

Article

H-PBFT: A Hierarchical and Credit-Aware PBFT Consensus Mechanism for Blockchain-Based Intelligent Transportation Systems

Zhenhua Wang ^{1,*}, Jiangang Hu ², Xinmeng Wang ¹, Zixuan Wang ¹, Haokun Yan ¹, Zheng Wu ¹, Mingxuan Liu ¹ and Zhen Tian ³

¹ College of Information Technology, Nanjing Police University, Nanjing 210023, China

² College of Public Security, Nanjing Police University, Nanjing 210023, China

³ James Watt School of Engineering, University of Glasgow, Glasgow G12 8QQ, UK

* Correspondence: wangzhenhua@aeu.edu.cn

Abstract

To address the risks of centralized single-point failures and data privacy leaks associated with massive data storage in Intelligent Transportation Systems (ITS) and Vehicle-to-Everything (V2X) environments, this paper proposes a distributed secure storage architecture based on blockchain. However, traditional Practical Byzantine Fault-Tolerant (PBFT) algorithms suffer from scalability bottlenecks in large-scale dynamic networks, such as high communication overhead and low consensus efficiency. Therefore, this paper designs a hierarchical and reputation-aware improved consensus mechanism (H-PBFT). First, this mechanism introduces a geographical location grouping strategy, dividing all network nodes into several local consensus groups and leveraging edge computing characteristics to achieve rapid consensus within each group. Second, a multi-dimensional reputation evaluation model (comprehensively considering historical behavior, performance, and availability) is constructed to dynamically elect representative nodes from each group to participate in global consensus, thereby significantly reducing the communication complexity from $O(N^2)$. Simulation results show that compared with standard PBFT, Q-PBFT, and APBFT, H-PBFT exhibits significant advantages in consensus latency, throughput, and view switching recovery time, and maintains high system robustness even in complex network environments with malicious nodes.

Keywords: H-PBFT; Practical Byzantine Fault-Tolerant; ITS; V2X

Academic Editor: Francesco Bucafurri

Received: 26 June 2026

Revised: 22 July 2026

Accepted: 27 July 2026

Published: 30 July 2026

Copyright: © 2026 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 \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

In modern intelligent transportation and vehicle-to-everything environments, massive amounts of vehicle terminals continuously generate multi-dimensional dynamic data containing vehicle location information, driving trajectories, and user identity data [1]. This type of data is highly time-sensitive and sensitive, forming a crucial foundation for intelligent scheduling, route planning, and traffic management [2]. However, once vehicle location data and user information are leaked or maliciously exploited, it not only exposes user travel patterns but may also pose serious risks such as personal safety and privacy violations. Therefore, how to ensure data availability while achieving secure data storage

and privacy protection has become one of the key issues urgently needing to be addressed in the V2X field [3,4].

Currently, most V2X data management systems still adopt a centralized architecture, with data stored and processed uniformly by cloud servers or traffic management centers [5]. While this model facilitates centralized management, it has significant structural flaws: on the one hand, if the central node is attacked or malfunctions, it can easily lead to a single point of failure, causing overall system service interruption; on the other hand, centralized storage results in excessive concentration of data control, making it difficult to prevent internal abuse or external tampering at the mechanism level, and data integrity and reliability cannot be effectively guaranteed [6,7]. Figure 1 illustrates the vulnerability of data storage in intelligent transportation systems to single points of failure. Against this backdrop, constructing a traffic data storage mechanism with decentralized, verifiable, and tamper-proof characteristics is of significant research importance.

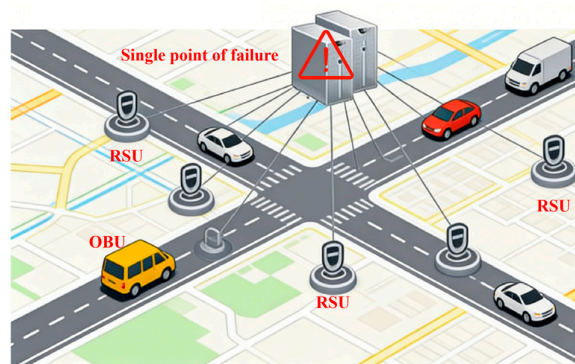


Figure 1. Diagram of Single Point of Fault Storage in Intelligent Transportation Systems.

Blockchain technology, as a distributed ledger technology, achieves the immutability and traceability of data through a chain-like data structure and cryptographic mechanisms, providing a new technological path to solve the trust problem in traditional centralized systems [8]. Participating nodes in a blockchain network jointly maintain a copy of the ledger through a consensus mechanism, thereby avoiding absolute control of data by a single entity. Currently, blockchain has been widely used in supply chain management, fintech, and the Internet of Things, and its decentralized nature makes it well-suited for secure data storage in the Internet of Vehicles environment [9]. Therefore, designing a blockchain data storage framework for the IoV environment has become an effective approach to improve traffic data security and system reliability.

However, the Practical Byzantine Fault Tolerance consensus algorithm, widely used in consortium blockchains or permissioned blockchains, faces the problem of rapidly increasing communication complexity as the number of nodes expands. PBFT's message exchange process requires multiple rounds of broadcasting between nodes, and its communication overhead increases quadratically with the number of nodes, leading to performance bottlenecks such as increased latency and decreased throughput in large-scale network environments. This problem is particularly prominent in vehicle-to-everything (V2X) scenarios with a large number of vehicle nodes and dynamically changing network environments, severely restricting the practical deployment of blockchain technology in traffic data systems.

To address this issue, this paper constructs a blockchain architecture for secure storage of V2X data, enabling decentralized management and distributed storage of vehicle location information and related user data. Building upon this, and addressing the low consensus efficiency of the traditional PBFT algorithm in large-scale node environments, an improved PBFT consensus mechanism is proposed. By optimizing node participation methods and message interaction processes, communication overhead is reduced and overall system performance is improved, thereby enhancing the scalability and practicality of blockchain in V2X scenarios. The main contributions are as follows:

1. To address the high communication overhead of the traditional PBFT consensus algorithm in large-scale node scenarios, a geographic location-based grouping H-PBFT architecture is proposed, transforming the traditional network-wide consensus into a two-layer cascaded mode of “intra-group plus inter-group”.
2. The traditional random or round-robin election method is abandoned, and a ternary evaluation model integrating historical credit, hardware performance (CPU/bandwidth), and online availability is constructed.
3. Simulation experiments comparing H-PBFT with Q-PBFT, PBFT, and APBFT consensus algorithms show that H-PBFT exhibits strong adaptability in dynamic network environments and under conditions of malicious node interference.

2. Related Work

The integration of blockchain technology into Intelligent Transportation Systems (ITS) and vehicular networks has attracted increasing research attention, particularly for enhancing data security, integrity, and distributed trust management. In such decentralized environments composed of autonomous vehicles and edge devices, consensus mechanisms play a pivotal role in maintaining system reliability. Among them, Practical Byzantine Fault Tolerance and its variants have been extensively studied due to their low latency and strong fault tolerance in permissioned blockchain settings. Several studies emphasize the importance of PBFT-based consensus in intelligent device ecosystems. Yu et al. [10] underline the demand for high-performance blockchain platforms tailored to intelligent devices and propose a PBFT-DPOC-based framework to support decentralized autonomy. Their work highlights the challenge of efficiently managing and utilizing large-scale IoT data, demonstrating that reliable consensus mechanisms are fundamental to sustaining secure and coordinated device interaction. Beyond general IoT systems, PBFT adaptations have also been explored in communication infrastructures. Tong et al. [11] develop a Trust-PBFT algorithm within a blockchain-enabled Cloud Radio Access Network (C-RAN) architecture for 5G environments. By incorporating trust evaluation and a tri-chain structure, their scheme mitigates insider attacks while improving scalability and transaction traceability, illustrating how PBFT variants can be extended to high-throughput and complex network scenarios. In ITS-specific applications, blockchain is increasingly leveraged to safeguard traffic data. Medel et al. [12] investigate blockchain-based ITS architectures to counter data integrity attacks, emphasizing the necessity of secure and dependable consensus protocols. Similarly, Lee et al. [13] design a two-layer blockchain credit mechanism for managing local traffic information, which reduces memory overhead and enhances system responsiveness. These studies confirm that consensus design is critical to maintaining trustworthy data exchange in transportation systems.

To address the unique requirements of vehicular networks, recent research has focused on improving the efficiency and security of PBFT-like algorithms. Xu et al. [14] propose SG-PBFT, a secure and high-efficiency consensus protocol tailored for the Internet of Vehicles, achieving performance gains while strengthening security guarantees. Kumar et al. [15] introduce R-PBFT, further emphasizing intelligent and secure consensus strategies for dynamic vehicular environments. These efforts reflect a shift toward domain-specific optimization of classical PBFT. PBFT improvements have also been applied in broader service and control system contexts. Zhang et al. [16] present an enhanced PBFT mechanism for Service Function Chain (SFC) deployment, reporting improvements in resource utilization, service quality, and system security. Although not limited to transportation, this work demonstrates the adaptability of refined PBFT protocols to complex, large-scale service orchestration scenarios. Moreover, blockchain-supported intelligent traffic management frameworks are emerging. Wang et al. [17] combine blockchain with federated reinforcement learning for traffic trajectory prediction. While their approach does not directly modify PBFT, it reflects the broader trend of integrating blockchain-based trust and coordination mechanisms into traffic analytics systems. Finally, V et al. [18] propose the EQPBFT algorithm for Vehicular Ad Hoc Networks (VANETs), enhancing network resilience through effective identification and isolation of malicious nodes during block generation. This study exemplifies how PBFT-inspired designs can strengthen trust management and security in vehicular communication environments.

Collectively, these studies demonstrate that PBFT-based and mainstream BFT protocols improve consensus through different mechanisms, including trust-based node selection, score-based grouping, malicious-node management, chained quorum certificates, and voting-power-based validation, as summarized in Table 1. Trust-PBFT, SG-PBFT, and EQPBFT mainly enhance trust management, grouping efficiency, and network security, whereas HotStuff and Tendermint focus on general-purpose scalability and deterministic finality. However, these protocols do not jointly consider the spatial locality of traffic data and the heterogeneous credit, performance, and availability of RSU nodes. To address this gap, H-PBFT combines physical geographical grouping with multidimensional representative-node selection and hierarchical local–global consensus for large-scale V2X environments.

Table 1. Comparison between H-PBFT and representative PBFT variants.

Protocol	Target Scenario	Grouping	Node-Evaluation Dimensions	Main Advantage
Trust-PBFT [11]	Blockchain-enabled 5G C-RAN	High-trust nodes are selected using PeerTrust	Primarily peer trust	Reduces the influence of unreliable or malicious internal nodes and improves consensus reliability
SG-PBFT [14]	Internet of Vehicles	Nodes are grouped using a score-based mechanism	Node score	Reduces the number of nodes participating in large-scale message exchange and improves consensus efficiency
EQPBFT [18]	VANETs	Quantified role-based node management	Role, message reliability, and security status	Enhances malicious-node detection, identity authentication, and network resilience
HotStuff [19]	General-purpose BFT state-machine replication and blockchain consensus	No explicit geographical or hierarchical grouping	No built-in multi-dimensional node score; replicas participate according to the	Achieves linear communication in the normal case and simplifies leader replacement

configured committee				
Tendermint [20]	Blockchain state-machine replication with a validator set	No explicit geographical grouping	Validator voting power; proposer selection is related to voting power	Provides deterministic finality and practical Byzantine fault tolerance for blockchain systems
H-PBFT	V2X and ITS	Physical geographical grouping followed by multidimensional representative election	Historical credit, network performance, and availability	Exploits traffic-data spatial locality while adapting representative selection to heterogeneous RSU conditions, thereby reducing unnecessary cross-region communication

3. Materials and Methods

3.1. Blockchain Overview

Blockchain is a decentralized, immutable distributed ledger technology (DLT) used to record, verify, and share data without relying on a centralized authority [21]. As a shared digital ledger, blockchain provides a single source of truth in the network, enabling transparent and traceable recording of transactions and asset transfers within a business network [22].

Blockchain technology was first proposed by Satoshi Nakamoto in 2008 and first implemented in the Bitcoin system [23]. As a decentralized peer-to-peer electronic cash system, the Bitcoin blockchain verified the feasibility of value exchange without a trusted third party and laid the foundation for the subsequent development of blockchain technology in a wider range of applications. From a technical architecture perspective, blockchain is essentially a distributed database system [24]. Ledger data is replicated and stored on multiple nodes in the network, making it difficult for any single point of failure or malicious tampering to affect the correctness of the overall system. Transactions within the system must be verified through a consensus mechanism, which ensures state consistency among nodes that are not fully trusted, thereby guaranteeing the integrity and consistency of the ledger data [25].

At the data structure level, the blockchain packages transactions into blocks in chronological order and links these blocks together using a cryptographic hash function, forming a chain structure [26]. Each block contains the hash value of the previous block, ensuring that any tampering with historical data will destroy the structure of subsequent blocks and be detected by the network [27]. This chain-like cryptographic structure gives the blockchain high tamper resistance and auditability, making it widely applicable in areas such as cryptocurrency, supply chain management, digital identity, the Internet of Things, and medical data sharing.

One of the key advantages of blockchain is its decentralized nature. The system establishes trust through algorithms and cryptographic mechanisms, without relying on traditional intermediaries such as banks, clearinghouses, or third-party platforms, thereby improving transaction transparency, reducing operational risks, and minimizing fraud and errors caused by human intervention. For industries with high requirements for data security and trustworthiness, such as finance, healthcare, energy, and government, blockchain technology has significant potential to improve process efficiency, enhance accountability, and reduce operating costs.

The classification of blockchain networks is essentially based on three core dimensions: the degree of openness to participation, the method of node permission control, and the network governance structure. First, regarding the openness of participants, a network can be completely open (anyone can join) or have restricted access. Second, from a permission management perspective, the system can control the identity of nodes, their consensus participation rights, and data access rights to varying degrees. Finally, in terms of governance models, the network may be managed by a single organization, jointly managed by multiple organizations, or completely decentralized and autonomous. Based on the differences in the combination of these three dimensions, different types of blockchains have emerged, such as public blockchains, private blockchains, permissioned blockchains, and consortium blockchains. This classification reflects the trade-off between the degree of decentralization and controllability of the system. Different types of blockchains cater to different trust environments and application needs, and are essentially a structured division of network trust models and management models. Table 2 shows a comparison of different blockchain network types.

Table 2. Comparison of Different Types of Blockchain Networks.

Type	Node Admission	Performance Characteristics	Typical Application Scenarios
Public Blockchain [28]	Permissionless participation	Strong decentralization but low throughput and high latency	Bitcoin, Ethereum, Decentralized Finance (DeFi)
Private Blockchain [29]	Internal authorization required	High performance, low latency, strong controllability	Enterprise auditing, internal data management
Permissioned Blockchain [30]	Identity authentication and authorization required	High security with flexible management	Enterprise blockchain systems, e-government platforms
Consortium Blockchain [31]	Admission approved by consortium members	Balance between performance and decentralization	Supply chain finance, energy data sharing, cross-institution collaboration

3.2. PBFT Consensus Algorithm Overview

The Practical Byzantine Fault Tolerance (PBFT) algorithm requires neither equity staking nor intensive computational resources, thereby significantly reducing the cost of malicious attacks [32]. By employing a three-phase voting protocol, PBFT can effectively tolerate Byzantine failures and maintain consensus correctness as long as the number of malicious nodes does not exceed one-third of the total network. Despite these advantages, the conventional PBFT algorithm suffers from excessive communication overhead and random master node selection, both of which substantially reduce consensus efficiency and limit its scalability [33].

The PBFT consensus protocol consists of three types of nodes: client nodes, a master node, and multiple replica nodes. Client nodes initiate transaction requests and receive the final consensus results. The master node is responsible for receiving client requests, assigning sequence numbers, and broadcasting signed requests to replica nodes. Replica nodes verify the received messages and collaboratively determine whether consensus has been reached. If the master node becomes faulty or fails to respond within a specified timeout, a view-change mechanism is triggered to elect a new master node, thereby ensuring continuous and reliable consensus execution.

The execution process of the PBFT algorithm can be divided into five main stages: Request, Pre-prepare, Prepare, Commit, and Reply. The overall process is shown in Figure 2, and is detailed below:

- (1) Request Phase: The client sends a consensus request message to the master node, in the format $\langle \text{Request}, m, t, c, \sigma_c \rangle$, where m represents the request content, t is the timestamp, c is the client identifier, and σ_c is the client signature.
- (2) Pre-prepare Phase: After receiving the request, the master node numbers and sorts it, then generates a pre-prepare message $\langle \text{Pre-prepare}, v, n, D(m), \sigma_p \rangle, m \rangle$, and sends it to all replica nodes. Here, v is the current view number, n is the message sequence number, $D(m)$ is the message digest, and σ_p is the master node signature.
- (3) Prepare Phase: After receiving the pre-prepare message, the replica nodes verify its validity and generate a prepare message $\langle \text{Prepare}, v, n, D(m), b_i, \sigma_b \rangle$ for broadcast, where b_i is the replica node identifier, and σ_b is the node's signature. In this phase, each replica node compares its generated preparation message with those sent by other replica nodes. When a node receives at least $2f + 1$ consistent preparation messages, it can proceed to the next phase. This phase helps screen for Byzantine malicious nodes in the network.
- (4) Commit Phase: After the preparation conditions are met, each node generates a commit message $\langle \text{Commit}, v, n, D(m), b_i, \sigma_b \rangle$ and broadcasts it. When a node receives at least $2f + 1$ commit messages consistent with its own, it indicates that consensus has been largely reached, and it can proceed to the response phase.
- (5) Reply (Response Phase): After confirmation in the commit phase, the node sends a response message $\langle \text{Reply}, v, n, \sigma_b, r \rangle$ to the client, where r represents the result after executing the request. The client considers its request to have been successfully completed when it receives at least $f + 1$ consistent response messages.

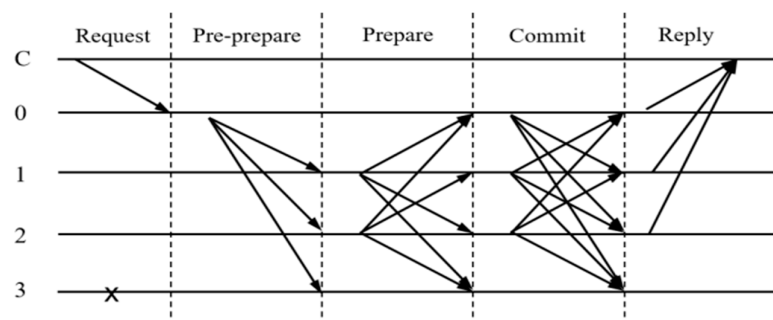


Figure 2. PBFT consensus protocol process diagram.

4. Blockchain-Based Intelligent Transportation System Architecture

In traditional traffic control systems, vehicle operation records are typically stored and managed in a centralized manner. While this architecture facilitates unified scheduling, it suffers from significant security flaws. If the central server is maliciously attacked or experiences a system failure, it is highly susceptible to serious problems such as data forgery, illegal tampering, and even complete data loss, with data recovery being extremely difficult. Furthermore, vehicle trajectories and user identity information are highly sensitive, and the lack of effective privacy protection mechanisms under a centralized management model makes them easy targets for attackers to steal and abuse. To address these issues, this paper introduces blockchain technology into the traffic signal control scenario. Blockchain employs a distributed storage architecture, with data maintained collaboratively by multiple nodes, avoiding the risk of single points of failure. Simultaneously, blockchain organizes data through a chain structure and timestamp mechanism.

Each block contains the hash value of the previous block and is encrypted using the SHA-256 algorithm. When a block of data is illegally modified, its hash value changes, affecting subsequent blocks, thus being quickly identified by other nodes in the network, ensuring the immutability and integrity of the data.

As shown in Figure 3, the proposed intelligent transportation system adopts a four-layer structure design, including an application layer, a contract layer, a consensus layer, and a data layer, to realize the complete process of data collection, on-chain processing, distributed consensus, and secure storage.

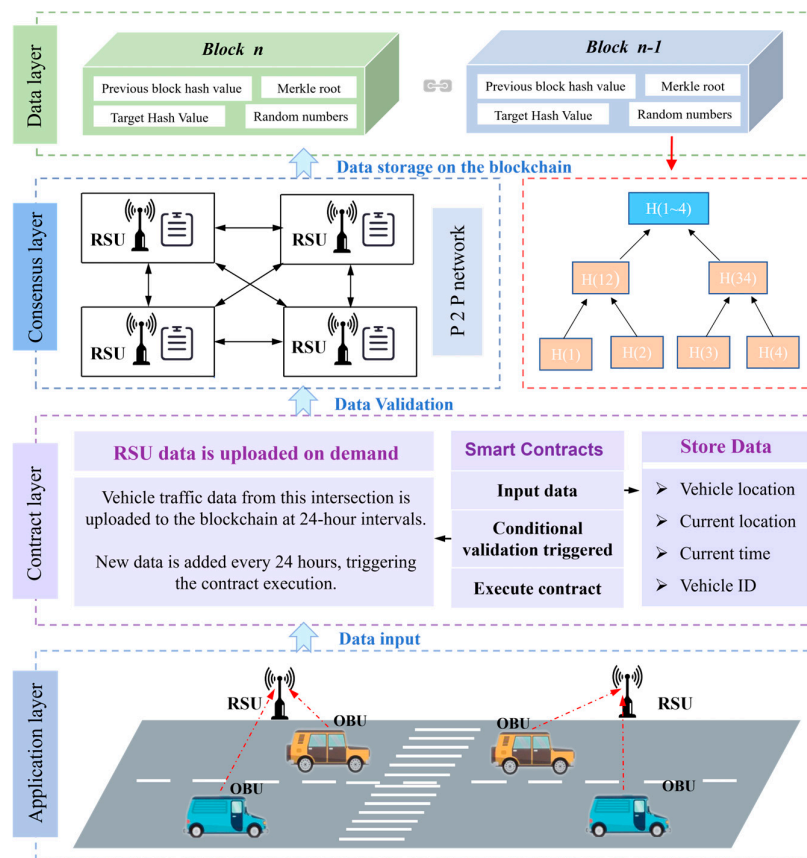


Figure 3. Architecture diagram of a blockchain-based intelligent transportation data storage system.

4.1. Application Layer

The application layer primarily consists of Roadside Units (RSUs) and On-Board Units (OBUs), serving as the system's data acquisition and upload entry point. As a key roadside device in the vehicle-road cooperative system, the RSU needs to maintain a unified communication protocol with the OBU to ensure real-time reception of dynamic data such as vehicle speed, location, and direction of travel. The OBU should possess local data recording and wireless transmission capabilities. To ensure stable system operation, the application layer must meet the following conditions:

A unified time synchronization mechanism should be implemented at the device level, along with necessary network security protection capabilities;

The software system must be compatible with existing traffic management platforms and comply with relevant technical standards and regulations;

Data formats should be standardized, and a regular maintenance mechanism should be established to ensure system interoperability and secure encrypted communication.

Assuming stable communication between the RSU and OBU, each intersection's RSU filters the collected vehicle data and uploads it as needed according to a predefined strategy. This paper sets the RSU to submit data to the blockchain network every 24 h for traffic condition analysis and traffic light cycle estimation. Considering the block size limit (approximately 1 MB per block), the system only uploads necessary fields, including vehicle identification ID, location coordinates, and timestamps.

4.2. Contract Layer

Traditional contract execution relies on trusted third-party supervision and arbitration, while smart contracts execute automatically on the blockchain through programmatic rules, possessing the characteristics of openness, transparency, and immutability. Participating nodes can complete trusted interactions based on trust in the code logic, thereby reducing human intervention and execution deviations. In this system, the data uploaded by RSUs is first verified for legality and integrity by the blockchain smart contract. As a core functional module of the blockchain, the smart contract layer also defines on-chain data access permissions, allowing only authorized nodes such as traffic management departments to retrieve vehicle data within a specific range. When new vehicle data is written to the chain, the smart contract automatically triggers the estimation process, realizing the automated operation of the system.

4.3. Consensus Layer

Traffic-related data must be unanimously approved by network nodes through a consensus mechanism before it can be written to the blockchain. This system adopts a consortium blockchain architecture and uses the Practical Byzantine Fault Tolerance (PBFT) algorithm as its consensus mechanism. Network nodes consist of RSU devices and entity nodes from traffic management departments. The PBFT algorithm standardizes the block generation and propagation process. Each node verifies the block data through multiple rounds of message interaction to ensure the authenticity of the data source and the integrity of the content. After consensus is reached, data copies are synchronously stored in the ledgers of all participating nodes, achieving distributed persistent storage and thus avoiding system crashes caused by single points of failure in centralized systems.

4.4. Data Layer

The data layer consists of the underlying blockchain database, responsible for storing data processed by smart contracts and confirmed through PBFT consensus, including basic vehicle information and estimated traffic light cycles. All data is written to the chain in block form, with copies maintained in each node's local ledger. To further enhance data integrity and security, each block contains a Merkle root hash, representing a summary of all transaction data within the block. Any minor modification to the block data will cause a change in the Merkle root, which will be detected and rejected by other nodes, ensuring data immutability and overall system security from a mechanism perspective.

5. Layered H-PBFT Consensus Algorithm Design

In intelligent transportation scenarios, roadside units typically handle data collection and participate in consortium blockchain consensus, such as traffic flow, accident events, congestion indices, and vehicle-to-everything perception results. This type of data is characterized by strong regionality, high timeliness, and significant dynamic load fluctuations: on the one hand, RSU observation data within the same region are highly correlated; on the other hand, data correlation between different regions is relatively weak, and cross-regional verification often only occurs at boundaries or in global statistical tasks. Therefore, incorporating all RSUs into a single-layer Practical Byzantine Fault Tolerance leads

to significant communication and computational redundancy. Assuming there are n RSU nodes in the system, PBFT within a view includes pre-prepare, prepare, and commit phases. For a broadcast message model, typical communication volume can be approximated as:

$$C_{\text{PBFT}}(n) = \delta n + \theta n(n-1) \in O(n^2) \quad (1)$$

where δn can represent the overhead of the primary node broadcasting the pre-preparation message, and $\theta n(n-1)$ represents the overhead of the full interconnect broadcast during the preparation/commit phase (the constant term is related to the specific implementation, but the growth order is $O(n^2)$).

In city-scale deployments, the number of RSU nodes n increases rapidly with growing RSU density, leading to several critical challenges, including excessive communication overhead, poor system scalability, redundant consensus operations, inefficient resource utilization, and insufficient incentive mechanisms. To address these issues, this paper proposes a Hierarchical Practical Byzantine Fault Tolerance consensus mechanism. The proposed H-PBFT framework decomposes the global consensus process into two hierarchical layers, namely intra-group (local) consensus and inter-group (global) representative consensus, by leveraging geographic grouping and representative node election. Exploiting the inherent spatial locality of traffic data, RSUs within geographically adjacent regions first achieve local consensus. Only the consensus summaries and corresponding cryptographic proofs are subsequently submitted to the global layer for final agreement. Through this hierarchical consensus design, the communication scale is effectively reduced from being dependent on the total number of nodes n to a function of the group size and the number of groups, thereby significantly mitigating communication complexity and improving scalability in large-scale intelligent transportation systems.

Figure 4 illustrates the layered architecture and execution flow of the H-PBFT consensus mechanism, aiming to balance consensus efficiency and security in the connected vehicle environment. This architecture consists of three core parts: the edge layer, the first-stage local consensus, and the second-stage global consensus. At the edge layer, raw data generated by intelligent vehicles and IoT sensors (indicated by yellow arrows) is transmitted in real time to Roadside Units. Based on a geographic location strategy, RSUs are divided into multiple local consensus groups, each composed of a leader and members selected by a reputation mechanism. The first-stage local consensus reduces latency and communication overhead through a simplified single-stage mechanism. The leader node initiates a proposal, and group members confirm and generate a local digest, ensuring data consistency. The second-stage global consensus is formed by a global consensus committee composed of representative nodes from each local group. Representative nodes participate sequentially in the entire process from Request to Reply (including Pre-prepare, Prepare, and Response stages), handling cross-regional transaction ordering and verification. Finally, the verified data is packaged into blocks, containing key information such as the previous block hash and Merkle root, achieving distributed storage.

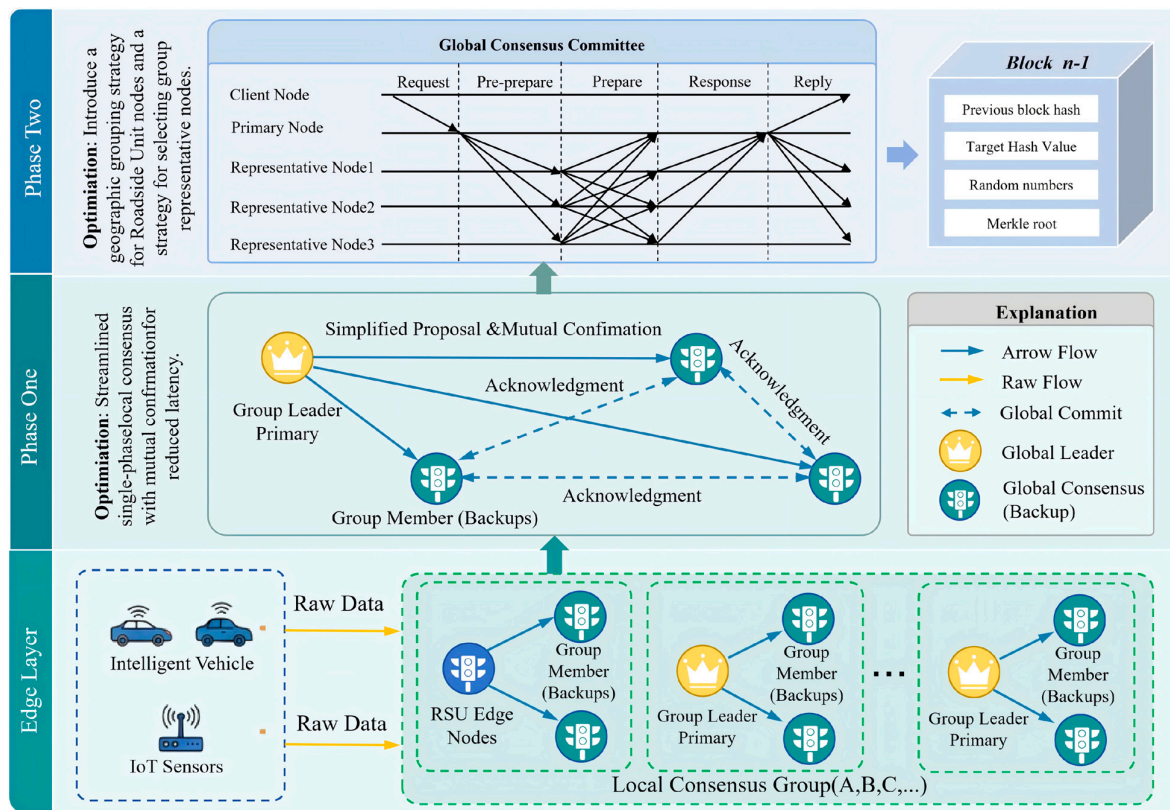


Figure 4. Layered architecture and execution workflow of the H-PBFT consensus mechanism.

5.1. RSU Node Geographic Grouping Strategy

This paper adopts a grid-based geographic grouping strategy, dividing the urban transportation network into several regions according to latitude and longitude or administrative boundaries, with each region corresponding to a consensus group. The geographic coordinates of RSU node r are denoted as $P_r = (lat_r, lon_r)$. The overall deployment area boundary is defined as:

$$lat = [min_lat, max_lat], \quad lon = [min_lon, max_lon] \quad (2)$$

Let the grid side length be Δ (corresponding to your original grid_size), then the number of grid rows and columns is:

$$N_{row} = \left\lceil \frac{max_lat - min_lat}{\Delta} \right\rceil, \quad N_{col} = \left\lceil \frac{max_lon - min_lon}{\Delta} \right\rceil \quad (3)$$

The grid index mapping for any RSU r is:

$$i(r) = \left\lfloor \frac{lat_r - min_lat}{\Delta} \right\rfloor, \quad j(r) = \left\lfloor \frac{lon_r - min_lon}{\Delta} \right\rfloor \quad (4)$$

Therefore, the group is defined as:

$$G_{ij} = \{r \in R \mid i(r) = i, j(r) = j\}, \quad Groups = \{G_{ij} > 0\} \quad (5)$$

The grid size Δ and the resulting group size are determined according to the spatial distribution and deployment density of RSUs. A smaller grid can reduce local communication distance but increases the number of representative nodes involved in global consensus, whereas a larger grid reduces the number of groups but increases intra-group communication overhead. Therefore, Δ is selected to balance local consensus efficiency and global communication cost while ensuring that each group satisfies the Byzantine fault-tolerance requirement.

5.2. Group Representative Node Selection Strategy

Each geographical group elects only a small number of representative nodes to participate in the global PBFT, in order to reduce the scale of global layer communication and improve throughput. However, if the representative nodes are not selected properly (weak performance, low online rate, poor credit), it will lead to increased global layer latency and even security risks. Therefore, this paper designs a multi-factor comprehensive scoring mechanism that takes into account security efficiency (performance/latency).

5.2.1. Node Scoring Model

For any node r within the group, define a comprehensive score:

$$S(r) = \omega_1 * CR(r) + \omega_2 * PR(r) + \omega_3 * AR(r), \quad (6)$$

$$\omega_1 + \omega_2 + \omega_3 = 1, \omega_i > 0. \quad (7)$$

The weights ω_1 , ω_2 , and ω_3 are normalized such that $\omega_1 + \omega_2 + \omega_3 = 1$. Their values reflect the relative importance of historical credit, hardware and communication performance, and online availability. In the considered ITS scenario, reliability and continuous availability are prioritized to reduce the probability of selecting unstable or malicious representative nodes, while the performance component ensures that the selected nodes can efficiently complete message verification and broadcasting.

To avoid “score manipulation” based on a single dimension, normalization (min-max or z-score) can be added to map each sub-indicator to [0, 100], and a minimum threshold can be set.

$$CR(r) > 0, AR(r) > 0. \quad (8)$$

5.2.2. Sub-Index Calculation

(1) Credit Score (CR)

Credit reflects a node’s historical behavior and data credibility. Let the observation window be the most recent H rounds of consensus or the most recent time period $[t - H, t]$. Let N_s be the number of successful consensus participations and N be the total number of participations, then:

$$\text{success_rate}(r) = \frac{N_s(r)}{N(r)}. \quad (9)$$

Response speed score can be calculated based on the average response delay $d(r)$ relative to the timeout threshold. $d_{\max}$ is normalized as follows:

$$\text{response_speed}(r) = \max(0, 1 - \frac{d(r)}{d_{\max}}) * 100. \quad (10)$$

Data quality score can be measured by data validation pass rate. Let $N_q(r)$ be the number of times the data reported by a node passes neighbor cross-validation, and $N_v(r)$ be the total number of validations:

$$\text{data_quality}(r) = \frac{N_q(r)}{N_v(r)} * 100. \quad (11)$$

The credit score can then be written as:

$$CR(r) = 35 * \text{success_rate}(r) * 100 + 35 * \text{response_speed}(r) + 30 * \text{data_quality}(r). \quad (12)$$

(2) Performance Rating (PR)

Performance reflects a node’s ability to perform aggregation, signature verification, and broadcasting. CPU, memory, bandwidth, and latency are normalized to [0, 100].

$$\text{bandwidth_scores} = \frac{B_r}{B_{\max}} * 100, \quad (13)$$

$$\text{latency_scores} = \max(0, 1 - \frac{L_r}{L_{\max}}) * 100, \quad (14)$$

$$\text{cpu_scores} = \max(100, \frac{f_r * c_r}{f_0 * c_0} * 100). \quad (15)$$

r denotes a candidate node (or resource provider); B_r is the available network bandwidth of node r ; $B_{\max}$ is the maximum observed bandwidth in the system (or a predefined benchmark upper bound). This metric captures the node's throughput capability in block or message dissemination. L_r denotes the average network latency or end-to-end response delay of node r ; $L_{\max}$ is the maximum tolerable latency threshold defined by the system. f_r is the CPU clock frequency of node r ; c_r is the number of CPU cores of node r ; f_0 and c_0 correspond to the clock frequency and core count of a reference (baseline) hardware configuration.

The formula for comprehensive performance scoring is:

$$\text{PR}(r) = 35 * \text{bandwidth_scores}(r) * 100 + 35 * \text{latency_scores}(r) + 30 * \text{cpu_scores}(r). \quad (16)$$

(3) Availability Rating (AR)

The availability rating measures the online persistence and operational reliability of a node over a given observation period.

(a) Failure Penalty Function

$$\phi(r) = \min(0.5, 0.1F(r)). \quad (17)$$

This function imposes a linear penalty for frequent failures, capped at 0.5, thereby preventing availability from being completely eliminated by extreme failure counts.

(b) Availability Score

$$\text{AR}(r) = \frac{t_{\text{on}}(r)}{T} * 100 * (1 - \phi(r)). \quad (18)$$

This metric jointly captures online time ratio and fault reliability, rewarding nodes that remain continuously online with minimal failures, where T denotes the total length of the observation period; $t_{\text{on}}(r)$ denote the cumulative online time of node r during T ; $F(r)$ denote the number of failures observed for node r within the same period.

5.3. Layered Consensus Process

H-PBFT divides consensus into two phases: intra-group consensus (local) and global consensus (representative layer). The local phase ensures that data within the same region is consistent first; the global phase ensures consistent ordering and on-chain confirmation across groups.

5.3.1. Phase 1: Intra-Group Raft Consensus

Let t_x denote the transaction or data payload awaiting consensus. Within a specific group G_i , the designated leader initiates the process by aggregating the intra-group data to formulate a local data block, denoted as D_i . This aggregation process is formalized as:

$$D_i = \text{Agg}(\{x_r \mid r \in G_i\}, t_x), \quad (19)$$

where x_r represents the local observations or evidences (e.g., event confidence, timestamps, or data digests) associated with the transaction by node r . The objective of the intra-group consensus is to ensure the consistency and correctness of D_i is confirmed by at least $2/3$ of the honest nodes within the group.

Assuming a group size of k_i under the Byzantine Fault Tolerance (BFT) model, the maximum number of malicious nodes f_i that the group can tolerate is defined as:

$$f_i = \left\lfloor \frac{k_i - 1}{3} \right\rfloor, \quad k_i \geq 3f_i + 1. \quad (20)$$

Local consensus is achieved when the leader collects no fewer than $2f_i + 1$ valid confirmation signatures (equivalently, $\lceil 2k_i/3 \rceil$ votes). Let v_i denote the set of collected votes. The condition for local commitment is given by:

$$v_i \geq 2f_i + 1 \Rightarrow D_i \text{ is locally committed}$$

5.3.2. Phase 2: Global Consensus

While the traditional PBFT protocol's three-phase commit (Pre-prepare, Prepare, Commit) effectively achieves consensus among nodes, it still has some shortcomings. The main issues include: when the primary node fails or switches over, incomplete requests may not be able to continue execution, affecting system stability; furthermore, the PBFT protocol assumes all nodes are honest, but in real-world scenarios, the existence of Byzantine nodes is unavoidable, leading to a decrease in protocol efficiency.

To improve the robustness and efficiency of the PBFT protocol, we propose the H-PBFT protocol and improve upon the three phases of PBFT, focusing on enhancing primary node election, fault tolerance, and the ability to filter Byzantine nodes. By introducing a the RSU Node Geographic Grouping Strategy, and Group representative node selection strategy, the protocol's fault tolerance and reliability are enhanced. The basic execution flow of the second phase protocol of the H-PBFT consensus algorithm is as follows:

- (1) Request phase. The client sends a request to the primary node. The request message is $\langle \text{Request}, m, t, c, \sigma_c \rangle$, where m is the content sent by the client to the primary node, t is the timestamp, c represents the client identifier, and σ_c is the client's signature.
- (2) Pre-prepare phase. After receiving the request message, the primary node orchestrates the request, assigns a sequence number n to the request, and generates a pre-prepare message $\langle \langle \text{Pre-prepare}, v, n, D(m), \sigma_p \rangle, m \rangle$ for the representative node. Here, v represents the view number, n represents the sequence number, $D(m)$ is the digest of the request content m , and σ_p represents the primary node's signature.
- (3) Prepare phase. At this time, the Representative node receive the pre-prepare message and generate a prepare message $\langle \text{Prepare}, v, n, D(m), b_i, \sigma_b \rangle$, where b_i represents the representative node's identifier, and σ_b represents the representative node's signature. In this phase, each representative node compares its generated preparation message with preparation messages sent from other representative node. When it receives at least $2f + 1$ preparation messages that match its own generated preparation message, consensus enters the determination phase. This phase can filter out a certain number of Byzantine nodes in the network.
- (4) Reply Phase. When each node receives at least $2f + 1$ consensus messages from different nodes in the determination phase, it forwards a reply message $\langle \text{Reply}, v, n, \sigma_b, r \rangle$ to the client, where r represents the client's request to execute the consensus result. If $f + 1$ nodes respond identically, it means the client's request was successful.

5.4. Credit Mechanism Design

The credit mechanism is a key component for improving the efficiency of the Hybrid Practical Byzantine Fault-Tolerant algorithm and incentivizing nodes. This system designs a comprehensive framework for credit evaluation, updating, and incentive allocation. The credit system plays a crucial role in ensuring nodes effectively fulfill their responsibilities and in the security and efficiency of the consensus process.

For newly added nodes, the initial credit score is set as follows:

Roadside Unit (RSU) node: Initial credit score = 60 points. Traffic management department node: Initial credit score = 80 points. Certified trusted node: Initial credit score = 70 points. These initial values reflect the node's role and trustworthiness in the system. Trusted entities such as traffic management departments and certified nodes will receive higher initial credit scores.

5.4.1. Credit Score Updates Dynamically

The credit score is dynamically updated based on node behavior. As shown in Tables 3. The formula for updating the credit score at each time step is as follows:

$$CR(t + 1) = \alpha * CR(t) + (1 - \alpha) * Reward(t). \quad (21)$$

where $CR(t)$: Credit score at time t . $Reward(t)$: Reward/penalty score at time t . The decay factor α is selected within the range of 0.7–0.9 to preserve the continuity of historical credit while allowing recent node behavior to influence the score in a timely manner. A value that is too high may cause outdated behavior to dominate the evaluation, whereas a value that is too low may make the credit score excessively sensitive to short-term fluctuations.

To translate the observed behavior of a node into a quantitative credit adjustment, both positive reward rules and negative penalty rules are defined. Table 3 summarizes the rewards assigned to nodes that correctly perform their consensus responsibilities. These rules are intended to encourage stable consensus participation, rapid responses, high-quality data submission, and continuous availability, thereby improving both consensus efficiency and transportation-data reliability.

Table 3. Reward Scene.

Reward Scenario	Reward Points	Trigger Conditions
Successful Consensus Participation	+2 points	Node successfully participates in the consensus process
Serving as a Representative Node	+5 points	Node is selected as a representative and successfully completes the task
Quick Response	+3 points	Response time is more than 50% faster than the average
High Data Quality	+4 points	Data verification pass rate > 95%
Continuous Online Presence	+1 point/day	Node is online for more than 20 h within a 24 h period

Corresponding to the reward mechanism, Table 4 presents the penalties imposed for abnormal or undesirable node behavior. The magnitude of each penalty is determined according to the potential effect of the behavior on consensus efficiency, data integrity, and system security. Minor events, such as occasional consensus timeouts or data-format errors, receive moderate penalties, whereas incorrect data submission, failure to perform representative duties, and Byzantine behavior receive stronger penalties.

Table 4. Penalty Scene.

Penalty Scenario	Penalty Points	Trigger Conditions
Consensus Timeout	−3 points	Response time exceeds the set threshold
Incorrect Data Format	−5 points	Submitted data does not meet required format
Submission of Incorrect Data	−10 points	Node submits invalid data or errors in data submission
Frequent Offline Behavior	−5 points/instance	Node is offline more than 3 times within a 24 h period
Malicious Behavior	−50 points	Node detected performing Byzantine behavior (malicious actions)
Failure to Perform Representative Duties	−15 points	Node fails to fulfill its duties when acting as a representative node

5.4.2. Credit Rating and Rights

Based on their credit scores, nodes are divided into five different tiers, each with different permissions and restrictions. These tiers are designed to ensure that nodes with higher credit scores are given priority for important roles, such as acting as representatives or accessing sensitive data. The classifications are shown in Table 5.

Table 5. Credit Levels and Rights.

Credit Level	Rights and Restrictions
Excellent (90–100 points)	1. Priority in selection as a representative node. 2. Enhanced data query permissions. 3. Voting rights in system governance.
Good (75–89 points)	1. Normal participation in consensus. 2. Standard data access rights. 3. Eligible to be selected as a representative node.
Qualified (60–74 points)	1. Normal participation in consensus. 2. Basic data access rights. 3. Less likely to be selected as a representative node.
Warning (40–59 points)	1. Participation in consensus with lower priority. 2. Restricted data access rights. 3. Needs to improve credit for recovery.
Disabled (<40 points)	1. Temporarily banned from participating in consensus. 2. No data access rights. 3. Requires manual review for reactivation.

5.4.3. Credit Recovery Mechanism

To prevent nodes from being permanently penalized due to temporary faults, the system provides a credit recovery mechanism. Nodes can recover their credit through the following means:

Natural Recovery:

1. After 7 consecutive days of normal operation without penalties, the credit value increases by 5 points.
2. After 30 consecutive days of good performance, the credit value increases by 15 points.

Active Recovery:

Submitting system improvement suggestions that are adopted increases the credit value by 8 points. Assisting in the deployment and debugging of new nodes increases the credit value by 5 points. Participating in system maintenance and upgrades increases the credit value by 10 points.

Appeal Mechanism:

1. Nodes can appeal unjust penalties.
2. An arbitration committee composed of high-credit nodes will review the appeals.
3. If the appeal is successful, the deducted credit points will be restored.

5.5. Primary Node Selection

In traditional PBFT and its many variants, master nodes are typically selected through round-robin or random rotation. While these strategies ensure fairness, they ignore differences in node reliability, historical behavior, and dynamic trustworthiness, potentially leading to frequent view changes, reduced consensus efficiency, and increased susceptibility to Byzantine behavior. This limitation is particularly pronounced in large-

scale dynamic intelligent transportation systems, where node performance and stability vary significantly.

To address this issue, this paper proposes a credit-aware master node selection mechanism integrated into the H-PBFT framework. Unlike traditional methods, this approach selects master nodes solely based on the credit value representing the node set, thereby improving consensus stability, security, and execution efficiency.

Let $R = \{r_1, r_2, \dots, r_m\}$ denote the set of representative nodes elected from different geographic groups after the local consensus phase, where M is the number of groups. Each representative node $r_i \in R$ maintains an up-to-date credit value $CR(i)$ at time i , which is dynamically adjusted according to its historical consensus behavior, availability, and data quality.

The primary node r_p for the global consensus phase is determined by selecting the representative node with the highest credit value. Formally, the primary node selection rule is defined as:

$$r_p = \arg \max_{r_i \in R} CR(i). \quad (22)$$

This deterministic selection strategy ensures that the node with the strongest historical reliability and trustworthiness assumes the role of the primary node. By prioritizing high-credit nodes, the proposed mechanism significantly reduces the probability of selecting unstable or malicious nodes as leaders.

6. Theoretical and Simulation Analysis

This experiment develops a Python-based simulation environment to evaluate the performance of PBFT, Q-PBFT [34], APBFT [35], and the proposed H-PBFT consensus algorithm. A distributed network model is adopted, in which each node is assigned an independent communication port, and multiple nodes participate concurrently in the consensus process through a multithreaded mechanism. Inter-node communication is implemented using the Socket protocol, while consensus messages are uniformly serialized in JSON format. The PyNaCl cryptographic library is employed for message signing and verification, thereby ensuring message authenticity, integrity, and resistance to tampering. It should be noted that the current system model assumes that RSUs have relatively stable computing resources and communication links. In realistic ITS environments, vehicle mobility, intermittent connectivity, packet loss, and time-varying network latency may delay local message aggregation, temporarily reduce the availability of representative nodes, and trigger additional retransmissions or view changes. Although the geographic grouping strategy and availability-aware node evaluation of H-PBFT can partially reduce the influence of unstable nodes and long-distance communication, the present simulation does not fully reproduce these dynamic network conditions. Therefore, the reported results mainly reflect the performance of H-PBFT under controlled network settings. Evaluating the protocol using realistic mobility traces, dynamic channel models, and roadside testbed environments will be an important direction of future work.

To approximate heterogeneous behaviors in complex network environments, the simulation includes normal nodes, delayed nodes, and faulty nodes, covering typical Byzantine fault-tolerant scenarios. All algorithms are implemented and evaluated under the same simulation framework to ensure a fair comparison. The detailed hardware, software, and experimental parameter configurations are presented in Table 6.

Regarding the proposed H-PBFT algorithm, this paper conducts a comparative analysis from four core performance indicators: consensus latency, communication overhead, transaction throughput, and view-switching recovery time. In addition, to further test the scalability of the algorithm, the experiment also constructed a large-scale node network scenario, focusing on the changing trends of consensus latency and communication complexity of the system under the condition of increasing number of nodes.

Table 6. Experimental configuration table.

Object	Configuration
CPU	13th Intel(R) i7-13620 H
Operating system	Window 11
Software Environment	PyCharm
GPU	NVIDIA GeForce RTX 4060
Memory	16 G
Programming language	Python

6.1. Communication Complexity Analysis

(1) Total Overhead of the PBFT Consensus Algorithm

As described in the PBFT consensus algorithm, messages need to be sent in three phases, with each node responding to the client's consensus message in the final phase. In the Pre-Prepare phase, the number of communications is $N - 1$; in the Prepare phase, the number of communications is $(N - 1)^2$; in the Commit phase, the number of communications is $N * N - 1$; and in the Reply phase, the number of communications is N . Therefore, the total communication overhead T_1 of the PBFT consensus algorithm is:

$$T_1 = (N - 1) + (N - 1)^2 + N * (N - 1) + N. \quad (23)$$

(2) Total overhead of the H-PBFT consensus algorithm

Assume there are N RSU nodes in the consensus network, divided into M groups, each group containing an average of $K = \frac{N}{M}$ nodes, and f is the number of Byzantine nodes the system can tolerate.

In intra-group consensus, concerning M groups in parallel, the message count per group is $O(K^2) = O((N/M)^2)$. In global consensus, in the proposed H-PBFT algorithm, the number of representative nodes participating in consensus is only M , and the algorithm optimizes the Commit and Reply phases of the consensus protocol. Therefore, the total communication overhead T_2 of the H-PBFT algorithm is:

$$T_2 = (M - 1) + (M - 1)^2 + M - 1 + N + M * \left(\frac{N}{M}\right)^2 = M^2 + N + \frac{N^2}{M} \quad (24)$$

When $M = \sqrt{N}$, the communication complexity is reduced to $O(N)$, which is significantly lower than the $O(N^2)$ of the traditional PBFT.

6.2. Consensus Delay Analysis

In blockchain systems, consensus latency is typically defined as the total time elapsed from when a client initiates a transaction request until all consensus nodes in the network complete state confirmation and return the final result to the client. This metric directly reflects the system's real-time responsiveness and collaborative efficiency.

To evaluate the latency performance of different algorithms under varying network scales, the experiment progressively expanded the node scale, increasing the number of participating nodes from 100 to 200, with an increment of 20. For each network scale configuration, 10 rounds of independent, repeated experiments were conducted to reduce the impact of random factors on the results and enhance statistical reliability. The node environment and network state were reinitialized in each round to ensure the independence

and fairness of the testing process. The final result is the arithmetic mean of the latency data measured in multiple experiments, used as the consensus latency characterization value for that scale.

Figure 5 shows a comparison of the average consensus latency of the PBFT, Q-PBFT, APBFT, and H-PBFT consensus algorithms under different node scales. Experimental data, as shown in the figure, ranged from 100 to 200 nodes. The latency of each algorithm increased with the number of nodes, but the rate of increase varied significantly. H-PBFT consistently maintained the lowest latency, demonstrating excellent stability. The latency of Q-PBFT and APBFT fell between that of H-PBFT and PBFT, and gradually increased with the number of nodes.

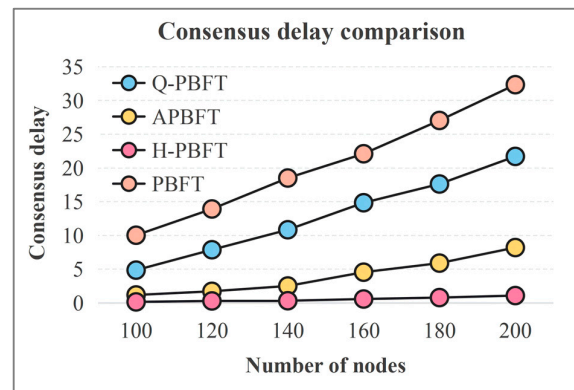


Figure 5. Comparison chart of consensus latency of various algorithms.

Horizontal comparisons allow for a direct analysis of the latency trends of each algorithm as the network size grows, thus evaluating their scalability and performance stability in large-scale distributed environments. The results show that H-PBFT performs best in terms of latency control.

6.3. Communication Overhead Analysis

Figure 6 compares the communication overhead of four consensus algorithms—Q-PBFT, APBFT, H-PBFT, and PBFT—at different node scales. As the number of nodes increases from 100 to 200, the communication overhead of each algorithm increases. PBFT's communication overhead is significantly higher than the other three improved algorithms, approaching 50,000 times when the node scale reaches 200, indicating a sharp increase in communication complexity with scale. In contrast, H-PBFT maintains the lowest communication overhead throughout the entire node range, remaining below 10,000 times even at 200 nodes, demonstrating its effective suppression of network load through hierarchical or redundancy reduction mechanisms. The communication overhead of Q-PBFT and APBFT falls between the two, steadily increasing with the number of nodes, reflecting their partial effectiveness in optimizing the traditional PBFT communication mode. Overall, the comparison of communication overhead further confirms that H-PBFT possesses superior scalability and communication efficiency in large-scale network environments.

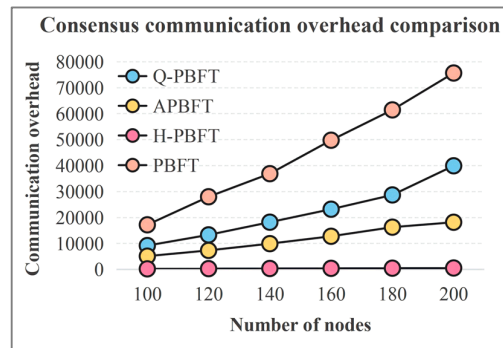


Figure 6. Comparison of consensus communication overhead for each algorithm.

6.4. Throughput Analysis

Throughput denotes the number of transactions successfully processed within a unit of time and is typically measured in TPS (transactions per second). It serves as a fundamental metric for evaluating and comparing the performance of different consensus algorithms, as it reflects the system's transaction processing capability under given network conditions. The calculation of TPS is defined as:

$$\text{TPS} = \frac{\text{Number of Transactions}}{\Delta t} \quad (25)$$

where Δt represents the total duration required to complete the transaction processing test, and Number of Transactions refers to the total transactions finalized during this interval. In the experimental setup, the client continuously generates requests, while the system records the number of requests successfully handled by the nodes per second to determine the overall throughput performance.

Figure 7 compares the system throughput performance of four consensus algorithms under different node scales. Experimental results show that as the number of participating nodes increases from 100 to 200, the throughput of all algorithms decreases to varying degrees. This aligns with the general rule that an increase in the number of nodes in distributed consensus typically leads to increased coordination costs. Among them, PBFT shows the most significant throughput decay, indicating its sensitivity to network scaling. H-PBFT exhibits the best throughput maintenance capability, with its curve showing the gentlest decline, demonstrating that its consensus mechanism can still effectively handle high-frequency transaction requests even with an increase in nodes. Q-PBFT and APBFT have throughput between PBFT and H-PBFT, outperforming traditional PBFT but lagging behind H-PBFT. This result further illustrates that H-PBFT also possesses better scalability and performance stability in terms of throughput.

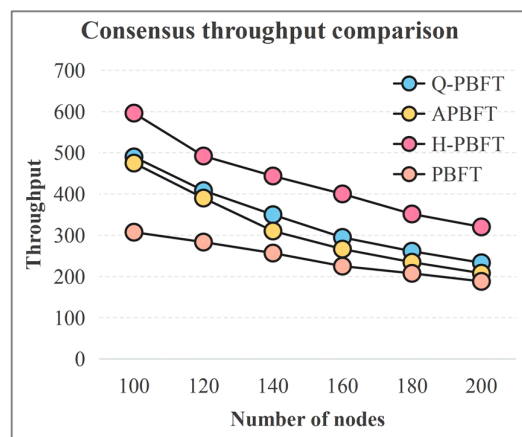


Figure 7. Throughput comparison of various algorithms.

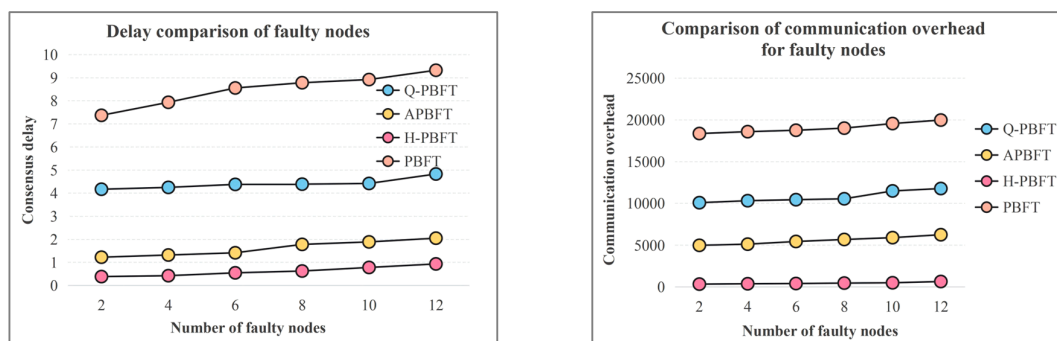
6.5. Fault-Tolerance Performance Analysis

Real-world blockchain networks are typically deployed in open or semi-open environments, where nodes may exhibit abnormal or even malicious behavior due to software vulnerabilities, network attacks, or resource constraints. Therefore, verifying consensus performance solely under ideal, fault-free conditions is insufficient to reflect the algorithm's true applicability. By introducing malicious nodes of varying sizes, this experiment quantifies the algorithm's fault tolerance, robustness, and performance degradation under abnormal environments, thereby verifying its engineering feasibility and security stability in complex network environments.

In this experiment, we constructed a consensus network environment with 100 nodes, including honest, slow, and malicious nodes, to simulate heterogeneous behavior in real distributed systems. Initially, the network consisted of 90 honest nodes, 8 slow nodes, and 2 malicious nodes. As consensus rounds progressed, the proportion of malicious nodes gradually increased, adding 2 malicious nodes per round until a total of 12 were reached, thus creating Byzantine fault scenarios of varying intensities.

This experiment aims to evaluate the consensus efficiency and stability of the H-PBFT algorithm under varying malicious node proportions by progressively increasing the adversarial environment, and to compare it with the PBFT, Q-PBFT, and APBFT algorithms. In the experiment, malicious nodes interfere with the decision-making process of honest nodes by sending forged or inconsistent messages, simulating deception and protocol violation behaviors that may occur in real-world blockchain networks, thus posing a challenge to the system's consistency maintenance mechanism.

Figure 8 shows a comparison of the average consensus latency and communication overhead of various consensus algorithms under the presence of malicious nodes. The experimental results show that as the number of faulty nodes increases, the consensus latency of all algorithms exhibits a certain upward trend, due to the additional verification and fault tolerance overhead required by the system. However, the latency increase in the H-PBFT algorithm is significantly smaller, remaining relatively stable overall. This is mainly because it can promptly identify and isolate malicious nodes, reducing the interference of invalid interactions on the consensus process, thereby maintaining high execution efficiency.

**Figure 8.** Performance comparison of various algorithms under malicious node conditions.

It should be noted that the present experiment mainly evaluates the robustness of H-PBFT under varying proportions of generic Byzantine nodes that send forged or inconsistent messages. More structured attack scenarios, including collusion attacks, Sybil attacks, representative-node compromise, leader manipulation, and network partition at-

attacks, are not explicitly considered in the current simulation. These attacks may affect representative selection, quorum formation, and inter-group communication in different ways. Therefore, the current results should be interpreted as evidence of robustness against general Byzantine-node interference rather than a comprehensive security evaluation under all attack models. A systematic investigation of these attack scenarios will be conducted in future work.

6.6. Scalability Analysis

To evaluate the scalability of the H-PBFT algorithm in a large-scale distributed environment, this paper constructs a high-node-density network simulation scenario and measures the consensus latency change as the number of nodes continuously increases. In the experiment, the number of nodes participating in consensus was gradually increased from 500 to 1000, with an increment interval of 100. To reduce interference from random factors and improve the reliability of the statistical results, 20 rounds of experiments were independently executed for each network size configuration. The average of multiple measurements was used as the consensus latency index for that size.

Figure 9 shows the latency comparison curves of four consensus algorithms—PBFT, Q-PBFT, APBFT, and H-PBFT—under different node sizes. Overall, as the network size increases, the consensus latency of all protocols increases to varying degrees. This is due to the increased message interaction frequency and verification load resulting from the increased number of nodes. Among them, the traditional PBFT protocol, due to its higher communication complexity, shows a more significant increase in latency with the increase in the number of nodes, demonstrating weaker scalability.

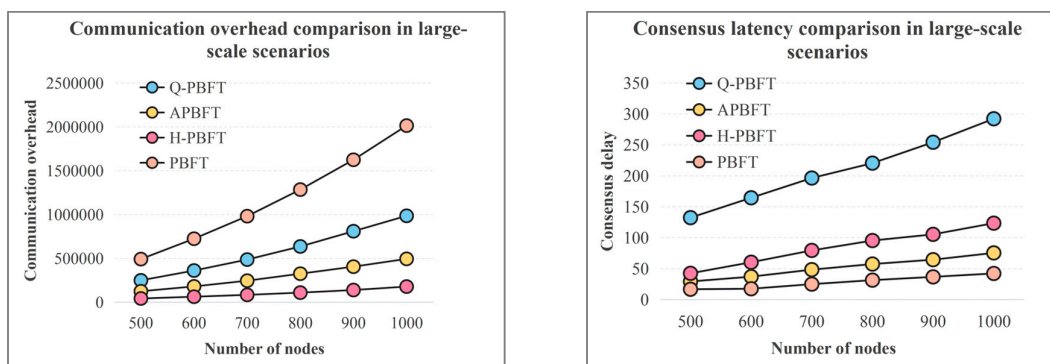


Figure 9. Performance comparison of various algorithms in large-scale node scenarios.

In comparison, APBFT and H-PBFT exhibit more stable performance in large-scale networks, with a relatively gradual increase in latency. Especially when the node scale expands to 1000, H-PBFT still maintains a low latency level, demonstrating its superior scalability design advantages in communication structure optimization and node management mechanisms, thus enabling it to maintain better consensus efficiency in high-density node environments. Moreover, the limited variation and consistent trends observed across the repeated experiments indicate that the scalability advantage of H-PBFT is stable and is unlikely to result from isolated experimental fluctuations.

6.7. View-Switching Analysis

View change recovery time measures a system's self-recovery capability in the event of a master node failure. It is defined as the time span from when a node initiates a view

change request to when a new master node is successfully elected and the consensus process restarts. This metric reflects the efficiency of the protocol in maintaining continuous service under abnormal conditions.

Different improved algorithms have varying time requirements for collecting the threshold $2f + 1$ view change requests due to inconsistent node sizes participating in consensus. Furthermore, the different designs of the master node election mechanisms in each protocol further affect the recovery speed after a failure.

The experiment constructed a network environment with 40 nodes and simulated a master node failure scenario. After the master node failed, the remaining consensus nodes triggered the view change process, submitting view change messages and completing the election of a new master node to initiate the next round of consensus. Figure 10 shows a comparison of the view change recovery times of Q-PBFT, APBFT, H-PBFT, and PBFT in this scenario.

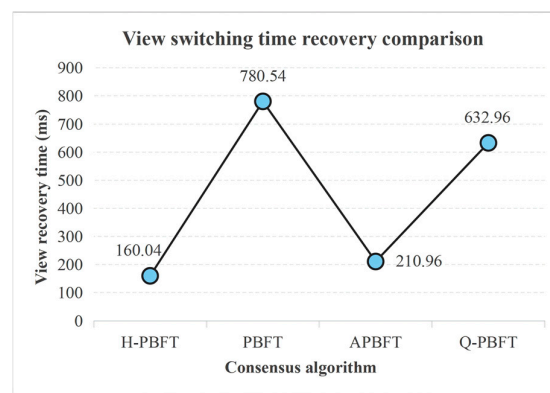


Figure 10. View change recovery time of various algorithms.

The experimental results show that H-PBFT and APBFT perform better in terms of recovery efficiency, with significantly shorter recovery times than the other two algorithms. This is primarily because the two protocols mentioned above limit the number of nodes participating in consensus to less than half of the total number of nodes, thereby shortening the time required to collect sufficient view change requests. Furthermore, H-PBFT elects only representative nodes within each group, reducing the scope of election competition and making the master node confirmation process more efficient, thus achieving the shortest recovery latency. The ablation results further show that the credit-enabled H-PBFT achieves a shorter view-change recovery time than H-PBFT-NoCredit. This is because the credit-based strategy preferentially selects representative nodes with better historical reliability, availability, and behavioral performance as the new primary, thereby reducing the probability of repeatedly selecting unstable or malicious nodes and accelerating consensus recovery.

In contrast, Q-PBFT and traditional PBFT require all nodes to participate in the consensus process. After a master node failure, they need to collect more view change requests and coordinate a wider range of nodes, increasing message interaction and verification overhead, resulting in a relatively longer overall recovery time.

6.8. Byzantine Fault Tolerance and Safety Analysis

The security analysis is based on authenticated communication channels, unforgeable digital signatures, collision-resistant hash functions, and fixed group membership during each consensus round. The credit mechanism affects only the priority of node-role selection and does not replace the Byzantine quorum requirements.

For a local consensus group G_i containing $k_i = 3f_i + 1$ nodes, at most f_i nodes may be Byzantine. A local result D_i is accepted only after it receives at least $2f_i + 1$ matching signatures from distinct group members. Let Q_i and Q_i' denote two local quorums corresponding to two conflicting local results. Their intersection satisfies

$$|Q_i \cap Q_i'| \geq 2(2f_i + 1) - (3f_i + 1) = f_i + 1. \quad (26)$$

Because at most f_i nodes are Byzantine, the intersection contains at least one honest node. An honest node signs at most one data digest for the same group, view, and sequence number. Therefore, two conflicting local results cannot both obtain valid local consensus certificates.

Similarly, assume that the global representative committee contains $M = 3F + 1$ representative nodes, of which at most F are Byzantine. A global result is committed only after at least $2F + 1$ matching Commit messages have been received. For any two global commit quorums Q_g and Q_g' ,

$$|Q_g \cap Q_g'| \geq 2(2F + 1) - (3F + 1) = F + 1. \quad (27)$$

Since at most F representatives are Byzantine, the intersection contains at least one honest representative. Because an honest representative does not sign two conflicting block digests for the same view and sequence number, two conflicting blocks cannot both obtain valid global commit certificates. Furthermore, the global layer accepts only local results carrying valid local quorum certificates. Therefore, H-PBFT preserves consensus safety under hierarchical composition, provided that the Byzantine fault bound is independently satisfied in every local group and in the global representative committee.

6.9. Liveness Analysis

The liveness of H-PBFT is analyzed under the partial-synchrony assumption. After an unknown network stabilization time, messages exchanged between honest nodes are delivered within a bounded delay. In each local group, at least $2f_i + 1$ responsive honest or non-faulty nodes are available to generate a valid local consensus certificate. If the local leader or representative node fails to respond before the timeout, the next eligible backup node is selected according to the ranked candidate list.

At the global layer, if the current primary node is faulty or unavailable, a timeout triggers the view-change procedure. Because Byzantine or repeatedly unavailable nodes are penalized by the credit mechanism and excluded from immediate reselection, an honest primary node is eventually selected. After network stabilization, the honest primary can collect at least $2F + 1$ matching Prepare and Commit messages, allowing the request to be committed and returned to the client. Therefore, H-PBFT satisfies liveness provided that the network is eventually synchronous, the Byzantine-node bounds are not exceeded, and the leader-replacement and view-change procedures eventually select honest nodes.

6.10. Credit Mechanism Analysis

To evaluate the independent contribution of the proposed credit mechanism, an ablation experiment was conducted to analyze the evolution of node credit values during continuous consensus rounds. Four representative node behaviors were considered, including reliable nodes, stable nodes, unstable nodes, and malicious nodes. The initial credit values were assigned according to their trust levels, and the credit scores were dynamically updated according to the reward and penalty rules described in Section 5.4.

As shown in Figure 11. During 20 consecutive consensus rounds, reliable nodes continuously obtained rewards due to stable participation and high-quality responses, whereas unstable nodes suffered gradual credit degradation because of frequent delays

and availability issues. Malicious nodes received severe penalties after performing Byzantine behaviors, resulting in a rapid decrease in credit values. The results demonstrate that the proposed credit mechanism can effectively distinguish heterogeneous nodes and provide a reliable basis for representative-node selection.

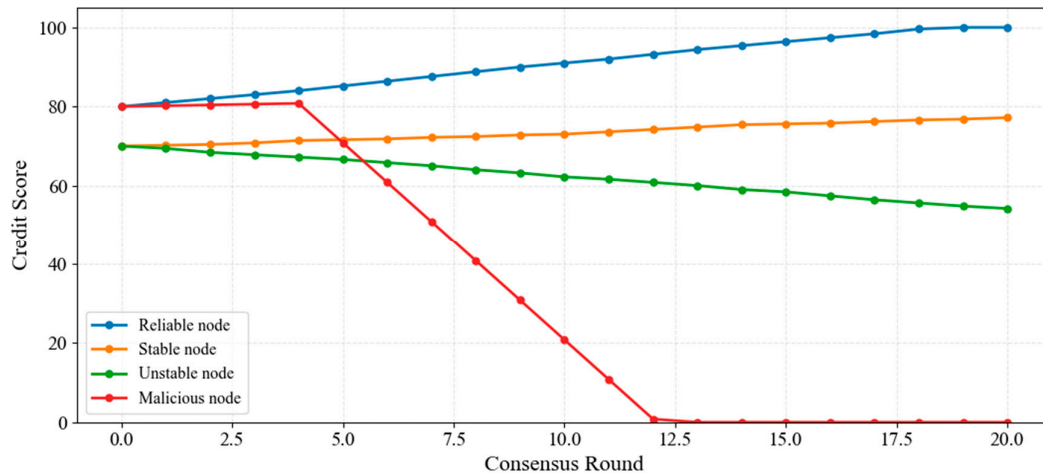


Figure 11. Credit evolution of different node types during consensus rounds.

7. Conclusions and Future Work

To address the scalability bottlenecks and security vulnerabilities faced by traditional blockchains in Intelligent Transportation Systems data storage, this paper proposes a layered, multi-dimensional reputation-aware improved consensus mechanism. This mechanism decouples the consensus process into edge-local consensus and global representative consensus by introducing a geographical grouping strategy. Furthermore, it optimizes node election by incorporating a reputation evaluation model that incorporates historical behavior and hardware performance, thereby significantly reducing communication complexity from $O(N^2)$ while enhancing the system's Byzantine fault tolerance. Simulation results demonstrate that H-PBFT exhibits significant advantages over traditional and existing improved PBFT algorithms (such as Q-PBFT and APBFT) in terms of consensus latency, throughput, and view-switching recovery efficiency in large-scale node and dynamic network environments. This provides a practical solution for building secure, efficient, and scalable distributed storage systems for the Internet of Vehicles.

Future research will extend the current simulation framework toward an integrated mobility–communication evaluation, with particular emphasis on heterogeneous roadside deployments, transient connectivity disruptions, and service continuity under dynamically changing node states. This extension will enable a more realistic assessment of the adaptability and recovery capability of H-PBFT in dynamic vehicular networks.

Author Contributions: Conceptualization, Z.W. (Zhenhua Wang) and J.H.; methodology, Z.W. (Zhenhua Wang) and X.W.; software, Z.W. (Zixuan Wang) and H.Y.; validation, Z.W. (Zhenhua Wang), Z.W. (Zheng Wu) and M.L.; formal analysis, X.W. and Z.W. (Zixuan Wang); investigation, J.H. and H.Y.; resources, J.H.; data curation, Z.W. (Zheng Wu) and M.L.; writing—original draft preparation, Z.W. (Zhenhua Wang); writing—review and editing, J.H., X.W., Z.W. (Zheng Wu) and Z.T. (Zhen Tian); visualization, Z.W. (Zixuan Wang) and H.Y.; supervision, J.H.; project administration, J.H. and Z.W. (Zhenhua Wang); funding acquisition, J.H. All authors have read and agreed to the published version of the manuscript.

Funding: The authors would like to express their sincere gratitude to the Fundamental Research Funds for the Central Universities Special Fund Project (Grant No. LGZD202609), the Youth Fund Project of Ministry of Education in China Humanities and Social Sciences Foundation (Grant No. 25YJC760104), the General Research Projects in Philosophy and Social Sciences of Colleges and Universities in Jiangsu Province (Grant No. 2025SJYB0084), the Research Project on Higher Education Teaching Reform of Jiangsu Association of Automation (Grant No. JSAAJG2025Y28), the General Project on Teaching Reform of Nanjing Police University (Grant No. YB26005), the Harbin Xinguang Optic-electronics Technology Co., Ltd. Horizontal Research Project (Grant No. 2024320107003397, 2025320107003049) for providing funds to support this study.

Data Availability Statement: The authors state that their research did not use any datasets and was conducted as an experiment simulating a blockchain consensus network.

Conflicts of Interest: The authors declare that this study received funding from the Harbin Xinguang Optic-electronics Technology Co., Ltd. Horizontal Research Project. The funder was not involved in the study design, collection, analysis, interpretation of data, the writing of this article or the decision to submit it for publication. The authors declare no conflicts of interest.

References

1. Zhao, X.; Liu, L.; Yin, D.; Su, Y.; Hu, T.; Zheng, Y.; Yang, T.; Shi, Q.; Tao, K. A vehicle-to-everything (V2X) interaction system for intelligent transportation based on energy harvesting–sensing cooperation triboelectric nanogenerator. *ACS Appl. Electron. Mater.* **2025**, *7*, 7419–7431.
2. Pawar, V.; Zade, N.; Vora, D.; Khairnar, V.; Oliveira, A.; Kotecha, K.; Kulkarni, A. Intelligent Transportation System with 5G Vehicle-to-Everything (V2X): Architectures, Vehicular Use Cases, Emergency Vehicles, Current Challenges and Future Directions. *IEEE Access* **2024**, *12*, 183937–183960.
3. Bohra, N.; Kumari, A.; Mishra, V.K.; Soni, P.K.; Balyan, V. Intelligence-Based Strategies with Vehicle-to-Everything Network: A Review. *Future Internet* **2025**, *17*, 79.
4. Tariq, U.; Ahanger, T.A. Enhancing Intelligent Transport Systems Through Decentralized Security Frameworks in Vehicle-to-Everything Networks. *World Electr. Veh. J.* **2025**, *16*, 24.
5. Yuan, F.; Lin, Z.; Tian, Z.; Chen, B.; Zhou, Q.; Yuan, C.; Sun, H.; Huang, Z. Bio-inspired hybrid path planning for efficient and smooth robotic navigation. *Int. J. Intell. Robot. Appl.* **2025**, *9*, 1934–1964.
6. Huang, T.; Liu, J.; Zhou, X.; Nguyen, D.C.; Azghadi, M.R.; Xia, Y.; Han, Q.-L.; Sun, S. Vehicle-to-everything cooperative perception for autonomous driving. *Proc. IEEE* **2025**, *113*, 443–477.
7. Zhang, K.; Yuan, F.; Jiang, Y.; Mao, Z.; Zuo, Z.; Peng, Y. A Particle Swarm Optimization-Guided Ivy Algorithm for Global Optimization Problems. *Biomimetics* **2025**, *10*, 342.
8. Liang, B.; Yuan, F.; Deng, J.; Wu, Q.; Gao, J. Cs-pbft: A comprehensive scoring-based Practical Byzantine Fault Tolerance consensus algorithm. *J. Supercomput.* **2025**, *81*, 859.
9. Yuan, F.; Zuo, Z.; Jiang, Y.; Shu, W.; Tian, Z.; Ye, C.; Yang, J.; Mao, Z.; Huang, X.; Gu, S.; et al. AI-driven optimization of blockchain scalability, security, and privacy protection. *Algorithms* **2025**, *18*, 263.
10. Yu, S.; Lv, K.; Shao, Z.; Guo, Y.; Zou, J.; Zhang, B. A high performance blockchain platform for intelligent devices. In *Proceedings of the 2018 1st IEEE International Conference on Hot Information-Centric Networking (HotICN)*; IEEE: Piscataway, NJ, USA, 2018; pp. 260–261.
11. Tong, W.; Dong, X.; Shen, Y.; Zheng, J. BC-RAN: Cloud radio access network enabled by blockchain for 5G. *Comput. Commun.* **2020**, *162*, 179–186.
12. Medel, D.K.S.; Perez, B.A.M.; Alpaño, P.V.; Pedrasa, J. Mitigation of Data Integrity Attacks using Blockchain-based Intelligent Transportation System. In *Proceedings of the 2021 26th IEEE Asia-Pacific Conference on Communications (APCC)*; IEEE: Piscataway, NJ, USA, 2021; pp. 13–18.
13. Lee, S.; Seo, S.H. Design of a two layered blockchain-based reputation system in vehicular networks. *IEEE Trans. Veh. Technol.* **2021**, *71*, 1209–1223.
14. Xu, G.; Bai, H.; Xing, J.; Luo, T.; Xiong, N.N.; Cheng, X.; Liu, S.; Zheng, X. SG-PBFT: A secure and highly efficient distributed blockchain PBFT consensus algorithm for intelligent Internet of vehicles. *J. Parallel Distrib. Comput.* **2022**, *164*, 1–11.

15. Kumar, A.; Vishwakarma, L.; Das, D. R-PBFT: A secure and intelligent consensus algorithm for Internet of vehicles. *Veh. Commun.* **2023**, *41*, 100609.
16. Zhang, P.; Yu, Y.; Liu, J.; Lv, C.; Tan, L.; Zhang, Y. Enhanced Practical Byzantine Fault Tolerance for Service Function Chain Deployment: Advancing Big Data Intelligence in Control Systems. *Comput. Mater. Contin.* **2025**, *83*, 4393–4409.
17. Wang, B.; Tian, Z.; Tang, F.; Pan, H.; She, W.; Liu, W. Blockchain-empowered asynchronous federated reinforcement learning for IoT-based traffic trajectory prediction. *IEEE Internet Things J.* **2025**, *12*, 17095–17109.
18. Chang, E. Blockchain driven trust management for intelligent transportation systems in VANETs. *Sci. Rep.* **2025**, *15*, 39702.
19. Wang, R.; Yuan, M.; Wang, Z.; Li, Y. Improved fast-response consensus algorithm based on HotStuff. *Sensors* **2024**, *24*, 5417.
20. Lagaillardie, N.; Djari, M.A.; Gürçan, Ö. A computational study on fairness of the Tendermint blockchain protocol. *Information* **2019**, *10*, 378.
21. Zheng, Z.; Xie, S.; Dai, H.N.; Chen, X.; Wang, H. Blockchain challenges and opportunities: A survey. *Int. J. Web Grid Serv.* **2018**, *14*, 352–375.
22. Belotti, M.; Božić, N.; Pujolle, G.; Secchi, S. A vademecum on blockchain technologies: When, which, and how. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 3796–3838.
23. Xu, M.; Chen, X.; Kou, G. A systematic review of blockchain. *Financ. Innov.* **2019**, *5*, 27.
24. Guo, H.; Yu, X. A survey on blockchain technology and its security. *Blockchain Res. Appl.* **2022**, *3*, 100067.
25. Dinh, T.T.A.; Liu, R.; Zhang, M.; Chen, G.; Ooi, B.C.; Wang, J. Untangling blockchain: A data processing view of blockchain systems. *IEEE Trans. Knowl. Data Eng.* **2018**, *30*, 1366–1385.
26. Li, X.; Jiang, P.; Chen, T.; Luo, X.; Wen, Q. A survey on the security of blockchain systems. *Future Gener. Comput. Syst.* **2020**, *107*, 841–853.
27. Zhang, R.; Xue, R.; Liu, L. Security and privacy on blockchain. *ACM Comput. Surv. (CSUR)* **2019**, *52*, 1–34.
28. Yang, R.; Wakefield, R.; Lyu, S.; Jayasuriya, S.; Han, F.; Yi, X.; Yang, X.; Amarasinghe, G.; Chen, S. Public and private blockchain in construction business process and information integration. *Autom. Constr.* **2020**, *118*, 103276.
29. Pahlajani, S.; Kshirsagar, A.; Pachghare, V. Survey on private blockchain consensus algorithms. In *Proceedings of the 2019 1st International Conference on Innovations in Information and Communication Technology (ICIICT)*; IEEE: Piscataway, NJ, USA, 2019; pp. 1–6.
30. Helliär, C.V.; Crawford, L.; Rocca, L.; Teodori, C.; Veneziani, M. Permissionless and permissioned blockchain diffusion. *Int. J. Inf. Manag.* **2020**, *54*, 102136.
31. Dib, O.; Brousmiche, K.L.; Durand, A.; Thea, E.; Hamida, E.B. Consortium blockchains: Overview, applications and challenges. *Int. J. Adv. Telecommun.* **2018**, *11*, 51–64.
32. Castro, M.; Liskov, B. Practical byzantine fault tolerance. *OsDI* **1999**, *99*, 173–186.
33. Yuan, F.; Huang, X.; Zheng, L.; Wang, L.; Wang, Y.; Yan, X.; Gu, S.; Peng, Y. The evolution and optimization strategies of a PBFT consensus algorithm for consortium blockchains. *Information* **2025**, *16*, 268.
34. Wu, X.; Fan, Y.; Gao, Z. An improved PBFT algorithm based on QoS values in cloud manufacturing service scenarios. *Comput. Integr. Manuf. Syst.* **2022**, *28*, 1767.
35. Riahi, K.; Abouaissa, A.; Idoumghar, L. Apbft: An adaptive pbft consensus for private blockchains. In *Proceedings of the GLOBECOM 2022—2022 IEEE Global Communications Conference*; IEEE: Piscataway, NJ, USA, 2022; pp. 1788–1793.

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.