Journal of Combinatorial Mathematics
and Combinatorial Computing
www.combinatorialpress.com/jcmcc

Combinatorial Press

Blockchain-based Decision Tree Optimization and Data Encryption Algorithm Design for Unmanned Equipment Environment Sensing System

Wei Chang[1,✉], Fuli Shi[1], Jianzhou Wang[2]

[1] *Equipment Management and Support College, Engineering University of PAP, Xi'an, Shaanxi, 710086, China*
[2] *Yichun Detachment, Heilongjiang Provincial Corps of PAP, Yichun, Heilongjiang, 153000, China*

ABSTRACT

In the context of rapid research and development of unmanned equipment products, how can we better design an environment sensing system suitable for unmanned equipment combat missions and combat tasks from the perspective of actual combat has become an important research topic. This paper explores the optimization scheme of unmanned equipment environment sensing system based on blockchain technology, proposes PBFT (DTPBFT) consensus algorithm based on C4.5 decision tree optimization, and combines with the full homomorphic encryption algorithm to put forward the shared data encryption scheme of unmanned equipment environment sensing system. The experimental results show that the classification accuracy of C4.5 decision tree is as high as 94.37%, which is better than other classification algorithms, indicating that the use of C4.5 decision tree can effectively improve the accuracy of the classification of the consensus nodes and the security of the PBFT algorithm. In the case of the same number of nodes, the throughput size of the DTPBRT algorithm proposed in this paper is always higher than that of the PBFT algorithm, and the consensus latency is higher than that of the PBFT algorithm only when there are Byzantine nodes inside the system, but the DTPBRT algorithm is able to effectively remove the Byzantine nodes inside the system, which verifies the superiority of this paper's algorithm. Comprehensive encryption and decryption time-consuming and throughput data, this paper's scheme in general can realize high data sharing efficiency and ensure the security of data sharing, which can provide technical support for the data security of unmanned equipment environment sensing system.

Keywords: pbft consensus algorithm, c4.5 decision tree, fully homomorphic encryption algorithm, blockchain, environment sensing system

✉ Corresponding author.
E-mail address: 18392130966@163.com (W. Chang).
Received 10 January 2024; Revised 15 March 2024; Accepted 25 December 2024; Published Online 16 April 2025.
DOI: 10.61091/jcmcc127b-496

1. Introduction

Decision tree algorithm is a commonly used machine learning algorithm, which is widely used in data mining, pattern recognition, intelligent recommendation and other fields. Its simple and intuitive characteristics make the decision tree algorithm become one of the popular research directions in the field of artificial intelligence [1-3]. However, the traditional decision tree algorithm is deficient in some problems, such as easy overfitting, difficult to deal with continuous attributes, etc. Therefore, decision tree optimization is of great significance. In traditional decision tree algorithms, the commonly used selection criteria are information gain and Gini index [4-7]. Information gain is based on the information entropy to assess the degree of uncertainty reduction in the data set, while the Gini index is to assess the difficulty of classification in the data set. In order to optimize the performance of the decision tree algorithm, it can be improved by choosing a suitable selection criterion [8-11]. In addition, improved pruning is a commonly used optimization method in decision tree algorithms. It reduces the complexity of the decision tree by deleting some unnecessary leaf nodes, so as to achieve the purpose of optimization. In addition, methods such as feature selection and continuous attribute processing are also commonly used in the optimization of decision tree algorithms [12-15].

Data encryption is an important information security technology, by encrypting the data can ensure the confidentiality and integrity of the data in the transmission and storage process. With the rapid development and popularization of the Internet, data transmission has become an important link in modern society [16-19]. However, the ensuing problem is the protection of data security. In the process of data transmission, data are easily threatened by hacking, stealing or tampering. Therefore, the design of data encryption algorithms has become necessary to ensure the security of data transmission [20-23].

Aiming at the data transmission encryption problem of unmanned equipment environment sensing system, this paper realizes the shared data encryption scheme design based on blockchain technology and fully homomorphic encryption algorithm. In order to improve the operation efficiency and security of the blockchain in the system, this paper optimizes the PBFT consensus algorithm using the C4.5 decision tree algorithm, and verifies the optimization effect of the algorithm in terms of throughput, consensus delay, security and other aspects. In the design of shared data encryption scheme, this paper utilizes the full homomorphic encryption algorithm, effectively combines it with the blockchain, and experimentally compares the performance of the scheme in terms of encryption and decryption time, throughput, etc., as well as data security with other schemes, to verify the validity of this paper's scheme.

2. Application of blockchain technology in unmanned equipment environment sensing system

2.1. Demand analysis of environmental awareness capabilities for unmanned ground equipment

In order to better realize the decision tree optimization and data encryption algorithm design for unmanned equipment environment sensing system, this paper firstly analyzes the environment sensing capability demand of ground unmanned equipment.

Ground unmanned equipment is characterized by “intelligence”, “autonomy” and “remote operation”, which requires the equipment to have the ability to adapt to a variety of complex battlefield environments and conduct reconnaissance and surveillance. In order to effectively fulfill the unmanned combat mission, the equipment is required to have the ability to adapt to various complex battlefield environments and conduct detection and surveillance. In order to effectively complete the unmanned combat tasks and improve the survivability of unmanned equipment, the ground unmanned equipment must have a full range of perception capabilities, build battlefield maneuvering maps, in order to correctly complete the obstacle avoidance and autonomous exploration, path

planning and other decision-making, to enhance situational awareness, and to effectively reduce the cognitive burden of the soldiers and the workload. Therefore, this paper will take the combat application as the traction, refine the ground unmanned equipment environment perception related ability demand, from the performance of single equipment, consider the role of unmanned equipment in the combat formation needs, combined with the mutual cooperation, mutual coordination, mutual complementary relationship with the manned equipment, ground unmanned environment perception related ability demand and specific equipment system/unit function, performance, to establish a rational The test model of environment sensing capability is established.

Refine the conditions and capabilities of environmental perception that unmanned ground equipment must meet in the process of performing combat missions, and complete the mapping from combat missions to the performance and functional indexes of weapons and equipment.

The operational perspective is based on the unmanned equipment in the process of performing combat tasks, analyzing the combat mission or task it may receive, combining its role in the combat formation, combat objectives, and combat scale and other factors, and refining the combat activities it needs to perform in order to complete the overall combat requirements. Take the U.S. Army ground unmanned equipment as an example, according to the weight of the ground unmanned system will be divided into micro, small, medium and large ground unmanned system, the different volume of unmanned equipment to use the occasions, in the joint formation of the role assumed by the role that needs to be accomplished naturally there will be a great deal of difference, so the need to be from the characteristics of the equipment from the formation of the task to carry out the operational perspective of the demand analysis.

The capability perspective is to abstract the capability that the unmanned equipment needs to have in the process of executing combat activities. The content analyzed under the capability perspective mainly includes the structure of the capabilities required to accomplish the mission, the supporting resources and the interactions between the capabilities. The near-term goal of current unmanned equipment is to utilize the endurance of unmanned vehicles to achieve long working hours, conduct continuous reconnaissance and surveillance in complex terrains or in situations where soldiers' reconnaissance and combat capabilities are limited, enter places that are inaccessible to human beings, improve survivability, reserve reaction time, and utilize artificial intelligence to reduce the cognitive burden of soldiers.

The system perspective starts from within the equipment, implements the operational requirements and capability needs into the physical equipment, and analyzes and searches for a variety of internal influencing factors that may affect the overall effectiveness of the equipment from the hardware equipment, software functions, and various aspects of system operation. At present, the internal factors related to the environment perception capability of unmanned equipment mainly include different types of sensor equipment, vehicle attitude perception equipment and its corresponding perception fusion algorithm.

Based on the DoDAF multi-view method, the extraction process of ground unmanned equipment environment perception capability is as follows:

1) Starting from the mission perspective, clarify the combat tasks, utilize the combat tasks to decompose the process of combat activities, and refine the combat tasks related to the environment perception capability. In the current research context of ground unmanned equipment, the combat tasks mainly include reconnaissance and surveillance tasks, communication relay tasks, and damage assessment tasks.

2) Capability perspective correlation, establish the mapping relationship between combat tasks/combat activities and capabilities, and form the preliminary environment-aware capability requirement elements.

3) System perspective refinement, the need to combine the current ground unmanned equipment equipment characteristics and system composition, further expansion and refinement of the

capabilities, the formation of a complete environmental awareness capabilities evaluation hierarchy relationship diagram. Such sorted out environmental awareness capability requirements, with top-down logical relationships and hierarchical relationships, will be abstract military needs, transformed into detailed equipment combat capabilities, mapped to specific equipment test indicators. The result of the ground unmanned equipment environment perception capability refinement is shown in Figure 1.

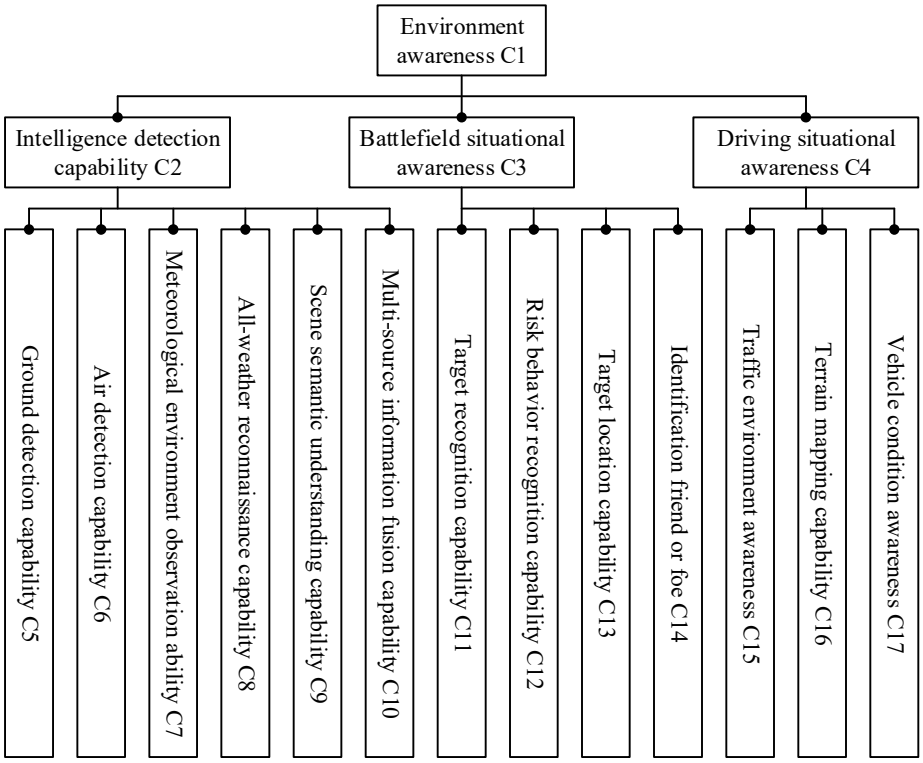


Fig. 1. Detailed environmental perception ability of ground special human equipment

2.2. Blockchain technology

Blockchain is, in a narrow sense, a distributed ledger that synthesizes a chained data structure in the form of blocks of data sequentially connected in chronological order and guaranteed by cryptographic means to be unforgeable and tamper-proof. Broadly speaking, blockchain is a new distributed infrastructure and computing paradigm that utilizes a block-chain data structure to verify and store data, a distributed node consensus algorithm to generate and update data, cryptographic means to ensure the security of data transmission and access, and smart contracts composed of automated script code to program and manipulate data.

A block is the basic data unit of a blockchain and contains information about all transactions. The block consists of two parts: the block header and the block body. The block header is mainly composed of the hash value of the parent block, timestamp, Merkle tree, etc. The block body generally contains a list of transactions, and the hash value of the parent block saved in each block header uniquely points to the parent block of the block, which constitutes a connecting relationship between the blocks, thus composing the basic data structure of the blockchain.

The tamperability of the blockchain is mainly ensured by the chain structure formed by the hash value of the parent block. The hash algorithm is a hashing algorithm that can generate a fixed-length string from an input of any length through computation, and the output fixed-length string is the hash value of the input. Through each block contains the hash value of the previous block of data in this layer nested way, connected to the block chain, to achieve tamper-proof. And through the hash algorithm to build Merkel tree, binary tree method to achieve rapid detection of data changes.

All nodes in the hash blockchain are involved in recording data, there is no central node, therefore, a consensus mechanism is needed to ensure that all nodes can record the same correct data. The current blockchain is divided into four main types of consensus algorithms: proof-of-work consensus algorithms, Po*'s credential-based consensus algorithms, Byzantine fault-tolerant algorithms, and consensus algorithms that incorporate trusted execution environments. The introduction of smart contracts has driven the development of blockchain in other areas. Smart contracts are promises defined in a digital program that are automatically executed by a computer program when certain conditions are met. Blockchain-based smart contracts need to include a complete state machine for receiving information about processing conditions, a thing processing mechanism, and a data storage mechanism. After the smart contract is deployed, it will be permanently stored on the blockchain and strictly enforced. When the preset conditions are triggered, the nodes in the network all respond and verify the content, and at the same time record the transaction data in the blockchain. The network architecture used in blockchain is P2P (peer-to-peer) network, which breaks the traditional centralized network structure.P2P network has no central node, and is maintained by the user group within the network.Any single or small number of node failures of the P2P network won't affect the operation of the system, so the reliability of the P2P network is very strong.

2.3. Application of blockchain in unmanned equipment environment sensing

The existing environment sensing unit, when providing intelligence information for the battlefield command nodes, basically provides information for the relevant intelligence processing or command nodes in a multi-point-to-point manner, and once the central node is destroyed or paralyzed, the corresponding environment sensing nodes basically lose their roles, with a fragile network structure and a weak flexible organization capability. The introduction of blockchain distributed network architecture in unmanned equipment environment sensing system has the following advantages:

1) Improve the destruction resistance of the intelligence system. The existing environment sensing unit and intelligence processing unit are in a star-type networking mode, and once the intelligence processing unit collapses or is destroyed, the corresponding environment sensing unit will be invalid. Environmental sensing unit on the chain, can greatly improve the destruction resistance.

2) Improve the real-time sharing of environmental sensing information. The sharing of environmental sensing information between environmental sensing units needs to be guaranteed by other node networks. The uplinking of environmental sensing units can greatly improve the real-time sharing of intelligence information.

3) Improve the efficiency of distributed processing of intelligence. Each environmental sensing unit can directly process, uplink and release the first discovered intelligence, and other units only carry out fusion and comparison to ensure the uniqueness of the information and reduce the amount of transmission of fusion processed information.

4) Improving the information processing quality of the environment sensing unit. Each environmental sensing unit can effectively improve the quality of released information by comparing the environmental sensing information of other units.

3. Optimization of PBFT consensus algorithm based on C4.5 decision tree

In this chapter, the optimization strategy of PBFT consensus algorithm is designed by using C4.5 decision tree algorithm, and the blockchain system based on the optimized PBFT algorithm is applied to the unmanned equipment environment sensing system in order to improve the operation efficiency and safety of the system.

3.1. Shortcomings of the PBFT consensus algorithm

The PBFT consensus algorithm [24] can be a good solution to the problem of blockchain chain forks, but the algorithm also has some limitations:

1) All consensus nodes have equal chances to act as a master node, i.e., a master node in View is elected by the master node selection formula $p = v \bmod |R|$. If the newly elected master node is a non-honest node, it will result in continuous master selection, and the external service of the whole blockchain system during the master node selection period will be greatly reduced or even not be able to provide normal external service.

2) The PBFT consensus algorithm transmits messages through message broadcasting, which includes ViewChange protocol, Timeout mechanism, etc., in addition to Pre-prepare, Prepare, Commit three-phase broadcasting, which makes the network bandwidth consumption aggravated by too much message broadcasting. In a blockchain network with a consensus node number of N, the number of messages that need to be delivered to complete a three-phase broadcast is Z, then the relationship between the two is:

$$Z = 2N^2 - 2N \tag{1}$$

3.2. C4.5 Decision Tree Algorithm

Decision tree is a very important classifier, which has the advantages of simple and easy to learn, high classification accuracy and low domain knowledge requirement. This paper focuses on the information entropy based C4.5 decision tree algorithm, C4.5 is improved on the basis of ID3 [25]. The process of categorizing consensus nodes according to the C4.5 decision tree algorithm process is as follows:

1) Calculate the information entropy. It measures whether the sample sets belong to the same category. Assuming that the proportion of samples of category k in the current training set D is P_k, then the category information entropy is calculated based on the training set D as:

$$Info(D) = -\sum_{k=1}^{K} P_k \log 2P_k \tag{2}$$

where K denotes the total number of categories in the sample.

2) Calculate the information gain. In the training set D, the attribute feature vector a is selected to divide the set D, and the feature vector a has a total of i possible values, then, taking the vth value is selected in the sample set D with the value of a^r, and is noted as D^v. The information entropy of D^v can be computed from Eqn. (2), and the branch nodes are assigned the weight $\frac{|D^r|}{|D|}$ to solve the problem of the number of samples contained in the branch nodes is different. Then the information gain of using attribute feature vectors a to partition the set D can be expressed as:

$$Gain(D,a) = Info(D) - \sum_{v=1}^{i} \frac{|D^v|}{|D|} Info(D^v) \tag{3}$$

However, the use of information gain to select the division attributes will cause it to favor the attributes with more selected values, thus affecting the generalization ability of the decision tree. Therefore, the C4.5 decision tree algorithm chooses the gain rate to select the division attributes.

3) Calculate the gain rate.The calculation of gain rate in C4.5 decision tree algorithm is extended on the basis of information gain.The training set D uses the gain rate of attribute feature vector a as shown in equation (4):

$$Gain_ratio(D,a) = \frac{Gain(D,a)}{IV(a)} \tag{4}$$

Among them:

$$IV(a) = -\sum_{v=1}^{i} \frac{|D^v|}{|D|} \log_2 \frac{|D^v|}{|D|} \tag{5}$$

3.3. Coalition Chain Consensus Environment

In the Hyperledger Fabric Alliance blockchain network, the PBFT consensus algorithm is used to complete the block consistency validation work, but since the PBFT consensus algorithm can only complete the consensus efficiently with no more than 100 consensus nodes. Therefore, among the many blockchain system participants, not all nodes need to participate in the block consensus verification work. With this, this paper defines the consensus nodes in the coalition chain network into the following two categories:

1) Consensus nodes. In the coalition blockchain network, the consensus node is composed of the master node and the slave node, both of which need to pack and save the verified data into the ledger maintained by themselves. The master node is responsible for broadcasting the requests sent by the client to the whole network, and verifying and processing the block and transaction information. Slave nodes are responsible for verifying and processing blocks and transaction information.

2) Non-consensus nodes. In a federated blockchain network, non-consensus nodes only need to synchronize the transaction data verified by the consensus nodes to the blockchain ledger maintained by themselves. It should be noted here that non-honest nodes may cause malicious attacks on the blockchain system only if they exist in the consensus nodes. Therefore, the focus of the optimization of the PBFT consensus algorithm in this paper is on the consensus nodes.

In a federated blockchain network, let set A be denoted as a node of total size i, set A_c be denoted as a consensus node of size j, and non-consensus nodes be denoted as set A_d, which is of size $i-j$. They satisfy the following aggregation relation among themselves:

$$A = A_c \cup A_d \tag{6}$$

And:

$$A_c \cap A_d = \varnothing \tag{7}$$

Suppose that the nodes in set A are numbered sequentially as $\{0,...,i-1\}$, where the krd node is denoted as A_k. It satisfies: for any A_k, $A_k \in A$ and k is in the range $\{k \mid 0 \le k \le i-1\}$.

And is satisfied:

$$A_k \in A_c, \{k \mid 0 \le k < j\} \tag{8}$$

$$A_k \in A_d, \{k \mid j \le k < i\} \tag{9}$$

In a federated blockchain network, there will be non-honest nodes. It is known that in set A_c with consensus node size j, suppose $\exists f$ dishonest nodes are denoted as set B_f. Then, in the practical Byzantine fault-tolerant algorithm, the non-honest nodes satisfy the following relationship with the total consensus nodes:

$$B_f \subseteq A_c \tag{10}$$

$$3f+1 \le j \tag{11}$$

Equation (10) shows that the set of dishonest nodes B_f is a true subset of the set of consensus nodes A_c. From Equation (11), it is easy to prove that the total number of consensus nodes must be greater than or equal to $3f+1$ in the coalition chain that solves the Byzantine General problem, and the value range of j is $\{j \mid 4 \le j \le 100\}$ in the PBFT consensus algorithm.

In a federated blockchain network, consensus nodes validate the transactions that need to be written into the blockchain through the appropriate mechanism, and only the validated data can be written into the blockchain network that they maintain. The non-consensus nodes update the data verified by the consensus nodes into the blockchain network maintained by them.

3.4. Optimization strategy of PBFT algorithm based on C4.5 decision tree

Aiming at the deficiencies of PBFT consensus algorithm, this paper designs the optimization strategy of PBFT algorithm based on C4.5 decision tree by introducing voting weight.

3.4.1. Holistic thinking. According to the consensus node trust evaluation classification will be $|R|$ consensus node in the blockchain network, according to the trust level of high and low sequential order numbered $\{0,\ldots,|R|-1\}$, if the current master node failure or overthrown, the use of the formula (12) for the re-selection of the master node.

$$p = v \bmod |R| \tag{12}$$

In addition, on the basis of using C4.5 decision tree to evaluate and classify the trust degree of consensus nodes in the blockchain network, this paper draws on the research idea of Tendermint algorithm, introduces the concept of voting right value, and adopts the idea of "the better the trust degree, the larger the voting right value" to reflect the differences between consensus nodes. The definition of consensus node's voting right value is as follows:

In a blockchain network with a number of consensus nodes of $|R|$, the reliability and responsiveness of the validation message provided by node k during the consistency consensus validation process is expressed in the form of a voting weight value denoted as $Weight_k$, which is shown in Equation (13), and $Weight_k \in (0, \frac{1}{3}]$:

$$Weight_k = \frac{1}{k+3} : \{k \mid 0 \le k \le |R|-1\} \tag{13}$$

Then, the sum of the voting power values of all consensus nodes in the blockchain, denoted as SUM_{weight}, is:

$$SUM_{weight} = \sum_{k=0}^{R-1} Weight_k \tag{14}$$

Here, this paper draws on the research methodology of PoW consensus algorithm, adopting the idea of "minority to majority", and setting the threshold value of the voting right value $Weight_{ibrechold}$ to be 0.5 times of the total voting right value, that is:

$$Weight_{ihreshold} = \frac{SUM_{weight}}{2} \tag{15}$$

3.4.2. Algorithm design. According to the overall idea of PBFT consensus algorithm based on C4.5 decision tree (DTPBFT), its detailed algorithm processing flow is shown below:

Step1: In the blockchain system, the client sends a request message (data on the chain) $< REQUEST, o, t, c >$ to the master node p.

Step2: When the blockchain network detects a new transaction request, it will first determine whether there are new consensus nodes joining or old consensus nodes exiting in the whole network. If there are joining or exiting consensus nodes, all consensus nodes are re-categorized for creditworthiness assessment according to the C4.5 decision tree classification model, and the consensus nodes are numbered, assigned with voting values and selected as master nodes according to the classification results. If no consensus node joins or quits then verify the status of master node p in the current view v.

Step3: If the master node p in the current view v is in the down state, trigger the ViewChange protocol to switch the master node. Otherwise, the master node p receives the request from the client and assigns the proposal number n to the request and prepares the message $< PRE\_PREPARE, v, n, digest >, message >$ to the network-wide multicast, where digest is the summary of the message.

Step4: After receiving the pre-prepared message, other slave nodes start to check the digest legitimacy and judge whether the value of n is correct or not. If the judgment result is true, it sends a ready message with its own vote value to other consensus nodes $< PREPARE, v, n, digest, k, Weight_k >$. Otherwise, the slave node does not do any operation.

Step5: The consensus node collects the ready messages from other nodes. If the consensus node collects more than $Weight_{\text{ulreshold}}$ ready messages (including its own voting right value), the node enters the ready state and multicasts the confirmation message to the whole network $< COMMIT, v, n, D(m), k, Weight_k >$. Otherwise, the consistency consensus validation work of this transaction request ends, and the node sends the consensus failure result $< FAILURE, v, t, c, k, Weight_k >$ to the client.

Step6: The consensus node collects the confirmation messages from other nodes. If the consensus node collects more than $Weight_{threshald}$ confirmation messages (including its own voting right value), the node enters the commit state, so it can execute the client request and return the successful result $< SUCCESS, v, t, c, r, k, Weight_k >$ to the client. Otherwise, the consistency consensus verification of this transaction request ends, and the node sends the consensus verification failure result $< FAILURE, v, t, c, k, Weight_k >$ to the client.

Step7: The client only receives the same execution results from different consensus nodes when the total voting power value is greater than the total voting power value of the consensus node to send the request failure results, the request is considered to be correctly executed successfully, and will be r as the final result of the request, the consensus node to update their own maintained ledger, the algorithm has been executed.

3.5. Experimentation and Analysis

3.5.1. Security analysis of consensus nodes. In this paper, C4.5 decision tree algorithm is used to classify the trust level of the nodes participating in the consensus of the coalition blockchain network in the unmanned equipment environment sensing system. Through this experimental dataset using C4.5 algorithm, KNN algorithm, plain Bayesian algorithm, Logistic, Perceptron and Maximum Entropy Classifier are compared and observed, and the results of the comparison of classification algorithm accuracy are obtained as shown in Fig. 2.

500 experimental data in this paper, including 400 training samples and 100 test samples. From the experimental results, it can be seen that C4.5 Decision Tree has the best classification accuracy of 94.37%, and the use of C4.5 Decision Tree can effectively improve the accuracy of classification of consensus nodes and improve the security of the algorithm.

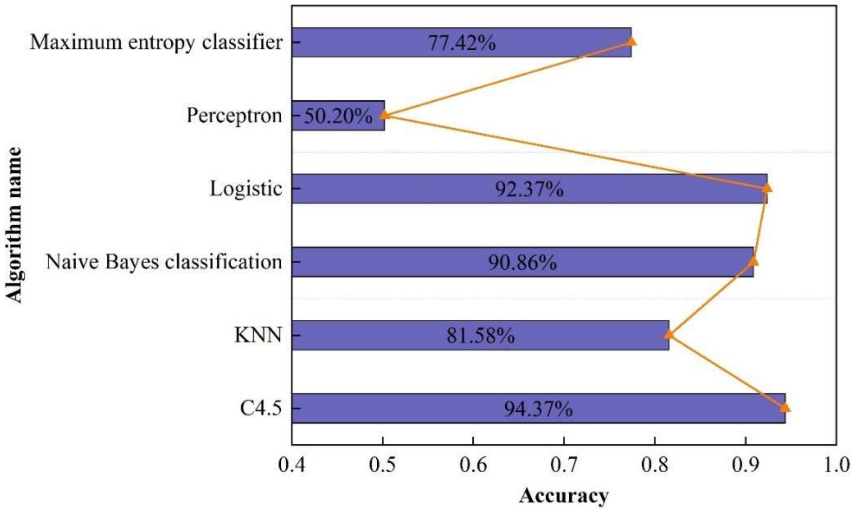


Fig. 2. Comparison of classification algorithm accuracy

3.5.2. Experiments and analysis of results. To test this paper's DTPBFT consensus algorithm based on C4.5 decision tree in a coalition chain scenario, the experiment is based on Hyperledger fabric, as well as its testing tool Hyperledger caplier, which builds a coalition chain network with 80 nodes, and runs PBFT and DTPBFT algorithms respectively. By sending different messages through the nodes, the Byzantine attack is simulated.

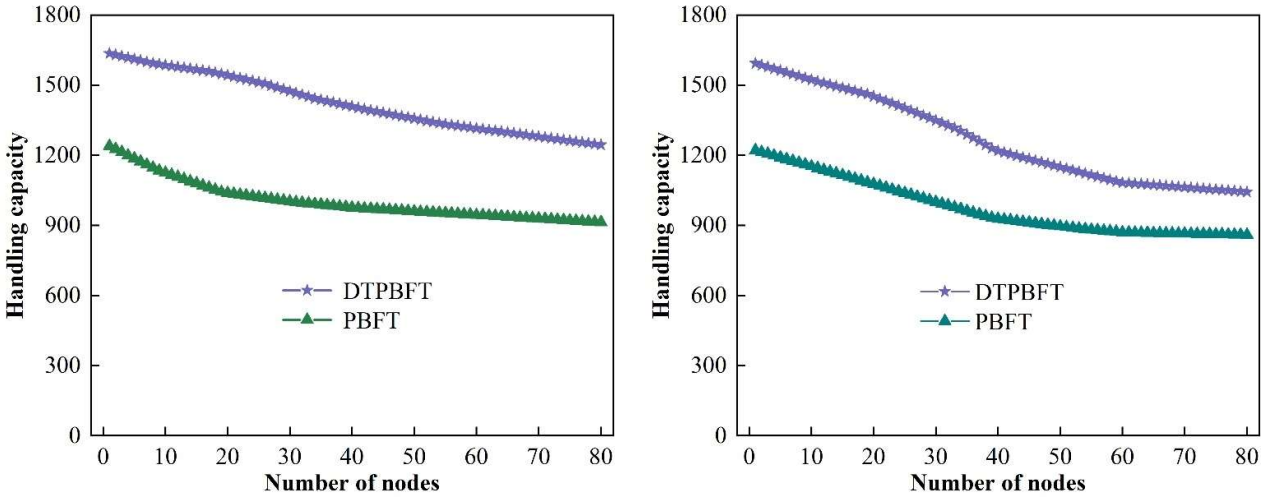
1) Throughput. Throughput refers to the number of transactions processed per unit of time, and in consensus algorithms, it refers to the number of transactions that can be successfully agreed upon per unit of time. Higher throughput proves that the consensus algorithm itself has higher performance, the throughput formula is:

$$S = \frac{N}{T} \tag{16}$$

where: S is the throughput, N is the number of successful consensus transactions, and T is the total time.

In the experiment, the client is set to send 1800 messages to record the number of transactions that can complete the consensus per second, and the test is carried out for different numbers of nodes in the presence and absence of Byzantine nodes within the system, respectively, and the results of the throughput experiments are shown in Fig. 3, with (a) and (b) showing the throughput when there are Byzantine nodes and absence of Byzantine nodes, respectively.

From the throughput line graph, it can be seen that the size of throughput is inversely proportional to the number of nodes participating in the consensus. In the presence of Byzantine nodes within the system, the throughput size of DTPBFT is always higher than the PBFT algorithm for the same number of nodes.



(a) Throughput (no Byzantium) (b) Throughput (existence of Byzantium)

Fig. 3. Throughput comparison

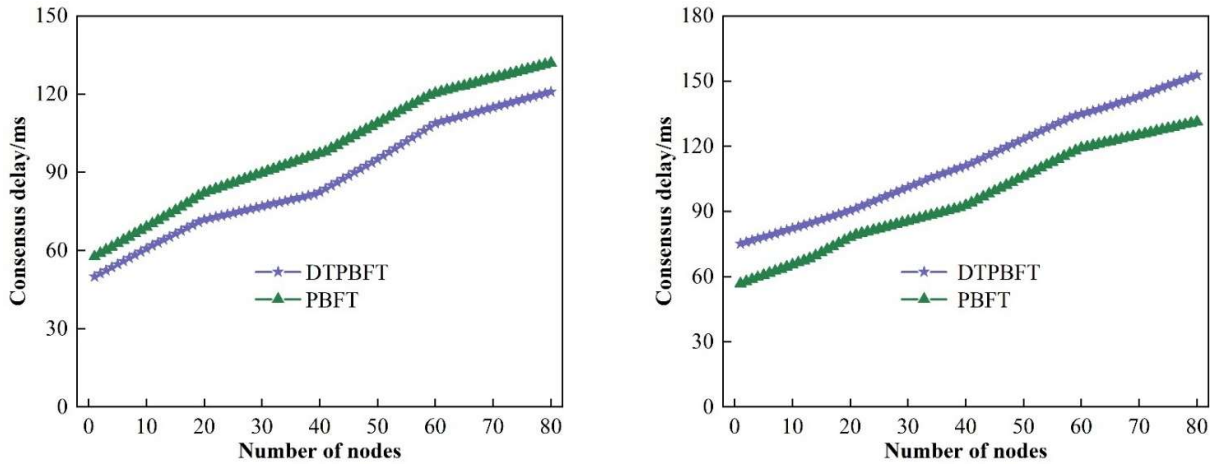
2) Consensus delay. Consensus latency, as an important measure of consensus algorithm, represents the time difference between the initiation and completion of a transaction in a blockchain. Lower latency makes the blockchain more available and secure. The test latency formula is:

$$DelayTime = T(submit) - T(authentication) \tag{17}$$

where: $T(submit)$ is the time when the consensus is completed and confirmed, and $T(authentication)$ is the time when the consensus authentication phase begins.

The experiments are averaged every 120 transactions and conducted under different numbers of nodes, and the results of the consensus delay experiments are shown in Fig. 4, with (a) and (b) showing the consensus delay when there are Byzantine nodes and when there are no Byzantine nodes, respectively

From the consensus delay comparison graph, it can be seen that the DTPBFT algorithm is faster than the PBFT algorithm when there is no existence of Byzantine nodes within the system, but when there is existence of Byzantine nodes within the system, the consensus delay is slightly higher than that of the PRET algorithm in the worst case scenario of the DTPBFT algorithm.



(a) Consensus delay (no Byzantium) (b) Consensus delay (existence of Byzantium)

Fig. 4. Comparison of consensus delay

3) Algorithm security test. Aiming at the problem that the PBFT algorithm cannot dynamically eliminate Byzantine nodes, the DTPBFT algorithm introduces the concept of voting weight value, which eliminates Byzantine nodes in the system by embodying the difference between consensus nodes to guarantee the security of the system. In the experiment, the number of preset nodes inside the system is 1200, in which there are 400 Byzantine nodes, and there will be Byzantine nodes joining in the process of system operation, running PBFT algorithm and DTPBFT algorithm, respectively, and then checking the number of Byzantine nodes inside the system according to the increase of the number of consensus rounds. The experimental results of security testing are shown in Fig. 5.

From the experimental results, it can be seen that the number of Byzantine nodes inside the system after running 12 rounds is 9, which shows that when the system tends to stabilize, the DTPBFT algorithm can remove the Byzantine nodes inside the system, and the probability of Byzantine nodes being elected as master nodes decreases, which ensures the security of the system. After a new Byzantine node joins, it cannot complete the consensus process and is removed by the system, while the number of Byzantine nodes in the PBFT algorithm is basically unchanged. Therefore, the experiment proves that the DTPBFT algorithm has high security performance.

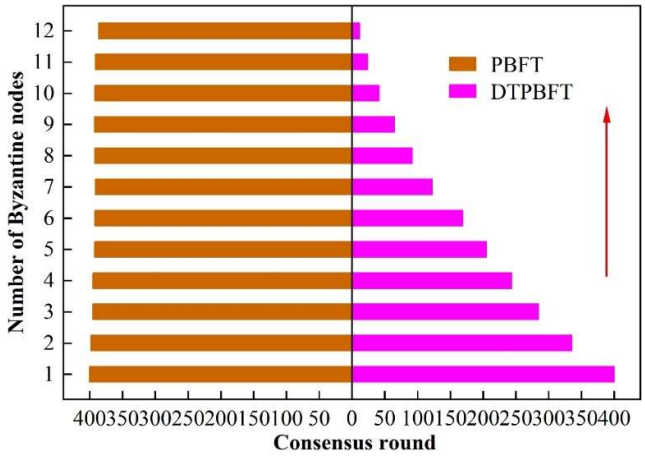


Fig. 5. Number of Byzantine nodes

4. Blockchain shared data encryption scheme for unmanned equipment environment sensing system

4.1. Overall Program Architecture

In order to realize the secure sharing of environmental sensing intelligence data in the unmanned equipment environmental sensing system, this paper designs a shared data encryption scheme based on blockchain and fully homomorphic encryption algorithm as shown in Fig. 6, and its specific process is divided into six stages: initialization, institutional teaming, key acquisition, data encryption, ciphertext computation and decryption and sharing [26]. The characters involved in the process and the meaning of the characters are shown below:

Env_{FHE} - Fully homomorphic encryption environment.

Env_{FHE_CAL} - Fully homomorphic cryptographic computing environment.

GID - Group unique identifier.

CID_{group} - User ID group.

DID_{group} - Data organization ID group.

DI_i - Data Organization I.

DID - Data organization ID.
$status$ - Grouping status.
PK_i - ECC encrypted public key of organization i.
SK_l - ECC encryption private key of the institution.
PK_{FHE} - Full homomorphic encryption public key.
SK_{FHE} - Fully homomorphic encryption private key.
$Addr_{group}$ - Ciphertext file address group.

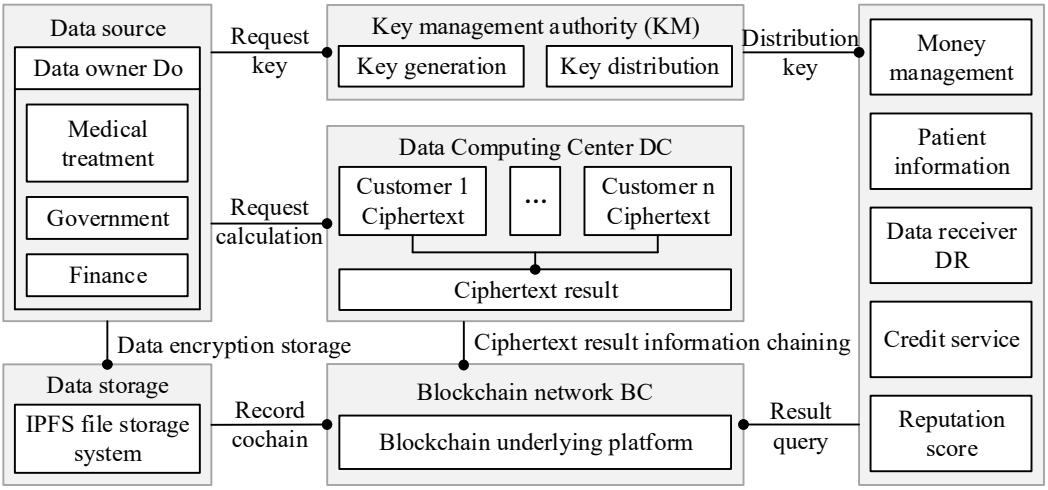


Fig. 6. Shared data encryption scheme

4.2. Composition of the program phase

4.2.1. Initialization phase. The execution subjects of the initialization phase include the key management organization KM and the data computation center DC.In this phase, KM and DC perform the following tasks:

Stop1: KM performs full homomorphic encryption parameter initialization and constructs an environment for full homomorphic encryption CKKS Env_{FHE}, and DC constructs a homomorphic computing environment Env_{FHE_CAL}.

Stop2: KM initializes a finite domain F_q and constructs the environment for Shamir secret sharing.

4.2.2. Institutional teaming phase. The execution subject of the organization teaming phase includes k data organization Dl, and one of the data organizations initiates the teaming application. The specific steps are as follows:

Stop1: All the institutions participating in data sharing will join the blockchain network as blockchain nodes, and the authentication of institutions will be carried out through the blockchain network to ensure that each data institution is legitimate.

Stop2: Data organization Dl_i applies for access to user data, and through the blockchain network, sends a teaming request transaction to other data organizations, with on-chain storage GID, CID_{group}, DID_i and the state $status$, $status$ of that teaming set to the default value of $false$. That is:

$$Request_{group} = \{GID, CID_{group}, DID_i\} \tag{18}$$

Stop3: The data organization that listens to the teaming request considers whether to agree to teaming based on CID_{group}. Specifically, the user ID group corresponding to the user dataset owned by the data organization is made to be CID. Judgment:

$$CID_{group} \subseteq CID \tag{19}$$

If Eq. (19) holds, the message agreeing to the grouping is returned, and vice versa for not agreeing. That is, $Resonse_{group} = \{true / false\}$.

Stop4: Data organization DI_i determines whether the teaming is successful or not based on the results returned by other data organizations. If it is successful, it sends a success message to the other data organization, and vice versa, it sends a failure message. That is $Resonse_{res} = \{true / false\}$.

Stop5: Data organization FI_i modifies the status of teaming stored on the chain to $true$ and writes it to the ID group of the data organization participating in the teaming, i.e., FID_{group}. Therefore, the fields stored on the chain at this time include:

$$Info_{group} = \{GID, CID_{group}, DID_{group}, DID_i, status\} \tag{20}$$

4.2.3. Key Acquisition Phase. The executing subjects of the key acquisition phase include k data organization DI and the key management organization KM.The specific steps of this phase are as follows:

Stop1: After the data institution joins the blockchain network, it utilizes the existing asymmetric encryption algorithm ECC, invokes the key generation algorithm $ECC.Keygen()$, applies for a public-private key pair PK_i and SK_i for itself, and stores PK_i in the blockchain network for subsequent encrypted transmission of the private key slice.

Stop2: The data organization DI_i sends a key application request $Request_{\text{key}} = \{GID, DID_i^{'}\}$ to the KM to apply for a fully homomorphic key for encrypting data, where $DID_t^{'}$ is the unique identifier of the data organization.

Stop3: After the KM receives the key application request $Request_{key}$, the KM uses the GID to view the information stored in the blockchain and obtains the grouping information. Judgment:

$$status = true \tag{21}$$

$$DID_i^{'} = DID_i \tag{22}$$

If the verification of Eq. (21) or (22) fails, the key request phase ends, otherwise the next step is carried out.

Stop4: KM calls $CKKS.Keygen(1^{\lambda})$ according to security parameter λ to generate fully homomorphic public-private key pairs PK_{FHE} and SK_{FHE}.

Stop5: KM slices the private key SK_{FHE} according to the number of data organizations k in DID_{group} and sets the number of minimum recovery keys to p. Select $p-1$ values $a_1, a_2, ..., a_{p-1}$ in F_q and construct the following polynomial where $f(0) = SK_{FHE}$:

$$f(x) = SK_{FHE} + a_1 \cdot x + a_2 \cdot x^2 + \cdots + a_{p-1} \cdot x^{p-1} \tag{23}$$

Stop6: Select k different non-zero elements $x_1, x_2, ..., x_k$ in F_q, and for each data organization DI_i, compute the value of the polynomial $f(x_i)$ according to Eq. (23) to obtain the secret slice $(x_i, f(x_i))$.

Stop7: The KM queries the corresponding public key PK_i from the blockchain network via the DID_i of each data organization, calls the encryption algorithm $ECC.Encrypt(PK_i, (x_i, f(x_i)))$, and encrypts the secret fragment $(x_i, f(x_i))$ using the PK_i to obtain the encrypted fragment CT_t; and then returns the fully homomorphic public key PK_{FHE} and the encrypted fragment CT_i to the respective data organization DI_i. The SK_{FHE} and polynomial $f(x)$ are destroyed.

4.2.4. Data encryption phase. After the key application is successful, the data encryption stage will be entered, and the main body of the execution of this stage is the k data organization FI. Its steps are described as follows:

Stop1: Each data organization DI_i queries the blockchain record based on the grouping unique identifier GID to get $Info_{group}$.

Stop2: Each data organization DI_t encrypts the user data in CID_{group}. First, encoding algorithm $CKKS.Ecd(Data, \Delta)$ is invoked to encode the data to obtain a plaintext polynomial. Specifically, for user u_t, the data $Data_{ut} = \{d_{t1}, d_{t2}, ..., d_{tn}\}$ is encoded to obtain plaintext $m(X) = \{m(X)_1, m(X)_2, \ldots, m(X)_n\}$. Then, the encryption algorithm $CKKS.Enc(M, PK_{FHE})$ is invoked to perform a fully homomorphic encryption operation utilizing a fully homomorphic public key PK_{FHE} to obtain a ciphertext $CT = \{CT_{u_1}, CT_{u_2}, ..., CT_{u_m}\}$, each as in Equation (24):

$$CT_{u_t} = \begin{cases} CKKS.Enc(m(X)_1, PK_{FHE}) \to CT_{t1} \\ CKKS.Enc(m(X)_2, PK_{FHE}) \to CT_{t2} \\ CKKS.Enc(m(X)_n, PK_{FHE}) \to CT_{tn} \end{cases} \tag{24}$$

Stop3: Each data organization will upload the encrypted ciphertext CT_{u_t} to IPFS and get the file address $herf_i$.

Stop4: The data organization DI_i initiated by the teaming will create the ciphertext address group $Addr_{group}$, add its own ciphertext address $herf_i$ to $Addr_{group}$, and then initiate the ciphertext upload transaction to store the GID and $Addr_{group}$ into the blockchain, i.e., the teaming information $Info_{group}$ on the chain is increased:

$$Addr_{group} = \{herf_1, herf_2, ..., herf_n\} \tag{25}$$

Stop5: The other data organization obtains the ciphertext address group $Addr_{group}$ through the blockchain network, adds the ciphertext address $herf_i$ to $Addr_{group}$, and then uploads it to the blockchain, and sends an upload success message to DI_i after the upload.

Stop6: DI_i After receiving messages from all the grouped data organizations, it initiates a data computation request $Request_cal = \{k, GID\}$ to the data computation center DC to request the computation of the ciphertext data.

4.2.5. Cryptographic computation phase. The main body of the ciphertext calculation phase execution is the Data Computation Center DC, and the specific steps of this phase are as follows:

Stop1: After DC receives $Request_{cal}$, it queries the blockchain record $Info_{group}$ based on the grouping unique identifier GID to obtain the stored ciphertext address group $Addr_{group}$, and gets a set of file addresses $herf_i$, where $i = 1, 2, ..., k$.

Stop2: DC queries the data from IPFS based on this set of file addresses $herf_t$, and obtains k the ciphertext.

Stop3: DC performs computation operations on each field data of k ciphertexts, and here, in order to simplify the description, it performs addition operations on them. According to the actual business scenario, different types of operations can be performed. That is, for user u_t, the result of the computed ciphertext is:

$$CT_{u_{tcal}} = \begin{cases} CKKS.Add(CT_{t1_1}, CT_{t1_2}, ..., CT_{t1_k}) \to CT_{t1_{cal}} \\ CKKS.Add(CT_{t2_1}, CT_{t2_2}, ..., CT_{t2_k}) \to CT_{t2_{cal}} \quad t = 1, 2, ..., m \\ CKKS.Add(CT_{t3_1}, CT_{t3_2}, ..., CT_{t3_k}) \to CT_{t3_{cal}} \end{cases} \tag{26}$$

The ciphertext result corresponding to m user is:

$$CT_{cal} = \{< u_1, CT_{u_{1cal}} >, < u_2, CT_{u_{2cal}} >, ..., < u_m, CT_{u_{mcal}} >\} \tag{27}$$

Stop4: The DC uploads the ciphertext data CT_{cal} to IPFS, returns the ciphertext file address $herf$, and then initiates a resultant ciphertext upload transaction, storing the GID and the address $herf$ where the computed ciphertext CT_{cal} is located in the blockchain.

Stop4: DC returns the computation completion message to DI_i.

4.2.6. Decryption sharing phase. After the data organization receives the message that the DC ciphertext computation is completed, it will enter the decryption sharing phase. The execution body of the decryption sharing phase includes the key management organization KM and the data organization DI, and the specific steps of this phase are as follows:

Stop1: DI_i After receiving the message that the ciphertext computation is completed, query the blockchain record according to the unique identifier GID of the group to obtain the ciphertext address $herf$.

Stop2: DI_i Send a request for key recovery to other teamed data organizations $Requestres\{GID, FID_i\}$.

Stop3: After receiving the request, the other teamed data organization queries the blockchain record through the teamed unique identifier GID to obtain the teamed organization group DID_{group}, and verifies:

$$DID_i \in DID_{group} \tag{28}$$

If the verification of formula (28) fails, it ends;

Stop4: If the verification of formula (28) succeeds, the other organizations use the public key of DI_i to encrypt the secret fragment to get the ciphertext CT_i and return it to data organization DI_i.

Stop5: After data organization DI_i obtains p the secret fragment, it can perform key recovery. First, DI_i calls the decryption algorithm $ECC.Decrypt(CT_i, SK_i)$ and decrypts the ciphertext fragment CT_i using its own private key SK_i to obtain the original p secret slice $(x_1, y_1), (x_2, y_2), ..., (x_p, y_p)$.

Stop6: The data mechanism DI_i constructs a polynomial according to Eq. (29) using the p secret fragment:

$$f(x) = \sum_{i=1}^{p} (y_i \prod_{1 \le j \le t, j \ne i} \frac{x - x_j}{x_i - x_j}) \tag{29}$$

Let $x = 0$, find $f(0)$, at this time can recover the full homomorphic private key $f(0) = SK_{FHE}$.

Stop7: DI_i obtains the data CT_{cal} of the user ciphertext from IPFS according to the address $herf$, calls the decryption algorithm $CKKS.Dec(CT_{cal} SK_{FHE})$, decrypts CT_{cal} using the private key, and obtains the statistically decrypted user ciphertext $m(X)$. Finally calls the decoding algorithm $CKKS.Dec(m(X), \Delta)$, and for the input decrypted text polynomial $m(X)$, computes the corresponding decrypted data Data.

4.3. Experimental validation of data encryption scheme

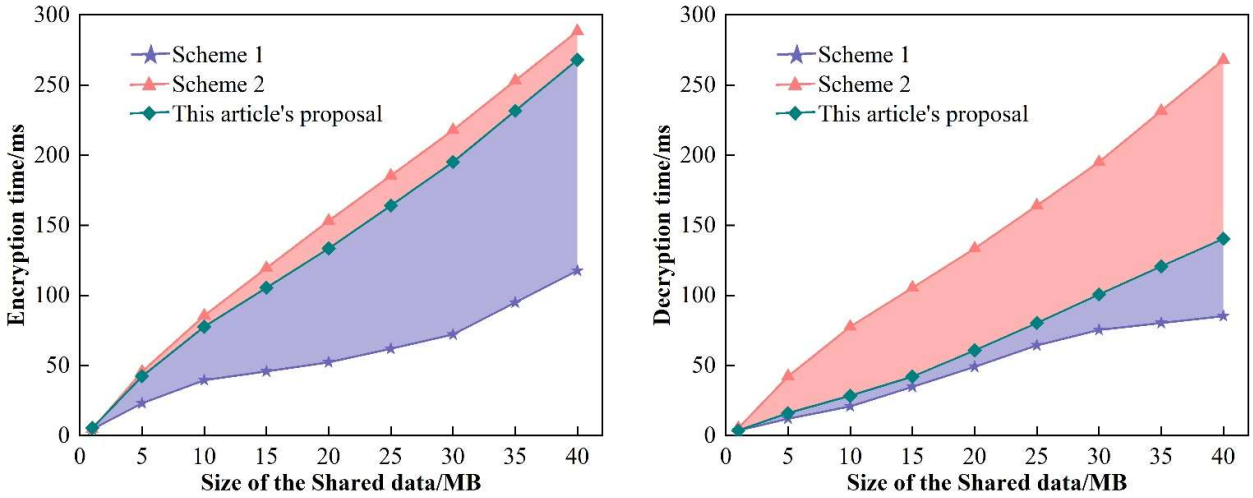
In the experimental phase of the blockchain shared data encryption scheme for the unmanned equipment environment sensing system, the test environment built in this paper uses an experimental platform equipped with an Intel Core i7-11800H processor (running at 3.40GHz) and 32GB RAM memory. The experiments are conducted in the Ubuntu operating system environment built based on VMware virtualization technology, which exists in the form of a virtual machine. In this environment, Hyperledger Fabric version 2.4 is used to build the blockchain as well as IPFS version 0.14 distributed file storage system as the off-chain cloud storage platform, which together serve the realization of data sharing.

4.3.1. Performance analysis:

1) Encryption and decryption time comparison. In the process of data sharing, performance consumption is inevitably involved, which mainly includes the cost of encrypting and decrypting the data as well as the cost incurred when transactions are conducted on the blockchain. In order to evaluate these costs comprehensively, this paper measures the encryption and decryption overhead of the original data, as well as the encryption and decryption overhead of the hash values in detail. In order to test the superiority of this scheme in the process of encryption and decryption of raw data, this paper analyzes it in comparison with two similar recent technological frameworks. In this case, scheme 1 uses AES symmetric encryption algorithm while scheme 2 recommends proxy re-encryption technique to protect data privacy. To ensure the reliability and accuracy of the experimental results, this paper performs each operation 80 times, and the resulting average values of the time taken to encrypt and decrypt the original data are shown in Fig. 7(a) and Fig. 7(b), respectively. Where, the shaded area indicates the difference in time consumed between different schemes.

The data from Fig. 7(a) and Fig. 7(b) shows that AES encryption and decryption incurs the lowest execution overhead cost, while proxy re-encryption technique incurs the highest execution overhead cost. This is mainly due to the fact that the AES encryption and decryption process utilizes an efficient byte cycling mechanism, which greatly accelerates the process of encrypting and decrypting the original data. However, the AES encryption method is prone to key leakage due to the fact that the data owner needs to share the key with the data demander in advance as the same key is used for two-way

operation. In contrast, the scheme in this paper does not need to share the key in the process of encrypting and decrypting the original data, and thus has advantages in data security. In addition, the proposed scheme in this paper is more efficient in encrypting and decrypting shared data compared to proxy re-encryption techniques. This is because the proxy re-encryption technique needs to convert the ciphertext into different ciphertexts through a proxy server to adapt to the decryption needs of different data demanders, which naturally increases the realization cost of the system.



(a) Comparison of Encryption Time

(b) Comparison of Decryption

Fig. 7. Comparison of encryption and decryption time

2) Throughput size comparison. Data throughput refers to the number of transactions completed per second on the blockchain, which can reflect the transaction cost overhead on the chain. In this paper, we conduct a detailed measurement of data throughput to evaluate the transaction cost of storing and querying on the chain, and compare this paper's scheme with Scheme 1, which does not include off-chain storage, and Scheme 2, which adopts the master-slave blockchain architecture to store data. In Scheme II, the slave chain stores the shared data and the master chain mainly stores the hash values of these shared data. To ensure the reliability of the experimental data, the experiment is repeated 30 times to get the average value, and the throughput size comparison results are shown in Figure 8.

From the data in Fig. 8, it can be seen that with the increase of data volume, the throughput of scheme 1, which does not have the function of off-chain storage, gradually declines, while the throughput of scheme 2 is even lower. Meanwhile, compared to these two schemes, the scheme proposed in this paper also maintains a stable and high throughput performance when the data volume increases, and is higher than the scheme without under-chain storage. This is mainly due to the fact that the master-slave blockchain architecture used in scheme 2 will frequently interact across the chain when storing and querying data, which reduces the efficiency of storing and querying data. In contrast, the scheme in this paper only stores the encrypted hash values on the chain, which occupies much less space than storing the complete data directly, and thus the proposed scheme shows obvious advantages in terms of data throughput.

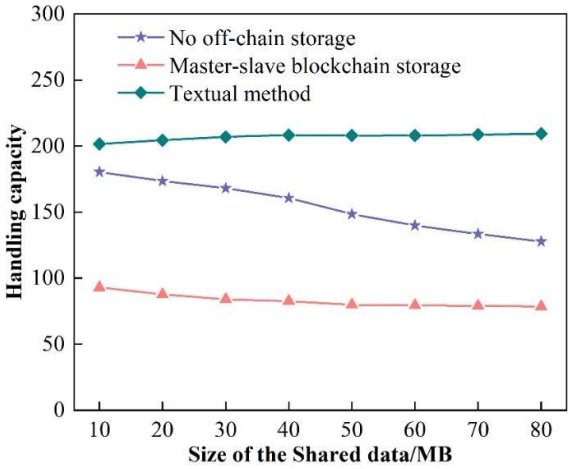


Fig. 8. Comparison of throughput sizes

4.3.2. Comparison of data security. The shared data encryption scheme for unmanned equipment environment sensing system designed in this paper is designed based on blockchain technology and fully homomorphic encryption algorithm, aiming to realize efficient and secure environment sensing data sharing under the premise of maintaining data privacy. In order to verify the reliability of the scheme, this section analyzes the security in terms of off-chain data security, on-chain data security, data integrity, trustworthiness, authorization and permission, authentication, reliability, and verifiability, and compares it with other related schemes. The security comparison results are shown in Table 1.

1) Off-chain data security. The data security protection scheme designed in this paper is based on homomorphic encryption technology, in which the data owner encrypts the original data by customizing the access structure, thus realizing fine-grained access control of the off-chain data, which in turn enhances the data security.

2) On-chain data security. This paper adopts full homomorphic encryption algorithm to store the data hash value as ciphertext on the blockchain, and at the same time, it supports the operation of the encrypted data, which realizes the operation of the data on the basis of guaranteeing the security of the data, and the user who owns the private key can solve the calculated data to get the result of the calculation, so as to enhance the security of the on-chain data.

3) Data Integrity. The shared data storage scheme proposed in this paper stores the off-chain data encrypted in IPFS, which can realize permanent storage. Meanwhile, the encrypted hash value stored on the chain cannot be tampered with, thus ensuring data integrity.

4) Trustworthiness.b The data security storage scheme in this paper stores the shared encrypted data through IPFS, while the hash value of the encrypted data is stored by the blockchain. Based on the uniqueness of the hash value, the data demander can compare the hash value of the data under the chain with the decrypted hash value on the chain to verify whether the data is trustworthy or not, and enhance the trustworthiness.

5) Authorization Permission. This study ensures that only authorized data demanders can decrypt the encrypted hash value stored on the blockchain and download the ciphertext of the data stored in IPFS. And the data ciphertext can be decrypted only when the attribute set of the data demander matches the attribute set set by the ciphertext decryption, thus ensuring data confidentiality.

6) Authentication. In the scheme designed in this paper, both the data owner and the demander must be authenticated by the system to access the system. The system will provide the key only after authentication.

Table 1. Security comparison

Peculiarity	Text scheme	Contrast scheme Ⅰ	Contrast scheme Ⅱ	Contrast scheme Ⅲ	Contrast scheme Ⅳ
Data integrity	Yes	Yes	Yes	Yes	Yes
Off-chain data security	Yes	No	No	No	No
On-chain data security	Yes	Yes	No	No	No
Believability	Yes	Yes	Yes	Yes	No
Authorized permission	Yes	Yes	Yes	No	No
Identity authentication	Yes	No	Yes	No	No
Reliability	Yes	Yes	No	Yes	No
Verifiability	Yes	Yes	Yes	No	No

5. Conclusion

This paper combines the PBFT optimization algorithm based on C4.5 decision tree (DTPBFT) and full homomorphic encryption algorithm to realize the design of shared data encryption scheme in unmanned equipment environment sensing system, which improves the efficiency and security of data transmission in unmanned equipment environment sensing system.

In this paper, the C4.5 decision tree algorithm is used to classify the trust level of the consensus nodes of the blockchain in the unmanned equipment environment sensing system, and comparing the classification algorithms such as the C4.5 algorithm, the KNN algorithm, the plain Bayesian algorithm, the Logistic, the perceptron, and the maximum entropy classifier, the C4.5 decision tree has the best classification accuracy (94.37%), i.e., the use of the C4.5 decision tree can effectively improve the accuracy of classification of consensus nodes and the security of PBRT consensus algorithm, which verifies the effectiveness of the optimization strategy of PBRT algorithm. Comparing the optimized DTPBFT algorithm with the PBFT algorithm, it can be seen that the DTPBFT algorithm is generally more effective in terms of throughput size and consensus delay, and can effectively remove the Byzantine nodes within the system, which has higher security performance.

In addition, comparing with other data encryption schemes, the shared data encryption scheme designed in this paper based on blockchain and fully homomorphic encryption algorithms achieves better experimental results in terms of performance indicators such as encryption and decryption time, throughput, and data security, and has higher operational performance and security, and is able to provide a guarantee of data security sharing in the unmanned equipment environment sensing system.

References

[1] Patel, H. H., & Prajapati, P. (2018). Study and analysis of decision tree based classification algorithms. International Journal of Computer Sciences and Engineering, 6(10), 74-78.

[2] Charbuty, B., & Abdulazeez, A. (2021). Classification based on decision tree algorithm for machine learning. Journal of Applied Science and Technology Trends, 2(01), 20-28.

[3] Gupta, B., Rawat, A., Jain, A., Arora, A., & Dhami, N. (2017). Analysis of various decision tree algorithms for classification in data mining. International Journal of Computer Applications, 163(8), 15-19.

[4] Damanik, I. S., Windarto, A. P., Wanto, A., Poningsih, Andani, S. R., & Saputra, W. (2019, August). Decision tree optimization in C4. 5 algorithm using genetic algorithm. In Journal of Physics: Conference Series (Vol. 1255, No. 1, p. 012012). IOP Publishing.

[5] Mienye, I. D., Sun, Y., & Wang, Z. (2019). Prediction performance of improved decision tree-based algorithms: a review. Procedia Manufacturing, 35, 698-703.

[6] Wang, H., Wang, T., Zhou, Y., Zhou, L., & Li, H. (2019). Information classification algorithm based on decision tree optimization. Cluster Computing, 22, 7559-7568.

[7] Zhang, H., & Zhou, R. (2017, July). The analysis and optimization of decision tree based on ID3 algorithm. In 2017 9th International Conference on Modelling, Identification and Control (ICMIC) (pp. 924-928). IEEE.

[8] Carreira-Perpinán, M. A., & Tavallali, P. (2018). Alternating optimization of decision trees, with application to learning sparse oblique trees. Advances in neural information processing systems, 31.

[9] Jati, W. K., & Muslim, L. K. (2020, November). Optimization of decision tree algorithm in text classification of job applicants using particle swarm optimization. In 2020 3rd International Conference on Information and Communications Technology (ICOIACT) (pp. 201-205). IEEE.

[10] Zharmagambetov, A., Hada, S. S., Gabidolla, M., & Carreira-Perpinán, M. A. (2021, July). Non-greedy algorithms for decision tree optimization: An experimental comparison. In 2021 International Joint Conference on Neural Networks (IJCNN) (pp. 1-8). IEEE.

[11] Chikalov, I., Hussain, S., & Moshkov, M. (2018). Bi-criteria optimization of decision trees with applications to data analysis. European Journal of Operational Research, 266(2), 689-701.

[12] Brunello, A., Marzano, E., Montanari, A., & Sciavicco, G. (2017). Decision tree pruning via multi-objective evolutionary computation. International Journal of Machine Learning and Computing, 7(6), 167-175.

[13] Mijwil, M. M., & Abttan, R. A. (2021). Utilizing the genetic algorithm to pruning the C4. 5 decision tree algorithm. Asian J. Appl. Sci, 9(1).

[14] Elmachtoub, A. N., Liang, J. C. N., & McNellis, R. (2020, November). Decision trees for decision-making under the predict-then-optimize framework. In International conference on machine learning (pp. 2858-2867). PMLR.

[15] Gomes Mantovani, R., Horváth, T., Rossi, A. L., Cerri, R., Barbon Junior, S., Vanschoren, J., & Carvalho, A. C. D. (2024). Better trees: an empirical study on hyperparameter tuning of classification decision tree induction algorithms. Data Mining and Knowledge Discovery, 1-53.

[16] Atadoga, A., Farayola, O. A., Ayinla, B. S., Amoo, O. O., Abrahams, T. O., & Osasona, F. (2024). A comparative review of data encryption methods in the USA and Europe. Computer Science & IT Research Journal, 5(2), 447-460.

[17] Wen, C., Li, X., Zanotti, T., Puglisi, F. M., Shi, Y., Saiz, F., ... & Lanza, M. (2021). Advanced Data Encryption using 2D Materials. Advanced Materials, 33(27), 2100185.

[18] Matta, P., Arora, M., & Sharma, D. (2021). A comparative survey on data encryption Techniques: Big data perspective. Materials today: proceedings, 46, 11035-11039.

[19] Gai, K., Qiu, M., & Zhao, H. (2017). Privacy-preserving data encryption strategy for big data in mobile cloud computing. IEEE Transactions on Big Data, 7(4), 678-688.

[20] Yang, P., Xiong, N., & Ren, J. (2020). Data security and privacy protection for cloud storage: A survey. Ieee Access, 8, 131723-131740.

[21] Zhang, D. (2018, October). Big data security and privacy protection. In 8th international conference on management and computer science (ICMCS 2018) (pp. 275-278). Atlantis Press.

[22] Mushtaq, M. F., Jamel, S., Disina, A. H., Pindar, Z. A., Shakir, N. S. A., & Deris, M. M. (2017). A survey on the cryptographic encryption algorithms. International Journal of Advanced Computer Science and Applications, 8(11).

[23] Oladoyinbo, T. O., Oladoyinbo, O. B., & Akinkunmi, A. I. (2024). The Importance Of Data Encryption Algorithm In Data Security. Current Journal of International Organization of Scientific Research Journal of Mobile Computing & Application (IOSR-JMCA), 11(2), 10-16.

[24] Xutong Zhu, Xiaoxuan Hu & Waiming Zhu. (2024). RGPBFT: A Reputation-Based PBFT Algorithm with Node Grouping Strategy. Arabian Journal for Science and Engineering(prepublish),1-14.

[25] Aprianti Aprianti, Sifai Izzatul Alifah, Nurjanah Nurjanah, Ratna Wulan Widya, Ratnawati Juli & Ahsan Abdillah. (2024). Early Detection for Determinant Factor of National Health Insurance Membership for Workers in Central Java Using C4.5 Algorithm. BIO Web of Conferences.

[26] Babenko L K & Tolomanenko E A. (2021). Development of algorithms for data transmission in sensor networks based on fully homomorphic encryption using symmetric Kuznyechik algorithm. Journal of Physics: Conference Series(1),012034-.