Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 5886024, 11 pages
https://doi.org/10.1155/2021/5886024

Research Article

Resource Sharing of Smart City Based on Blockchain

Tao Huang

The University of Queensland, St Lucia Campus, Brisbane, St Lucia QLD 4072, Australia

Correspondence should be addressed to Tao Huang; tao.huang2@uq.net.au

Received 30 August 2021; Revised 23 September 2021; Accepted 25 September 2021; Published 20 October 2021

Academic Editor: Deepak Gupta

The concept of smart city refers to the improvement of the quality of life of the city by making full use of idle resources by sharing. However, limited by the technical level, the current resource sharing system mostly adopts centralized data storage mode. Systems managed in this way are vulnerable to multiple threats. The tested blockchain technology with the characteristics of decentralization and tamper resistance can effectively prevent various risks. Starting with the architecture of blockchain intelligent contract, this paper puts forward a structural optimization factor model of intelligent contract. To optimize the structure of blockchain intelligent contract, the gas optimization theory is put forward by changing the order, reducing the use of costly EVM data fields, reducing redundant fields, and optimizing intelligent contract codes. Experimental analysis of the proposed model is carried out, and the effectiveness of the proposed method is verified by comparing the transaction execution time of cost calculation with the cost of executing gas, which can provide reference for the selection of intelligent contract organization structure of smart city resource sharing system.

1. Introduction

The concept of smart city refers to the improvement of the quality of life of the city by making full use of idle resources by sharing. Due to the limitation of technical level, most of the current resource sharing systems adopt centralized data storage mode. Systems managed in this way are vulnerable to multiple threats. In addition, the traditional sharing method cannot establish an objective and true credit mechanism for bad users, which leads to potential safety hazards. The tested blockchain technology with the characteristics of decentralization and tamper resistance can effectively prevent various risks. This paper proposes an algorithm to predict transaction execution time and gas execution cost and verifies the effectiveness of the algorithm by experiments, which can provide reference for the selection of intelligent contract organization structure in smart city resource sharing system.

Literature [1] proposes a method based on network tomography to realize low-cost, scalable, and flexible monitoring deployment of the road network system. It can complete the path matching of smart city system. Literature [2] studies the specific positioning and components of data center. Based on the construction requirements of data center, it constructs an architecture of smart city, related contents, and construction ideas that data center should contain in it and discusses the application of data center in smart city. Literature [3] proposes an infrastructure based on blockchain to realize sustainable Internet of Things (IoT) sharing economy in super smart cities, so as to support space-time intelligent contract services for security and privacy. Literature [4] constructs the smart city blockchain vehicle information network system. By running the system in a decentralized way, a multipoint collaborative transportation management system can be built. Literature [5] proposes an efficient, scalable, and blockchain-based distributed smart city network architecture-light-fidelity (Li-Fi) communication technology. In order to solve the rapid increase in the number and diversity of smart devices connected to the Internet, literature [6] establishes a new multi-interconnection system framework of smart cities. Through the design of hybrid architecture, the architecture is divided into two parts: core network and edge network, which improves the efficiency and solves the limitations of the current architecture. Literature [7] proposes a multivariate system model which is divided into core network and edge network and puts forward a working proof scheme in the model. Literature [8] uses blockchain technology to take a global view of the security policy in the system and integrate it into the FIWARE platform. Literature [9] deeply explores the association between sensor networks and blockchain smart contracts. The concept of "rolling blockchain" is put forward, which can be used as a network node to build wireless sensor networks with the participation of smart cars. Literature [10] puts forward a workflow chart of technical experiment to explore how blockchain technology can protect the integrity of sensor data when the Internet of Things is the infrastructure. A data management framework of Internet of Things is constructed, which can strengthen data-driven operation. Literature [11] identifies the basic elements of smart cities, then provides a detailed literature on the existing technologies for realizing smart cities, highlights their shortcomings, and explains how blockchain can help the effective implementation of these technologies. Literature [12] proposes a decentralized architecture using blockchain, which can run ledger services on distributed networks. Literature [13] proposes a blockchain-based power transaction (B-ET) ecosystem and designs an intelligent contract to ensure that the transaction is conducted in a safe and reliable way. Literature [14] proposes an infrastructure based on blockchain to support space-time intelligent contract services for security and privacy, which can be used for sustainable sharing economy in megasmart cities. Literature [15] proposes a smart city information exchange blockchain network, which is helpful to establish sharing among several untrusted organizations. The proposed approach enables one organization to safely use data from another organization to participate in business operations without having to access the data.

According to the research status proposed in this paper, there is no description of the differences in contracts in blockchain. However, there is no intelligent contract method for resource sharing mechanism. The intelligent contract method proposed in this paper can reduce the operating cost and improve the overall performance.

2. Brief Introduction of Blockchain Intelligent Contract

2.1. Brief Introduction of Blockchain Technology

2.1.1. Blockchain Technology. Blockchain is a distributed ledger, which can provide verifiable evidence of transactions between sender and receiver. It has no central repository, and it is continuously maintained by many investors (possibly individuals or organizations).

From the perspective of discipline boundary, the research topics of blockchain include mathematics, cryptography, data network, and many other directions. In practical application, blockchain plays its sharing function, which can be applied to the network architecture where multiple individuals interact, and has antitampering function. Blockchain can reduce the probability of omitting all transaction information in the network and complete the necessary repair and maintenance work at the necessary time. Blockchain is essentially a distributed database. Bitcoin, Ethereum, etc. use blockchain as the underlying technology and use the Hasi specification method to generate a series of data fields. These data blocks all have data messages that generate the next block, and they all contain a record of the complete error process, which is used to verify whether the information is valid and ensure its security.

The three types of blockchain are private chain, alliance chain, and public chain. All data in the public chain is accessible to the public, including Bitcoin and Ethereum. However, alliance chain and private chain will have access restrictions in Table 1.

The relationship between blockchain and multiple data blocks can be understood as the relationship between record list and record chain. Blockchain network can be regarded as a decentralized account book of public network. The ledger is also publicly shared among users. Blockchain creates a constant database for transactions between individuals in the network. Each item stores data with a timestamp and is linked to the previous item, and each digital record or transaction can be reorganized into a block linked to a particular participant. At a certain level, these individual data blocks together constitute an information network that is difficult to modify and attack, and only when there is consensus among individuals can it be maintained and updated. Blockchain technology ensures the credibility of information in the matching system by this method. In short, you can think of blockchain as a record-based repository, which can replicate data on a large peer-to-peer computer network instead of a central server.

Blockchain uses the Secure Hash Algorithm (SHA) or Hash Encryption Method. Hashing algorithms do not use secrets such as passwords or keys to provide security. A hashing algorithm can convert any part of information data such as numbers (such as text and pictures) into data fields having a prescribed length. The National Institute of Standards and Technology (NIST) has developed a hash specification, which is publicly available to government and private users.

2.1.2. P2P Networks. P2P (peer-to-peer) network is a peer-to-peer network architecture. At present, the dominant network structure is client-server mode. In this network structure, there are two roles: client and server, and the server serves the client as a center. Through the P2P network, the traditional top-down structure is broken, but each node provides services to each other, and each node is both a client and a server. This decentralized approach allows permissions to be distributed over multiple network nodes instead of centralizing on a single central server.

Because of the decentralized propagation of the P2P network, the blockchain technology can distribute data fields to each node through intelligent contracts to realize distributed accounting. In this manner, multiple copies of the data fields are provided and stored in each node where consensus is reached. When the ledger is threatened by tampering, attack, etc., only the data nodes stored on a single node are maliciously attacked, and the whole blockchain network will not be affected. Therefore, by increasing the geometric

Table 1: Differences between block chains.

Property	Private chain	Alliance chain	Public chain
Participant	Members within the organization	Alliance member	Anyone
Read permission	Restricted	Public or restricted	Public
Consensus mechanisms	Solo and PBFS	PBFS, Kafka, Raft	PoW, PoS, and DpoS
Efficiency	High	Low	Low
Represents	Multichain	Hyperledger fabric	Bitcoin and Ethereum

multiples of attack cost and difficulty, the blockchain network meets the requirements of tamper resistance and transaction security, and the blockchain network has the characteristics of decentralization and extensibility.

Each node in a peer-to-peer network has two functions, one is to maintain its own data information, and the other is to verify and propagate the data information of other nodes. Blockchain spreads the transaction data to individuals in each regional network, and after being tested by preset judgment conditions, it continuously spreads effective information in the network. It is this mechanism that ensures the accuracy and effectiveness of data information stored in blockchain network. Individuals in the regional chain network are distinguished by their own accommodation limits and are divided into all nodes with all information data from the establishment of the whole network to the current time, and light nodes with relatively weak accommodation capacity, such as various hard disks, which can only retain some information associated with themselves. There is interactive function between full node and light node. The decision mechanism of each individual in the local area network is formulated by the asymmetric encryption algorithm, so as to judge the transaction data. During the transaction, the public key (used to validate the transaction data) is paired with the private key (used to encrypt the transaction data).

The node uses the private key to sign the transaction information and then initiates the transaction. Once the transaction is accepted by the corresponding node, the initiator's public key is used to verify the received transaction. In blockchain technology, two parties who do not know each other do not need to be certified by centralized certification bodies but only need to trust the algorithm-based trading rules to establish mutual trust and reach consensus.

The emergence of the P2P technology provides an efficient and fast way for users to download network resources. Adding recommendation algorithm to the P2P sharing platform can enable users to capture the resources they are interested in in a small range, so as to effectively improve the utilization rate of resources and realize the rapid use and sharing of resources.

2.1.3. Consensus Mechanism. The blockchain network has decentralized and scalable data storage mode, which requires high trust between nodes, which needs to be realized by different consensus methods. At present, common consensus methods mainly include proof of rights and interests, proof of workload, proof of entrusted rights and interests, and practical byzantine fault tolerance in Table 2.

Among the four common blockchain consensus mechanisms, this paper adopts the consensus mechanism of entrusted rights and interests proof. In cryptocurrency technology, the consensus algorithm of entrusted rights proof is used to ensure the security and reliability of the whole blockchain network.

2.2. Smart Contracts. Intelligent contract is the core of resource sharing system. The research on intelligent contract operation mechanism and Ethernet gas mechanism can provide reference for the selection of intelligent contract organization structure of smart city resource sharing system. Smart contract can help make decisions, provide verification and execution functions, store data and trade functions for the system, and realize the applications needed by a variety of resource sharing systems. Figure 1 shows how smart contracts work. With the spread of transaction data, intelligent contracts are constantly taking effect at various nodes in the blockchain network. Smart contracts can be preinstalled with relevant condition settings and corresponding rules. When the conditions are met, corresponding operations can be performed when relevant functions are automatically triggered, thus realizing comprehensive management and control of physical and digital assets.

2.3. Encryption Algorithm. Encryption can protect data from theft or tampering. This key can be used for user authentication. There are three general and widely used encryption schemes: symmetric encryption, asymmetric encryption, and hash function. This paper introduces the homomorphic encryption algorithm.

Homomorphic encryption is regarded as a bright pearl in the field of cryptography, which is a method of processing data without accessing the data itself, that is, the result of processing ciphertext in encrypted state is the same as that of corresponding plaintext operation. Homomorphic encryption can realize ciphertext addition and multiplication at any time. In 2009, Pascal Paillier proposed a provable additive homomorphic secure encryption system. This scheme allows any computable function to operate the encrypted data, but it needs to be improved in practical application.

Assuming that M represents a set of plaintext and C represents a set of ciphertext, for a given key k satisfying formula (1), it is called homomorphic encryption scheme.

$$\forall m_1, m_2 \in M, E(m_1 \odot_M m_2) \longleftarrow E(m_1) \odot_C E(m_2). \quad (1)$$

Among them, the symbol can be calculated directly in

Table 2: Common blockchain consensus mechanism.

Consensus method	Proof of rights and interests	Proof of workload	Proof of entrusted rights and interests	Practical byzantine fault tolerance
Consensus efficiency	General	Low	High	High
The trust environment used	Untrusted	Untrusted	Untrusted	Semitrusted
Computing power/resource consumption	General	High	Low	Low
The ratio of fault-tolerant nodes	As the case may be	Less than or equal to 25%	As the case may be	Less than or equal to 33.3%
Security threat	Candidate cheating	Computing power centralization	Candidate cheating	Master node failure

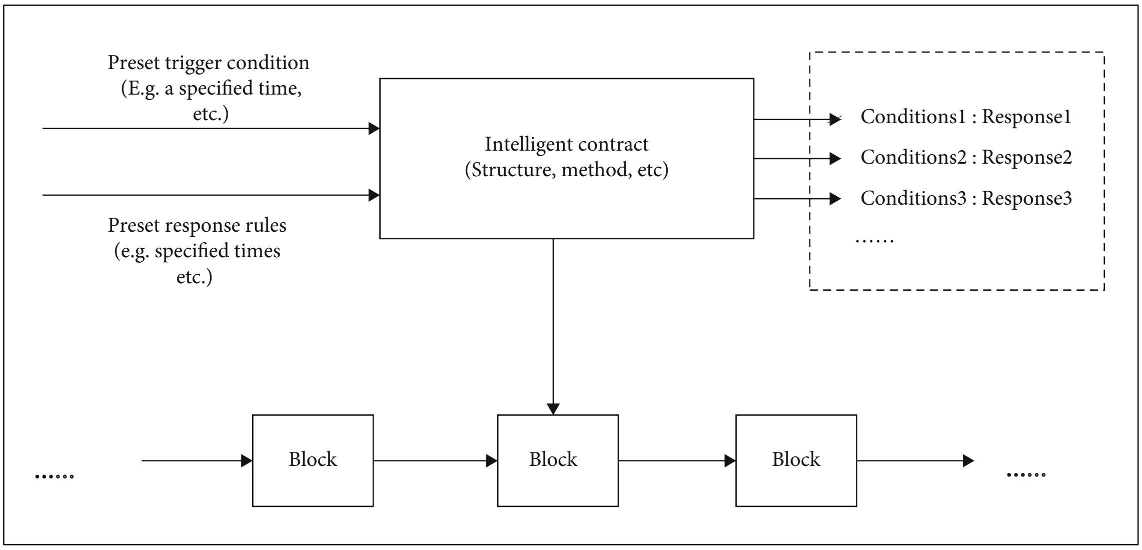
Preset trigger condition (E.g. a specified time, etc.)
Preset response rules (e.g. specified times etc.)
Intelligent contract (Structure, method, etc)
Conditions1 : Response1
Conditions2 : Response2
Conditions3 : Response3
......
......
Block
Block
Block
......

Figure 1: Operation mechanism of smart contract.

the set M and can be calculated directly in the set C, and neither calculation process needs any encryption and decryption.

At present, the additive homomorphism schemes are mainly Paillier, the multiplicative homomorphism schemes are RSA and ElGamal, the hybrid encryption schemes are BGV, EHC, NEHE, etc. and are given, and the ciphertext of + can be calculated. The principle of the algorithm is as follows:

Key generation section is as follows:

Step 1. p and q are two independent large prime numbers randomly selected, and $\gcd(pq, (p-1)(q-1)) = 1$, where gcd denotes finding the greatest common divisor.

Step 2. Calculate the parameter $n = pq$, $\lambda = \text{lcm}\ (p-1)\ (q-1)$, where lcm is the least common multiple.

Step 3. Choose a random integer i, and $i \in Z_{n^2}^*$.

Step 4. Calculate $\mu = (L(i^\lambda \bmod n^2))^{-1} \bmod n$, where $L(u) = (u-1)/n$.

Step 5. The public key is (n, i), and the private key is (λ, μ).

Data encryption process is as follows:

Step 1. Choose P as plaintext, and $0 \le p < n$.

Step 2. Choose the random number r, and $0 < r < n, r \in Z_n^*$.

Step 3. Calculate the ciphertext $C = i^{\mathrm{P}} \bullet r^n \bmod n^2$.

Homomorphic computation is as follows:

Step 1. Additive homomorphism of ciphertext: $D(E(m_1, r_1) \cdot E(m_2, r_2) \bmod n^2) = m_1 + m_2 \bmod n$.

Step 2. The ciphertext is multiplicative homomorphism with the constant k: $D(E(m_1, r_1)^k \bmod n^2) = kmk_1 \bmod n$.

Decryption process is as follows:

Step 1. The data ciphertext is $C \in Z_{n^2}^*$.

Step 2. The data plaintext $P = L(C^\lambda \bmod n^2) \cdot \mu \bmod n$ is calculated.

3. Research on the Factor Model of Structural Optimization of Smart Contract

3.1. Research on Gas Optimization Theory. The execution of each transaction in Ethereum has the attribute of transaction cost, that is, the gas consumed by transaction execution. When a transaction contains data fields, transaction costs consist of execution costs and additional costs per byte in the data fields. If the transaction cost is G_l, the execution cost is G_e, and the data cost is G_d, there are

$$G_l = G_e + G_d. \tag{2}$$

Ethereum foundation predefines the consensus rule of gas benchmark measurement according to opcodes, so the gas consumed by each EVM opcode is fixed. The execution cost is available in the transaction log and can be debugged in a debug manner, such as in the online compilation environment Remix, to obtain all the opcodes during the execution of the transaction in sequence. Let each opcode cost *Gas* be G, and there are

$$G_e = \sum G_o. \tag{3}$$

In the data cost, except that each transaction has a fixed value *Gas* cost, the *Gas* cost is different according to the byte type of the data field. The fixed cost per transaction is 21,000 *Gas*, 68 *Gas* is required for each nonzero byte of data or code in the transaction, and 4 *Gas* is required for each zero byte of data or code. Then, let the number of nonzero bytes and zero bytes of the additional data field of the transaction be N_x and N_y; then there are

$$G_d = 21000 + 68N_x + 4N_y. \tag{4}$$

According to formula (1), the transaction cost of each transaction in Ethereum is positively correlated with the data cost and execution cost. From formula (2) and formula (3), we know the formation mode of data consumption and execution consumption in Ethernet Square and put forward the optimization methods from these two aspects to reduce gas consumption.

According to the above theory, the smart contract code is optimized by reducing gas cost from three aspects: changing the order of variables, reducing the use of expensive EVM opcodes, and reducing redundancy and unreasonable design of smart contract codes.

3.1.1. Change the Order of the Quantities. The EVM virtual machine executes in a 32-byte group each time, so the compiler will try to merge variables into 32 bytes for execution. However, the compiler cannot automatically optimize variable grouping, and it divides 32 bytes into groups according to variables of static size. According to the different byte types of different variables, the access order can be adjusted to optimize.

3.1.2. Reduce the Use of Expensive EVM Opcodes. Gas of EVM opcode has been defined in advance. For example, adding ADD opcode consumes 3 *Gas*, dividing DIV opcode consumes 5 *Gas*, while SSTORE opcode means writing data, which requires 20000 *Gas* for the first writing and 5000 *Gas* for remodification. Therefore, taking SSTORE opcode as an example, it is expensive to modify it again, and the optimization direction needs to avoid repeated writing, that is, it can write as much data as possible at one time.

```
contract SingleContractModel{
uint cnt;
struct Dealing{
  uint dealingValue; }
mapping(uint => Dealing) private Dealings;
function buildDealing(uint _dealing Value){
    cnt += 1;
    var newDealing = Dealing({
      dealingValue:_dealingValue });
    Dealings[ent] = newDealing;
  }}
```

Code 1: Simplified code for a single smart contract.

Table 3: Offsets between predicted events and test time.

Trading volume	200000	400000	600000	800000	1000000
Test time (s)	0.289	0.265	0.214	0.336	0.346
Forecast events (s)	0.318	0.336	0.346	0.354	0.360
Offset value (s)	0.029	0.071	0.132	0.018	0.014

3.1.3. Optimize Smart Contract Code to Reduce Redundancy and Unreasonable Design Code. Reducing redundant and unreasonable code design can reduce the gas cost caused by code transmission and also reduce the use of EVM opcodes, thus reducing the gas cost more effectively. Therefore, it is necessary to try our best to remove redundant and unreasonable codes in the system with intelligent contract as the core.

3.2. Algorithm Establishment. Different from Bitcoin, the Ethernet platform uses Merkle Patricia tree instead of Merkle tree. By modifying the block structure, it avoids the fact that the original node can only obtain the global state of the system through the processed transactions, so that each region can clearly give the global state of the system. The structure of the MPT tree (that is, the Merkle Patricia tree) is the organizational structure of the Patricia tree. Ethernet uses the MPT tree to store the account Trie for special processing of key values and uses a special hexadecimal prefix (HP) encoding for key values, which corresponds to 16 characters in the alphabet. And before the node is inserted into the MPT tree, the key value is hashed once again by sha3 ( ), so that the key value is random, so the MPT tree used in the global state of Ethernet can be regarded as a random 16-fork Patricia tree. According to the relationship between tree height expectation and node size of randomly asymptotic Patricia tree, the relationship between tree height expectation and node number of binary Patricia tree is as

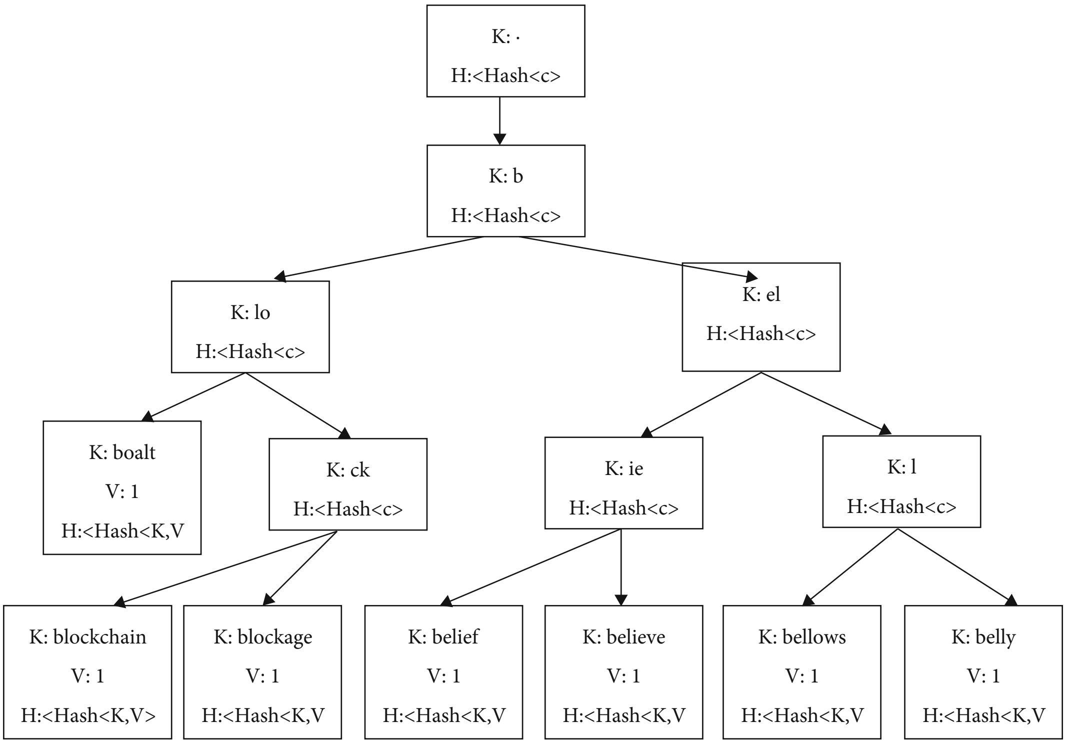

Figure 2: Ethernet MPT tree in gas theory.

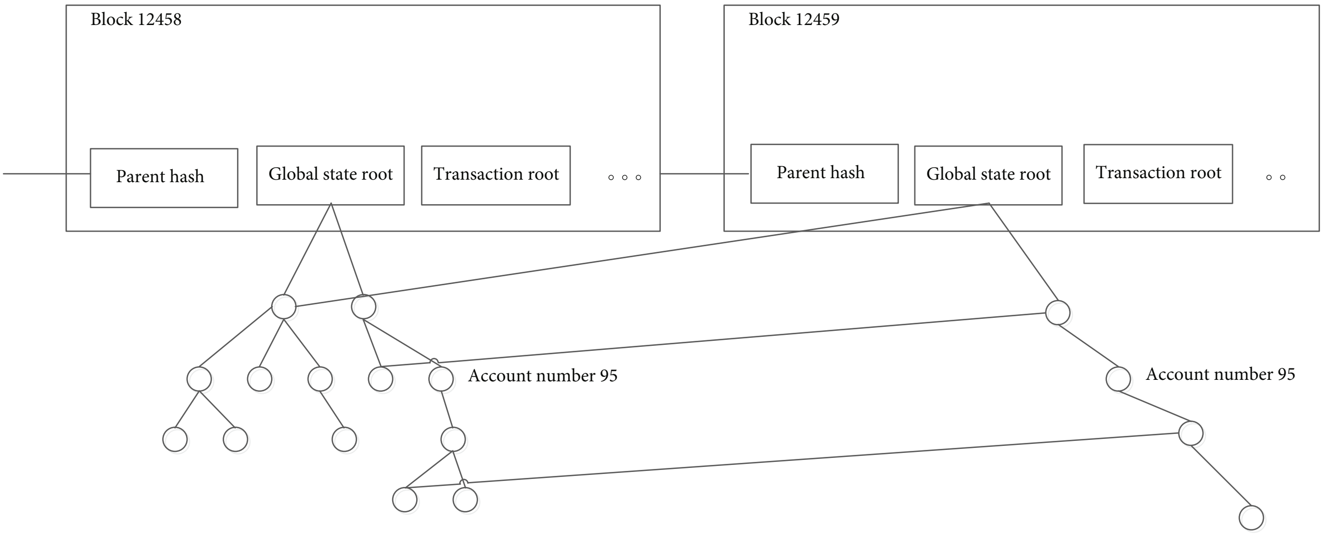

Figure 3: Transition of global state Trie of Ethereum.

follows:

$$E\{H_n\} \sim c \log n,$$
$$\text{where } c = \frac{2}{\log_2\left(1/\sum_j P_j^2\right)}. \quad (5)$$

Since the MPT tree in the global state can be regarded as a random Patricia tree with 16 forks ($P_j = 1/16$, $1 \le j \le 16$), according to formula (4), we can deduce the following:

$$E\{H_n\} \sim \log(n)/2. \quad (6)$$

When an intelligent contract executes a new transaction, i.e., inserting or modifying leaf nodes in an MPT tree, all hash values on the path will be updated because of the characteristics of the Merkle tree, that is, all nonleaf node values of tree height will be updated. And because the MPT node is stored in LevelDB, the database needs to be read and updated. Assuming that the average time of each node in

MPT is t (including the time of calculating Hash and accessing database), then when the transaction scale reaches r and then executes another transaction, it can be concluded that the average time required to modify MPT tree is

$$T_{mpt}(n) = t \log (n)/2. \tag{7}$$

Set the deployment intelligent contract consumption gas to G_{deploy} and the one-time transaction consumption gas to G_{tran}; then the total consumption gas when the transaction amount is n:

$$G(N) = G_{deploy} + \sum G_{tran}. \tag{8}$$

By analyzing the MPT tree of the underlying structure of Ethernet Square as a 16-fork Patricia tree, the relationship between transaction volume and tree height is determined. Since updating the transaction node will result in updating the log n node, the relationship between transaction volume and time is derived. Gas cost, additional data size, and EVM opcode have definite calculation formulas, so gas cost can also be derived from transaction volume.

When the data is dispersed into the newly created smart contract, the read time is positively correlated with the height of the global state tree. But with it, there is a lot of gas expenditure, which is used to create new smart contracts. Therefore, according to the actual system, the transaction execution time and gas consumption can be predicted from the transaction scale, and these two factors can be considered to choose different intelligent contract organization structures for the system.

3.3. Algorithm Verification. Any one of the three organizational architectures of intelligent contract is based on MPT tree, so it can be used for algorithm verification. This verification chooses a single intelligent contract. Considering the execution time and execution cost, the smart contract with code simplification is adopted, as shown in Code 1; this code uses the structure Dealing, links the structure with a mapping index, and then executes build Dealing, saving the Dealing information in the Single Contract Model's storage space, and the newly added Dealing storage will enter the Single Contract Model's storage Trie.

If the transaction volume is n, the successful execution time of the transaction is t (including calculating the sha30 hash function and the time of database access), and a data structure is changed in one transaction; the average time of the next transaction after n transactions that have been executed is

$$T(n) = t \log (\mathrm{an})/2\, a > 0. \tag{9}$$

```
Struct Dealing {
        adress userAddress;
        Uint256 date;
        Uint64 dealingNo;
        Uint64 publishingNo;
        Address publisherAddress;
        Uint64 price;
        State state;
        String newTime;
      }
Enum State {Created, Pending, Completed, Stopped}
```

Code 2: before manual optimization of specified structure.

```
'Struct Dealing {
        Uint64 dealingNo;
        Uint64 publishingNo;
        Uint64 price;
        Address publisherAddress;
        adress userAddress;
        State state;
        String newTime;
        Uint256 date;
    }
Enum State {Created, Pending, Completed, Stopped}
```

Code 3: Code after manual optimization of specified structure.

Each transaction changes the data structure $a = 1$, and the average time to access the database $t = 0.12\,\mathrm{ms}$ (since LevelDB random access dramatically degrades performance with the increase of data volume, t will continue to increase, which is only an average in the simple use test here). When $G_{deploy} = 119799$ and the control input data is always 1, then $G_{tran1} = 62319$ and $G_{tran} = 47319$ are measured. The next transaction after n transactions are executed:

$$T_{predict}(n) = 0.06 \log (n). \tag{10}$$

After the transaction volume is n, the consumption gas is

$$G_{predict}(n) = 47319n + 87480. \tag{11}$$

In this experiment, a dedicated Ethernet chain is built on the server, and 10 Ethernet nodes are deployed. Deploy the above simplified contract, which generates transaction structure data every time it is called. Execute transactions randomly in the system. The Ethernet node prints logs internally and records the consumption of transaction time, gas consumed by transactions, and transaction volume in the global intelligent contract account. When the transaction volume reaches millions, stop the above experiment, write data analysis scripts, and extract data such as processing log time. Under different transaction volumes, the offset value between the forecast time and the test transaction execution time is shown in Table 3.

By comparing the error between the predicted value and the test time, the blockchain method in this paper can have little correlation with the actual business occurrence time. It shows that the test time is safe and reasonable, which can represent the good performance of the system.

Because the prediction curve in this paper is calculated with a fixed t value, the t value in the real environment is

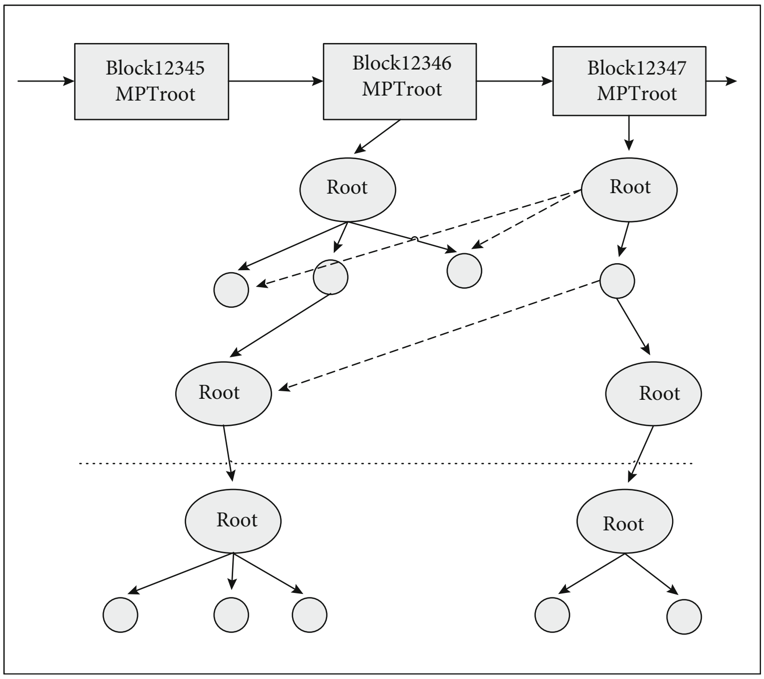

Figure 4: Organizational architecture of smart contract.

from small to large, which is mainly determined by the reading performance of LevelDB. Within 200000 data, the predicted value is significantly larger than the experimental data, because the actual time is smaller than the fixed value used in calculation. With the increase of trading volume, the distribution of experimental data on both sides of the curve is more obvious, and the prediction is effective. The overall offset value is less than 0.1 s, which is acceptable and proves the effectiveness of the algorithm. Once the smart contract structure is fixed and the input data is fixed, it can be seen from the above that the consumption of gas is fixed and can be completely calculated, so the algorithm is correct.

After the above verification, it can be seen that the prediction algorithm is effective. Then, according to the system transaction volume and the implementation of the specific system intelligent contract code, we can choose the time and gas cost and then choose different intelligent contract organization modes.

4. Performance Test

Gas is the consumption unit for deploying and executing smart contracts in Ethereum, which represents the amount of computation required to execute transaction operations. Therefore, the constraint of gas factor must be considered when designing related business process contracts. Gas mechanism can encourage computing nodes to participate in negotiation mechanism to create transactions, thus ensuring that Ethernet can continue to be decentralized without trust under increasingly complex intelligent contracts in Figure 2.

Gas is an important performance index in Ethernet Square system. Therefore, the next step will be to analyze and test from the perspective of gas consumption. First, the function code was specified for the gas optimization analysis, the detailed optimization method was elaborated, and then for the function for the Gas test, and the Gas trend graph and optimization percentage were drawn to verify whether the gas optimization method is effective. Finally, according to the optimization theory, the code of the whole intelligent contract is processed by these three methods in turn, and the optimization results are checked by comparing the gas test values before and after optimization in Figure 3.

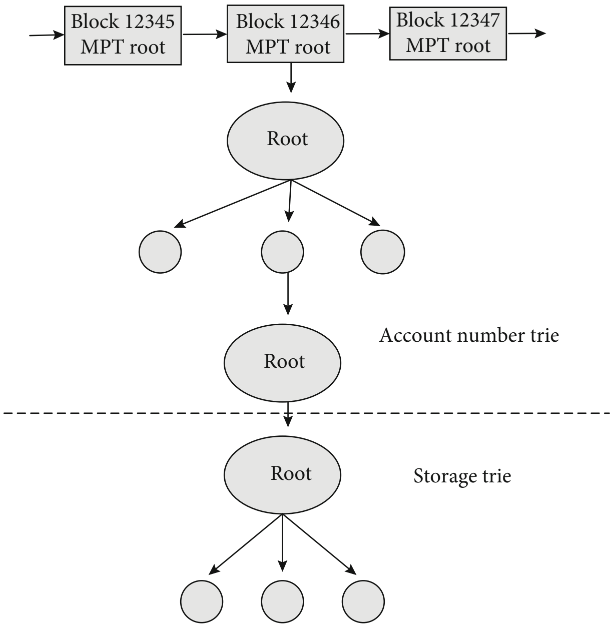

Figure 5: Account Trie and storage Trie.

Table 4: Formulating function gas consumption.

Gas consumption	Transaction cost	Execution cost	Date cost
Original distribution	84177	62905	21272
Change sequence variable	83576	61347	21274
Reducing high energy consumption directive	76862	56727	21272
Redundant code cut	72022	51750	20270

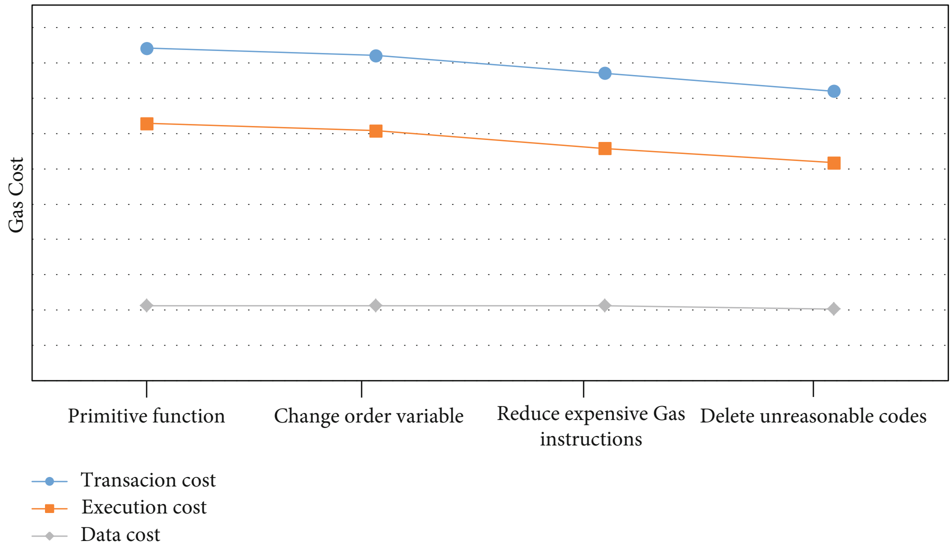

Figure 6: Line chart of gas consumption of function.

4.1. Gas Optimization Analysis. According to the gas optimization theory proposed in section 3, the intelligent contract code is optimized by reducing gas cost from three aspects: changing the order of variables, reducing the use of expensive EVM opcodes, and reducing redundancy and unreasonable design of intelligent contract codes. The following is a detailed explanation and example of manual optimization for specific functions.

4.1.1. Change the Order of the Quantities. The EVM virtual machine can only read sequentially and process variables in a group of 32 bytes. The declared variable type unit64 occupies 64 bits, that is, 8 bytes, unit256 occupies 256 bits, that is, 32 bytes, the variable type address occupies 160 bits, that is, 20 bytes, string is a variable length variable, that occupies different bytes according to the stored variable length, and state occupies 8 bits, that is, 1 byte according to the defined type of enumeration. Then, considering the unreasonable order of variables, try to sort them in a group of 32 bytes, and put the variables updated at the same time together to effectively reduce gas consumption. The manually optimized code is shown in Codes 2 and 3.

4.1.2. Reduce the Use of Expensive EVM Opcodes. Call the function personAllDealing(), tempNo as a temporary variable, get the total number of orders for the user, and write that number to the return value result. The global variables cntDealing and tempNo are counted in a self-growing way and are used in their functions. However, cntDealing has a better way, that is, after tempNo++ gets the final result, it directly assigns a value to cntDealing once. That is, cntDealing = tempNo is used instead of cntDealing for optimization purposes in Figures 4 and 5.

4.1.3. Optimize Smart Contract Code to Reduce Redundancy and Unreasonable Design Code. The function of tempNo is to obtain all the orders of the user. This operation can be calculated after the client obtains all the order information. There is no need to consume gas calculation in the intelligent contract, so this part of unreasonable design code is removed to reduce gas consumption.

As shown in Table 4 and Figure 6, it is the change trend of consuming gas by manually optimizing the specified function personAllDealingO in three ways in turn and initiating the same order data information query. It can be seen from the chart that the consumption of gas has dropped significantly. After three optimizations, the transaction cost of gas has dropped by 12.9%, so it can be proved that it is effective to optimize smart contracts from three aspects: changing sequence variables, reducing expensive gas instructions, and reducing unreasonable codes.

4.2. Gas Optimization Test. Functions in blockchain intelligent contract architecture consume gas with the selection of associated bytes or the distributed output of contracts. Therefore, by integrating the information contained in contracts into EVM function codes, the gas consumption of contract-related function codes is tested and improved.

Table 5: Gas expenditure for smart contract deployment.

Gas consumption	Transaction cost	Execution cost	Date cost
Original distribution	4873802	3670188	1203614
Variable sequence	4863592	3662582	1201010
Control modify variable operation	4860415	3662063	1198352
Reduce useless code	4473174	3357074	1116100

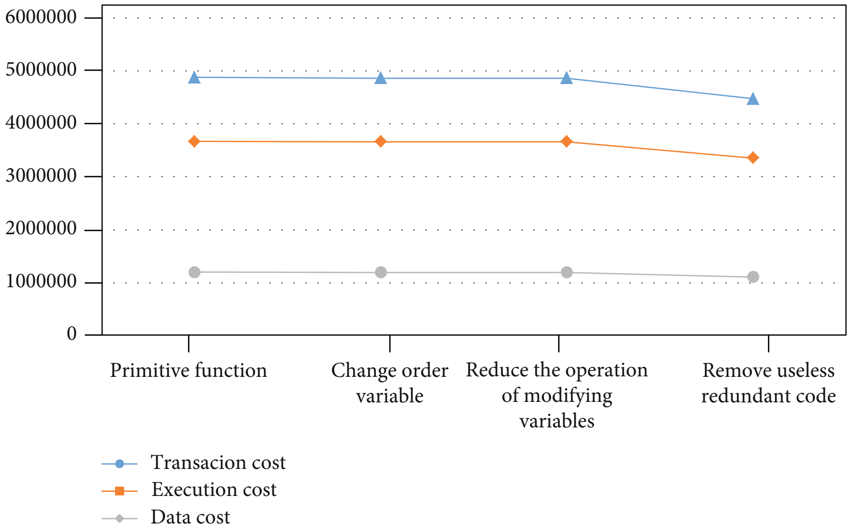

Figure 7: Gas consumption trend chart of smart contract deployment.

Table 5 and Figure 7 show the trend chart of gas cost for deploying smart contracts. It can be seen that the consumption of gas in millions has decreased. Because the smart contract has been written as concise and reasonable as possible, the cost of gas transaction has decreased by 8.22% after the final three methods, thus completing the gas optimization and testing of smart contract.

As can be seen in Figure 7, the execution cost can be reduced a lot through blockchain. Among all indicators, the execution cost of blockchain is lower than the conversion cost. The greater the change, the better the performance, and the lower the cost of the proposed method.

5. Conclusion

In this paper, the optimization factors of intelligent contract structure of smart city sharing system are studied. The structure of intelligent contract can be divided into two situations: the code structure of intelligent contract and the organizational structure of intelligent contract. The key factor affecting the performance of intelligent contract includes gas cost. Firstly, based on the composition of gas cost of transaction, the gas optimization theory is put forward, and the way to optimize intelligent contract code by changing the order of variables, reducing the use of expensive EVM opcodes, and reducing redundant codes is put forward, which provides a basis for the optimization of intelligent contract code in the final system. In addition, an algorithm for predicting transaction execution time and gas execution cost is proposed. According to the different transaction volume of specific systems and the implementation mode of intelligent contracts, the reference of intelligent contract organization structure is given, and the effectiveness of the algorithm is verified by experiments, which provides the implementation basis for intelligent contract organization structure and the basis for the implementation of smart city sharing system.

Data Availability

The experimental data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The author declared no conflicts of interest regarding this work.

References

[1] R. Zhang, S. Newman, M. Ortolani, and S. Silvestri, “A network tomography approach for traffic monitoring in smart cities,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 7, pp. 2268–2278, 2018.

[2] J. Su, “The research of data center construction for smart city,” *IOP Conference Series: Materials Science and Engineering*, vol. 490, pp. 1–6, 2019.

[3] M. A. Rahman, M. M. Rashid, M. S. Hossain, E. Hassanain, M. F. Alhamid, and M. Guizani, “Blockchain and IoT-based

cognitive edge framework for sharing economy services in a smart city," *IEEE Access*, vol. 7, pp. 18611–18621, 2019.

[4] P. K. Sharma, S. Y. Moon, and J. H. Park, "Block-VN: a distributed blockchain based vehicular network architecture in smart city," *Journal of Information Processing Systems*, vol. 13, no. 1, pp. 184–195, 2017.

[5] S. P. Kumar, R. Shailendra, and P. J. Hyuk, "DistArch-SCNet: blockchain-based distributed architecture with li-fi communication for a scalable smart city network," *IEEE Consumer Electronics Magazine*, vol. 7, no. 4, pp. 55–64, 2018.

[6] P. K. Sharma and J. H. Park, "Blockchain based hybrid network architecture for the smart city," *Future Generation Computer Systems*, vol. 86, pp. 650–655, 2018.

[7] S. Guo, Y. Qi, P. Yu, S. Shao, and X. Qiu, "Edge network resource synergy for mobile blockchain in smart city," in *2020 International wireless communications and Mobile computing (IWCMC)*, pp. 1272–1277, Limassol, Cyprus, 2020.

[8] C. Esposito, M. Ficco, and B. B. Gupta, "Blockchain-based authentication and authorization for smart city applications," *Information Processing & Management*, vol. 58, no. 2, p. 102468, 2021.

[9] S. Fan, L. Song, and C. Sang, *Research on Privacy Protection in IoT System Based on Blockchain*, vol. 11911, Springer, Cham, 2019.

[10] U. Saxena, J. S. Sodhi, and R. Tanwar, "Augmenting smart home network security using blockchain technology," *International Journal of Electronic Security and Digital Forensics*, vol. 12, no. 1, p. 99, 2020.

[11] K. B. Adolphe, "Privacy protection issues in blockchain technology," *International Journal of Computer Science and Information Security*, vol. 17, pp. 119–123, 2020.

[12] J. Liu, M. Xie, S. Chen, C. Ma, and Q. Gong, "An improved DPoS consensus mechanism in blockchain based on PLTS for the smart autonomous multi-robot system," *Information Sciences*, vol. 575, no. 12, pp. 528–541, 2021.

[13] N. Kawaguchi, "Application of blockchain to supply chain: flexible blockchain technology," *Procedia Computer Science*, vol. 164, pp. 143–148, 2019.

[14] M. H. Joo, Y. Nishikawa, and K. Dandapani, "Cryptocurrency, a successful application of blockchain technology," *Managerial Finance Managerial Finance*, vol. 46, no. 6, pp. 715–733, 2020.

[15] G. Lin and J. Tao, "A privacy protection method of lightweight nodes in blockchain," *Security and Communication Networks*, vol. 2021, no. 10, 17 pages, 2021.