future internet

MDPI

Review

Research on Progress of Blockchain Consensus Algorithm: A Review on Recent Progress of Blockchain Consensus Algorithms

Huanliang Xiong, Muxi Chen, Canghai Wu *, Yingding Zhao and Wenlong Yi

Software College, Jiangxi Agriculture University, Nanchang 330045, China; xionghuanliang@jxau.edu.cn (H.X.); chenmuxi@stu.jxau.edu.cn (M.C.); zhaoyingding@jxau.edu.cn (Y.Z.); yiwenlong@jxau.edu.cn (W.Y.)
* Correspondence: wucanghai@jxau.edu.cn

Abstract: Blockchain technology can solve the problem of trust in the open network in a decentralized way. It has broad application prospects and has attracted extensive attention from academia and industry. The blockchain consensus algorithm ensures that the nodes in the chain reach consensus in the complex network environment, and the node status ultimately remains the same. The consensus algorithm is one of the core technologies of blockchain and plays a pivotal role in the research of blockchain technology. This article gives the basic concepts of the blockchain, summarizes the key technologies of the blockchain, especially focuses on the research of the blockchain consensus algorithm, expounds the general principles of the consensus process, and classifies the mainstream consensus algorithms. Then, focusing on the improvement of consensus algorithm performance, it reviews the research progress of consensus algorithms in detail, analyzes and compares the characteristics, suitable scenarios, and possible shortcomings of different consensus algorithms, and based on this, studies the future development trend of consensus algorithms for reference.

Keywords: blockchain; consensus algorithm; distributed system; byzantine failures

check for updates

Citation: Xiong, H.; Chen, M.; Wu, C.; Zhao, Y.; Yi, W. Research on Progress of Blockchain Consensus Algorithm: A Review on Recent Progress of Blockchain Consensus Algorithms. *Future Internet* **2022**, *14*, 47. https://doi.org/10.3390/fi14020047

Academic Editors: Xu Wang, Bin Shi and Yili Fang

Received: 26 December 2021
Accepted: 22 January 2022
Published: 30 January 2022

1. Introduction

Blockchain technology is one of the most promising emerging technologies in the 21st century. It has significant advantages such as decentralization, non-tampering, non-forgery, and traceability [1]. It is very suitable for important anti-counterfeiting data storage, data security, and other realities scenes [2,3]. Blockchain technology can solve the security problems of data tampering and data loss existing in traditional centralized endorsement agencies, as well as the inefficiency of transaction processing [4–6], which can be found in finance [7–9], medical [10], Internet of Things [11], property rights protection [12], privacy protection [13] and other fields. Blockchain technology has produced great value, so it has been attracting attention since it appeared [14]. Blockchain technology originated from the paper "Bitcoin: A Peer-to-Peer Electronic Cash System" written by the scholar "Satoshi Nakamoto" in 2008 [15]. In 2009, the creation of the genesis block in Bitcoin marked the birth of the blockchain. The emergence of blockchain technology announced the birth of a new scientific field and an innovative distributed technology.

With the gradual application of blockchain technology, it is increasingly known and understood, and some of its inherent drawbacks were gradually exposed by people. For example, in terms of performance, the ability to handle high concurrent transactions is too low, the data throughput is too small, and there is insufficient capacity to defend from malicious nodes [16] of a chain, etc. In terms of energy consumption, there are problems such as waste of power resources caused by excessive computing power consumption. These problems have greatly hindered the development of blockchain technology. So, how to solve these thorny problems has become a hotspot in the research of blockchain. The consensus algorithm is one of the core technologies of blockchain, which is essential to the efficiency, security, and stability of blockchain data processing [17]. When the blockchain consensus algorithm promotes the consensus of blockchain nodes, it also needs to pay great attention to the security, scalability, resource consumption and performance efficiency of the algorithm, and it can constantly pursue high performance, low energy consumption, high security and scalability. In fact, the consensus algorithm has always been one of the key research directions of blockchain.

The research work of consensus algorithms precedes the concept of blockchain. It can be traced back to the research work carried out by Edmund Eisenberg et al. in 1959 [18]. That is, a special probability space represents a collection of event probabilities and includes sample space, event space, and probability function, a group of subjective individual probability distributions, and how to reach a consensus probability distribution. According to whether or not to consider the existence of malicious nodes in a blockchain network, consensus algorithms can be divided into non-Byzantine fault-tolerant algorithms and Byzantine fault-tolerant algorithms [19]. Most of the early consensus algorithms did not consider the possibility of malicious nodes in the system, so they are generally non-Byzantine algorithms, such as the VR (Viewstamped Replication) consensus algorithm proposed by Brian M. Oki et al. in 1988 [20], the Paxos algorithm proposed by Leslie Lamport in 1989 [21], the PBFT (Practical Byzantine Fault Tolerance) algorithm proposed by Barbara Liskov in 1999 [22], and so on. Non-Byzantine algorithms such as VR, Paxos, and PBFT have certain limitations, which are only applicable to blockchains with a limited number of nodes and relatively trustworthy nodes. In 2008, Satoshi Nakamoto proposed the concept of "Bitcoin". The Bitcoin system he designed uses the POW (Proof of Work) consensus algorithm proposed by Markus Jakobsson [23], which fully considers the possible existence of malicious nodes in a blockchain. Since then, Byzantine algorithms began to develop rapidly. A series of excellent consensus algorithms have been proposed, such as the PoS (Proof of Stake) consensus algorithm [24], DPoS (Delegated Proof of Stake) algorithm [25], Ripple consensus algorithm, PoSV (Proof of Stake Velocity) consensus algorithm [26], POA (Proof of Activity) algorithm [27], POB (Proof of Burn) algorithm [28], HoneyBadger BFT consensus algorithm [29], VBFT algorithm [30], Snowflake to Avalanche algorithm [31], and 99% Fault Tolerant Consensus algorithm [32].

This article first introduces a brief blockchain concept and the current status of the development of consensus algorithms, and then, Section 2 explains the working principle and architecture of the blockchain and gives a brief description of the typical division of the blockchain. Section 3 gives the concept of the consensus algorithm in blockchain and conducts an in-depth analysis of the typical consensus algorithm, including the working principle of the algorithm, consensus efficiency, security, scalability, energy consumption, etc.; then, it makes a detailed comparative analysis of them. Section 4 discusses the challenges faced by the blockchain consensus algorithm and predicts its future development trends.

2. Blockchain Overview

This section introduces the basic architecture of blockchain technology, explains the working principle of blockchain, and analyzes the core technology of blockchain.

2.1. Basic Architecture of Blockchain Technology

Blockchain often adopts a chained data structure, and its different data blocks are linked in ascending order of generation time. The first block of the chain is called the genesis block. The transaction information in the blockchain is the data structure of valuable data that expresses the value transfer through the signature operation. The block is the collection of transaction data. The data block consists of a block header and a block body. The block header generally includes the following:

1. *Version*: used to track software/protocol updates.
2. *PrevBlock Hash*: The current block records the hash value of the previous block. A unique hash value is generated based on the block information in an irreversible way through the hash algorithm. The hash value, which is short and fixed in length, is

used to uniquely mark the block. The hash value of the previous block is stored in the current block to ensure that the current block is linked to its previous block.

3. *Merkle Root*: Records the hash value of the Merkle tree root of this block.
4. *Timestamp*: The creation timestamp of the block is recorded. The timestamp ensures that the data in the blockchain can be stored in the order of the recorded generation time of the block, so that the source of the data in the chain can be traced according to the time stamp of the block.
5. *Difficulty Target*: The difficulty coefficient to be solved for the current block.
6. *Nonce*: The value calculated by the node based on computing power; generally, *nonce* is less than the *target*.

The block body stores the transaction content and related information. Each transaction information is marked with a digital signature. The digital signature mechanism is used to ensure the security of the block data. The block body generally includes the following:

1. *Num TransactionsBytes*: Records the number of bytes occupied by the *num* transactions.
2. *Num Transactions*: Records the number of transactions in the block.
3. *Transactions*: Records multiple transaction data in the block.

All transaction information is processed in the structure of the Merkle tree [33] in the block body. The transaction information is stored in the leaf nodes of the Merkle tree [34]. The leaf nodes are paired by hash calculation and combined to generate a hash value until the root node of the Merkle tree is obtained. The transaction information on the tree node can be queried by the nodes of the entire network. The hash value of the Merkle tree root is sensitive to the node information of the entire network. Once any transaction information is maliciously tampered with, the hash value of the Merkle tree root will be changed [35]. The Merkle tree structure ensures the information security of the blockchain to a certain extent. The structure of the Merkle tree is shown in Figure 1.

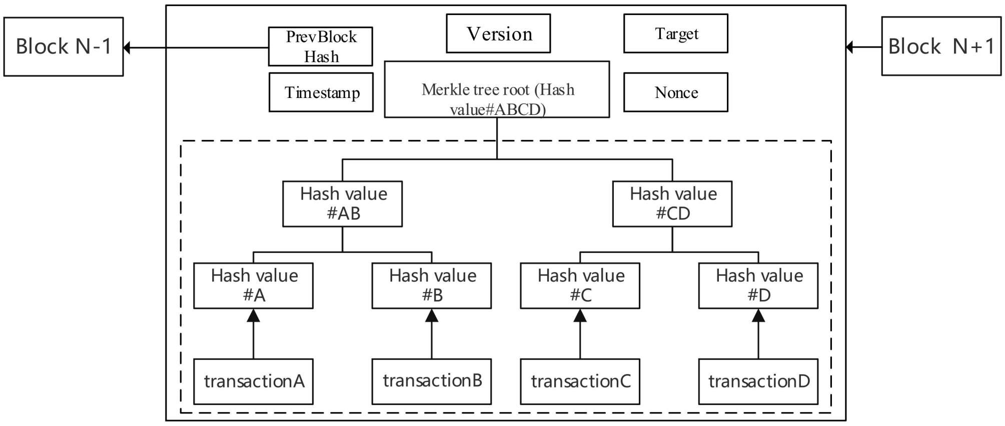

Figure 1. Merkle tree structure.

A node in a blockchain refers to a computing device that participates in a peer-to-peer network and provides computing power. A blockchain network is a set of nodes that implements a specific blockchain P2P protocol, and the entire network orchestrates and coordinates the operations of each node in the network in a fully federated, decentralized, and distributed manner. The basic task of a blockchain node is to verify whether the information in the block is correct and valid and to store the correct information. According to the function of the nodes in the blockchain network, they can generally be divided into three categories: a broadcast node that only sends out transaction information and accepts a small amount of blockchain information; a complete node that stores the entire blockchain information, whose function is to issue transactions, propagate transactions, and verify the consistency of data, etc.; and mining nodes with powerful computing power to create new blocks and publish and disseminate transactions, which are also known as "miners". These nodes support the stable and efficient operation of the blockchain system.

In the next section, we will explain the working principle of the blockchain according to its composition.

2.2. How Blockchain Works

Blockchain technology is a comprehensive innovative technology that combines a series of cutting-edge science and technology such as peer-to-peer network architecture [36], cryptography [37], mathematical knowledge, consensus algorithms, and smart contracts [38]. The blockchain uses a peer-to-peer network to realize the interconnection of all nodes. Each node in the network has an equal status and can provide various services for the blockchain system. Blockchain can essentially be understood as a distributed ledger, that is, a set of blocks with a specific structure. Its data blocks are stored in the order of generation time, and cryptographic technology is used to encrypt and verify the data stored in blockchain, so that the data cannot be tampered with or forged. Both appending data block to the blockchain and updating the data in blockchain are achieved through consensus algorithms.

The working process of blockchain can generally be divided into three stages: block generation, consensus verification, and ledger maintenance.

(1) In the block generation stage, the nodes in the blockchain collect the transaction information broadcast in the network and compete for the accounting rights of the block based on computing power. The nodes with accounting rights can package transaction information into blocks and obtain rewards preset by the reward mechanism in the blockchain protocol. These rewards often have economic benefits and can motivate nodes to provide a steady stream of computing power for the blockchain network. (2) In the consensus verification stage, the accounting node broadcasts the packaged block to the blockchain network, The entire network nodes accept a large number of blocks and verifies the content of the blocks according to the consensus algorithm, checks the correctness of the block content, and records the result in the blockchain ledger. (3) In the ledger maintenance phase, the node can store the data verified in the consensus verification phase for a long time and perform retrospective verification of the data according to the timestamp and hash value in the block, so as to provide the upper-layer application with an access interface for ledger information query. The nodes in the blockchain continue to provide computing power for the blockchain network, making the blockchain a decentralized, open, stable, honest, and credible system.

2.3. Blockchain Structure Model and Classification

Although different blockchain systems have different implementations, most blockchain systems can be abstracted as the data layer, network layer, consensus layer, incentive layer, contract layer, and application layer [39], as shown in Figure 2.

The data layer of the blockchain includes data blocks, a linked list, Merkle trees, and other data structures. It uses technologies such as timestamps, hash functions, and cryptography. It is the firm foundation for a series of functions such as blockchain organization, management, and data storage. The network layer of the blockchain uses the peer-to-peer network mechanism to connect all nodes in the chain to support the realization of transactions between nodes, data transmission, data verification, and other functions. The network layer ensures the communication between the nodes in blockchain and the interconnection with the entire network. The consensus algorithm is the core technology of the blockchain, determining the nodes in the chain that have accounting rights, enabling blockchain nodes to quickly reach a consensus on transactions stored in a data block, ensuring the consistency and security of block data, and improving the computational efficiency of the blockchain. This article focuses on the research of blockchain consensus algorithms. The incentive layer of the blockchain is to integrate some rewards and punishments into the distribution mechanism of the blockchain system to incentivize the activity of the nodes in blockchain to provide services so that the blockchain network remains active, and each node of the blockchain not only abides by the rules but also continuously contributes effective computing power. The contract layer of the blockchain encapsulates some smart contracts and algorithms. From the perspective of a computer, a smart contract can be understood as a piece of computer program involving related commercial transactions and algorithms, and from the perspective of the public, a smart contract is a related agreement. Once the preset conditions of the contract are triggered, the smart contract will be automatically verified and executed. Smart contracts are some pre-compiled code programs that will be activated and automatically executed when the preset conditions are met. Smart contracts realize the intelligence and customization of transactions in blockchain, which lay the foundation for building the application layer in a blockchain system framework. The application layer integrates the underlying structure, script code, and smart contracts in blockchain. So, blockchain can be applied to various scenarios in the real world. Blockchain can confirm, measure, and store valuable information in reality, thereby realizing data anti-counterfeiting, anti-tampering, and traceable *targets*. At present, blockchain systems are widely used in popular fields such as finance, medicine, security, anti-counterfeiting, the Internet of Things, games [40], etc.

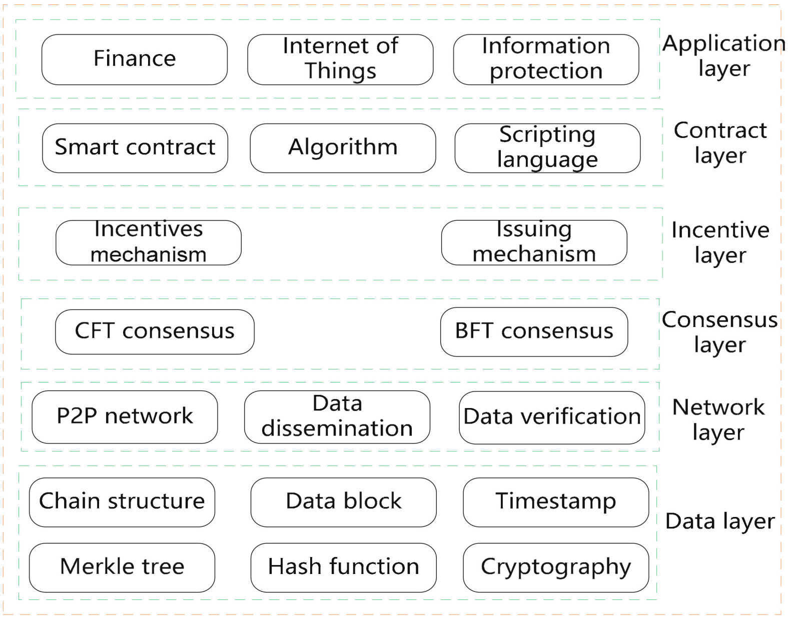

Figure 2. Blockchain structure model.

In terms of data access authority, blockchain can be divided into three modes: public chain [41], consortium chain [42], and private chain [43]. The main feature of the public chain is that nodes can freely join and exit the blockchain network without the need for review and approval by the licensing agency. Although the permissions of the public blockchain system are extremely open, the transaction processing speed of the system is slow, and the overall performance is low. Current popular applications such as Bitcoin and Ethereum all adopt the public chain model [44]. The consortium chain requires nodes to register their identity in advance and can join the blockchain network after review and approval by the central organization. The degree of openness of the consortium chain is lower than that of the public chain, and it is more suitable for some enterprises or companies that form consortia for the same purpose. The nodes in the consortium chain do not completely trust each other, and consensus algorithms are needed to reach agreement. The consortium chain presets the authority of the nodes, and only the nodes in the consortium chain can participate in accounting. The identity of each node on the consortium chain is known, and its data processing performance is high. The Hyperledger Fabric [45] developed by the LINUX Foundation adopts the consortium chain model. The private chain is controlled by a single internal node, which only provides services to the outside. The nodes in the chain usually belong to a single organization, and the access rights of the nodes are restricted to read and write. Among the three types of blockchains, the private chain has the lowest degree of decentralization, but its transaction processing speed is the fastest. R3 Corda [46] is a typical application that adopts the private chain model. The comparison of the blockchain characteristics based on three different models is shown in Table 1.

Table 1. Comparison of the blockchain characteristics based on three different models.

Type Feature	**Public Chain**	**Consortium Chain**	**Private Chain**
Participant	Any node	Consortium members only	Individual or company
Accounter	All participating nodes	Determined by negotiation among consortium members	Determined within the organization
Degree of centralization	Decentralized	Polycentric	Polycentric
Advantage	Tamper-proof, reliable	High efficiency, low cost	High efficiency, low cost
Application scenario	Digital currency	Payment, transaction	Audit, issue

3. Blockchain Consensus Algorithm

This section gives the definitions of some important concepts in the blockchain consensus algorithm and uses the literature review method to check the relevant literature and blockchain project websites. Then, some typical consensus algorithms are analyzed and discussed, including the principle, advantages, and disadvantages of the algorithm. Finally, we compare, analyze the various algorithms, and organize them into a table for a brief and intuitive explanation.

3.1. Consensus Algorithm Concept

In 1959, Edmund Eisenberg and David Gale carried out a subjective probability consensus study: that is, how many individuals with subjective consciousness in the same space can reach a consistent consensus probability distribution under certain conditions. In fact, this is the early consistency problem, and in some cases, it is also called the consensus problem, and this article uniformly expresses it as the consistency problem.

Definition 1. *(**Consistency problem** [47,48]): It refers to a series of operations performed by multiple nodes in blockchain network, and it is difficult for the data results obtained and saved by each node to reach consistency to a certain extent.*

The early consistency problem only focused on scenarios where the number of nodes was limited and trustworthy, but such scenarios had greater limitations and were not suitable for open Internet scenarios. The scholar Satoshi Nakamoto extended the consistency problem to the Internet environment with open scenes and massive nodes in the Bitcoin system he designed, and he proposed a distributed consensus mechanism that considers the possibility of malicious nodes in the environment, that is, a consensus algorithm that focuses on the Byzantine Generals problem, which has trans-era significance and scientific research value.

Definition 2. *(**Byzantine failures** [49]): It refers to the problem of how to reach a cooperative consensus among various nodes distributed in the network environment, with possible malicious nodes. In the blockchain, Byzantine error is a concrete manifestation of the Byzantine Generals problem. It refers to the fact that independent components in a blockchain network can do arbitrary or malicious behaviors, and these components may cooperate with other faulty components, thereby affecting the correct results of node calculations in the blockchain.*

The consensus algorithm is a means to ensure the data consistency of each node for a certain proposal. However, there are always differences in the computing power, storage, and other machine configurations of each node in a blockchain. Different consensus algorithms and consensus mechanisms have different capabilities to ensure that nodes obtain balanced accounting rights. If some nodes always easily obtain the right to accounting, it will undoubtedly cause other nodes to lose the activity to participate in the competition for accounting, thus making the entire blockchain network inactive.

Definition 3. *(**Activeness**): It refers to the proportion of nodes in blockchain network that actively participate in the competition for accounting rights of the data block, block information verification, as well as other work per unit time. Note that the number of blockchain network nodes is **n**, and the number of nodes actively participating in the work for the blockchain network within unit time **t** is **m**; then, the blockchain network activeness in this period is **a** = **m**/(**nt**).*

An excellent consensus algorithm can ensure the activeness of the blockchain network and a steady stream of effective computing power for the whole network, while a poorly designed consensus algorithm cannot guarantee that the nodes in chain actively provide effective computing power, causing the whole network to become easily paralyzed when attacked. According to whether to consider the Byzantine Generals problem, consensus algorithms can be divided into non-Byzantine fault-tolerant algorithms and Byzantine fault-tolerant algorithms.

3.2. Non-Byzantine Fault-Tolerant Consensus Algorithm

The concept of non-Byzantine error has been put forward very early. It generally refers to the occurrence of system failures such as machine downtime and node reporting errors in a distributed system, but there is no malicious node in the system that interferes with the distributed system. When malicious nodes in the system perform malicious activities such as tampering with data, non-Byzantine fault-tolerant algorithms cannot guarantee data security and system stability. Therefore, non-Byzantine fault-tolerant algorithms are difficult to apply to an open network with diverse nodes and low credibility. They are mainly used in internal environments with a certain degree of closure and relatively high credibility between nodes, such as consortium chains or private chains. The advantages of these algorithms are their high performance and strong ability to tolerate non-Byzantine errors.

3.2.1. VR Algorithm

The Viewstamped Replication algorithm is called the VR algorithm, which is a non-Byzantine fault-tolerant algorithm dedicated to solving the consistency problem in 1988. The working principle of the VR algorithm is to perform log replication operations between each node and to ensure the consistency between nodes in the blockchain network by replicating the state. The VR algorithm can tolerate no more than f nodes failing at the same time in a network with a total of $2f + 1$ nodes. During the working process of the VR algorithm, one node will be selected as primary, and the other nodes will be used as backup. The most important thing in the work process is to let all nodes execute commands in the same order; the primary node determines the order of commands, and other backup nodes accept the order determined by the primary node. When the primary node fails, the VR algorithm will perform a process called view change, which selects a new primary node. The election of the new primary node requires voting. At least $f + 1$ nodes need to communicate and vote to determine the new primary node. The new primary node will perform the new consensus process and replace the old primary node. The VR algorithm stipulates that a node that is down or has an error in the network will be set to the recovering state after restarting. In this state, it will not participate in any voting. The node will generate a Recovery request with a unique number and send it to all working nodes in the network. All nodes can respond to the request. The primary node replies to the error node with the content of the replicated node. The requesting node needs to receive at least $f + 1$ responses, receive the information of the primary node, and then update its own log information to return to the normal state.

The VR algorithm comes with a main election phase, and it requires that only one leader exists in each consensus process, and the leader is responsible for operations such as sequencing. This greatly simplifies the difficulty of algorithm design and implementation, and it is an early consensus algorithm with relatively simple implementation.

3.2.2. Paxos Algorithm

The Paxos algorithm was proposed by Lamport, an expert in the field of distributed technology, in 1990. This algorithm can make a distributed network that may have network failures, machine downtime, message delay or loss, etc., quickly achieve data consistency, and ensure the security of the distributed network. The Paxos algorithm aims to solve the distributed consistency problem, so that multiple participating nodes in the distributed system can reach a consensus after multiple steps.

In the Paxos algorithm, node roles are divided into three categories: proposer, acceptor, and learner. A node does not just have a single role, but it can play multiple roles, and there is a certain transformation relationship. There may be one or more Proposers, which are responsible for making proposals. There must be multiple acceptors. They vote on the specified proposal, accept the proposal if they agree, and reject it if they disagree. The learner collects the proposals accepted by each acceptor and promotes the formation of the final proposal based on the principle of the minority obeying the majority. In practical applications, the Paxos algorithm can be roughly divided into two working phases: the preparation phase and the submission phase.

In the preparation stage, the proposer first sends multiple proposals, including the number and proposal value of each proposal, to each acceptor. The acceptor saves the various proposals sent by the proposer, compares them, and saves the proposal with the maximum proposal number; at the same time, the acceptor returns the maximum proposal value received to the proposer and discards proposals with a proposal number less than the maximum number after checking the original or received proposal numbers.

In the submission stage, after the proposer receives more than 1/2 of the total number of acceptors, the proposers will return the received proposal and send them to all other acceptors. In this process, the recipient continues the work in the preparation phase, compares the proposal numbers sent, saves the largest proposal number, and updates the proposal value, and the learner forwards it after obtaining the voting result. When more than half of the recipients in the system have the same proposal value, the consensus is reached, and the consensus result is that multiple nodes participate in the work and get the same proposal value. The Paxos algorithm flow is shown in Figure 3.

The Paxos algorithm was first applied to prove that the system is safe and efficient. The communication between nodes does not need to verify the identity and signature, and the design is rigorous and logical. Due to the multi-role participation message passing mechanism adopted by the Paxos algorithm, the algorithm is non-Byzantine fault-tolerant. Scenarios such as the failure of less than half of the acceptors or the failure of the proposer will not affect the operation of the algorithm. After the proposal number or the content of the proposal is confirmed, even if less than half of the proposers fail, the result will not be affected. However, the required accuracy and conditions for the Paxos algorithm implementation are also relatively harsh. It is only applicable to private chains and cannot be applied to environments with malicious nodes. Subsequently, based on the Paxos algorithm, researchers successively proposed improved Paxos-like algorithms such as Multi Paxos [50] and Cheap Paxos [51]. These algorithms have obvious improvements in computational efficiency and security. For example, the Multi-Paxos algorithm accelerates the consensus among nodes by strengthening the authority of the leader. The Cheap-Paxos algorithm proposes to replace the faulty node with a less expensive auxiliary node to ensure the safe operation of the blockchain. The Mencius algorithm [52] reduces the work delay and achieves load balancing through the master–slave rotation system of replicas. The

Generalized Paxos algorithm [53] achieves high concurrency and low latency by processing multiple proposals at the same time.

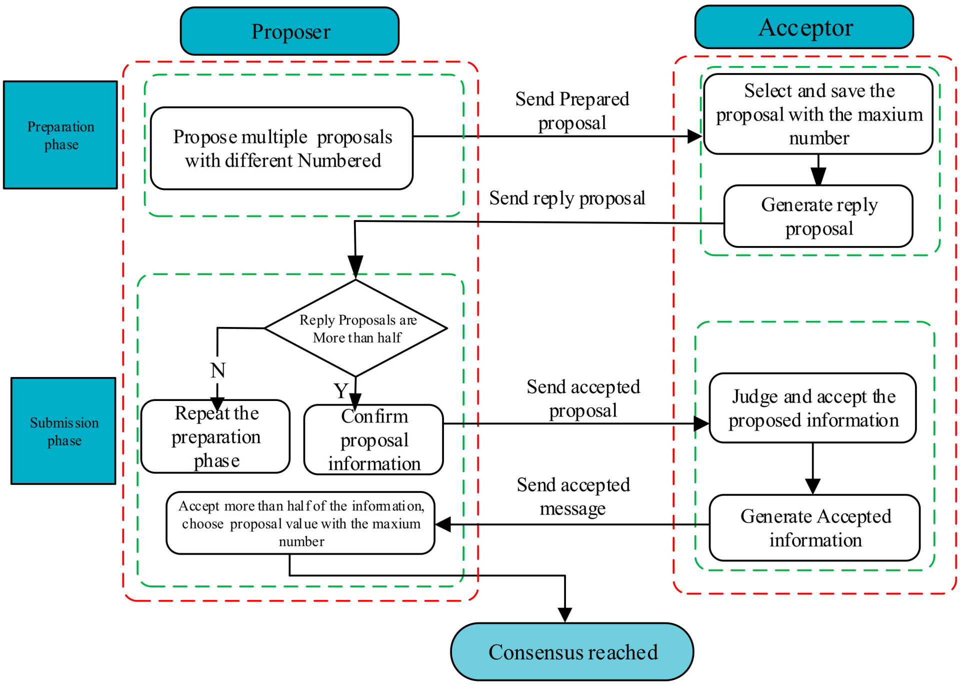
Proposer
Acceptor
Preparation phase
Submission phase
Propose multiple proposals with different Numbered
Send Prepared proposal
Select and save the proposal with the maxium number
Generate reply proposal
Send reply proposal
Reply Proposals are More than half
N
Y
Repeat the preparation phase
Confirm proposal information
Send accepted proposal
Judge and accept the proposed information
Generate Accepted information
Send accepted message
Accept more than half of the information, choose proposal value with the maxium number
Consensus reached

Figure 3. Paxos algorithm flow.

Although Paxos algorithms have high computational efficiency, the difficulty of algorithm implementation is relatively high, and the requirements for reaching consensus are relatively harsh. The Raft algorithm has the same advantages as the Paxos algorithm. At the same time, the algorithm is easier to realize and understand for readers.

3.2.3. Raft Algorithm

In 2013, Diego et al. [54] proposed the Raft algorithm. The Raft algorithm uses the log replication of each node in the system to achieve unification of transactions and consensus. The nodes in the system are also divided into three categories: namely, leaders, followers, and candidates. The three types of identities can be transformed into each other. The leader in the algorithm is responsible for processing interactive information, log replication, log distribution, and other tasks. There is usually only one leader during each consensus process. Followers are similar to voters in voting activities, generally one-way passively accepting the log information sent by the leader. Candidates can be transformed from followers and have a probability of being elected as the new leader. There are two main working phases in the Raft algorithm: the leader election phase and the log synchronization phase.

At the beginning of the leader election phase, except for an initial leader node, other nodes are initialized as followers. The initial leader first sends logs one-way to all followers, and followers passively receive the log information sent by the leader. Each follower has a random timeout period. If the message sent by the leader is not received within this time, the follower will turn into a candidate and initiate a leader election request. Candidates transformed from followers will vote for themselves and send other followers to request voting applications. If the candidate receives more than half of the votes in the process, the identity is converted to leader. If the leader information from other nodes is received in the middle of the election, it indicates that other candidates pre-empted to become the leader, and the election failed. If a candidate does not receive more than half of the votes, it also means that the election has failed and needs to wait until the next election request. The process of Raft election is shown in Figure 4.

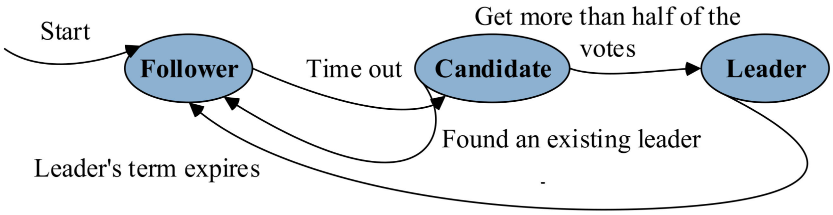

Figure 4. The process of Raft leader election.

When the leader is elected, it enters the next stage of log replication. In this stage, the leader will create new log entries into the leader's log, update the log content in real time, and send the new log content to the followers through log replication. After receiving the log information, followers update the locally stored log information and return a confirmation message. Once the leader receives more than half of the confirmation information, the log reaches a consensus. If there is a system failure, so that the leader cannot access most of the followers, the leader can only update the accessed followers' information, but other followers in the system cannot get log updates. In this case, the system will automatically execute a leader election, and the new leader will continue to work as a leader. When the system failure is repaired, the leader before the failure will become a follower. All updated information proposed by the old leader during the failure will be rolled back. At the same time, the old leader will receive information updates from the new leader, so that the consistency and security of the system can be guaranteed. The process of the Raft log replication phase is shown in Figure 5.

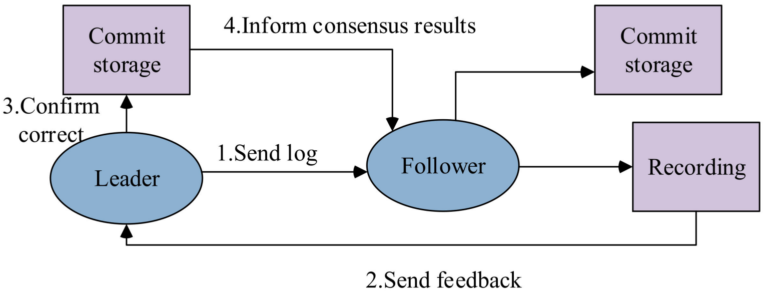

Figure 5. The process of Raft log replication.

The Raft algorithm is easier to understand and realize than the Paxos algorithm proposed earlier. The algorithm is more efficient compared to the Paxos algorithm. The Leader election and log replication mechanism are adopted by the Raft algorithm; the Raft algorithm relies on the strong leadership of the leader. A single leader sends log information to other nodes for management. Even if the leader fails due to a failure, a new leader will be selected immediately to ensure the safety and stability of the system.

Since the leader node can only support a limited number of node networks, and the fault tolerance is not very good, the Raft algorithm cannot support public chains with many nodes that are open to the outside world. The Raft algorithm is only applicable to private chains and cannot deal with malicious nodes. For this reason, researchers have tried to improve and perfect the Raft algorithm.

In 2019, Wang et al. [55] proposed the KRaft algorithm. Based on the Raft algorithm, the KRaft algorithm optimizes the process of election and consensus by establishing a new node relationship in the consensus protocol. It uses multiple candidate nodes to replicate logs in parallel, which improves scalability and transaction throughput. In 2017, Zhang et al. [56] proposed the Network-Assisted Raft algorithm using network assistance. Under the premise of ensuring correctness and scalability, by unloading the raft algorithm from the consensus layer to the programmable switch in the network layer, the working process of forwarding and response in the algorithm is optimized to improve the working performance. In 2021, Wang et al. [57] proposed the hhRaft algorithm on the basis of the Raft algorithm; a new monitor role is introduced to supervise candidate nodes and leader nodes, the leader node election phase and log replication phase are optimized, fault tolerance is enhanced, consensus delay is reduced, and transaction throughput is also improved. The hhRaft algorithm is superior to the Raft algorithm in anti-Byzantine failure ability, and it is more suitable for high real-time and highly adversarial environments.

3.3. Byzantine Fault-Tolerant Consensus Algorithm

The above-mentioned non-Byzantine fault-tolerant algorithms such as the Paxos algorithm and Raft algorithm have the advantages of higher efficiency and can be applied in distributed environments. However, there are a series of limitations, such as the inability to deal with malicious nodes and it is difficult to apply to the open Internet environment [58]. To this end, researchers have designed a series of Byzantine fault-tolerant consensus algorithms. The Byzantine fault-tolerant consensus algorithm can solve any type of error in the distributed system to a certain extent and ensure the security and stability of the distributed system. Byzantine fault-tolerant consensus algorithms can cope with the presence of malicious nodes in the environment and ensure the security and activity of the system [59].

3.3.1. PBFT Algorithm

The PBFT algorithm is a practical Byzantine fault-tolerance algorithm, which was first proposed by Barbara Liskov et al. in 1999. Inspired by the solution of the Byzantine general problem based on the oral message and signature message proposed by Lamport in 1982 [49], the algorithm reduces the complexity of the Byzantine fault-tolerant algorithm from the exponential level to polynomial level, and it increases the fault-tolerant performance by one-third on the premise of ensuring the stability and security of the original algorithm, making the Byzantine fault-tolerant algorithm more suitable for practice. The PBFT algorithm focuses on solving the problem of transaction ordering in distributed nodes that may have Byzantine errors. Under the condition that the number of Byzantine nodes with Byzantine errors accounts for up to $1/3$ of the total, the state of non-Byzantine nodes in the system can be consistent.

The nodes in the PBFT algorithm include the master node and other nodes. The master node is selected by the system and is responsible for distributing the number and processing information to other nodes in blockchain as well as checking the validity of the information. The distributing number is used for the ordering of the request message and the processing information includes the current view, the number n of the current request, the summary of the message content, and the message content. After other nodes accept the information sent by the master node, they will verify the information. Suppose there are $3f + 1$ nodes in the working process of the PBFT algorithm; among $3f + 1$ nodes, there is 1 master node, $2f$ honest nodes, and f malicious nodes. The working phase of the PBFT algorithm can be divided into three phases: pre-preparation phase, preparation phase, and submission phase. In the pre-preparation phase, the master node sends the assigned sequence number and pre-preparation information to other nodes. After the other nodes receive the information sent by the master node and confirm, the preparation phase begins. In the preparation phase, each node sends preparation information to all nodes except itself. When each node receives $2f + 1$ pieces of information, all consensus nodes are regarded as ready. Then, it is in the confirmation phase. All other nodes will broadcast a confirmation message to the entire network. When each node receives $2f + 1$ confirmation messages, the confirmation work phase is completed, and the correct log information is returned; then, they will be recorded in the blockchain. The workflow of the PBFT algorithm is shown in Figure 6.

The PBFT algorithm greatly improves the efficiency of the Byzantine fault-tolerant algorithm by reducing the complexity of the algorithm. When the number of Byzantine nodes in the system is less than $1/3$ of the total, the PBFT consensus algorithm can operate correctly and ensure the reliability of the blockchain system, thus greatly improving the security performance. These advantages make the Byzantine fault-tolerant algorithm more suitable for practical application scenarios. However, the PBFT algorithm also has certain limitations. For example, because the complexity of communication is the square of the number of nodes, it is difficult to support large-scale network nodes, so its scalability is poor. It cannot be directly used in the public chain, which may include a large number of nodes and is unable to undertake large-scale communication between nodes. So, it is only applicable to consortium chains and private chains, and it is not suitable for public chains with public authority. The PBFT algorithm also has certain limitations in communication problems. If the number of nodes is too large, the performance will be limited. In the case of poor network conditions, there will also be problems such as high processing delay.

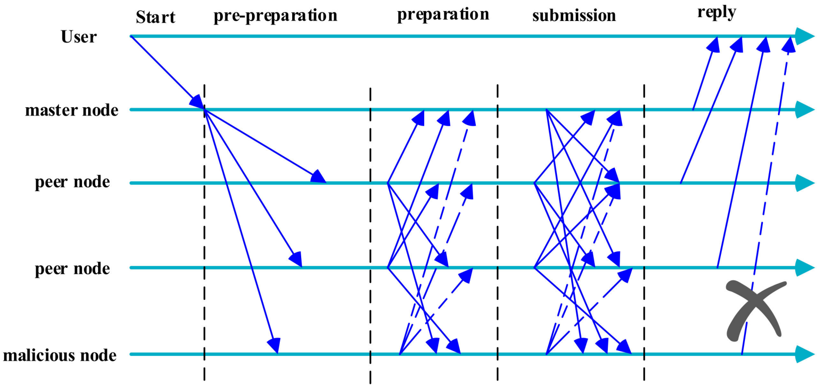

Figure 6. PBFT algorithm workflow.

Some optimization algorithms based on the PBFT algorithm have also been proposed. For example, in 2007, the Zyzzyva algorithm proposed by Kotla [60] simplifies the communication process of PBFT and reduces the difficulty of reaching consensus among blocks. In 2016, the HoneyBadgerBFT algorithm proposed by Miller [29] realizes the asynchronous Byzantine error and solves the time limit problem caused by the network delay in blockchain network. In 2018, the SBFT algorithm proposed by Gueta [61] optimized the centralization problem in the PBFT algorithm; at the same time, it enhanced the throughput of the system and improved the efficiency of the system in processing data. In 2018, the Thunderella algorithm proposed by Pass [62] replicates the operations of honest nodes in the blockchain through state machine replication, reducing repetitive identification work and greatly improving transaction speed. In the same year, the SG-PBFT algorithm proposed by Xu [63] reformed the PBFT algorithm by using a fractional grouping mechanism, which greatly improved the consensus efficiency and enhanced defensiveness.

3.3.2. PoW Algorithm

The PoW algorithm, namely the proof of work algorithm, was originally designed to resist spam attacks. That is, the mail sender must spend a certain amount of computing power to solve a mathematical problem before sending the mail, which greatly increases the cost of sending spam, thereby ensuring the security of the system. However, it is the application of “Satoshi Nakamoto” in the Bitcoin system that promotes the PoW algorithm. In the Bitcoin white paper, Satoshi Nakamoto officially used the PoW algorithm in distributed applications. This is also the first time that a Byzantine fault-tolerant consensus algorithm has been used in an open and public blockchain network.

The core idea of the PoW algorithm is to ensure the consistency of data and the security of consensus through computing power competition. Each node in the blockchain calculates a random nonce value through its own computing power. This value is compared with the solution value *target* of a mathematical problem of the blockchain system. If the nonce is less than the *target*, the node obtains the accounting right of this block and broadcasts the packaged block to the entire network. After receiving the broadcast packaged block, other nodes will verify the correctness of the block information. The PoW algorithm automatically adjusts the *target* to change the computational complexity, in order to ensure that the nodes in the chain will not lose their work enthusiasm because of too difficult evaluation and will not monopolize the accounting rights due to strong computing power of some nodes. As a result, the entire blockchain network remains relatively stable. This process of competing for accounting rights and being recognized and accepted by network nodes is also called "mining". Only the node that first solves the nonce value that meets the *target* condition can get the accounting right and obtain the rewards in the incentive mechanism. After receiving the broadcast, other nodes will give up solving the block problem, but they will verify the block information. During the verification process, other nodes cannot account, but will quickly verify the hash relationship information of the block information, to ensure the uniqueness and correctness of the data in the ledger. The working principle of the PoW algorithm is shown in Figure 7.

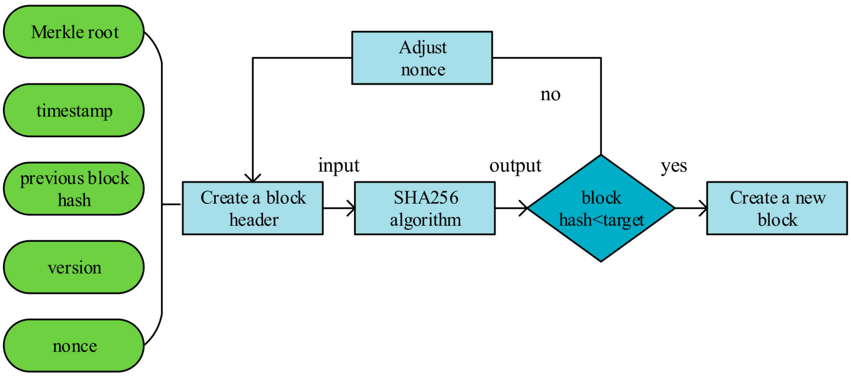

Figure 7. The working principle of the PoW algorithm.

The PoW algorithm is suitable for the public chain, and in the participation process, it is not necessary to introduce an identity verification mechanism for identity verification. It is the first Byzantine fault-tolerant algorithm that is actually applied in the public chain. In the PoW algorithm, the node obtains the nonce value through exhaustive calculation, and the algorithm is less difficult to implement. However, the PoW algorithm also has many problems. For example, it relies heavily on computing power in the process of solving *nonce* [64], which consumes huge computing resources, the PoW consensus agreement period is too long [65], it currently takes around 10 min to solve a math problem, and the transaction throughput is too low [66]. In response to these problems, the researchers improved the PoW algorithm and proposed some new algorithms originating from PoW. The Bitcoin-NG algorithm innovatively uses key blocks and micro blocks to unitize the running time of the PoW algorithm, which greatly improves the system throughput. The Byzcoin algorithm is based on the Bitcoin-NG algorithm and combines the PoW algorithm with the Byzantine fault-tolerant algorithm. While improving the system throughput, it also strengthens the security performance of the system. The GHOST algorithm changes the longest chain rule and proposes a new concept of block generation. Compared with the PoW algorithm, it reduces computing power consumption and improves the activity of nodes in the chain to participate in services. In 2021, the PoH consensus algorithm proposed by Arjomandi-Nezhad [67] changed the accounting rights competition mechanism based on the calculation power used in the PoW algorithm to the mechanism of the node's donation contribution to the chain. The more nodes donate to the entire chain, the easier it is to obtain the right to accounting. This algorithm is more suitable for social taxation scenarios such as educational funds or charities. In 2021, the CW-POW algorithm proposed by Kara [68] changed the single round workload proof of the PoW algorithm into a multi-round problem solution game, eliminated the nodes that did not solve the solution round by round, enhanced the robustness of the algorithm against attacks, and significantly reduced energy consumption

3.3.3. PoS Algorithm

Due to the obvious shortcomings of the PoW algorithm, such as waste of computing power and low efficiency in reaching consensus, the PoS algorithm was proposed in 2011.

The PoS algorithm is the proof-of-stake algorithm. As soon as it was proposed, it was practically applied in the digital currency “Peercoin” in the following year, which is enough to show that the PoS algorithm has a certain application value.

Different from the nodes of the PoW algorithm to obtain the accounting right through the competition of computing power, the PoS algorithm selects the node with the highest stake in the system as the accounting node. The PoS algorithm puts forward the concept of tokens. The stake of a node can be calculated based on the number and time of tokens held by it. The more tokens a node holds and the longer it holds, the higher its equity is. The higher the equity is, the lower the difficulty of “mining” is, and the higher the efficiency of finding the *target* value of random number. After the node successfully obtains the accounting right of the block, its stake will be cleared, and the next round of stake accumulation will begin. The working principle of the PoS algorithm is shown in Figure 8.

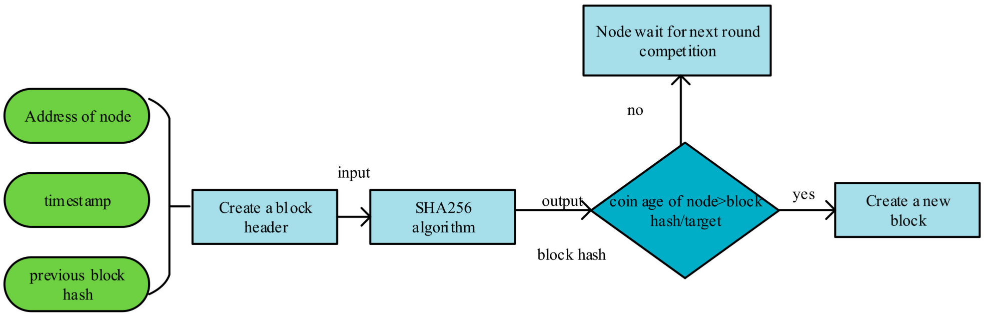

Figure 8. The working principle of the PoS algorithm.

The advantage of the PoS algorithm is that it does not need to go through the complicated “mining” process and only needs to pass the proof of stake to obtain the accounting right; this in turn reduces the block time and transaction processing time and greatly saves the time for consensus reaching, and the consensus efficiency is significantly improved [69]. In addition, the PoS algorithm also saves and improves the consumption of computing resources compared with the PoW algorithm [70]. In the method based on the proof of stake, when a cyber-attack affects the stake of the entire system, it will also damage the stake of the attacker. The disadvantage of the PoS algorithm is that nodes with higher equity tend to obtain higher accounting rights, which reduces the enthusiasm of users with low equity to participate, thereby reducing the activeness of the entire blockchain.

For subsequent improvements to the PoS algorithm, there is the Ouroboros algorithm based on security proofs, the Ouroboros Praos algorithm that adds identity confidentiality mechanism [71], and the Ouroboros Genesis algorithm that adds a self-starting process to prevent long-range attacks [72]. In addition, there is the Algorand consensus algorithm with high decentralization and low energy consumption [73]. This algorithm greatly reduces transaction time and is widely used in digital currencies. In 2014, the Tendermint algorithm proposed by Kwon [74] was optimized in transaction processing, with shorter transaction confirmation time and higher transaction throughput. In 2017, the Casper algorithm proposed by Buterin [75] combines the PoS algorithm with the PoW algorithm to ensure security and reduce energy consumption, but the transaction speed is not as good as the aforementioned algorithms such as Algorand and Ouroboros. In 2021, the LaKAS algorithm proposed by Reijsbergen [76] is a new type of PoS algorithm, which greatly improves the transaction speed and reduces the risk of long-range attacks.

3.3.4. DPoS Algorithm

The core idea of the DPoS algorithm, namely the Delegated Proof of Stake Algorithm,,is similar to the representative election system. The number of currencies held by each node in the system is the basis for the number of votes allocated to it. Through voting, nodes that they consider to be more credible are elected as the decision makers in the consensus process. According to the number of votes obtained, multiple decision makers are allocated, and decision makers take turns to obtain the accounting rights of the block. Every node in the system may become a decision maker. If the current decision maker violates the blockchain protocol, the decision-making identity will be cancelled. The system will allocate new decision makers to join the team of decision makers. Through the multi-decision makers mechanism, the accounting rights will not be too concentrated on a single node, which can prevent excessive centralization. The working principle of the DPoS algorithm is shown in Figure 9.

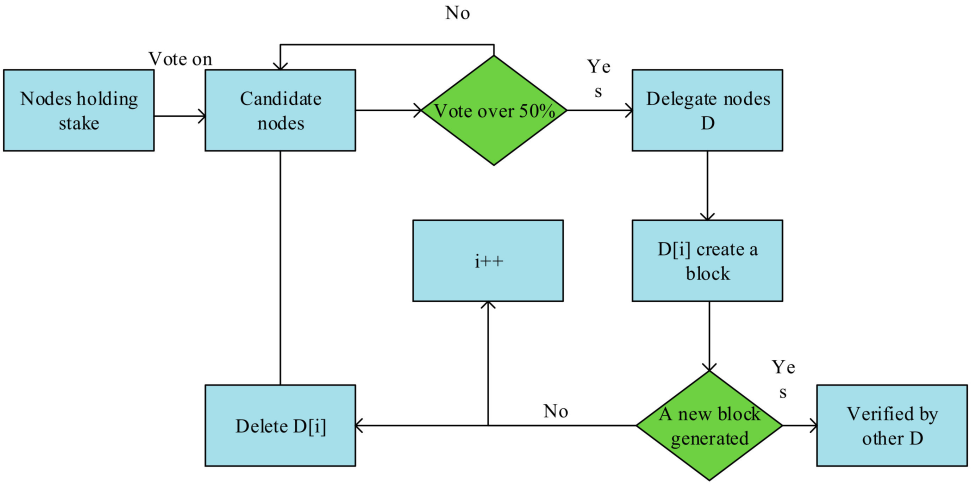

Figure 9. The working principle of the DPoS algorithm.

The DPoS algorithm combines the advantages of the PoS and PoW algorithms, respectively. Based on the multi-decision maker mechanism with accounting right in turn, the communication speed between nodes in the DPoS consensus is faster, and the nodes can quickly complete block packaging, broadcasting, and verification, and significantly improve the system transaction throughput. DPoS does not rely on computing resources and accordingly reduces energy consumption. Due to the election of decision makers, the transaction delay of the system is reduced, and the transaction processing speed has been significantly improved. At the same time, it can accommodate a large number of nodes to enter and exit the blockchain system freely, and the change of the node size will not affect the performance of the system, or the impact will be small, and the scalability is strong. The disadvantage of the DPoS algorithm is that it relies too much on voting rules. Poor voting rules will not only reduce the enthusiasm of participating nodes but also are not conducive to the realization of the decentralization of the entire system. Therefore, in 2020, Nir Bitansky [77] proposed the VRF consensus algorithm, which is based on the relatively highly centralized DPoS algorithm and adds virtual nodes to increase the uncertainty in the election process. This algorithm not only retains the characteristics of DPoS's high efficiency and low energy consumption but also improves the degree of decentralization.

3.3.5. PoH Algorithm

The PoH algorithm and CW-PoW algorithm are new algorithms proposed in recent years, the purpose of them is to improve the traditional consensus algorithm for some specific application scenarios. Next, we will introduce them from the aspects of working principle and improvement effect.

The full name of the PoH algorithm is Proof of Humanity; it is designed for gathering society-related taxes. The process of algorithm consensus is not similar to the PoW algorithm that relies on node computing power to compete; the probability of a node becoming the leader with accounting rights depends on the amount of its donation to the blockchain network. The more a candidate leader node contributes to the blockchain network, the more likely it is to become the leader of a new block. In this algorithm, participating nodes can generate smart contracts or transactions, and nodes will generate transaction fees during transactions. During the working phase of the algorithm, a smart contract is created containing a payable address and a reputation score. The payable address is the address at which participating nodes can make donations to the entire network. The reputation score represents the public's trust in the entire blockchain network, and the reputation score affects the weight of the node donation amount. In the leader selection stage of the algorithm, the amount of donations made by nodes to the blockchain network determines the leader's identity, and donations generally use recognized cryptocurrencies. The probability of a node becoming a leader is equal to its total donations as a percentage of the total donations received by the blockchain network. In order to maintain the safe and stable operation of the blockchain network, the PoH algorithm uses a distributed random variable generation algorithm to ensure that the leader randomly selected is neither predictable nor adjustable. When the leader node exceeds the preset time but does not send a block, other nodes with a high donation share will replace it as the leader.

The PoH algorithm replaces the computing power of the traditional algorithm through the donation system to compete for the right of accounting. Nodes only need to pay a relatively small fee to participate in the accounting and do not need to purchase expensive computing equipment to compete for computing power, which greatly reduces costs and energy consumption. In addition, through donations, most nodes in the blockchain network can benefit, and the nodes are also willing to maintain the stable operation of the blockchain network.

3.3.6. CW-PoW Algorithm

The CW-PoW algorithm means Compute and Wait in the PoW consensus algorithm. The algorithm is based on the PoW algorithm, and the nodes need to solve the *target* value to compete for the accounting right to reach a consensus. However, multiple rounds of calculation and evaluation are set up, and the difficulty of solving the *target* value in each round is much lower than that of the PoW algorithm; through layer-by-layer competition, the node with the smallest standard deviation from the *target* value in each round is selected and elected as the accounting node. During the algorithm process, the nodes are screened by preset rounds according to the calculation and solution situation, only the nodes that satisfy the current round of solving are eligible to enter the next round of solving, nodes that lag behind other nodes in a certain round of computing time will withdraw from the competition to avoid wasting computing resources. In the last round of competition, the first node that satisfies the *target* value and has the smallest deviation value in each round will be the accounting node.

The multi-round solution mode and standard deviation evaluation criteria are introduced by the CW-PoW algorithm. Compared with the PoW algorithm, the multi-round evaluation mode reduces the difficulty of solving the problem of the *target* and refines the entire evaluation problem into multiple rounds, and nodes that do not meet the evaluation conditions are eliminated early, which can greatly reduce the wasted computing power of nodes. In addition, the CW-PoW algorithm has also improved in terms of security performance. The multi-round evaluation mode can reduce the working speed of malicious nodes so that they cannot continue to perform sabotage; the standard deviation value will eliminate some malicious nodes so that they have no chance to do evil. Through experimental tests, the CW-PoW algorithm has significant improvements in energy consumption and robustness against 51% and Sybil attacks.

3.3.7. Analysis and Comparison of Typical Consensus Algorithms

By analyzing and comparing the characteristics of the above-mentioned various classic consensus algorithms, it can be known that the PoW algorithm and the PoS algorithm are suitable for public chains with a relatively open environment. The PoW algorithm has strong decentralization ability, high fault tolerance, and relatively simple implementation, but it has problems such as high energy consumption, slow transaction processing speed, and small throughput. The PoS algorithm achieves a consensus by electing the nodes with the highest equity, which consumes less energy but has a higher degree of centralization than the PoW algorithm and lacks security. The DPoS algorithm is optimized on the basis of the PoS algorithm, and the number of nodes participating in the work is optimized. The data processing speed has been significantly increased, and the security has also been improved, but the decentralization aspect needs further optimization. The Paxos algorithm, Raft algorithm, and PBFT algorithm are more suitable for permission chains where nodes such as consortium chains and private chains require identity verification and have a limited node size. These several consensus algorithms can provide excellent information processing capabilities, low resource consumption, and can tolerate interference from non-Byzantine errors. However, the implementation of the Paxos consensus algorithm is more difficult, and it is only suitable for consortium chains and private chains with relatively high security. The Raft algorithm is more feasible in practical applications, and its performance is similar to Paxos, but its shortcomings are similar to the Paxos consensus algorithm, and it is only suitable for consortium chains and private chains with relatively high security. The PBFT consensus algorithm has strong data processing capabilities and high security, and it is only applicable to consortium chains and private chains. In addition, the PoH and CW-PoW consensus algorithms proposed in 2021 are improved and optimized on the typical consensus algorithm, and they have distinctive features such as high computing efficiency, low energy consumption, safety, and reliability. The comparative analysis results of the above-mentioned various classic consensus algorithms are shown in Table 2.

Table 2. Comparative analysis of typical blockchain consensus algorithms.

Type	Time Proposed	Application Scenario	Efficiency	Energy Consumption	Core Principle	Advantage	Disadvantage
VR	**1988**	Consortium chain/Private chain	**High**	**Low**	**Log replication protocol**	**Simple, efficient and easy to implement**	**Insufficient fault tolerance**
Paxos	1990	Consortium chain/Private chain	High	Low	Based on message passing, a proposal can be consistent with high fault tolerance	Strong ability to tolerate non-Byzantine errors, high efficiency	Difficult to implement, unable to accommodate malicious nodes
Raft	2013	Consortium chain/Private chain	High	Low	Leader election, log replication mechanism	Strong ability to tolerate non-Byzantine errors, strong leader mechanism, high efficiency	Low applicability, unable to accommodate malicious nodes
PBFT	1999	Consortium chain/Private chain	High	Low	A consensus based on the leader election mechanism	High efficiency, strong security	Low degree of decentralization
PoW	2008	Public chain	Low	High	Competing for accounting rights based on computing power	High fault tolerance, simple logic, easy to implement	Long time for consensus, low throughput, high energy consumption
PoS	2012	Public chain	Relatively high	Relatively low	High-stakes nodes have the right to account	Higher efficiency than PoW algorithm, lower energy consumption	Insufficient security
DPoS	2014	Public chain	High	Low	The leader elected by the voting has the accounting right	High efficiency	Insufficient decentralization
PoH	2021	Public chain	High	Low	Donation contribution mechanism	High efficiency, high safety	Adapt to a single scene
CW-PoW	2021	Public chain	High	Relatively low	Multiple rounds of proof, node competition elimination mechanism	High safety and low energy consumption	Part of the computing power is wasted in node competition

4. Future Prospects of Consensus Algorithms

With the rapid development of blockchain technology, the requirements of practical applications for blockchain consensus algorithms are also increasing. Traditional consensus algorithms often have certain limitations and are difficult to meet the actual needs of blockchain applications. Therefore, improving and innovating traditional consensus algorithms are an important research direction of blockchain technology. This section analyzes and judges the future development trends of consensus algorithms.

4.1. Improvements Based on Traditional Consensus Algorithms

Traditional consensus algorithms have more or less various problems and shortcomings. In response to these problems and shortcomings, researchers have improved and perfected traditional consensus algorithms. While retaining the advantages of the original algorithm, trying to make up for and overcome the shortcomings of the original algorithm is also an important path for the development and evolution of consensus algorithms.

4.1.1. Continuous Improvement of PoW Algorithm

The PoW algorithm has problems such as low data throughput, high consumption of computing power, and too long time for consensus. So, the PoW algorithm can be improved from many aspects to make the PoW algorithm more perfect. Regarding the problem of low data throughput of the blockchain, expansion plans such as increasing the block size and reducing the block generation interval can be used to improve it. The Bitcoin-NG [16] protocol provides a new idea, namely introducing the concept of key blocks and micro-blocks. Key blocks are only used to elect leaders and do not store transaction information; micro-blocks are used to store transaction information. The generation of blocks does not need to consume computing power. After the node generates the key block, it is set as the leader, and the leader generates multiple micro blocks with less consumption and faster efficiency. By generating a large number of micro-blocks for storing data, the storage capacity of the blockchain can be greatly increased, and the transaction response time can be reduced. Aiming at the problem of too high computing power consumption, the Ethash algorithm proposed a solution, that is, to introduce I/O blocking and a directed acyclic graph to improve the solution of the *target* value in PoW. It uses a directed acyclic graph to create a large dataset and a small one, and the small dataset is used to verify block information, the large dataset is generated by calculation of the small dataset, and miners can only save the large dataset in order to mine faster. Through I/O blocking, the overhead of system resources is reduced. The Ethash algorithm makes mining evaluation more suitable for general-purpose computers with large memory capacity without other special requirements. Through dynamic adjustment, the production speed of the data block is improved, and the transaction time is shortened.

4.1.2. Continuous Improvement of PoS Algorithm

The PoS algorithm has the exclusive accounting rights of high-stakes nodes, which leads to network centralization problems. In response to these problems, researchers have done a lot of exploration and improvement. There are two main directions to improve the PoS algorithm. One direction is to imitate the PoW algorithm. Virtual mining technology refers to a set of different mining methods with one common feature: that only a small amount of computing resources are required for participating nodes. It replaces the computing power competition and new blocks are selected, which not only guarantees the randomness in the election process and reduces the risk of centralization but also avoids waste of computing power, reduces the transaction verification delay, and increases the fairness of mining. The other direction is to combine the PoS algorithm with the Byzantine fault-tolerant algorithm. In the voting election of the Byzantine fault-tolerant algorithm, the votes are set different weights according to the stake, and those of the main nodes are given higher weights. If the weight of the voting result exceeds two-thirds of the overall weight, a consensus can be reached. Based on this idea, Algorand and Ethereum proposed the BA protocol and the Casper Friendly Finality Gadge protocol. In addition, through the introduction of a reward mechanism, rewards are provided for the honest nodes in a blockchain to provide effective computing power, so that the behavior of each node is close to a Nash equilibrium state that does not increase its own revenue due to changes in its own strategy. This method can improve the security of the PoS algorithm. The Ouroboros algorithm in the ADA coin system is a case of this improvement.

4.2. Improvement of Consensus Algorithms Focusing on Specific Performance

4.2.1. Improvement of Consensus Algorithms Focusing on Efficiency

Whether an algorithm is excellent depends largely on the efficiency of the algorithm. A good consensus algorithm should have the characteristics of high computational efficiency, small delay, safety, and stability. Although the traditional consensus algorithm can ensure the smooth operation of the blockchain system, most of its efficiency etc. is flawed. For example, the PoW consensus algorithm requires at least 10 min for each hash calculation, and the confirmation delay of a single transaction is about 1 h. Such work efficiency of the PoW can no longer meet the practical needs of frequent computing. The algorithm efficiency greatly affects the application and promotion of blockchain in real-world scenarios, and the improvement of algorithm efficiency is also an important development direction of the blockchain consensus algorithm. An improved MPoW algorithm based on the PoW algorithm [78] optimizes the hash algorithm and uses matrix calculations to reduce the block time of the algorithm by about 30%, which significantly improves the efficiency of the algorithm. The PoT algorithm [79] introduces a trust-proof mechanism to dynamically authorize trust for nodes in blockchain. The higher the trust, the higher the possibility of accounting. This algorithm reduces the network delay in the chain, decreases the consensus reaching time, and greatly improves the efficiency of the system.

4.2.2. Improvement of Consensus Algorithms Focusing on Scalability

The scalability of an algorithm is a measure of the computing and processing power of the algorithm. The high-scalability algorithm can ensure the vigorous vitality of the blockchain nodes. At the same time, it can accommodate various behaviors of nodes in the chain, achieve a linear increase in the processing capacity of the system, and achieve the effect of reducing transaction delay and improving throughput. The scalability of traditional algorithms has limitations, and it is often impossible to achieve efficient sharing of information in a chain. The PBCM algorithm [80] optimizes this problem. The algorithm adopts a double-layer chain to realize a master–slave and multi-chain structure, it connects each main block through a global block, and the transaction throughput is significantly improved. The ELGamal consensus algorithm proposed by Min Xinping in 2018 [81] is based on the PBFT algorithm, and it improved the ring signature technology to solve the problem of dynamic entry and exit of nodes in a blockchain network. The fault tolerance rate is improved, and the scalability is also well improved.

4.2.3. Improvement of Consensus Algorithms Focusing on Security

The security of an algorithm is an important criterion for whether an algorithm can stably meet the practical application. A consensus algorithm with excellent security performance is to ensure that the blockchain continues to work efficiently while also resisting malicious attacks, maintaining the stability of the blockchain network system, and ensuring the correctness and uniformity of the blockchain ledger data [82]. At present, the more common blockchain attacks [83] include selfish mining attacks, block interception attacks, and "double-spending attacks" that *target* revenue attacks and reduce the effective computing power in blockchain, evasion of eclipse attacks, witch attacks, etc. from the constraints of the blockchain consensus algorithm. The blockchain consensus algorithm prevents "double-spending attacks" by setting up fork detection, performs identity verification by identifying the computing power performance in blockchain, and monitors the effective computing power in blockchain to prevent attacks. By setting up a relay node mechanism and buffering it, the attack of camouflaged nodes is prevented [84]. There is also an identity authorization mechanism and the addition of access control permissions to protect data to achieve the goal of preventing data leakage and greatly enhance the data security of the blockchain system [85]. These schemes use different methods to significantly improve the security performance of the blockchain.

4.3. Hybrid Consensus Algorithms Aggregating the Advantages of Other Ones

In real application scenarios, a single specific type of consensus algorithm often has greater limitations, such as the resource consumption problem of the PoW algorithm, and the problem of the PBFT algorithm that only applies to consortium chains and private chains, not public chains, etc. With the progress of consensus algorithm research, scholars have discovered that there are more or less complementary features between consensus algorithms. Therefore, integrating the advantages of various algorithms into one can undoubtedly achieve the purpose of maximizing strengths and avoiding weaknesses. This also provides a new idea and reference for the improvement of consensus algorithms in the future.

For example, the "Peercoin" system mentioned earlier combines the PoW algorithm with the PoS algorithm. In the early stage of mining, the PoW algorithm is used to distribute the stake to the miners relatively fairly, so that each miner has the same probability of obtaining token, and the PoS algorithm is used to calculate the stake of the miners in the later stage. Adjusting the difficulty of mining based on the stake can effectively alleviate the problem of waste of computing power, shorten the time of reaching a consensus, and greatly improve the efficiency of the system.

The well-known blockchain software system EOS [86] is currently the encrypted digital currency project with the highest market value except for Bitcoin and Ethereum systems. The consensus mechanism it uses is BFT-DPoS, which is a delegated stake proof algorithm with Byzantine fault tolerance. In the consensus process, the nodes are first voted to determine the decision maker through the DPoS algorithm, and then, the decision makers communicate with each other to formulate the block sequence of the system, and six blocks are continuously generated at an interval of 0.5 s. This minimizes the delay of continuous block propagation, increases the speed of block generation, and greatly increases the number of transactions, so as to achieve the goal of using blockchain technology to support a million-level customer base [87].

5. Conclusions

The consensus algorithm is one of the core technologies of the blockchain. Through continuous research, improvement, and development of the consensus algorithm, the overall performance of the blockchain has gradually been significantly improved. This article starts with the principles of blockchain technology and elaborates on the structure of the blockchain and the related technologies used. Then, based on the access mode and permission of the blockchain, the different classifications of the blockchain are summarized, namely, public chain, consortium chain, and private chain. In-depth analysis and comparison of the characteristics of various blockchains have been carried out. Different types of blockchain networks have their own suitable application scenarios. Regardless of the type of blockchain, the consensus mechanism and consensus algorithm are an important part of it. This article takes the time of consensus algorithm development as the main axis and successively expounds the working principles of a variety of classic consensus algorithms, as well as their advantages and disadvantages. The shortcomings and limitations of consensus algorithms include too long consensus time, low throughput, and large transaction delays. In view of the deficiencies of these algorithms, researchers are constantly exploring, improving, and perfecting. The article summarizes the direction and ways of improving consensus algorithms and makes predictions on the future development trend of consensus algorithms. The subsequent research work of the new consensus algorithm is based on the traditional consensus algorithm, inherits and develops the excellent characteristics of the traditional algorithm, and improves the shortcomings of the algorithm. The improvement of the new consensus algorithm can also be studied from a certain feature of the algorithm, focusing on algorithm efficiency, scalability, security, and other aspects to improve the work efficiency of the consensus algorithm. In addition, through the mixed use of various consensus algorithms, innovative algorithms can be proposed by synthesizing the advantages of various excellent consensus algorithms. Those are the article's prediction and summary of the future development trend of consensus algorithms. Through the continuous development and optimization of the consensus algorithm, the computing efficiency has been continuously improved, and the transaction throughput has continued to increase. Blockchain technology can be better applied to actual scenarios with increasing demand and generate more practical application value.

Author Contributions: Investigation, M.C.; resources, Y.Z.; writing—original draft preparation, M.C.; writing—review and editing, H.X.; supervision, Y.Z.; project administration, C.W.; funding acquisition, W.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National key R & D projects, grant number 2020YFD1100600, National Nature Fund Project, grant number 61762048, Scientific research project of Jiangxi Provincial Department of Education, grant number GJJ190180, GJJ200428, Scientific research fund project of Jiangxi Agricultural University, grant number 9232307210.

Data Availability Statement: Not Applicable, the study does not report any data.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Han, X.; Yuan, Y.; Wang, F.Y. Security problems on blockchain: The state of the art and future trends. *Acta Autom. Sin.* **2019**, *45*, 206–225.
2. Berdik, D.; Otoum, S.; Schmidt, N.; Porter, D.; Jararweh, Y. A survey on blockchain for information systems management and security. *Inf. Process. Manag.* **2021**, *58*, 102397. [CrossRef]
3. Cai, X.Q.; Deng, Y.; Zhang, L.; Shi, J.C.; Chen, Q.; Zhen, W.L.; Guo, M.Y. The principle and core technology of blockchain. *Chin. J. Comput.* **2019**, *42*, 1–15.
4. Pu, H.P.; Yu, G.; Zhang, Y.F.; Bao, Y. Survey on Blockchain Technology and Its Application Prospect. *Comput. Sci.* **2017**, *44*, 1–7.
5. Yuan, Y.; Ni, X.C.; Zeng, S.; Wang, F. Blockchain Consensus Algorithms: The State of the Art and Future Trends. *Acta Autom. Sin.* **2018**, *44*, 2011–2022.
6. Belchior, R.; Vasconcelos, A.; Guerreiro, S.; Correia, M. A survey on blockchain interoperability: Past, present, and future trends. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–41. [CrossRef]
7. IBM Financial Services. 2019. Available online: https://www.ibm.com/blockchain/industries/financial-service (accessed on 25 December 2021).
8. Ant Financial Blockchain. 2019. Available online: https://tech.antfin.com/blockchain (accessed on 25 December 2021).
9. Chen, L.; Cong, L.W.; Xiao, Y. A Brief Introduction to Blockchain Economics. In *Information for Efficient Decision Making: Big Data, Blockchain and Relevance*; 2021; pp. 1–40. Available online: https://www.worldscientific.com/doi/abs/10.1142/9789811220470_0001 (accessed on 25 December 2021).
10. IBM Supply Chain. 2019. Available online: https://www.ibm.com/blockchain/industries/supply-chain (accessed on 25 December 2021).
11. Government Blockchain Association. 2019. Available online: https://www.gbaglobal.org/ (accessed on 25 December 2021).
12. Jing, N.; Liu, Q.; Sugumaran, V. A blockchain-based code copyright management system. *Inf. Process. Manag.* **2021**, *58*, 102518. [CrossRef]
13. Cryptocurrency Market Cap. 2019. Available online: https://coinmarketcap.com/ (accessed on 25 December 2021).
14. Javaid, M.; Haleem, A.; Singh, R.P.; Khan, S.; Suman, R. Blockchain technology applications for Industry 4.0: A literature-based review. *Blockchain: Res. Appl.* **2021**, *2*, 100027. [CrossRef]
15. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System. Available online: http://bitcoins.info/bitcoin.pdf (accessed on 25 December 2021).
16. Eyal, I.; Gencer, A.E.; Sirer, E.G.; Van Renesse, R. Bitcoin-NG: A Scalable Blockchain Protocol. In Proceedings of the 13th USENIX Symposium on Networked Systems Design and Implementation, Santa Clara, CA, USA, 16–18 March 2016; pp. 45–59.
17. Liu, A.D.; Du, X.H.; Wang, N.; Li, S.Z. Research progress of blockchain technology and its application in information security. *J. Softw.* **2018**, *29*, 2092–2115.
18. Eisenberg, E.; Gale, D. Consensus of subjective probabilities: The pari-mutuel method. *Ann. Math. Stat.* **1959**, *30*, 165–168. [CrossRef]

19. Bano, S.; Sonnino, A.; Al-Bassam, M.; Azouvi, S.; McCorry, P.; Meiklejohn, S.; Danezis, G. SoK: Consensus in the Age of Blockchains. In Proceedings of the 1st ACM Conference, Online, 21 October 2019.
20. Oki, B.M.; Liskov, B.H. Viewstamped replication: A New Primary Copy Method to Support Highly-Available Distributed Systems. In Proceedings of the 7th Annual ACM Symposium on Principles of Distributed Computing, Toronto, ON, Canada, 15–17 August 1988; ACM: New York, NY, USA, 1988; pp. 1–7.
21. Lamport, L. The part-time parliament. *ACM Trans. Comput. Syst.* **1998**, *16*, 133–169. [CrossRef]
22. Castro, M.; Liskov, B. Practical Byzantine Fault Tolerance. In Proceedings of the Third Symposium on Operating SystemsDesign and Implementation, New Orleans, LA, USA, 22–25 February 1999; USENIX Association: Berkeley, CA, USA, 1999; pp. 173–186.
23. Jakobsson, M.; Juels, A. Proofs of Work and Bread Pudding Protocols (Extended Abstract). In *Secure Information Networks*; Springer: Berlin/Heidelberg, Germany; Boston, MA, USA, 1999; pp. 258–272.
24. Proof of Stake. Available online: https://en.bitcoin.it/wiki/Proof_of_Stake (accessed on 25 December 2021).
25. BitShares. Delegated Proof of Stake. 2021. Available online: http://docs.bitshares.org/bitshares/dpos.html (accessed on 25 December 2021).
26. Ren, L. Proof of Stake Velocity: Building the Social Currency of the Digital Age. 2021. Available online: https://assets.coss.io/documents/white-papers/reddcoin.pdf (accessed on 25 December 2021).
27. Proof of Burn. 2021. Available online: https://en.bitcoin.it/wiki/Proof_of_burn (accessed on 25 December 2021).
28. Bentov, I.; Lee, C.; Mizrahi, A.; Rosenfeld, M. Proof of Activity: Extending Bitcoin.s Proof of Work Via Proof of Stake. 2021. Available online: http://eprint.iacr.org/2014/452 (accessed on 25 December 2021).
29. Miller, A.; Xia, Y.; Croman, K.; Shi, E.; Song, D. The Honey Badger of BFT Protocols. In Proceedings of the Acm Sigsac Conference on Computer & Communications Security, Vienna, Austria, 24–28 October 2016; ACM: New York, NY, USA, 2016; pp. 31–42.
30. Verifiable Byzantine Fault Tolerance. 2021. Available online: https://ont.io/wp/Ontologytechnology-white-paper-ZH.pdf (accessed on 25 December 2021).
31. Snowflake to Avalanche: A Novel Metastable Consensus Protocol Family for Crypto. 2021. Available online: https://ipfs.io/ipfs/QmUy4jh5mGNZvLkjies1RWM4YuvJh5o2FYopNPVYwrRVGV (accessed on 25 December 2021).
32. Buterin, V. A Guide to 99% Fault Tolerant Consensus. 2021. Available online: https://vitalik.ca/general/2018/08/07/99_fault_tolerant.html (accessed on 25 December 2021).
33. Merkle, R.C. A Digital Signature Based on a Conventional Encryption Function. In Proceedings of the Conference on the Theory and Application of Cryptographic Techniques, Amsterdam, The Netherlands, 16–20 August 1987; pp. 369–378.
34. Peng, L.; Feng, W.; Yan, Z.; Li, Y.; Zhou, X.; Shimizu, S. Privacy preservation in permissionless blockchain: A survey. *Digit. Commun. Netw.* **2020**, *7*, 295–307. [CrossRef]
35. Gupta, S.; Sadoghi, M. Blockchain transaction processing. *arXiv* **2021**, arXiv:2107.11592.
36. Milojicic, D.S.; Kalogeraki, V.; Lukose, R.; Nagaraja, K.; Pruyne, J.; Richard, B.; Xu, Z. Peer-to-Peer Computing. HP Lab. Palo Alto. 2002. Available online: https://www.cs.kau.se/cs/education/courses/dvad02/p2/seminar4/Papers/HPL-2002-57R1.pdf (accessed on 25 December 2021).
37. Katz, J.; Lindell, Y. *Introduction to Modern Cryptography (Chapman & Hall/Crc Cryptography and Network Security Series)*; Chapman & Hall/CRC: Boca Raton, FL, USA, 2007.
38. Szabo, N. Formalizing and Securing Relationships on Public Networks [EB/OL]. 2018. Available online: https://ojphi.org/ojs/index.php/fm/article/view/548 (accessed on 25 December 2021).
39. Yuan, Y.; Wang, F.Y. Blockchain: The state of the art and future trends. *Acta Autom. Sin.* **2016**, *42*, 481–494.
40. Shiqin, Z.; Ru, H.; Tao, H.J.L.; Shuo, W.A.N.G.; Wei, F.E.N.G. Survey of blockchain: Principle, progress and application. *J. Commun.* **2020**, *41*, 134.
41. Bitfury Group, J.G. Public versus Private Blockchains. *SSRN Electron. J.* **2015**. Available online: https://bitfury.com/content/downloads/public-vs-private-pt1-1.pdf (accessed on 25 December 2021).
42. Vukolić, M. Rethinking Permissioned Blockchains. In Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts, Online, 2 April 2017; pp. 3–7.
43. Zheng, Z.; Xie, S.; Dai, H.; Chen, X.; Wang, H. An overview of blockchain technology: Architecture, consensus, and future trends. *IEEE Int. Congr. Big Data* **2017**, 557–564. [CrossRef]
44. Buterin, V. Ethereum Whitepaper. 2020. Available online: https://github.com/ethereum/wiki/wiki/White-Paper (accessed on 25 December 2021).
45. Androulaki, E.; Barger, A.; Bortnikov, V.; Cachin, C.; Christidis, K.; De Caro, A.; Yellick, J. Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains [EB/OL]. 2018. Available online: https://dl.acm.org/doi/abs/10.1145/3190508.3190538 (accessed on 25 December 2021).
46. Brown, R.G.; Carlyle, J.; Grigg, I.; Hearn, M. Corda: An Introduction. 2016. Available online: http://r3cev.com/s/corda-introductory-whitepaper-final.pdf (accessed on 25 December 2021).
47. Lamport, L. Proving the Correctness of Multiprocess Programs. *IEEE Trans. Softw. Eng.* **1977**, *3*, 125–143. [CrossRef]
48. Gilbert, S.; Lynch, N. Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. *Acm Sigact News* **2002**, *33*, 51–59. [CrossRef]
49. Lamport, L. The Byzantine Generals Problem. *Acm Trans. Program. Lang. Syst.* **1982**, *4*, 382–401. [CrossRef]

50. Rao, J.; Shekita, E.J.; Tata, S. Using Paxos to Build a Scalable, Consistent, and Highly Available Datastore. *Proc. VLDB Endow.* **2011**, *4*, 243–254. [CrossRef]
51. Lamport, L.; Massa, M. Cheap Paxos. In Proceedings of the International Conference on Dependable Systems and Networks, New York, NY, USA, 28 June–1 July 2004; IEEE: Piscataway, NJ, USA, 2004; pp. 307–314.
52. Mao, Y.; Junqueira, F.P.; Marzullo, K. Mencius: Building Efficient Replicated State Machines for WANs. In Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation, OSDI'08, Berkeley, CA, USA, 8–10 December 2008; pp. 369–384.
53. Lamport, L.B. Generalized Paxos. U.S. Patent 7,698,465, 13 April 2010.
54. Ongaro, D.; Ousterhout, J. In Search of an Understandable Consensus Algorithm. In Proceedings of the USENIX Annual Technical Conference, Philadelphia, PA, USA, 19–20 June 2014; pp. 305–319.
55. Wang, R.; Zhang, L.; Xu, Q.; Zhou, H. K-Bucket Based Raft-like Consensus Algorithm for Permissioned Blockchain. In Proceedings of the 2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS), Tianjin, China, 4–6 December 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 996–999.
56. Zhang, Y.; Han, B.; Zhang, Z.L.; Gopalakrishnan, V. Network-Assisted Raft Consensus Algorithm. In Proceedings of the SIGCOMM Posters and Demos, Online, 22 August 2017; pp. 94–96.
57. Wang, Y.; Li, S.; Xu, L.; Xu, L. Improved Raft Consensus Algorithm in High Real-Time and Highly Adversarial Environment. In *International Conference on Web Information Systems and Applications*; Springer: Cham, Switzerland, 2021; pp. 718–726.
58. Tian, G.H.; Hu, Y.H.; Chen, X.F. Research progress on attack and defense techniques in block-chain system. *Ruan Jian Xue Bao/J. Softw.* **2021**, *32*, 1495–1525.
59. Schneider, F.B. Implementing Fault-Tolerant Services Using the State Machine Approach: A Tutorial. *ACM Comput. Surv.* **1990**, *22*, 299–319. [CrossRef]
60. Kotla, R. Zyzzyva: Speculative Byzantine Fault Tolerance. In *Acm Sigops Symposium on Operating Systems Principles*; ACM: New York, NY, USA, 2007.
61. Golan Gueta, G.; Abraham, I.; Grossman, S.; Malkhi, D.; Pinkas, B.; Reiter, M.K.; Tomescu, A. SBFT: A Scalable Decentralized Trust Infrastructure for Blockchains. **2018**, *4*, 50–62. Available online: https://ui.adsabs.harvard.edu/abs/2018arXiv180401626G/abstract (accessed on 25 December 2021).
62. Pass, R.; Shi, E. *Thunderella: Blockchains with Optimistic Instant Confirmation*; Springer: Cham, Switzerland, 2018; Volume 10821, pp. 3–33.
63. Xu, G.; Liu, Y.; Xing, J.; Luo, T.; Gu, Y.; Liu, S.; Vasilakos, A.V. SG-PBFT: A Secure and Highly Efficient Blockchain PBFT Consensus Algorithm for Internet of Vehicles. *arXiv* **2021**, arXiv:2101.01306.
64. De Vries, A. Bitcoin's growing energy problem. *Joule* **2018**, *2*, 801–805. [CrossRef]
65. Rudden, J. Average Confirmation Time of Bitcoin Transactions. Available online: https://www.statista.com/statistics/793539/bitcoin-transaction-confirmation-time/ (accessed on 25 December 2021).
66. Rosenfeld, M. Analysis of Hashrate-Based Double Spending. *arXiv* **2014**, arXiv:1402. 2009.
67. Arjomandi-Nezhad, A.; Fotuhi-Firuzabad, M.; Dorri, A.; Dehghanian, P. Proof of humanity: A tax-aware society-centric consensus algorithm for Blockchains. *Peer—Peer Netw. Appl.* **2021**, *14*, 3634–3646. [CrossRef]
68. Kara, M.; Laouid, A.; AlShaikh, M.; Hammoudeh, M.; Bounceur, A.; Euler, R.; Laouid, B. A Compute and Wait in PoW (CW-PoW) Consensus Algorithm for Preserving Energy Consumption. *Appl. Sci.* **2021**, *11*, 6750. [CrossRef]
69. Saleh, F. Blockchain without waste: Proof-of-stake. *Rev. Financ. Stud.* **2021**, *34*, 1156–1190. [CrossRef]
70. Sunny, K.; Scott, N. PPcoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake [EB/OL]. Available online: https://decred.org/research/king2012.pdf,2012-8-19 (accessed on 25 December 2021).
71. David, B.; Gaži, P.; Kiayias, A.; Russell, A. Ouroboros Praos:An Adaptively-Secure, Semi-Synchronous Proof-of-Stake Blockchain. In *International Conference on the Theory and Applications of Cryptographic Techniques*; Springer: Cham, Switzerland, 2018; pp. 66–98.
72. Badertscher, C.; Gaži, P.; Kiayias, A.; Russell, A.; Zikas, V. Ouroboros Genesis:Composable Proof-of-Stake Blockchains with Dynamic Availability. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, New York, NY, USA, 15–19 October 2018; ACM Press: New York, NY, USA, 2018; pp. 913–930.
73. Gilad, Y.; Hemo, R.; Micali, S.; Vlachos, G.; Zeldovich, N. Algorand: Scaling Byzantine Agreements for Cryptocurrencies. In Proceedings of the 26th Symposium on Operating Systems Principles, New York, NY, USA, 14 October 2017; ACM: New York, NY, USA, 2017; pp. 51–68.
74. Kwon, J. TenderMint: Consensus without Mining. **2014**, *1*, 1–10. Available online: https://cdn.relayto.com/media/files/LPgoWO18TCeMIggJVakt_tendermint.pdf (accessed on 25 December 2021).
75. Buterin, V.; Griffith, V. Casper the Friendly Finality Gadget. *arXiv* **2017**, arXiv:1710.09437, 1–10.
76. Reijsbergen, D.; Szalachowski, P.; Ke, J.; Li, Z.; Zhou, J. LaKSA: A Probabilistic Proof-of-Stake Protocol. In *Proceedings 2021 Network and Distributed System Security Symposium*; Internet Society: Reston, VA, USA, 2021.
77. Nir, B. Verifiable Random Functions from Non-interactive Witness-Indistinguishable Proofs. *J. Cryptol.* **2020**, *33*, 459–493.
78. Zeng, L.; Xin, S.; Xu, A.; Pang, T.; Yang, T.; Zheng, M. Seele's New Anti-ASIC Consensus Algorithm with Emphasis on Matrix Computation. *arXiv* **2019**, arXiv:1905.04565.
79. Huang, J.H.; Xia, X.; Li, Z.C. Proof of Trust: Mechanism of Trust Degree Based on Dynamic Authorization. *J. Softw.* **2019**, *30*, 2593–2607.
80. Fang, Y.; Deng, J.Q.; Cong, L.H. An improved scheme for PBFT blockchain consensus algorithm based on ring signature. *Comput. Eng.* **2019**, *45*, 32–36.

81. Min, X.P.; Li, Q.Z.; Kong, L.J.; Zhang, S.D.; Zheng, Y.Q.; Xiao, Z. Permissioned Blockchain Dynamic Consensus Mechanism Based Multi-Centers. *Chin. J. Comput.* **2018**, *41*, 16.
82. Qian, W.; Shao, Q.; Zhu, Y.; Jin, C.; Zhou, A. Research Problems and Methods in Blockchain and Trusted Data Management. *J. Softw.* **2018**, *29*, 150–159.
83. Singh, S.; Hosen, A.S.; Yoon, B. Blockchain security attacks, challenges, and solutions for the future distributed iot network. *IEEE Access* **2021**, *9*, 13938–13959. [CrossRef]
84. Tsai, W.T.; Yu, L.; Wang, R.; Liu, N.; Deng, E.Y. Blockchain Application Development Techniques. *J. Softw.* **2017**, *28*, 1474–1487.
85. Shi, N.; Tan, L.; Li, W.; Qi, X.; Yu, K. A blockchain-empowered AAA scheme in the large-scale HetNet. *Digit. Commun. Netw.* **2021**, *7*, 308–316. [CrossRef]
86. Larimer, D. Eos. Io Technical White Paper V2 [EB/OL]. Available online: https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md (accessed on 25 December 2021).
87. Ismail, L.; Materwala, H. Article A Review of Blockchain Architecture and Consensus Protocols: Use Cases, Challenges, and Solutions. *Symmetry* **2019**, *11*, 1198. [CrossRef]