symmetry

MDPI

Article

Research on Blockchain Architecture and Operating Principles Based on H-DAG

Jie Huang [1,2,*], Changsheng Liu [2] and Joseph Harding [3]

[1] Hunan Provincial Engineering Research Center for Missile Maintenance, Changsha 410024, China
[2] Department of Aviation Electronic Equipment Maintenance, Changsha Aeronautical Vocational and Technical College, Changsha 410024, China; lcs888_2002@163.com
[3] Division of Information Technology, Southern University, Baton Rouge, LA 70813, USA; joseph.harding@sus.edu
* Correspondence: huangjie918@163.com

Abstract: A hybrid blockchain structure (hybrid directed acyclic graph, or H-DAG) is proposed in this article to solve the existing problem of blockchain architectures using symmetric key encryption technology by combining the characteristics of single-chain blockchains and DAG distributed ledgers. By improving the block and transaction structures and optimizing the consensus mechanism, the H-DAG confirmed transaction orders while maintaining the high-throughput characteristics of a DAG, thus solving the transaction order dependence problem. We introduced a lightweight PoW mechanism to the H-DAG to improve the anti-fork ability of the blockchain. An incentive mechanism was adopted in our model to compel honest nodes to be more enthusiastic about participating in, maintaining, and enhancing the security of a given network. The blockchain states achieved strong levels of consistency, and their transaction confirmation times were predictable. We evaluated the performance of the H-DAG by comparing and analyzing multiple experiments, and we modeled a forking attack strategy, verifying the resistance of the H-DAG to this attack strategy. The experimental results demonstrated that the order of transactions in the H-DAG was globally consistent, and the confirmation time of transactions was predictable. The H-DAG improved the anti-fork ability and enhanced the security of the blockchain to ensure a degree of decentralization of the blockchain system. Therefore, the system throughput was enhanced by improving the block structure using symmetric key technology.

Keywords: blockchain; cryptographic technology; directed acyclic graph; throughput

Citation: Huang, J.; Liu, C.; Harding, J. Research on Blockchain Architecture and Operating Principles Based on H-DAG. *Symmetry* **2023**, *15*, 1361. https://doi.org/10.3390/sym15071361

Academic Editors: Achyut Shankar and José Carlos R. Alcantud

Received: 12 May 2023
Revised: 20 June 2023
Accepted: 20 June 2023
Published: 4 July 2023

1. Introduction

In this research, we employed cryptography technology to propose a hybrid blockchain structure; namely, a hybrid-DAG (H-DAG) blockchain. This was based on the characteristics of single-chain blockchain architecture and DAG distributed ledgers. Sa and Zhang proposed a scheme combining ABE and blockchains to hide the access strategy of their model [1,2]. These two schemes prevented the leakage of data relevant to the privacy of users and achieved secure access control using blockchain technology to realize the process of data storage and sharing without the involvement of a third party. Pournaghi et al. used double-chain architecture to improve the efficiency of remote sharing and treatment while supporting the protection of user privacy [3,4]. Gao et al. linked a small part of critical data, matching data storage, data sharing, and key information with two blockchains to reduce the load pressure on the blockchain. The two blockchains played a role in reducing the access control pressure of the single-chain system, thus improving its efficiency [5]. Research on multi-chain, cross-chain, side-chain, and fragmentation technology has improved blockchain scalability [6–9], but all these solutions were on the blockchain. Different schemas of blockchain architectures have significant potential to break through the current

blockchain performance bottleneck, providing new directions to improve the consensus speed and throughput of blockchain systems.

In this paper, we improved a currently used blockchain structure and consensus mechanism to propose new options. We verified the performance and reliability of the system through theoretical proof and experiments to achieve a blockchain system architecture with superior throughput, security, and scalability.

The reference relationship between units in a DAG distributed ledger is used in a blockchain transaction to improve the concurrency and throughput of the blockchain. This simultaneously builds the dependency relationship between transactions [10] to avoid the problem of transaction order dependency. Regarding the consensus mechanism of a blockchain, a proper accounting competition is performed by means of a computing power competition. Thus, all the nodes of an entire network can quickly reach a consensus to ensure decentralization. The confirmation time of each transaction can be expected to improve the reliability, robustness, and anti-attack ability of the blockchain system, achieving the global order of all trades [11].

In terms of the blockchain structure [12], we combined the chain structure of a traditional blockchain with a DAG structure, which solved the security problems caused by the transaction decisions related to conventional blockchain order miners while retaining high-throughput characteristics [13]. A single-chain blockchain architecture was combined with a DAG distributed ledger structure to improve the system throughput. By improving the PoW (proof of work) consensus mechanism and introducing a lightweight PoW mechanism for transactions, the anti-fork ability of the blockchain was enhanced [14]. Thus, the hybrid blockchain system architecture based on a DAG proposed in this paper achieved high throughput, security, and scalability.

2. Hybrid Blockchain System Architecture

2.1. Blockchain Network Structure

The H-DAG network structure of the hybrid blockchain based on a DAG is shown in Figure 1. The nodes in a network are divided into full nodes and light nodes. Full nodes store all blockchain information, collect all transactions on the web, and restrict blocks from participating in mining. They are also called miner nodes [15]. Light nodes send transactions in blockchain networks and include mobile phones, sensors, and other terminal devices. These devices often do not have high computing power and storage capacity and cannot save all the information of a blockchain to participate in mining [16]. Therefore, light nodes must interact with whole nodes to participate in a blockchain network.

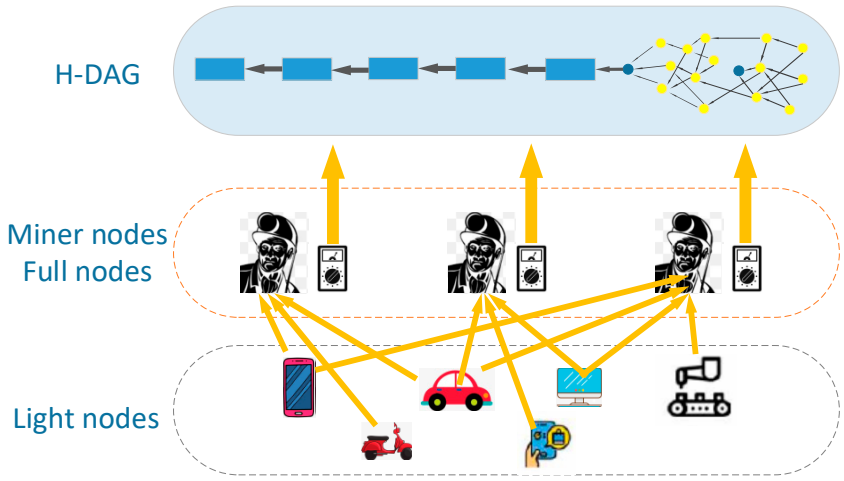

Figure 1. H-DAG network structure.

2.2. Blockchain System Framework

This framework is based on the open-source framework of Ethereum and modifies its trading pool, trading structure, block structure, and consensus mechanism, as well as other parts. The trading structure adopts a DAG structure based on the reference relationship. The consensus algorithm modifies the PoW of Ethereum. The overall architecture of an H-DAG is shown in Figure 2, which is divided into application, data, consensus, and network layers [17].

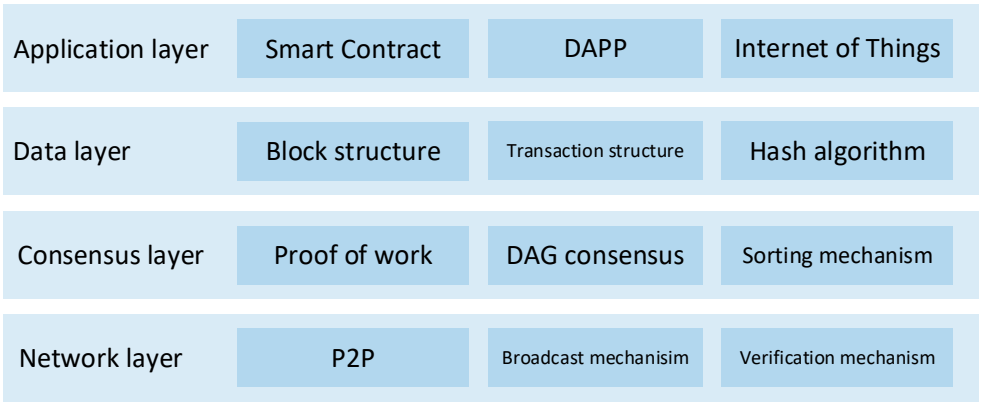

Figure 2. H-DAG architecture.

The application layer provides services for H-DAG application scenarios. Users apply cryptography to generate accounts. External accounts can request smart contracts by sending transactions to contract accounts. Functions such as DAPP and the Internet of Things are implemented based on smart contracts. The global order of transactions through the H-DAG structure and consensus mechanism can ensure that all transactions correctly execute intelligent contracts, solving the problem of transaction order dependence and significantly improving blockchain security. Therefore, IoT scenarios with high-throughput and security requirements are more suitable, such as vehicle networks and the Industrial Internet [18].

With the data layer, the data storage of an H-DAG adopts the block storage form of a single-chain blockchain. The block is composed of the block header and transaction data. The block header contains the basic information of the league and is used to verify the validity of the block [19]. All transaction information in the current block is recorded in the transaction data. In the process of consensus, only the hash value of the transaction is required to be written into the league [20]. Miners construct an index according to the hash value when verifying the block to obtain the complete transaction information. This improves bandwidth utilization, thereby improving the system throughput [21].

The consensus layer contains the PoW for the block consensus as well as the DAG consensus mechanism for the trading consensus. Similar to the consensus process of Bitcoin and Ethereum, the block consensus adds the computing power of a lightweight PoW in each transaction to improve the security of the blockchain. In the process of the transaction consensus, the entire node accepts and verifies the transaction [22], and then saves it in the local DAG and simultaneously broadcasts the transaction. In the process of complete node verification, if no transaction referenced by an exchange is found, it indicates a transaction in the current network that is not locally stored. It is then required to request other nodes to synchronize the transaction information. This mechanism also ensures that the structure of a DAG composed of transactions in all full-node stores is consistent [23].

With the network layer, a P2P network is formed among all nodes. All nodes are equal and jointly participate in and maintain the blockchain system [24]. Light nodes in the network are primarily devices with low computing power and storage capacity, such as mobile phones and sensors [25,26], which mainly participate in the network in the form of sending transactions. Thus, it is necessary to rely on all nodes to provide partial information to participate in the network [27].

3. Improved Consensus Mechanism Based on a PoW

3.1. Consensus Mechanism Process

In a single-chain blockchain, miners first select transactions from the mining pool in the process of block packaging. To maximize benefits, miners choose transactions according to their mining strategies for the package, usually prioritizing transactions with high commissions. This results in uncertainty when dealings with low commissions are packaged into the block. As the mining strategy of each miner is different, the trades that are packaged in each block are unpredictable. In the DAG represented by IOTA and Byteball, transactions are required to be verified and confirmed by the reference of subsequent transactions. There is no clear strategy for quoting transactions in their consensus mechanism. Therefore, the time when each transaction is mentioned cannot be determined.

To improve the scalability of the system and ensure the randomness of selecting miner nodes, we improved the consensus mechanism PoW, which is the most widely used mechanism in the current public chain. The H-DAG blockchain is divided into two parts: the block of packaged transactions and the DAG connected by the Genesis node. The sender selects the appropriate Genesis node and secures the transaction in the subsequent DAG. Miners package the DAGs of transactions into blocks, which are then broadcast to the blockchain network, where other nodes verify them and reach a final consensus.

The process of nodes joining a blockchain network for the first time is shown in Figure 3. The nodes first request the neighboring nodes in the network to synchronize the blockchain data. After verifying and synchronizing all block data, the nodes compete for computing power and participate in mining as full nodes. The part of the DAG connected by the current Genesis node is packaged, and the Merkle tree is calculated according to the transactions in the DAG. The Merkle root, block hash value, difficulty value, timestamp, and version information are then packaged into the block header. The hash value of the current block is then calculated. If the hash value of the block exceeds the difficulty value, the random number must be changed and recalculated until the hash value is less than the difficulty value and a legal block is found. This process is shown in Figure 4. In the competition for computing power, the whole node is required to monitor the information in the network, accept and verify the transactions on the web, and then broadcast the transactions.

When the whole node receives the new block broadcast by other nodes, it verifies it. When the new block is legitimate, that round of computing power competition is abandoned. The latest block is recorded on the local blockchain and broadcast, and then the next round of computing power competition begins. The validation process of a block is shown in Figure 5. First, the block header information is verified, and then the local complete transaction information is added to the league through the transaction hash value contained in the Merkle tree. If there is a transaction whose reference transaction is empty during the transaction search, the adjacent node is requested to synchronize the transaction information.

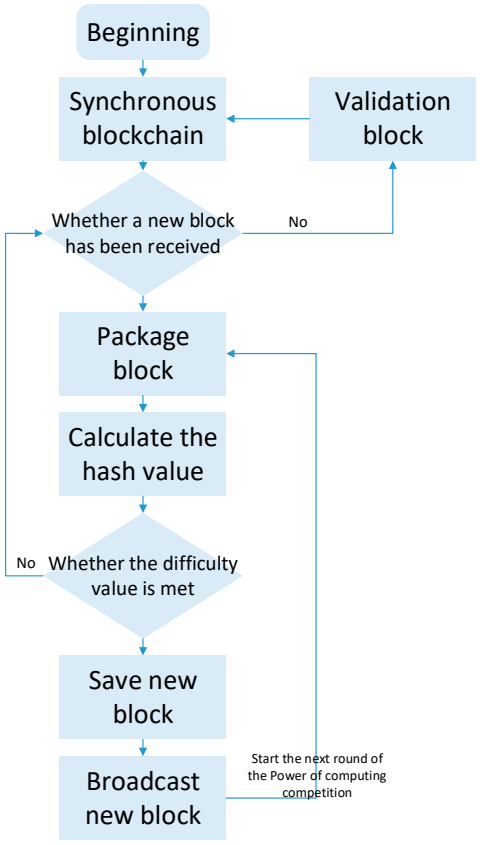

Figure 3. Process for adding a node to a network for the first time.

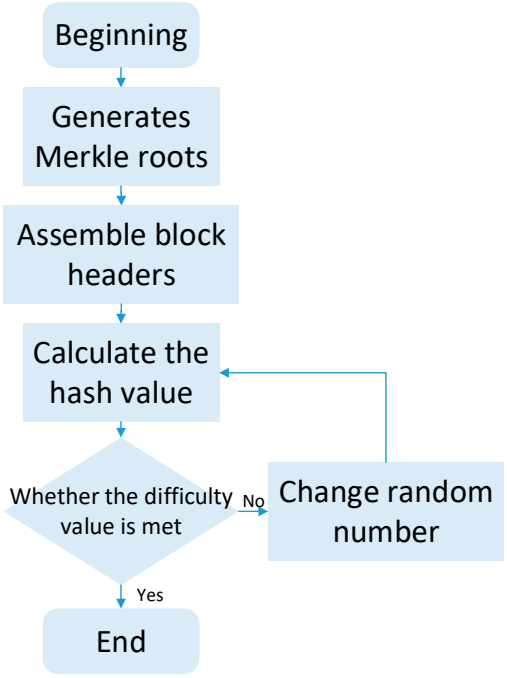

Figure 4. PoW workflow.

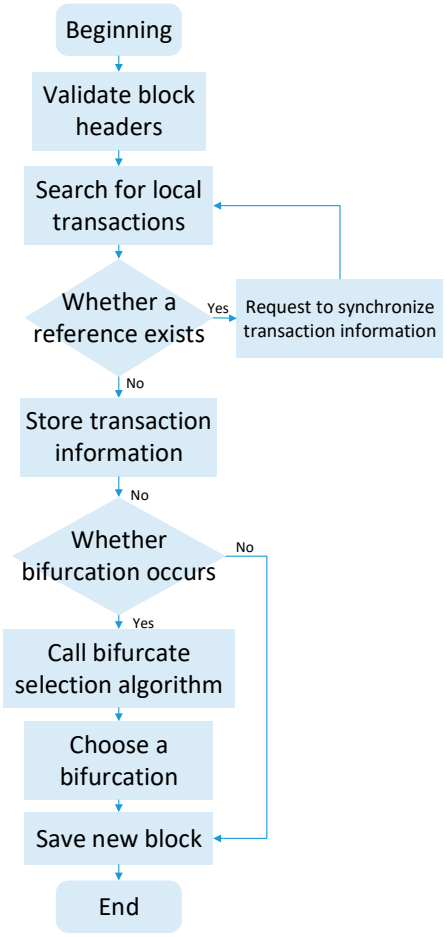

Figure 5. Block validation process.

When a miner receives two blocks of the same height at the same time or a synchronous blockchain, the difficulty of the value of all partnerships in each fork and the synthesis of the difficulty value of the block containing the transactions are counted. The chain with the most significant difficulty value is obtained as the main chain. The process is shown in Algorithm 1.

Algorithm 1. Bifurcation selection algorithm.

Input: Conflict block B_1; B_2
1. Initialize the difficulty value D_1; D_2
2. While B_1's parent block is Different from B_2:
3. D_1 + = B_1 Block difficulty value + B_1 Transaction total difficulty value
4. D_2 + = B_2 Block difficulty value + B_2 Transaction total difficulty value
5. B_1 = B_1's parent block; B_2 = B_2's parent block
6. End While
7. If $D_1 > D_2$:
8. return: Bifurcation of Block B_1
9. Else
10. return: Bifurcation of Block B_2

3.2. Trading Weight Mechanism

Every time a whole node discovers a legal block and it is agreed upon by the entire network, it obtains a block reward. The block reward includes the mintage reward for

the discovery of a new partnership as well as the commission for all transactions with the league block. The commission for a single transaction is shown in Formula (1).

$$\text{Transaction renewal fee} = \text{transaction fee} - min(log_{\alpha}^{w} * k, maxcharge) \quad (1)$$

where *maxcharge* is the maximum reduction in commission charges, k is the adjustment coefficient, and the transaction weight w is the sum of the number of directly and indirectly referenced transactions. The calculation method is shown in Algorithm 2. The commission function is similar to a margin. The probability of being quoted increases after quoting the correct trade, which reduces the commission of the work. Choosing an incorrect business does not reduce, or only slightly reduces, the commission. Through this incentive mechanism, honest nodes are guided to reference unreferenced transactions when sending transactions to ensure the stable extension of the DAG structure. Adding a lightweight PoW mechanism for transactions increases the cost of sending spam transactions while providing computing power to maintain H-DAG network security.

Algorithm 2. Weight calculation.

Input: Transaction t
1. Initializes the transaction set L = {}; L_s = {}; transaction weight w
2. Add t to set L
3. While $\boldsymbol{L} \neq 0$:
4. Add two reference transactions of transaction $L(1)$ to set L
5. Move $L\{1\}$ out of set L
6. If $\boldsymbol{L} \notin \boldsymbol{L_s}$:
7. Add L to Set L_s
8. W+=1
9. End While
10. Add all transaction sets L_c to set L that referenced transaction t
11. While $\boldsymbol{L} \neq 0$:
12. Add all transaction sets L_c to set L that referenced transaction *L(1)*
13. Move $L(1)$ out of set L
14. W+=1
15. End While
16. Return: W

3.3. Difficulty Value Adjustment Mechanism

The difficulty value reflects that the node is consumed by force. First, the hash value must finish to quantify the workforce, which is converted into a binary format. The hash value of several consecutive zeros of the front-end for n is the difficulty of the importance of 2 n tradings. The object is, in theory, to discover the first n zeros and attempt to calculate the maximum times required to force the standard for an evaluation. After receiving or packaging the new block, miners compete for the next block. The difficulty value of the next block should be included in the newly generated block, as shown in Formula (2).

$$Difficultyvalue = Currentdifficulty \times \frac{Time\ spent\ in\ the\ past\ n\ blocks}{n \times t} \quad (2)$$

where t is the block production time set by the blockchain. The average block production time of n blocks in the past is followed and the difficulty value is dynamically adjusted to ensure that the average block production time is stable at time t. When the current time is t1, the whole network agrees upon block 7, and miners compete for the permissions to block 8. The timestamp of Genesis node 10 contained in block 7 is then t1 + 2 ∗ t. In an ideal situation, the interval between the timestamp of Genesis node 9 and block 7 is t. After time t, the current time is t1 + t, and the mining competition time of block 8 is t. Ideally, the whole network should calculate and agree upon block 8. In this way, all nodes in the asynchronous system receive the majority of the transaction information. When the

network status is incorrect, the latest Genesis node can be selected to ensure that all nodes in the network have enough broadcast time to receive the transaction. Transactions sent by each node are packaged in fixed blocks. For example, transactions connected at Genesis node 8 are packaged in block 8; therefore, only block 8 is required to be read to identify if a transaction has been confirmed.

As mobile phones, sensors, and other lightweight devices often only have a tiny amount of computing power and storage capacity, they cannot save all the blockchain information and participate in mining. Therefore, this aspect of network participation is called a light node. The light nodes in a network are only required to hold information about the block sizes in all the blocks. When a transaction is sent, the DAG to which the current Genesis node is connected is first requested from the entire node. Two transactions are then referenced and a lightweight PoW is performed. The transaction is broadcast after calculating a hash that meets the difficulty value.

Thus, the DAG hybrid blockchain structure consensus mechanism PoW can be improved based on the data structure of an H-DAG. The process of nodes—from joining the network to participating in mining—has been explained, the consensus mechanism has been introduced in detail, and the specific methods of all nodes in a network regarding packaging blocks, verification blocks, and computing power competition have been described through algorithms and flow charts.

4. System Implementation

4.1. Experimental Environment

This experiment simulated a blockchain network by creating virtual machines. There were 20 virtual machines in total, of which 10 were virtual machines operating as full nodes and were responsible for packaging transactions and maintaining the operation of the blockchain network, and 10 were virtual machines operating as light nodes and were responsible for sending transactions. We ran the virtual machine software on three servers using VMware Workstation Pro 16.

4.2. System Deployment

This experiment used Geth, an Ethereum client written by Golang, to modify part of the code of Ethereum. We deployed the modified code to 20 virtual machines. We initialized all nodes using a configuration file and ran the client program. The startup parameters and functions of Geth are shown in Table 1.

Table 1. Geth startup parameter description.

Boot Option	Function
datadir	Specifies the data directory for the database and keystore key
networkid	Network ID
nod discover	The node discovery mechanism is disabled
allow-insecure-unlock	Allows an account to be unlocked
rpc	Enables the HTTP–RPC server
rpcapi	This is based on the API provided by the HTTP–RPC interface
ws	WS–RPC server is enabled
wsorigins	WebSocket Source allowed for the request

By using RPC to start the HTTP–RPC server, the default listening was on port 8545. Using WS to start the WS–RPC server, the default listening was on port 8546, which was used for web3.js to communicate with the Ethereum client through HTTP and the WebSocket protocol. After the node began the Ethereum client, its query was encoded and written into the same static-nodes.json file. This was then distributed to the directory specified by the data-dir of each node. Thus, the node could automatically connect to other nodes in the network when it started up and try to reconnect when it disconnected. This ensured network stability.

In terms of the block time, H-DAG uses a consensus mechanism based on an improvement of the PoW to maintain the average value of the block time at a fixed value by dynamically adjusting the difficulty value. We set different block times for the control experiments.

This experiment adopted Truffle's innovative contract framework to develop, debug, test, and deploy smart contracts. We used the interface provided by Truffle to simplify the development process and used the command line tool provided by Truffle to invoke the output results to directly debug the intelligent contracts.

4.3. Analysis of Experimental Results

Using a theoretical analysis and an experimental comparison, we analyzed the performance of the H-DAG from three aspects: the difficulty of the bifurcation attack, the transaction confirmation time, and the commission fee.

4.3.1. Fork Attack Difficulty

In our scenario, the attacker wanted to launch a forking attack, starting with the block that began the forking. The following two blocks used the Genesis node published by the previous two blocks; therefore, republishing the block required all the transactions connected to the Genesis block to be packaged. Starting from the third block, this creation node was not used by other miners because the block mined by the attacker defined the creation node. Thus, no transactions referenced this creation node. The attacker would have been required to create many transactions and pack them into the block to increase the total difficulty value of the block. Therefore, forking attacks need to be considered in two stages. In the first stage, the fork height was less than 3. The blockchain state at this stage is shown in Figure 6. The probability of success of the attacker is shown in Formula (3).

$$q_z \begin{cases} 1 & p \le q \\ (q/p)^z & p > q \end{cases} \tag{3}$$

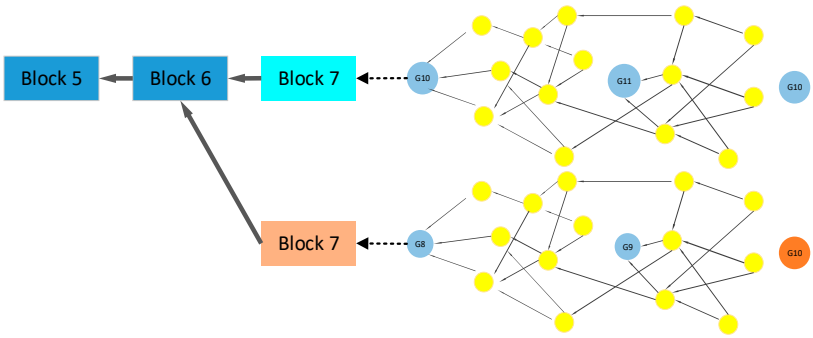

Figure 6. Bifurcation height less than 3.

p: The probability that the honest node finds the next block.

q: The probability that the attacker finds the next block.

q_z: The probability of the attacker catching up with the honest chain from z blocks behind ($z \le 2$).

When the fork height was greater than 3, the Genesis node at this time was generated by the block mined by the attacker. The attacker was required to simultaneously fight against the final force of other miners as well as all transaction senders in the network by connecting a large number of transactions behind its Genesis node to increase the block difficulty value. The blockchain state at this time is shown in Figure 7. The probability that the attacker successfully forked into the main chain is shown in Formula (4).

$$p_l = The\ computational\ power\ of\ all\ transaction\ senders$$
$$q_z \begin{cases} 1 & p + p_l \le q \\ \left(\frac{q}{p}\right)^2 * \left(\frac{q}{p+p_l}\right)^{z-2} & p + p_l > q \end{cases} \tag{4}$$

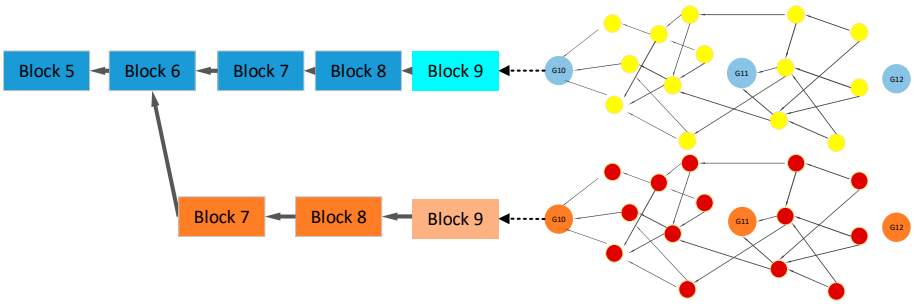

Figure 7. Bifurcation height greater than 3. (The red transactions are added by the attacker to increase the computing power).

In a payment scenario, the time it takes for a new transaction to progress from recording to the blockchain to determining that the transaction is not double-spent must be considered. For example, when a second block is published, the attacker has mined the first two blocks and has begun to compete with the honest node for the third block. In the third block, the probability of the attacker mining blocks is a Poisson process. T is an extended period; therefore, in T time, the honest node mines p blocks and the attacker mines q blocks. The time t used by the honest node to mine x blocks then satisfies $z = \left(\frac{p+p_l}{T}\right)t$. The time used by the attacker to mine x blocks satisfies $x = \left(\frac{p}{T}\right)t$, $x = \frac{q}{p+p_l}z$. Thus, the number of mines secretly mined by the attacker satisfies the Poisson distribution P(x) of $\lambda = \frac{q}{p+p_l}z$. The probability of attack success is then as follows:

$$Q_z = \sum_{x=2}^{+\infty} P(x) * q_z(z-x) = \sum_{x=2}^{+\infty} \frac{\lambda^x e^{-\lambda}}{x!} * \begin{cases} 1 & x > z \\ \left(\frac{q}{p}\right)^2 * \left(\frac{q}{p+p_l}\right)^{z-x-2} & x \le z \end{cases} \quad (5)$$

According to Formulas (3)–(5), when the computing power ratios between the attacker and the honest node are 3:7 and 1:1, the probability that the attacker can catch up with the main chain is calculated, as shown in Tables 2 and 3, respectively. The computing power of the light node is 0, which is the success rate of a traditional PoW computing mechanism, such as a Bitcoin blockchain fork attack. As seen in the above data, the H-DAG could operate normally and ensure safety, even when malicious nodes accounted for half of the computing power of all miner nodes. The main reason for this was that although all transaction senders and light nodes did not directly participate in mining, they still contributed computing power to maintain network security. Their computing power was transferred to honest miners. This was a significant boost to the bifurcation difficulty for malicious attackers.

Table 2. Results when the power ratio of the attacker to the honest miner node was 3:7.

Block Number/ Calculation Ratio	0	0.25	0.5	0.75
1	0.627759	0.469143	0.309697	0.152646
2	0.445716	0.257055	0.11503	0.0285795
3	0.324583	0.145217	0.0442274	0.00555341
4	0.239126	0.083148	0.0172527	0.00109569
5	0.177351	0.0479671	0.0067836	0.000217937
6	0.132110	0.0278042	0.00268046	4.35635×10^{-5}
7	0.0987124	0.00161692	0.00106263	8.73635×10^{-6}
8	0.0739243	0.00942538	0.00042226	1.75606×10^{-6}
9	0.0554571	0.00550406	0.000168097	3.53583×10^{-7}
10	0.0416604	0.00321865	6.70056×10^{-5}	7.12852×10^{-8}
20	0.0024803	1.56432×10^{-5}	7.04034×10^{-9}	8.16637×10^{-15}
30	0.000152235	7.82213×10^{-8}	7.59298×10^{-13}	8.78355×10^{-17}

Table 3. Results when the power ratio of the attacker to the honest miner node was 1:1.

Block Number/ Calculation Ratio	0	0.25	0.5	0.75
1	1	0.780476	0.522311	0.256961
2	1	0.662661	0.315443	0.0798236
3	1	0.572867	0.196114	0.0256902
4	1	0.499526	0.123511	0.00839166
5	1	0.437833	0.078356	0.00276311
6	1	0.385092	0.0499425	0.000914305
7	1	0.339557	0.0319347	0.000303542
8	1	0.299976	0.0204681	0.00010101
9	1	0.2654005	0.0131421	3.36716×10^{-5}
10	1	0.235101	0.0084502	1.12397×10^{-5}
20	1	0.0725322	0.000106114	2.00088×10^{-10}
30	1	0.023062	1.37197×10^{-6}	3.57754×10^{-15}

4.3.2. Transaction Confirmation Time

As two transactions are referenced before each transaction is sent, the transactions they refer to eventually point to the Genesis node. Therefore, after each Genesis node is packaged into a block and agreed upon by the whole network, all transactions connected to the node are packaged into the league. As the block output under the PoW mechanism is a random event, the block output time of each block cannot accurately be predicted. In our experiment, we used the 15 s block-out interval in Ethereum and the 10 min block-out interval in Bitcoin, respectively, to test the distribution of the block-out frequency under these two block-out times. The experimental results are shown in Figures 8 and 9.

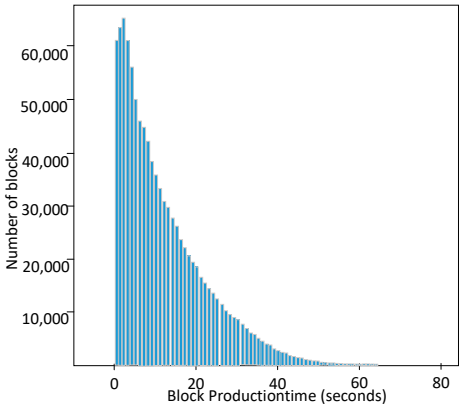

Figure 8. An actual block production time of 15 s.

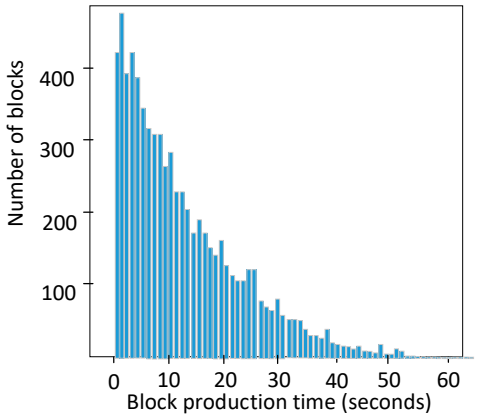

Figure 9. An actual block production time of 10 min.

We observed that under the PoW consensus mechanism, the block-out time presented an apparent exponential distribution. When this was combined with the conclusion of [20], we judged that the block-out time followed the exponential distribution of λ = (average block-out time). In our structure, sending a transaction required the creation node to be selected; thus, the transaction only appeared on a particular block. The time a transaction was recorded on the blockchain could be estimated based on the probability distribution of the block production. In an H-DAG, when a transaction is sent at any time, the miners of the network compete for the last block. After selecting a Genesis node and sending the marketing, they are required to wait for one or two blocks before they can query their transaction in the following block. At this time, the transaction is recorded on the chain. If the available closest creation node is selected, the transaction is recorded on the chain two blocks later. After the transaction is sent, two blocks are created in the blockchain, and the transaction is recorded in the second block. Due to the block time obeying an exponential distribution, the risk of multiple blocks should abide by the gamma distribution. This should confirm the relationship between the time interval and a piece of $f\ (x\ \beta, \alpha)$.

$$f(x, \beta, \alpha) = \frac{\beta^{\alpha}}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}, \quad x > 0 \tag{6}$$

where the random variable x is the time required to generate α blocks. The beta is λ in the exponential distribution formula, which we approximated as the average block generation time. We analyzed the block distribution when the block production time was set at 10 min and 15 s, as shown in Figures 10 and 11.

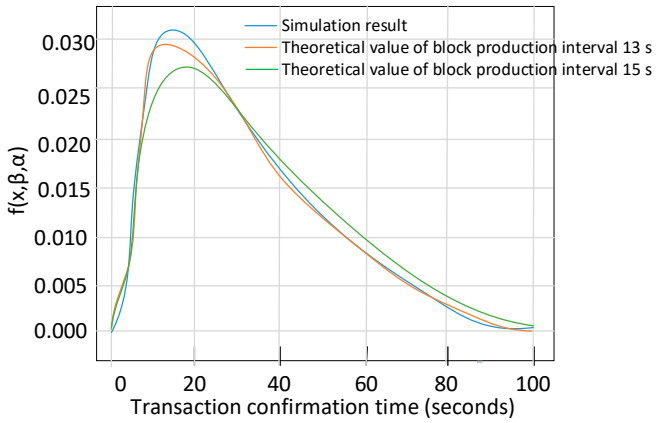

Figure 10. The theoretical value of a block production interval of 15 s.

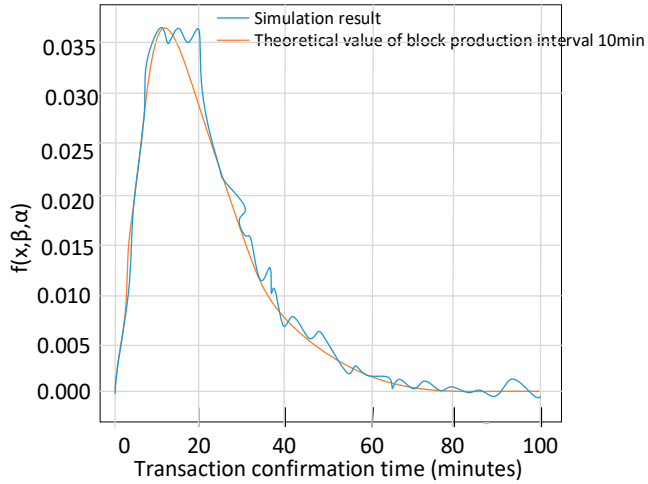

Figure 11. The theoretical value of a block production interval of 10 min.

When the block output interval was set to 15 s, the average value of the actual block output interval was approximately 13 s due to computational power fluctuations, network delays, etc. When the block output interval was set to 10 min, the average value of the block output interval was close to 10 min. Compared with Figures 10 and 11, the distribution function curve gradually smoothed as the number of blocks increased. The block distribution followed the gamma distribution and was consistent with the theoretical value. In our system, the confirmation time of a transaction can be predicted based on the gamma distribution. For example, when the block time was 15 s, we chose the current node and selected the nearest creation trading time distribution of the chain for f (x, 0.0667, 2); in 34 s, a 90% probability was confirmed.

4.3.3. Fee

Introducing fees can guide submitters to refer to appropriate transactions, reduce their costs by referring to proper transactions, and enable the DAG to expand and enhance blockchain security. In our experiment, the arrival of a trade to the blockchain was modeled using the standard method of the Poisson process. This set the transaction rate to a fixed value of λ. The relationship between the sum of obtained transaction references and the number of citations, as well as the number of transactions contained in the current block, was assessed.

Figure 12 shows the ratio of the total number of quoted transactions and quoted transactions of each exchange in 1000 blocks to the total number of all transactions contained in the block. With an increase in the number of transactions, the number of quoted transactions of each transaction accounted for an increasing proportion of the total number of transactions in the block. The number of unacknowledged transactions at the end of the DAG fluctuated around a constant. Thus, the appropriate trades were referenced by most exchanges while ensuring that the directed acyclic graph formed between transactions had extended.

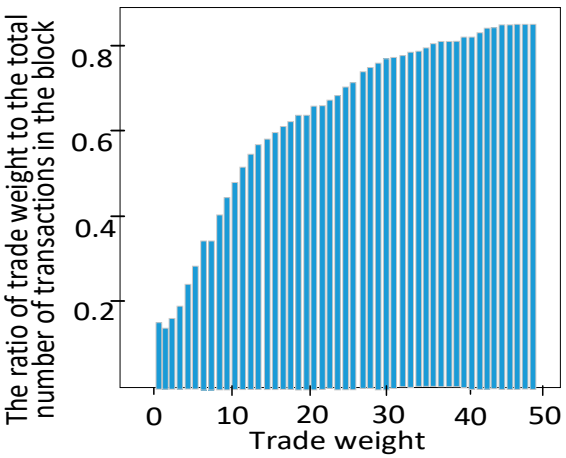

Figure 12. Trade weight.

5. Conclusions

By modifying the Ethereum client, a data structure based on an H-DAG and the overall architecture of the blockchain, encryption technology, and consensus mechanisms were realized. The performance and safety of the H-DAG were verified by theories and experiments. The results demonstrated that the H-DAG realized the confirmation of a transaction order while maintaining the high-throughput characteristics of a DAG distributed ledger. The anti-fork ability of the blockchain was improved, the transaction order dependence problem in blockchain smart contracts was solved, and the anti-double-blossom and anti-private-mining abilities were improved. In the network, the ratio of the computing power of the light node to all miners was 1:3. The computing power ratio of the attack node to the honest node was 3:7. The success rate of a forking attack after six blocks was

0.027%, significantly lower than the success rate of 0.13% for a Bitcoin blockchain attack (Table 2). This proved that the H-DAG dramatically improved security compared with a traditional blockchain.

Author Contributions: Conceptualization, J.H. (Jie Huang) and C.L.; methodology, J.H. (Jie Huang); software, J.H. (Jie Huang); validation, C.L.; formal analysis, C.L.; investigation, J.H. (Jie Huang); resources, J.H. (Joseph Harding); data curation, J.H. (Jie Huang); writing—original draft preparation, J.H. (Jie Huang) and C.L.; writing—review and editing, J.H. (Jie Huang), C.L. and J.H. (Joseph Harding); visualization, C.L.; funding acquisition, J.H. (Jie Huang). All authors have read and agreed to the published version of the manuscript.

Funding: This study was funded by the Natural Science Foundation of Hunan Province in 2022: "Design and application of cloud storage security architecture based on blockchain." (Research funder: Jie Huang; grant number 2022JJ60091.)

Institutional Review Board Statement: This article contains no studies with human participants or animals performed by the authors.

Data Availability Statement: The datasets used and analyzed during the current study are available from the corresponding author upon reasonable request.

Conflicts of Interest: We all declare that we have no conflicts of interest in this paper.

References

1. Pournaghi, S.M.; Bayat, M.; Farjami, Y. MedSBA: A novel and secure scheme to share medical data based on blockchain technology and attribute-based encryption. *J. Ambient. Intell. Humaniz. Comput.* **2020**, *11*, 4613–4641. [CrossRef]
2. Zhang, L.; Peng, M.; Wang, W.; Su, Y.; Cui, S.; Kim, S. Secure and efficient data storage and sharing scheme based on double blockchain. *Comput. Mater. Contin.* **2021**, *66*, 499–515.
3. Gao, J.; Yu, H.; Zhu, X.; Li, X. Blockchain-Based Digital Rights Management Scheme via Multiauthority Ciphertext-Policy Attribute-Based Encryption and Proxy Re-Encryption. *IEEE Syst. J.* **2021**, *15*, 5233–5244. [CrossRef]
4. Liu, S.; Wu, J.; Long, C. IoT Meets Blockchain: A Parallel Distributed Architecture for Data Storage and Sharing. In Proceedings of the 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom), and IEEE Smart Data (SmartData), Halifax, NS, Canada, 30 July–3 August 2018; pp. 1355–1360.
5. Qasse, I.A.; Abu Talib, M.; Nasir, Q. Inter Blockchain Communication: A Survey. In Proceedings of the ArabWIC 6th Annual International Conference Research Track, Rabat, Morocco, 7–9 March 2019; pp. 1–6.
6. Jiang, Y.; Wang, C.; Wang, Y.; Gao, L. A cross-chain solution to integrating multiple blockchains for IoT data management. *Sensors* **2019**, *19*, 2042. [CrossRef] [PubMed]
7. Li, W.; Feng, C.; Zhang, L.; Xu, H.; Cao, B.; Imran, M.A. A scalable multi-layer pbft consensus for blockchain. *IEEE Trans. Parallel Distrib. Syst.* **2020**, *32*, 1146–1160. [CrossRef]
8. Huang, C.; Wang, Z.; Chen, H.; Hu, Q.; Zhang, Q.; Wang, W.; Guan, X. Repchain: A reputation-based, secure, fast, and high-incentive blockchain system via sharding. *IEEE Internet Things J.* **2020**, *8*, 4291–4304. [CrossRef]
9. Qiao, R.; Luo, X.Y.; Zhu, S.F.; Liu, A.D.; Yan, X.Q.; Wang, Q.X. Dynamic autonomous cross consortium chain mechanism in e-Healthcare. *IEEE J. Biomed. Health Inform.* **2020**, *24*, 2157–2168. [CrossRef] [PubMed]
10. Berger, C.; Penzenstadler, B.; Drogehorn, O. On Using Blockchains for Safety-Critical Systems. In Proceedings of the 4th International Workshop on Software Engineering for Smart Cyber-Physical Systems (SEsCPS 2018), Gothenburg, Sweden, 27 May–3 June 2018; ACM: New York, NY, USA, 2018; pp. 30–36.
11. Wu, S.; Du, J. Electronic Medical Record Security Sharing Model Based on Blockchain. In Proceedings of the 3rd International Conference on Cryptography, Security and Privacy (ICCSP 2019), Kuala Lumpur, Malaysia, 19–21 January 2019; ACM: New York, NY, USA, 2019; pp. 13–17.
12. Xu, X.; Jianhua, H.; Hong, Z.; Ruicong, T. An Optimal Stability Matching Algorithm for DAG Blockchain Based on Matching Theory. *Chin. J. Electron.* **2021**, *30*, 367–377. [CrossRef]
13. Biswas, S.; Sharif, K.; Li, F.; Maharjan, S.; Mohanty, S.P.; Wang, Y. PoBT: A Lightweight Consensus Algorithm for Scalable IoT Business Blockchain. *IEEE Internet Things J.* **2020**, *7*, 2343–2355. [CrossRef]
14. Zhu, X.; Li, Y.; Fang, L.; Chen, P. An Improved Proof-of-Trust Consensus Algorithm for Credible Crowdsourcing Blockchain Services. *IEEE Access* **2020**, *8*, 102177–102187. [CrossRef]
15. Du, M.; Chen, Q.; Ma, X. MBFT: A new consensus algorithm for consortium blockchain. *IEEE Access* **2020**, *8*, 87665–87675. [CrossRef]

16. de Oliveira, M.T.; Reis, L.H.; Medeiros, D.S.; Carrano, R.C.; Olabarriaga, S.D.; Mattos, D.M. Blockchain Reputation-Based Consensus: A Scalable and Resilient Mechanism for Distributed Mistrusting Applications. *Comput. Netw.* **2020**, *179*, 107367. [CrossRef]
17. Bai, Q.; Zhou, X.; Wang, X.; Xu, Y.; Wang, X.; Kong, Q. A Deep Dive Into Blockchain Selfish Mining. In Proceedings of the ICC 2019—2019 IEEE International Conference on Communications (ICC), Shanghai, China, 20–24 May 2019; pp. 1–6.
18. Sukhwani, H.; Martínez, J.M.; Chang, X.; Trivedi, K.S.; Rindos, A. Performance modeling of pbft consensus process for permissioned blockchain network (hyper ledger fabric). In Proceedings of the 2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS), Hong Kong, China, 26–29 September 2017.
19. Lerner, S.D. Dag Coin: A Cryptocurrency without Blocks. 2015. Available online: https://bitslog.files.wordpress.com/2015/09/dagcoin-v41.pdf (accessed on 17 October 2017).
20. Baird, L. The Swirls Hash Graph Consensus Algorithm: Fair, Fast, Byzantine Fault Tolerance; Technical Report SWIRLDS-TR-2016-01, Swirlds Tech Reports; Dallas, TX, USA. 2016. Available online: https://eclass.upatras.gr/modules/document/file.php/CEID1175/Pool-of-Research-Papers%5B0%5D/31.HASH-GRAPH.pdf (accessed on 17 October 2017).
21. Le Mahieu, C. Nano: A Feeless Distributed Cryptocurrency Network. 2018. Available online: https://nano.org/en/whitepaper (accessed on 17 October 2017).
22. Chen, T.Y.; Huang, W.N.; Kuo, P.C.; Chung, H.; Chao, T.W. DEXON: A highly scalable, decentralized DAG-based consensus algorithm. *arXiv* **2018**, arXiv:1811.07525.
23. Miller, A. Gas Economics. Available online: https://github.com/LeastAuthority/ethereum-analyses/blob/master/GasEcon.md (accessed on 17 October 2017).
24. Alharby, M.; van Moorsel, A. The Impact of Profit Uncertainty on Miner Decisions in Blockchain Systems. *Electron. Notes Theor. Comput. Sci.* **2018**, *340*, 151–167. [CrossRef]
25. Atzei, N.; Bartoletti, M.; Cimoli, T. A Survey of Attacks on Ethereum Smart Contracts SoK. In Proceedings of the 6th International Conference on Principles of Security and Trust, Uppsala, Sweded, 22–29 April 2017; Springer: Berlin/Heidelberg, Germany; Volume 10204, pp. 164–186.
26. Li, X.; Jiang, P.; Chen, T.; Luo, X.; Wen, Q. A Survey on the Security of Blockchain Systems. *arXiv* **2018**, arXiv:1802.06993. [CrossRef]
27. Yang, R.; Chang, X.; Mišić, J.; Mišić, V.B. Assessing Blockchain Selfish Mining in an Imperfect Network: Honest and Selfish Miner Views. *Comput. Secur.* **2020**, *97*, 101956. [CrossRef]