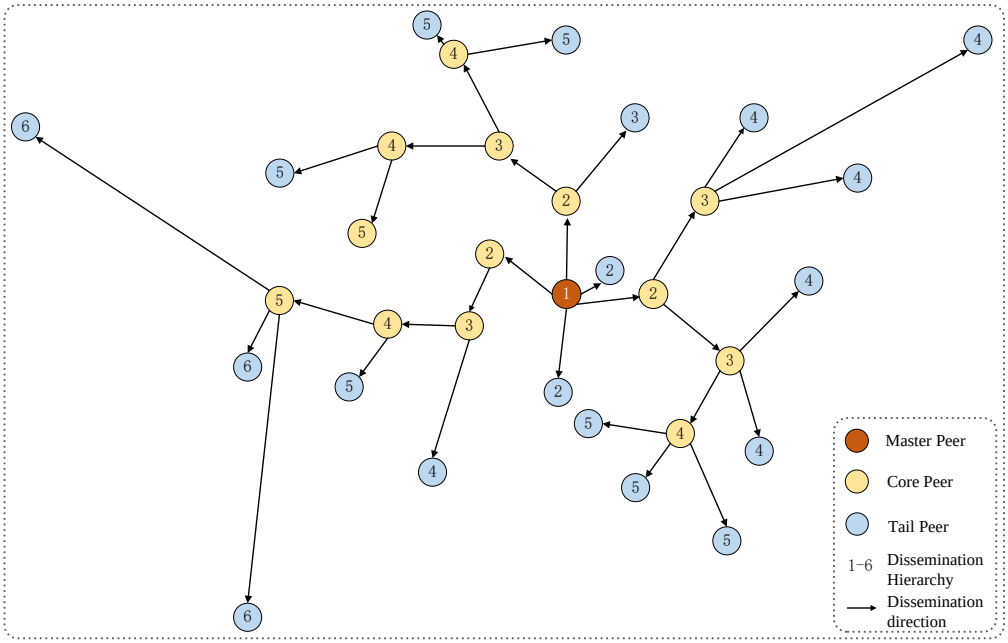


Figure 3. The process of pushing block in density structure.

3.3.2. Dynamic Network Self-Healing

Dynamic changes are inevitable during network operation, such as peers entering an inactive state or new peers requesting to join the network. Instability in the network naturally arises from changes in peer states, prompting DC-SoC to implement measures to maintain the stability of the propagation structure. Collectively, these maintenance measures constitute self-healing mechanisms.

Blockchain networks typically utilize a heartbeat mechanism to monitor peer offline status [29], and DC-SoC adheres to this practice. However, unlike traditional approaches, we posit that most offline peers are only temporarily in a pseudo-dead state and will return to normalcy within a period of time. Hence, to minimize network overhead, DC-SoC designates the offline peer a as deactivated, temporarily removing it from the propagation structure. If the heartbeat mechanism detects that peer a is back online within a fixed time period, it reinstates its propagation relationship within the cluster class in which it is located, instead of directly rejoining the network. If peer a does not come back online after the fixed time period, it is formally expelled from the network.

Changes in intra-cluster propagation relationships resulting from changes in peer states also require attention. If the deactivated peer a is a core peer, it becomes essential to transfer its core neighborhood affiliation to another peer within the same neighborhood and inform all parent peers of peer a in the propagation structure about the change in the core relationship of that neighborhood. The new core peer is then determined through an election mechanism, with users having the option to select either voting or automatic election. If the automatic election is used, the peer with the largest social value with peer a is selected as the new core peer. Upon peer a's return online within a predefined timeframe, it assumes the role of a regular leaf peer within the corresponding neighborhood. In the case of leaf peers, when a new peer requests to join the network, it is initially added randomly to a neighborhood within a cluster. During the subsequent network update phase, this new peer will be allocated to the appropriate cluster class and neighborhood.

3.3.3. Recovery Mechanism

Upon successful joining of a new peer to the network or reactivation of an inactivated peer, DC-SoC initiates a recovery mechanism to maintain ledger consistency with other peers. The traditional Gossip protocol includes two mechanisms, pull and recovery, for maintaining ledger consistency among peers. However, since the pull mechanism is activated periodically to compare the ledger with specific peers in the organization for missing blocks, and DC-SoC has eliminated isolated peer occurrences in the push phase through structured network topology, frequent activation of the pull mechanism has become unnecessary, and redundant. Therefore, in DC-SoC, only the recovery mechanism is retained. Upon activation of the recovery mechanism, DC-SoC selects multiple peers from the current organization and other organizations to request a batch of their missing blocks. The selection criterion for requesting peers is to choose the peer with the highest social value within a specific range. The recovery mechanism offers the advantage of reducing redundant operations through on-demand startup and improving fault tolerance by initiating requests to different organizations.

3.3.4. Network Update

Given the continuous changes in peer states within the network, the integrity of the existing propagation structure and the appropriateness of social value allocation progressively deteriorate. Therefore, DC-SoC defines a lifecycle called du. After each du, the currently online peers are considered, undergo re-clustering to update the propagation structure, and their social values are reallocated based on the current transaction records of each peer. Nevertheless, the frequency of social value updates does not necessarily align with propagation structure updates, given that changes in the community attributes of network peers occur more frequently than changes in their individual states. Additionally, a particular scenario must be taken into account during network updates: when a du cycle

concludes, if certain peers are still engaged in propagation tasks, the update process awaits their completion before proceeding. In the interim, if the master peer receives a new block while awaiting, it temporarily caches the block until the new propagation structure is established, after which it systematically pushes the cached blocks.

4. Social Mechanism

DC-SoC utilizes a community detection-based approach to allocate social credibility to individual nodes, thereby strengthening the network's overall robustness. Social credibility consists of two components: trustworthiness and economic interest value. Trustworthiness is further categorized into local and global dimensions. Local trustworthiness assesses the node's performance within its local neighborhood, using historical interactions, including data propagation success rate and response timeliness, as evaluation criteria. Global trustworthiness evaluates the node's performance across the broader network, considering interactions with nodes in different clusters and the success rate of cross-cluster propagation. Economic interest value reflects the node's overall contribution and resource investment within the network. Nodes that successfully propagate data or defend against malicious actions receive economic incentives, proportional to their role in the process. Social credibility represents the overall reliability of the node and the quality of the data it forwards, which directly influences the probability of other nodes engaging with it. A node's trustworthiness is dynamically adjusted based on its performance within the network, particularly when it participates in cross-cluster interactions or experiences changes in communication patterns. This dynamic adjustment mechanism enables the network to resist malicious nodes, such as Sybil and Byzantine attacks, thereby ensuring secure and accurate data propagation. The detailed calculation process is provided below.

4.1. Local Trustworthiness

Peers belonging to the same cluster foster localized trust. This is because these peers are part of the same neighborhood in the network topology and share a direct transactional relationship, facilitating the generation of trust. It is important to note that the term "transaction" here encompasses not only the exchange of valuable assets but also the dissemination and reception of data in general.

Prior to calculating the local trustworthiness, it is essential to determine the similarity between the two sets of transactional peers. Numerous social network researchers employ the number of neighboring peers as a criterion for assessing the similarity between peers and utilize the Pearson correlation coefficient method to quantify this similarity [26]. However, due to the utilization of the density clustering network structure in this study, which is not a pure P2P structure, certain scenarios involve unidirectional data propagation. For instance, when a core object disseminates blocks to the leaf peers within the cluster. These leaf peers, which serve as the tail peers in the network, typically have significantly fewer neighboring peers compared to the core object. Consequently, when applying the aforementioned calculation method under such circumstances, the resulting similarity between the two sets is likely to be lower. To mitigate this issue, this study utilizes the count of peers belonging to the same cluster class as the similarity discrimination parameter. The calculation method is illustrated in (1):

$$r(a,b) = \begin{cases} 0 & CL_{ab} = 0 \\ \frac{1}{1+e^{-x+t2}} & CL_{ab} > 0 \end{cases} \quad (1)$$

The variable $r(a,b)$ in (1) denotes the similarity between peer a and peer b, where $r(a,b) \in [0,1)$. Similarly, CL_{ab} represents the number of clusters that both peer a and peer b belong to. As the number of shared cluster classes increases, their similarity to each other also increases. The sigmoid function can represent the nonlinear relationship between these two variables.

Once the similarity between the trading parties is determined, the local trustworthiness can be assessed based on the number of successful and failed transactions between peers. The calculation method is presented in (2):

$$PT_{ab} = \begin{cases} r(a,b)\frac{S_{ab}-US_{ab}}{S_{ab}+US_{ab}}, & S_{ab}! = 0 \;||\; US_{ab}! = 0 \\ 0, & S_{ab} = 0 \;\&\&\; US_{ab} = 0 \end{cases} \tag{2}$$

Equation (2) defines $PTab$ as a measure of local trustworthiness, in which Sab represents the number of successful transactions, and $USab$ represents the number of failed transactions. Transaction failures may occur due to a peer becoming inactive or deliberately disseminating distorted messages. Crucially, the trustworthiness calculation is unidirectional and focuses solely on the message sender. Specifically, peer a exclusively records both Sab and $USab$ to assess the trustworthiness of peer b. Conversely, peer b logs only those parameters related to trustworthiness that pertain to peer a's role as the message sender, such as Sba, $USba$, $PTba$, etc. Furthermore, PT_{ab} is assigned a value of zero when there are no transaction records between the two peers.

4.2. Global Trustworthiness

The purpose of computing the global trustworthiness is to determine the overall trustworthiness of a peer within the social network. At this stage, peer b, regardless of its adjacency to peer a, can iteratively visit other peers to calculate its trustworthiness within the entire network using the density connectivity relationship established by DBSCAN. The specific calculation process is outlined in Algorithm 2.

Algorithm 2 Global Trustworthiness Calculation

Input: Algorithm's access object n, computation peer b, successful transaction count S_{sc}, iteration upper limit I_{ul};
Output: Global trust value GT_{ab};
Initialization: $sc \leftarrow 0$, $ul \leftarrow 0$, $GT_{ab} \leftarrow 0$, $n \leftarrow a$, weight $W_{tr} \leftarrow 1$;

```
Begin
while sc ≤ S_sc && ul < I_ul do
    NT ← 0, count ← 0
    for i ∈ neighbors of n do
        if PT_ib > 0 then
            NT ← NT + PT_ib
            sc ← sc + 1, count ← count + 1
            invQueue.push(pair(1, W_tr/2))
        end if
    end for
    NT ← NT ∗ (W_tr/count)
    GT_ab ← GT_ab + NT
    ul ← ul + 1
    if sc ≤ S_sc && ul < I_ul then
        n = invoQueue[0].first
        W_tr = invoQueue[0].second
        invokeQueue.erase(begin)
    end if
end while
return GT_ab
```

In addition to enhancing the fairness of the trust value mechanism, computing global trust plays another crucial role in facilitating the recovery mechanism, as discussed in the upcoming section. In order to ensure the integrity of the block content obtained through the recovery mechanism, DC-SoC typically initiates requests to peers from other organizations. Given that distinct organizations within the channel lack direct interactions, except for

communication facilitated through anchor peers, they maintain identical copies of the ledger. Hence, it is crucial to filter suitable peers, based on the computation of global trust, in order to initiate the recovery request. The final trustworthiness can be obtained by combining the local and global trusts using the calculation method specified in (3):

$$T_{ab} = \lambda \times PT_{ab} + (1 - \lambda)GT_{ab} \tag{3}$$

The confidence factor λ represents the weighting factor for local trustworthiness, allowing adjustment of the relative proportion of local trustworthiness and global trustworthiness. Typically, λ is assigned a value between 0.5 and 1. In practical applications, the value of λ can be adjusted based on specific requirements and needs.

4.3. Economic Incentive Value

Incentive mechanisms have been extensively employed in blockchain data-sharing schemes to enhance data reliability. In DC-SoC, economic interest value exhibits two distinct differences compared to trust value. Firstly, unlike the passive evaluation of peers in trust value, economic interest value assumes an "active" role for peers. This arises because each user, functioning as an independent economic entity, possesses self-interested attributes within the social network. By distributing benefits, the incentive mechanism motivates participants to enhance service quality and data reliability, leading to increased profits [27] and fostering a positive cycle of data sharing. Secondly, the economic benefits value emphasizes the degree of data authenticity rather than the stability of peers. For instance, in the coalition chain system, if a peer propagates distorted data, this behavior is quantified as a penalty value that is incorporated into the economic value calculation for that particular peer.

To comprehend the specific calculation of the economic benefits value, it is essential to introduce the definition of the combined trust value in (4):

$$CT_b = \frac{\sum_{i \in N} T_{ib}}{n} \tag{4}$$

N represents the set of neighboring peers of peer b, including peers with which it has had interactions, and n denotes the number of peers in set N. CT_b refers to the combined trust value of peer b, which is calculated as the average of the sum of trust values between peer b and all its interacting peers.

Afterward, DC-SoC defines the value of economic interest for a peer using (5):

$$E_{B_b} = \begin{cases} \frac{F}{\sum_{i \in N} CT_i} \cdot CT_b & CT_b > 0 \\ 0 & CT_b \leq 0 \end{cases} \tag{5}$$

EB_b denotes the value of the economic benefit associated with peer b. F is an initial benefit coefficient set to 1. As peer b disseminates increasingly distorted data in the network, F gradually converges to 0. Conversely, if peer b maintains honesty, F grows and ultimately converges to a constant value m. The specific value of m depends on the application. The variation of F, as determined by the curve, is typically described using sigmoid or arctan functional expressions. In summary, economic interest differs from trust value as it is a global variable. Regardless of the current transaction's counterpart, the economic interest of the peer remains constant and is updated in real-time based on the transaction.

Once the trust value and economic benefit value have been derived, the final socialization value can be calculated using (6):

$$SV_{ab} = EB_b * T_{ab} \tag{6}$$

SV_{ab} represents the social value resulting from the interaction between peer a and peer b. S consists of peer a's neighboring peers and the peers it has interacted with. Finally, by normalizing $SVab$, it can be transformed into the transaction probability between peer a

and peer b, which represents the likelihood of peer a accepting a transaction request from peer b.

The algorithm dynamically adjusts both a node's local and global trustworthiness based on its historical behavior and current performance. Local trustworthiness is derived from interactions with neighboring nodes, primarily considering the number of successful data transmissions, data accuracy, and response times. The more frequently a node interacts with highly trustworthy nodes while consistently providing reliable data, the higher its local trustworthiness becomes. In contrast, global trustworthiness is evaluated through cross-cluster interactions, where the success rate and stability of a node's data propagation across different clusters determine its global influence. Nodes demonstrating consistent performance gain higher global trustworthiness. Moreover, economic incentives are used to reward nodes that contribute to network security and propagation efficiency, which in turn enhances their social credibility. Conversely, malicious nodes experience a decrease in trustworthiness, which diminishes their influence within the network. Through this dynamic trustworthiness allocation mechanism, the system preserves reliability and security, even in network environments subject to frequent changes in node availability.

Social mechanisms are extensively employed in blockchain networks. Apart from identifying malicious peers, they can also serve the purpose of marking frequently inactive peers, guiding the assignment of cluster classifications to outlier peers, and recommending request objects to initiators of recovery mechanisms. The DC-SoC model's trust-based evaluation system is designed to mitigate the effects of Sybil attacks by requiring malicious nodes to build trust with multiple peers before gaining influence in the network. This makes it difficult for attackers to create multiple fake identities to compromise the system. Furthermore, the model's economic incentive mechanism discourages Byzantine behavior by penalizing peers who spread false information, ensuring that the network can maintain integrity and security even in the presence of adversarial nodes. Unlike traditional mechanisms, which often lack dynamic trust evaluation and robust economic incentives, the DC-SoC model ensures that malicious nodes cannot easily compromise the network's integrity.

5. Evaluation

This chapter focuses on evaluating the performance enhancement of DC-SoC in the Fabric federation chain system. In addition to the native Gossip protocol, this paper selects BlockP2P-EP and DC-Gossip for experimental comparison. The former utilizes the K-Means algorithm for peer clustering, while the latter, which is the predecessor of this paper, employs the OPTICS algorithm for peer clustering. Additionally, this paper employs the DC-SoC-DB algorithm. To objectively evaluate the performance differences among the different methods in Fabric network communication, this paper compares and validates the time consumed for peer clustering, the number of malicious transactions, the propagation delay, and the rate of incomplete block propagation across multiple dimensions. Among these, the incomplete propagation rate measures the proportion of blocks lost due to the propagation mechanism during a broadcast round.

5.1. Experimental Setup

The NS-3 [22] network simulator was employed to develop the simulation environment for this experiment, replicating a blockchain network environment running on Hyperledger Fabric version 2.2. The program was deployed in a virtual machine hosted on a server with an Intel Core i5-10400F CPU and 32 GB of DDR4 3200 MHz RAM. The virtual machine was operating on Ubuntu 20.04, with 8 GB of RAM and a dual-core processor allocated to it. At the experiment's onset, the network comprised 100 nodes, all connected in a peer-to-peer architecture, representing nodes within the same member cluster in Hyperledger Fabric.

Table 1 summarizes the basic parameters of the simulation experiment, including the network form, bandwidth, block size, interval of block sending, the initial number

of peers, latency range, etc. Specifically, the initial number of peers refers to the count of peers present within the channel at the start of the blockchain network. The latency range delineates the span of propagation delay values between Peers in the channel, signifying the magnitude of a vector space with time delay as the Euclidean distance. Moreover, *eps* and *minPts* correspond to the neighborhood radius and the minimum aggregate count of objects within the neighborhood in the DBSCAN algorithm, respectively. As this experiment is simulated using software, its fidelity to the real Fabric network is constrained by environmental conditions, resulting in certain parameter settings that may deviate from actual values. Nevertheless, the parameters presented in the Table 2 are grounded on authentic Fabric parameters and represent the most suitable values determined through iterative experiments and comparisons, thereby enabling effective observation of the pros and cons of distinct methods.

Table 2. Basic parameters setup of the experiment.

Bandwidth	Block Size	Interval of Block Sending	Initial Number of Peers	Latency Range	Forwarding Delay	Block Generation Interval	*eps*	*minPts*
5 Mbps	1 kb	15 ms	100	1–30 ms	0–3 ms	15 ms	3	10

5.2. Performance Analysis

From Figure 4, it is evident that the BlockP2P-EP data propagation model, derived from the DBSCAN clustering algorithm, and DC-Gossip, based on the OPTICS clustering algorithm, demonstrates greater clustering complexity. Furthermore, as the network size expands, the clustering time increases at an accelerated rate. In contrast, DC-SoC-DB substantially reduces clustering time within the DC-SoC model by employing a lower sampling rate for core objects. When the number of nodes reaches 300, its time consumption is only 68.2% of that of OPTICS, another density-based clustering algorithm. It is important to note that these experiments are conducted on a dynamic network to simulate real-world network scale-up scenarios. The experiments involve increasing both the number of Peers and the latency range between Peers. Specifically, the interval between each new peer joining ranges from 8 to 15 ms, the interval between inactivated peers appearances ranges from 10 to 20 ms, and the interval between re-connections of inactivated peers ranges from 15 to 25 ms. Please refer to Table 3 for additional parameter settings.

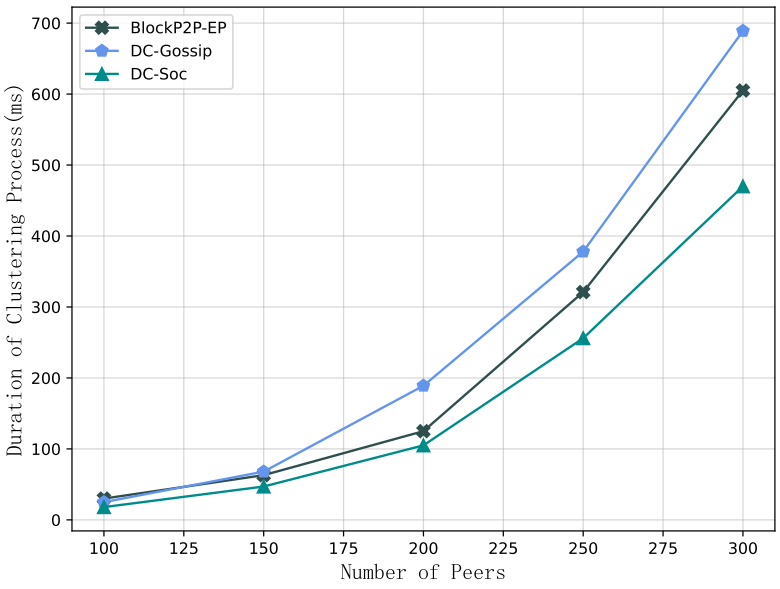
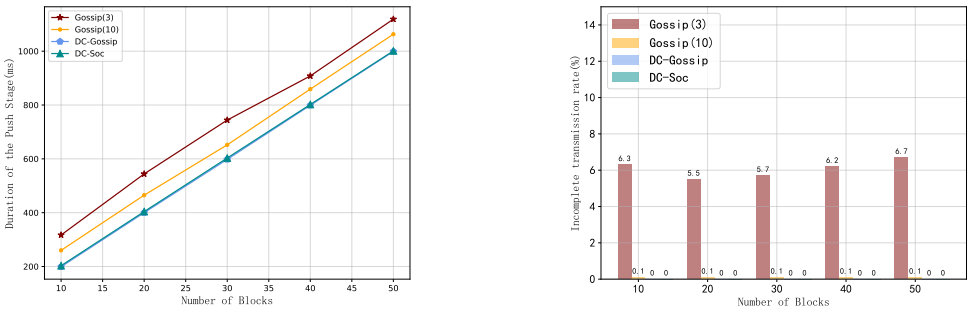


Figure 4. Comparison of clustering time consumption across different models.

Table 3. Parameters setup of network scale expanding.

Number of Peers	Latency Range	Number of New Peers	Number of Inactive Peers	Number of Blocks
100	1–30 ms	15	10	10
150	1–45 ms	20	15	10
200	1–57 ms	35	20	10
250	1–75 ms	30	25	10
300	1–95 ms	35	30	10

After confirming the advantages of DC-SoC in terms of clustering efficiency, it is important to assess the usability of the propagation structures it generates. Subsequent experiments will compare the performance of each propagation structure in different network scenarios, including static, dynamic, and varying network sizes. Figure 5 presents the test results in a static network. As the network state remains unchanged during the propagation process, this set of experiments enables a visual comparison of the performance advantages and disadvantages in the push phase. The analysis of Figure 5a clearly demonstrates that the network structure generated by the density clustering algorithm exhibits significantly faster pushing times compared to the network structure under the native Gossip protocol. Even when the Gossip protocol sets the f_{out} parameter to 10, requiring each peer to push blocks to 10 other peers, the process of pushing 50 blocks still takes approximately 10% longer than with the density structure. In practice, the f_{out} parameter is typically not set to 10 to prevent overloading individual peers within the network. Figure 5b illustrates that the native Gossip protocol encounters a considerable percentage of block loss during the push phase due to randomness propagation. In contrast, the density structure effectively eliminates this problem, ensuring the absence of any missed blocks. Notably, the performance of the identical density clustering algorithm closely resembles that of the resulting network structure obtained through OPTICS and DC-SoC-DB in a static network, displaying only a marginal disparity. Consequently, as depicted in Figure 5a, the curves of the two approaches exhibit an almost identical overlap.

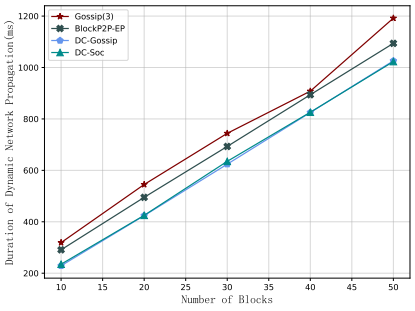


(**a**) Comparison of block propagation latency (**b**) Comparison of incomplete propagation rates

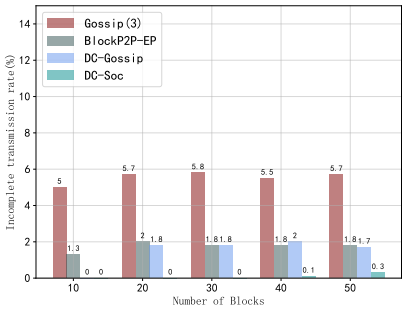
Figure 5. Performance comparison of the static network.

The test results in a dynamic network are presented in Figure 6. Figure 6a reveals that the propagation elapsed time for both density structures remains roughly equivalent during dynamic network states, with overlapping folding trends. Notably, both density structures exhibit shorter elapsed times compared to the use of K-Means and the native Gossip protocol. However, the propagation loss probability clearly demonstrates the superiority of the DC-SoC model under DC-SoC-DB dominance. In Figure 6b, it is evident that OPTICS exhibits a certain level of incomplete propagation rate in dynamic networks, roughly comparable to the level observed with K-Means. In contrast, DC-SoC-DB demonstrates an incomplete propagation rate of only 0.3% in the worst case, resulting in a significantly lesser negative impact on the network's normal operation compared to other algorithms.

Table 4 displays the remaining parameter settings for each group of performance tests conducted in the dynamic network.



(**a**) Comparison of block propagation latency



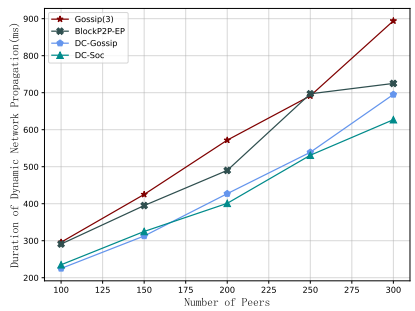
(**b**) Comparison of incomplete propagation rates

Figure 6. Performance comparison of the dynamic network.

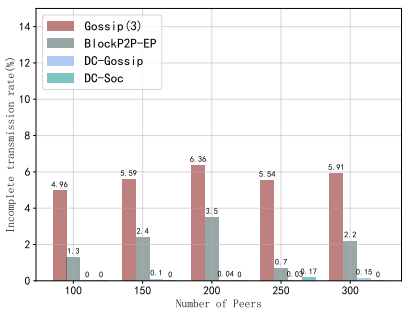
Table 4. Parameter setup for a dynamic network.

Number of Blocks	Number of Newly Joined Peers	Number of Inactive Peers
10	15	10
20	20	15
30	25	20
40	30	25
50	35	30

The test results under continuous changes in network size are depicted in Figure 7. The change in network size encompasses variations in both the number of peers and the distribution range of peers, with the distribution range referring to the range of propagation delay between peers. Figure 7a demonstrates that as the number of peers in the network increases, the variation in peer propagation elapsed time is considerably more stable under the density clustering structure compared to other structures, resulting in significantly lower elapsed times. Notably, DC-SoC-DB shows a clear advantage, particularly when the number of peers exceeds 300. The significant change observed in K-Means can be attributed to its implementation of secondary clustering within clusters to accommodate the larger number of peers in the network. Figure 7b highlights the propagation reliability of the DC-SoC model in large-scale networks. The incomplete propagation rate remains consistently near zero, demonstrating that DC-SoC maintains a high level of reliability, regardless of network size. The specific settings for the required parameters in this set of experiments are provided in detail in Table 3.



(**a**) Comparison of block propagation latency

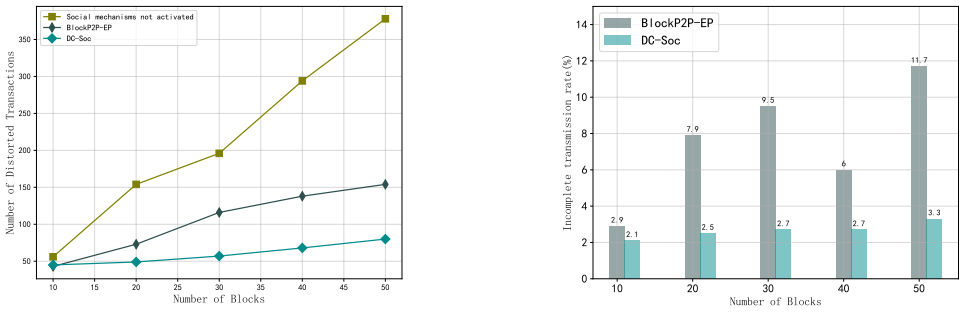


(**b**) Comparison of incomplete propagation rates

Figure 7. Performance comparison of network scale changing.

However, the assessment of data propagation reliability in a network cannot solely rely on the incomplete propagation rate. Given that the blockchain consists of numerous genuine user peers, the presence of malicious attackers in real-world applications is highly probable. To validate the robustness of the DC-SoC model against sophisticated attacks such as Sybil and Byzantine behavior, we conducted a series of experiments simulating environments with varying proportions of malicious nodes. A distortion transaction refers to a scenario where a peer accepts a false message from a malicious peer. Subsequently, upon successful propagation of the distorted message, the malicious peers collaborate to exploit the inconsistency in the ledger content and block sequence, thereby compromising the system's security. To enhance the complexity of the broadcasting environment and obtain objective test results, this set of experiments involves 300 peers in a dynamic network, utilizing the DC-SoC-DB clustering algorithm. Regarding the determination of the number of malicious peers, this paper adopts the experimental setup of malicious user detection researcher Yu et al. [30], wherein 1 malicious user is set for every 20 users. These malicious peers disseminate distorted information within the network at a specific frequency.

The results of the malicious behavior resistance test for the first set of experiments are depicted in Figure 8. Discerning the presence of malicious peers is challenging during the initial network startup phase due to the absence of transactions between peers. Consequently, the social mechanism requires a certain timeframe for the transaction data collection phase to become gradually effective. Based on the aforementioned premise, this experiment considers the initial 10 blocks of transaction propagation as the transaction data collection phase. Subsequently, upon the conclusion of the collection phase, the social value is formally employed to activate malicious blocking. Consequently, in Figure 8a, the number of distorted transactions in the network is identical when the social mechanism is activated and when it is not activated, at 10 blocks. Subsequently, it becomes evident that the activation of the social mechanism exerts a substantial defensive and preemptive effect against malicious behavior. In the absence of the enabled social mechanism, the number of distorted transactions attained by malicious peers in the network will rapidly escalate, thereby severely impeding the normal network operation. Enabling the social mechanism in DC-SoC reduces the number of distorted transactions in the network to approximately 21% of the original count after 50 blocks of propagation. Conversely, within the same network structure, the number of missed distorted transactions under the malicious peer detection mechanism employed in BlockP2P-EP is up to three times greater than that of DC-SoC. Additionally, as depicted in Figure 8b, DC-SoC consistently maintains the incomplete propagation rate at approximately 3% when confronted with malicious attacks. Conversely, the incomplete propagation rate of BlockP2P-EP exhibits an upward trend with the transactions, occasionally exceeding 10%. This discrepancy arises due to BlockP2P-EP's deficiency in effective measures to counter conspiracy attacks, leading to erroneous identification of certain normal peers as malicious ones and subsequent interception and failure of numerous secure transactions.

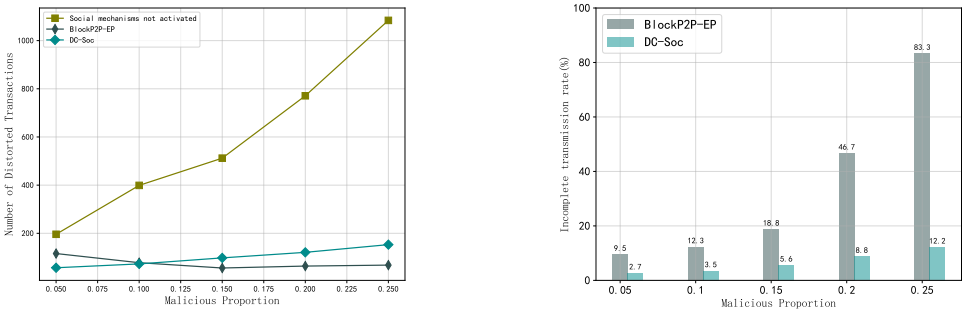


(**a**) Comparison of distorted transaction counts (**b**) Comparison of incomplete propagation rates

Figure 8. Comparison of malicious behavior resistance as the number of blocks rises.

When the ratio of malicious peers is low, most of the methods effectively detect and mitigate malicious behavior, thereby ensuring minimal impact on the overall network. Consequently, the second set of experiments employs the malicious peer ratio as the independent variable to assess whether the methods can maintain network availability as the malicious ratio increases.

The results of the malicious behavior resistance test for the second set of experiments are depicted in Figure 9. This group of experiments considers the propagation of 30 blocks as a complete transaction process. Analysis of Figure 9a reveals that in the absence of the enabled social mechanism, the number of distorted transactions accelerates as the proportion of malicious peers in the network gradually increases, with the count exceeding 1000 when the malice proportion reaches 25%. Conversely, DC-SoC effectively limits the count to 200, with a success rate of 85.9% in intercepting malicious attacks. While Figure 9a reveals that BlockP2P-EP demonstrates greater effectiveness in controlling distorted transactions, its performance deteriorates rapidly as the ratio increases. However, Figure 9b reveals that BlockP2P-EP lacks resilience against conspiracy attacks. As the malicious ratio surpasses 20%, the absence of countermeasures results in a breakdown of transaction relationships across the blockchain network. When the malicious ratio reaches 25%, all but five transactions are unable to conclude due to the lack of a robust defense mechanism. In contrast, DC-SoC demonstrates greater stability and reliability as the proportion of malicious nodes increases. The DC-SoC model not only maintains a low count of distorted transactions but also ensures a high propagation success rate, blocking over 72% of malicious transactions. This demonstrates DC-SoC's strong resistance to Sybil and Byzantine attacks, ensuring network reliability even under adverse conditions. Thus, while BlockP2P-EP may show superior initial performance, DC-SoC offers superior long-term resilience and security, particularly as the network scales or faces higher malicious peer ratios.



(**a**) Comparison of distorted transaction counts (**b**) Comparison of incomplete propagation rates

Figure 9. Comparison of malicious behavior resistance as the malicious ratio increases.

6. Conclusions

In this paper, we propose a new model of a federation chain network called DC-SoC, which incorporates the concepts of density clustering and social networks. The model is built upon the Hyperledger Fabric system and the Gossip broadcast protocol. DC-SoC, while ensuring network reliability, effectively enhances block propagation efficiency and coverage between nodes. By integrating trust-based and economic incentive mechanisms, it effectively prevents malicious behavior, mitigates Sybil and Byzantine attacks, enhances network security, and significantly improves the performance of the Fabric network. However, DC-SoC has certain algorithmic and experimental limitations, which will be addressed and optimized in future work. For example, the gap period required before enabling the current social mechanism could be shortened or omitted. Additionally, statistical analysis methods or machine learning techniques could be employed to conduct an in-depth analysis of peer behavioral characteristics when the proportion of malicious peers increases or when malicious behaviors become more complex, thereby enhancing network security.

Furthermore, it is crucial to deploy the network model in an actual Fabric environment to validate its practical applicability in real-world industrial settings.

Author Contributions: Conceptualization, X.D. (Xinhua Dong) and Z.X.; Methodology, X.D. (Xiaohong Dang) and K.Y.; Software, K.Y. and E.Z.; Validation, X.D. (Xinhua Dong) and H.H.; Formal analysis, H.H. and E.Z.; Investigation, X.D. (Xiaohong Dang) and K.Y.; Resources, X.D. (Xinhua Dong) and Z.X.; Data curation, X.D. (Xinhua Dong), X.D. (Xiaohong Dang) and K.Y.; Writing—original draft, X.D. (Xinhua Dong) and X.D. (Xiaohong Dang); Writing—review & editing, Z.X. and H.H.; Visualization, X.D. (Xinhua Dong), X.D. (Xiaohong Dang) and E.Z.; Supervision, Z.X. and H.H.; Project administration, X.D. (Xinhua Dong) and Z.X.; Funding acquisition, X.D. (Xinhua Dong), Z.X. and H.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Key-Area Research and Development Program of Hubei Province 2022BAA040; the Key-Area Research and Development Program of Guangdong Province (2020B1111420002); the Science and Technology Project of the Department of Transport of Hubei Province 2022-11-4-3; and the Innovation Fund of the Hubei University of Technology BSQD2019027, BSQD2019020, and BSQD2016019.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used to support the findings of this study can be obtained from the corresponding author upon request.

Conflicts of Interest: The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

1. Qammar, A.; Karim, A.; Ning, H.; Ding, J. Securing federated learning with blockchain: A systematic literature review. *Artif. Intell. Rev.* **2023**, *56*, 3951–3985. [CrossRef] [PubMed]
2. Hyperledger. Hyperledger Fabric v2.2 Documentation. 26 December 2020. Available online:https://hyperledgerfabric.readthedocs.io/zh_CN/release-2.2/ (accessed on 13 March 2024).
3. Yazdinejad, A.; Dehghantanha, A.; Parizi, R.M.; Srivastava, G.; Karimipour, H. Secure intelligent fuzzy blockchain framework: Effective threat detection in iot networks. *Comput. Ind.* **2023**, *144*, 103801. [CrossRef]
4. Chandan, R.R.; Balobaid, A.; Cherukupalli, N.L.S.; HL, G.; Flammini, F.; Natarajan, R. Secure modern wireless communication network based on blockchain technology. *Electronics* **2023**, *12*, 1095. [CrossRef]
5. Khan, Z.A.; Amjad, S.; Ahmed, F.; Almasoud, A.M.; Imran, M.; Javaid, N. A blockchain-based deep-learning-driven architecture for quality routing in wireless sensor networks. *IEEE Access* **2023**, *11*, 31036–31051. [CrossRef]
6. Xu, G.; Yao, T.; Zhang, K.; Meng, X.; Liu, X.; Xiao, K.; Chen, X. An Optimized Byzantine Fault Tolerance Algorithm for Medical Data Security. *Electronics* **2023**, *12*, 5045. [CrossRef]
7. Assiri, B. A Modified and Effective Blockchain Model for E-Healthcare Systems. *Appl. Sci.* **2023**, *13*, 12630. [CrossRef]
8. Spengler, A.C.F.; de Souza, P.S.L. The impact of using couchdb on hyperledger fabric performance for heterogeneous medical data storage. In Proceedings of the 2021 XLVII Latin American Computing Conference (CLEI), Cartago, Costa Rica, 25–29 October 2021; pp. 1–10.
9. Marson, G.A.; Andreina, S.; Alluminio, L.; Munichev, K.; Karame, G. MITOSIS: Practically scaling permissioned blockchains. In Proceedings of the 37th Annual Computer Security Applications Conference, Virtual Event, 6–10 December 2021; pp. 773–783.
10. Yi, H.; Li, Y.; Wang, M.; Yan, Z.; Nie, Z.An efficient blockchain consensus algorithm based on post-quantum thresholdsignature. *Big Data Res.* **2021**, *26*, 100268. [CrossRef]
11. Lu, J.; Chen, Y.; Cao, S.; Chen, L.; Wang, W.; Xin, Y. LEAP: Optimization Hierarchical Federated Learning on Non-IID Data with Coalition Formation Game. *arXiv* **2024**, arXiv:2405.00579.
12. Seid, A.M.; Lu, J.; Abishu, H.N.; Ayall, T.A. Blockchain-enabled task offloading with energy harvesting in multi-UAV-assisted IoT networks: A multi-agent DRL approach. *IEEE J. Sel. Areas Commun.* **2022**, *40*, 3517–3532. [CrossRef]
13. Demers, A.; Greene, D.; Hauser, C.; Irish, W.; Larson, J.; Shenker, S.; Terry, D. Epidemic algorithms for replicated database maintenance. In Proceedings of the Sixth Annual ACM Symposium on Principles of Distributed Computing, Vancouver, BC, Canada, 10–12 August 1987; pp. 1–12.
14. Kan, J.; Zou, L.; Liu, B.; Huang, X. Boost blockchain broadcast propagation with tree routing. In Proceedings of the International Conference on Smart Blockchain, Tokyo, Japan, 10–12 December 2018; pp. 77–85.
15. Hao, W.; Zeng, J.; Dai, X.; Xiao, J.; Hua, Q.; Chen, H.; Jin, H. BlockP2P: Enabling fast blockchain broadcast with scalable peer-to-peer network topology. In *Green, Pervasive, and Cloud Computing: 14th International Conference, GPC 2019, Uberlândia, Brazil, 26–28 May 2019, Proceedings 14*; Springer International Publishing: Berlin/Heidelberg, Germany, 2019; pp. 223–237.

16. Hao, W.; Zeng, J.; Dai, X.; Xiao, J.; Hua, Q.S.; Chen, H.; Jin, H. Towards a trust-enhanced blockchain P2P topology for enabling fast and reliable broadcast. *IEEE Trans. Netw. Serv. Manag.* **2020**, *17*, 904–917. [CrossRef]
17. Berendea, N.; Mercier, H.; Onica, E.; Riviere, E. Fair and efficient gossip in hyperledger fabric. In Proceedings of the 2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS), Singapore, 29 November–1 December 2020; pp. 190–200.
18. Oktian, Y.E.; Stanley, B.; Lee, S.-G. Building Trusted Federated Learning on Blockchain. *Symmetry* **2022**, *14*, 1407. [CrossRef]
19. Yang, W.; Hou, C.; Zhang, Z.; Wang, X.; Chen, S. Secure and Efficient Data Sharing for IoT Based On Blockchain and Reputation Mechanism. *IEEE Internet Things J.* **2024**, *11*, 20631–20647. [CrossRef]
20. AlSobeh, A.M.; Magableh, A.A. BlockASP: A Framework for AOP-Based Model Checking Blockchain System. *IEEE Access* **2023**, *11*, 115062–115075. [CrossRef]
21. AlSobeh, A. OSM: Leveraging Model Checking for Observing Dynamic 1 behaviors in Aspect-Oriented Applications. *arXiv* **2024**, arXiv:2403.01349. [CrossRef]
22. ns-3. ns-3 Tutorial [EB/OL]. 27 March 2023. Available online: https://www.nsnam.org/docs/tutorial/html/ (accessed on 14 March 2024).
23. Li, L.; Wan, J.; Liu, C. Access Control Strategy for the Internet of Vehicles Based on Blockchain and Edge Computing. *Electronics* **2023**, *12*, 4057. [CrossRef]
24. Razaque, A.; Bektemyssova, G.; Yoo, J.; Alotaibi, A.; Ali, M.; Amsaad, F.; Amanzholova, S.; Alshammari, M. Malicious Vehicle Detection Using Layer-Based Paradigm and the Internet of Things. *Sensors* **2023**, *23*, 6554. [CrossRef] [PubMed]
25. Xu, Z.; Ye, K.; Dong, X.; Han, H.; Yan, Z.; Chen, X.; Wang, H. DC-Gossip: An enhanced broadcast protocol in hyperledger fabric based on density clustering. In Proceedings of the International Conference on Wireless Algorithms, Systems, and Applications, Dalian, China, 24–26 November 2022; pp. 3–19.
26. Zhang, M.; Li, W.; Zhang, L.; Jin, H.; Mu, Y.; Wang, L. A Pearson correlation-based adaptive variable grouping method for large-scale multi-objective optimization. *Inf. Sci.* **2023**, *639*, 118737. [CrossRef]
27. Wu, S. X.; Wu, Z.; Chen, S.; Li, G.; Zhang, S. Community detectionin blockchain social networks. *J. Commun. Inf. Netw.* **2021**, *6*, 59–71. [CrossRef]
28. Jang, J.; Jiang, H. DBSCAN++: Towards fast and scalable density clustering. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 10–15 June 2019; pp. 3019–3029.
29. Shen, M.; Duan, J.; Zhu, L.; Zhang, J.; Du, X.; Guizani, M. Blockchain-based incentives for secure and collaborative data sharing in multiple clouds. *IEEE J. Sel. Areas Commun.* **2020**, *38*, 1229–1241. [CrossRef]
30. Yu, J.; Wang, K.; Li, P.; Xia, R.; Guo, S.; Guo, M. Efficient trustworthiness management for malicious user detection in big data collection. *IEEE Trans. Big Data* **2022**, *8*, 99–112. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.