mathematics

MDPI

Systematic Review

A Systematic Review of Consensus Mechanisms in Blockchain

Sisi Zhou [1,2], Kuanching Li [1,2,*], Lijun Xiao [1,2], Jiahong Cai [1,2], Wei Liang [1,2] and Arcangelo Castiglione [3]

[1] School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan 411201, China; sisizhou@mail.hnust.edu.cn (S.Z.)
[2] Hunan Key Laboratory for Service Computing and Novel Software Technology, Xiangtan 411201, China
[3] Department of Computer Science, Università degli Studi di Salerno, 84084 Fisciano, Italy
* Correspondence: aliric@hnust.edu.cn

Abstract: Since the birth of Bitcoin, blockchain has shifted from a critical cryptocurrency technology to an enabling technology. Due to its immutability and trustworthiness, blockchain has revolutionized many fields requiring credibility and high-quality data for decision making. Particularly in business intelligence and business process management, users can use blockchain to build their blockchain-enabled collaboration and data-sharing ecosystem with their partners. In this paper, we present the development process of blockchain and consensus mechanisms, where important blockchain consensus mechanisms are introduced. The consensus mechanism is the kernel among various blockchain components to ensure security and performance. Again, we present a comparison of these consensus mechanisms from different perspectives. We take the blockchain-enabling business as an example and analyze the relationship between blockchain and business process characteristics and the ideas and principles for selecting consensus mechanisms. Finally, we describe the differences and connections among many consensus mechanisms while laying a foundation for selecting appropriate consensus mechanisms for different scenarios and fields of application.

Keywords: blockchain; BFT; business intelligence; distributed systems; hybrid consensus mechanisms

MSC: 68U99

Citation: Zhou, S.; Li, K.; Xiao, L.; Cai, J.; Liang, W.; Castiglione, A. A Systematic Review of Consensus Mechanisms in Blockchain. *Mathematics* **2023**, *11*, 2248. https://doi.org/10.3390/math11102248

Academic Editor: Basilis Boutsinas

Received: 11 March 2023
Revised: 8 May 2023
Accepted: 8 May 2023
Published: 11 May 2023

1. Introduction

Blockchain technology was introduced in 2008 with the release of Bitcoin by a pseudonymous creator S. Nakamoto [1] and had its first real-world application in January 2009. Following Bitcoin, several projects emerged, such as Litecoin [2], NameCoin [3], PrimeCoin [4], and Ripple [5]. Research on cryptocurrency pushed the development of the international monetary system and led to rapid blockchain technology development. The blockchain platform Ethereum [6], proposed in 2013, and the concept of Web 3.0 mark the beginning of the Blockchain 2.0 era, where the most significant difference between Ethereum and Bitcoin is that Ethereum is an application platform. The most important innovation introduced by Ethereum has been smart contracts, which expanded the application scope of blockchain from the digital currency field to other financial areas involving contract consensus. Later, some enterprise blockchain platforms appeared, such as Hyperledger Fabric [7], Quorum [8], and Codra [9]. Due to its immutability and trustworthiness, blockchain has become an enabling technology and has led to the building of a credible digital environment, marking the era of blockchain 3.0. Ideally, blockchain is perceived as a trusted machine or data-sharing method that enjoys widespread support. This perception has led blockchain technology to affect many fields and scenarios, such as business intelligence [10], business process [11], copyright transactions [12], data privacy protection [13,14], data transmission in the Internet of Things (IoT) systems [15], data authentication in the Internet of vehicles (IoV) [16,17], Metaverse [18], NFTs [19], online social networks [20], public and social services [21], and supply chain management [22], among others.

The scope of blockchain applications has gradually expanded, and the requirements for blockchain have proportionally increased. Different types of blockchain platforms and application scenarios adopt different consensus mechanisms, especially in the financial and business intelligence fields, where high security, scalability, high throughput, and low latency are required to optimize business processes and reduce operating costs. For instance, Vukolić [23] highlighted the importance of selecting an appropriate consensus mechanism for a given application, taking into account factors such as security, efficiency, and scalability in 2015. Next, using the properties given by Vukolić, Zheng et al. [24] provided an overview of blockchain architecture and a comparison of different consensus mechanisms. Later, Viriyasitavat et al. [25] analyzed the consensus mechanisms that had been exercised, which can promote collaboration, knowledge sharing, and collective decision making in the blockchain-enabling business process 4.0. Concurrently, Biswas et al. [26] proposed a proof of block and trade (PoBT) consensus mechanism for IoT blockchain to make business processes efficient and smart. Later, in 2022, Xu et al. [27], and Wang et al. [28] combined federated learning algorithms to optimize blockchain technology and the consensus mechanism. Therefore, a remarkable research effort has been devoted to solving the problem that traditional consensus mechanisms cannot simultaneously satisfy the need for high security, high scalability, high throughput, and low latency in financial technology scenarios. The innovation of consensus mechanisms has become the key to promoting blockchain applications in these fields, requiring credibility and high-quality data for decision making. It is still a hotspot in research.

The concept of consensus mechanism starts from distributed systems. The research on classic distributed consensus mechanisms began in the 1980s. Typical mechanisms such as Paxos [29], Raft [30], and PBFT [31] have been widely used and continuously optimized. These mechanisms are secure in the classical model. A mature consensus system can achieve higher throughput in practical applications, and its transaction confirmation time is short. According to the access authority, consensus mechanisms can be divided into permissioned and permissionless. Permissioned mechanisms require every node in the consensus system to know each other's identity, which is also conducive to objective supervision. In permissionless mechanisms, the node does not need access to join the consensus system, allowing more significant consensus-participating nodes, a higher degree of decentralization, and better privacy. Permissionless agreements are mostly mechanisms for the public blockchain. Taking the proof of work (PoW) [1] in Bitcoin as an example, the total number of nodes is unlimited, and any node can join or exit at any time. However, every miner must carry the corresponding workload proof when proposing a new block, which requires many computing resources. So far, many consensus mechanisms have been proposed, and researchers are still exploring effective attacks, defense methods, and security models for different consensus mechanisms.

There is no best consensus mechanism but only the most suitable for a given application scenario. Research on consensus mechanisms still needs to be carried out from multiple perspectives. In this context, Zheng et al. [24] presented a comprehensive overview of blockchain technology and compared the typical consensus mechanisms used in different blockchains. However, consensus mechanisms have yet to be deeply analyzed and studied. Nguyen et al. [32] categorized some of the consensus mechanisms used in blockchain into two types, namely, proof-based and vote-based; for each type, they outlined advantages and drawbacks. Later, Fu et al. [33] categorized blockchain consensus algorithms into four modes, namely the leader-based mode, the voting-based mode, the committee+voting mode, and the fair accounting mode. Although such classification is very detailed, there is a blending among the categories which could be more friendly to beginners. Based on the above observation, in this study, we combined the development of distributed consensus mechanisms, analyzed the traditional consensus mechanisms as well as blockchain consensus mechanisms, and categorized them into four types, namely crash fault tolerance (CFT) consensus mechanisms, Byzantine fault tolerance (BFT) consensus mechanisms, proof of something (PoX) series consensus mechanisms, and hybrid consensus mechanisms. Finally,

we developed ideas and suggestions for selecting concerns algorithms under the business process scenario.

The remainder of this paper is organized as follows: Section 2 introduces the basic concepts underlying blockchain; Section 3 presents and analyzes some significant blockchain consensus mechanisms; Section 4 compares consensus mechanisms from different perspectives and explores the ideas and principles for selecting consensus mechanisms in the blockchain-enabling business process; and, finally, Section 5 presents the conclusion of the paper as well as future research directions.

2. Basic Concept of Blockchain

What is blockchain? In a narrow sense, it is an open and shared distributed ledger or database [34]. In a broad sense, it is an entirely new infrastructure and distributed computing paradigm. Blockchain is decentralized and uses encrypted and linked data structures to verify and store data, utilizing distributed nodes and a consensus mechanism to generate and update the data [35,36]. Notably, it is based on automated script code, such as smart contracts, to program and manipulate data. The term blockchain hints at the unique structure of blockchain. Whenever a participant in a blockchain adds some data, such as a transaction or a record, it creates a new block. The blocks are chronologically and sequentially stored, starting the chain. Taking the Bitcoin block, for example, the structure of the blockchain is shown in Figure 1.

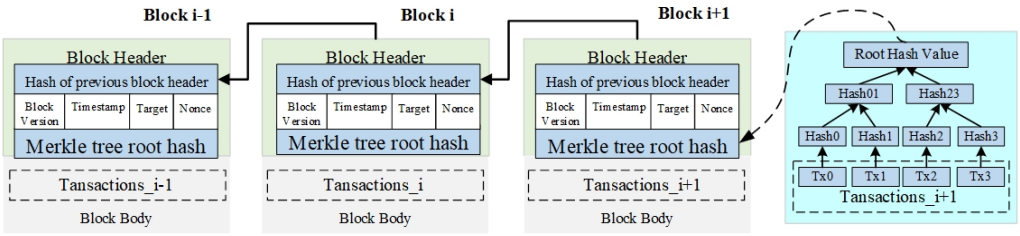

Figure 1. The blockchain structure.

Two main components, namely, the block header and the block body, are included in each block. In particular, the block body uses the Merkel tree data structure to store transaction data packed by miners, and the block header stores some nontransaction data, which are listed as follows:

- Block version: indicates which set of block validation rules to follow;
- The hash of the previous block header: a 256-bit hash value that used to implement links between blocks by pointing to the last block;
- Merkle tree root hash: records the hash values of all transactions in the block;
- Timestamp: used to record the creation time of blocks to ensure that they are created within a limited time;
- Target: current hashing target in a compact format;
- Nonce: a 4-byte field that usually starts with zero and increases for every hash calculation.

As an analogy to the OSI seven-layer model, the basic architecture of a blockchain typically consists of six layers: data, network, consensus, incentive, contract, and application. The basic blockchain architecture is shown in Figure 2.

- The data layer encapsulates the blockchain's underlying encryption technology and data storage method;
- The network layer involves the distributed peer-to-peer network and the transport mechanisms required to connect and operate among network nodes;
- The consensus layer includes various consensus mechanisms, which are combined with incentive mechanisms to achieve data consistency among nodes;
- The contract layer is a programmable implementation of blockchain technology;

- The application layer draws support from the underlying technology to implement various application scenarios and cases.

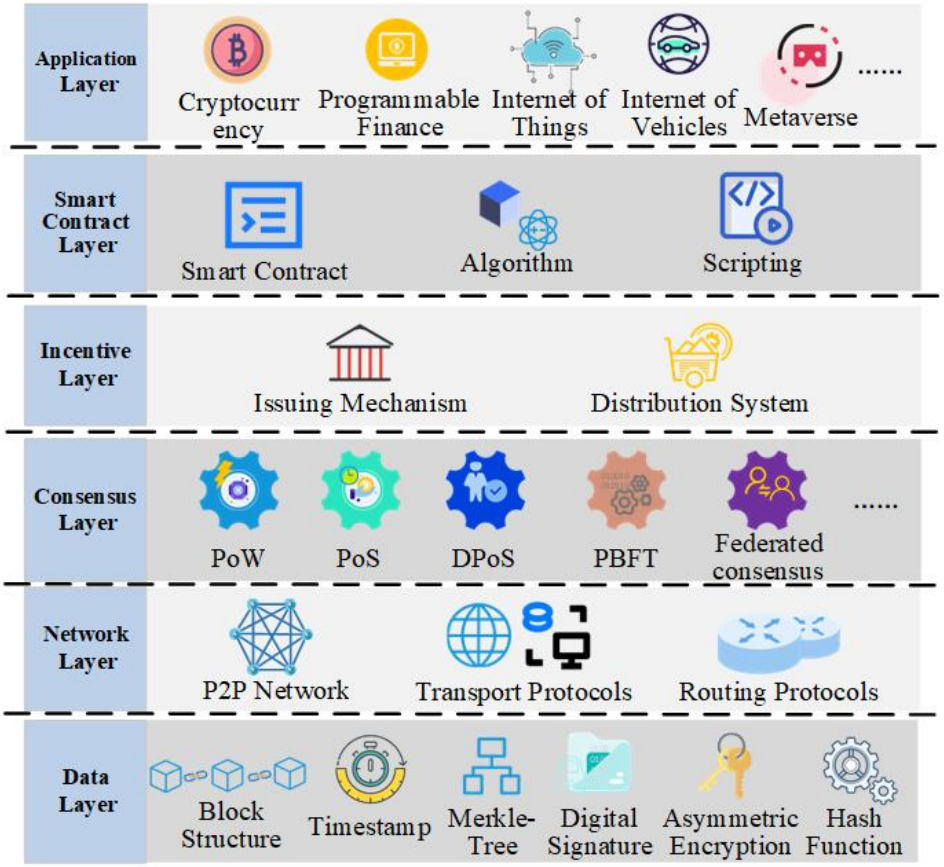

Figure 2. The basic architecture of blockchain.

There are four main types of blockchain: public, private, consortium, and hybrid. Each type has its own advantages and disadvantages that primarily drive its ideal uses. The four types of blockchains are shown in Figure 3.

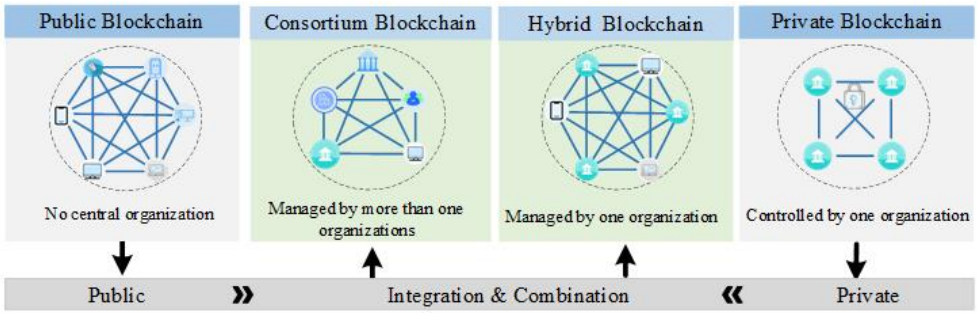

Figure 3. The four types of blockchain.

A public blockchain is a decentralized blockchain platform that connects unrelated parties. Also considered a permissionless blockchain, anybody can participate in the network's operation, consensus, and data validation, but nobody can alter the information once put on-chain. The verification of transactions in a public blockchain follows a predefined consensus, such as proof of work (PoW) in Bitcoin [1], proof of stake (PoS) in PPCoin [37], and delegated proof of stake (DPoS) in Bitshare [38]. This type of blockchain is perfect for the industry that can benefit from increased transparency and a shared source of truth.

The most important use of public blockchains is for cryptocurrencies, such as Bitcoin [1], Litecoin [2], NameCoin [3], and many other altcoins.

A single organization controls a private blockchain. It allows entries of only verified participants, with the central operator having the right to override, edit, or delete entries as required. This is very similar to a distributed database. It belongs to a completely centralized, permissioned blockchain. This type of blockchain, suitable for an organization that intends to leverage the technology to revamp its internal operations, can be used as the public blockchain and consortium blockchain test chain. Examples of private blockchains are Hyperledger Fabric [7] and Corda [9].

A consortium blockchain is managed by more than one organization. It is semidecentralized and belongs to a permissioned blockchain. Examples of consortium blockchains are Quorum [8], Tendermint [39], and Energy Web Foundation [40].

Hybrid blockchains are similar to consortium blockchains, but the difference between them is striking. A hybrid blockchain combines private and public blockchains, allowing users to seamlessly integrate a private blockchain with several public ones. Unlike consortium blockchains with multiple participants collectively helping to support the network, a hybrid blockchain can have a single entity network administrator. Hybrid blockchains work better when an organization needs some of its data to be open access and others kept private for in-house use. Examples of hybrid blockchains are IBM Food Trust [41], Dragonchain [42], and the Ripple network [43].

Different blockchains with different designs show differences in the security, accessibility, and sustainability of the underlying blockchain. A comparison of blockchains is provided in Table 1.

Table 1. The comparison of different types of blockchain.

Property	Public	Private	Consortium	Hybrid
Definition	Open to everyone	Controlled by one authority	Managed by two or more individuals, companies, or business outfits	Managed by one authority with some permissionless processes
Permission	Permissionless	Permissioned	Permissioned (Pre-selected nodes)	Permissioned (Public and Private)
Decentralized	Yes	No	Partial	Partial
Efficiency	Low	High	High	High
Consensus Determination	All miners	One authority	Pre-selected participants	One authority
Consensus Mechanisms	PoW, PoS, DPoS	PBFT	PBFT, Raft	PBFT
Auditability	Trustable	Lack of auditability	Trustable	Trustable
Energy	Energy consumption	More environmental	More environmental	More environmental

3. Blockchain Consensus Mechanisms

3.1. Origin

The concept of consensus starts with distributed systems. The critical problem addressed by distributed consensus mechanisms is implementing certainty and consensus to return reliable consensus results across the entire distributed network. Before the introduction of blockchain, the consensus mechanisms to solve the problem of consistency and consensus of the distributed system could be considered classic distributed consensus mechanisms. The traditional distributed consensus mechanisms and related technologies have laid a solid theoretical foundation for blockchain consensus mechanisms. The theoretical bases of blockchain consensus mechanisms are shown in Figure 4.

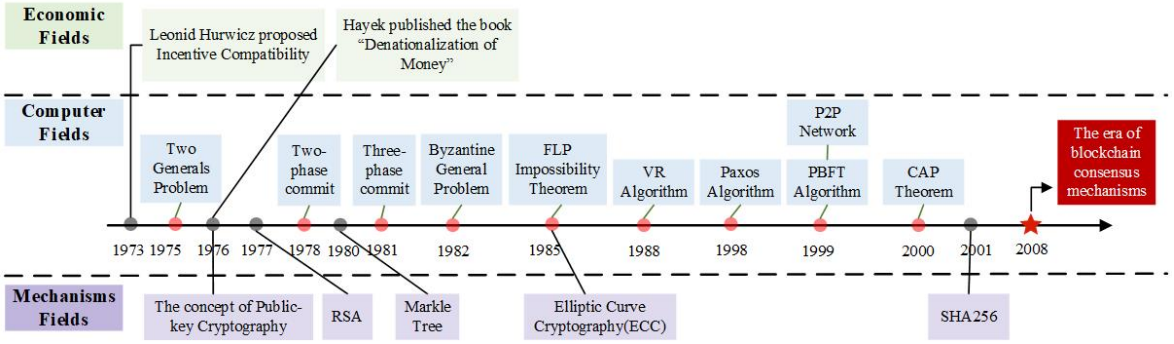

Figure 4. The theoretical bases of blockchain consensus mechanisms.

The main theoretical foundations of blockchain consensus mechanisms come from different fields, such as distributed systems from computer science, cryptography from mathematics, and game theory from economics.

The incentive mechanism of blockchain technology conforms to incentive compatibility theory [44]. This theory was proposed by the economist Leonid Hurwicz in 1973. Its main idea is to admit the selfishness of human nature and then achieve agreement between individual and collective interests with the help of a particular mechanism. The incentive mechanism provides motivation and guarantees for reaching a consensus. Subsequently, the economist Hayek published the book Denationalization of Money in 1976, considered the ideological source of digital currency [45].

Blockchain technology uses a large number of cryptographic technologies, such as key exchange [46], RSA encryption [47], elliptic curve cryptography [48], elliptical surface digital signature algorithm [49], SHA256 hash function [50], and Merkle tree data structure [51]. These technologies provide technical support for the blockchain consensus mechanism and ensure data security.

From the perspective of a distributed system, the key achievements of consensus mechanisms, such as the two generals Pproblem [52]; the two-phase commit protocol (2PC) [53], the three-phase commit protocol (3PC) [54]; the Byzantine generals problem [55]; the Fisher, Lynch, and Paterson (FLP) impossibility theorem [56]; the wiewstamped replication (VR) [57]; the Paxos [29]; the PBFT [31]; and the consistency availability partition tolerance (CAP) theorem [58], have laid a foundation for the development of blockchain consensus mechanisms.

A. The Byzantine Generals Problem

The Byzantine generals problem originated from Lamport's paper [55] in 1982. This problem describes how to make honest generals reach a consensus when there are a certain number of malicious generals. Lamport provided two algorithms to solve this problem. One is a solution with an oral message, and the other is a solution with a signed message. Although these two algorithms are difficult to use owing to their high communication complexity, they paved the way for the emergence of practical Byzantine fault tolerance (PBFT).

Based on the Byzantine generals problem, the faults in a distributed network can be divided into two categories: crash and Byzantine faults.

- Crash faults are also called non-Byzantine faults. They arise from errors such as delay and loss and do not cause other malicious actions. This type of fault is the most basic and common type to be solved in distributed systems. The consensus mechanisms tackling this type of fault only are called crash fault tolerance (CFT) consensus mechanisms;
- Byzantine faults can cause malicious actions, such as deliberately delaying messages and deceiving other nodes. This type of fault is more complex to address. The consensus mechanisms tackling this type of fault are called Byzantine fault tolerance (BFT) consensus mechanisms.

The main difference between BFT and CFT is the assumption of an adversary model. A universal conclusion proposed in Lamport's paper [55] is that if the number of traitors is f, then the total number of generals we need is at least $3f + 1$ if we want to reach a consistent decision in a BFT consensus mechanism. In a CFT consensus mechanism, the total number of generals we need is at least $2f + 1$.

B. FLP Impossibility Theorem

Fisher, Lynch, and Paterson proposed the FLP impossibility theorem in 1985 [56], covering three aspects of properties: safety, liveness, and fault tolerance.

- Safety: the values reached by nodes in a distributed system are consistent and valid;
- Liveness: the nodes in a distributed system must reach an agreement in a bounded time;
- Fault tolerance: a system must also be effective in case of node failures.

The FLP impossibility theorem indicates that no consensus mechanism can simultaneously meet the three properties in an asynchronous network. Because node failures are almost inevitable in a distributed system, fault tolerance must be considered. Therefore, only one of liveness or safety can be chosen. However, this is only a theoretical assumption, and if some constraints can be relaxed, the practical and feasible solutions from the engineering field can be found.

C. CAP Theorem

Brewer first advanced the CAP Theorem during a talk on distributed computing in 2000 and was later defined by Gilbert et al. in 2002 [58]. CAP theorem indicates that a distributed system can ensure only two of the following desired characteristics: consistency, availability, and partition tolerance. The CAP theorem is very similar to the FLP impossibility theorem, though they are not the same. In practice, some tradeoffs based on actual business scenarios are needed to achieve improvements.

Above all, ensuring the system's consistency is relatively straightforward when there are only crash faults. Researchers have developed relatively mature algorithms to solve this problem, such as Paxos [29], Raft [30], and other mechanisms applied to centralized distributed database systems. However, in the blockchain system, individuals or groups have different identities behind every node. To maximize their interests, they may perform some malicious actions. Therefore, the blockchain system must often consider the Byzantine fault tolerance problem. For this reason, much research on improving consensus mechanisms has been conducted. Notably, PoW in Bitcoin has pushed consensus mechanisms in a new direction. Since then, the era of blockchain consensus mechanisms had arrived.

Based on BFT and CFT, the BFT consensus mechanisms can be further divided into classical BFT consensus mechanisms, proof of something (PoX) series consensus mechanisms, and hybrid consensus mechanisms. Some popular blockchain consensus mechanisms are shown in Figure 5.

CFT consensus mechanisms are based on traditional distributed systems and only apply to non-Byzantine scenarios, such as downtime and network delay. These mechanisms' advantage, such as Paxos and Raft, is their high performance, but their disadvantage is poor fault tolerance. Again, these mechanisms must be modified before they can be applied to the blockchain.

BFT consensus mechanisms were developed from traditional distributed consensus mechanisms and are suitable for Byzantine scenarios. There are three ways to deal with Byzantine faults. The first one uses the classical BFT consensus mechanisms, mainly used in permissioned blockchains, allowing a certain number of malicious nodes in the system to reach consensus, for example, PBFT. The second one uses the PoX series consensus mechanisms, which are specific to the blockchain, mainly applying to the fully decentralized permissionless blockchain and reaching consensus by increasing the cost of publishing a block and relaxing the requirements for final consensus confirmation, e.g., PoW. The third one is using the hybrid consensus mechanisms. These consensus mechanisms combine the two kinds of mechanisms mentioned above.

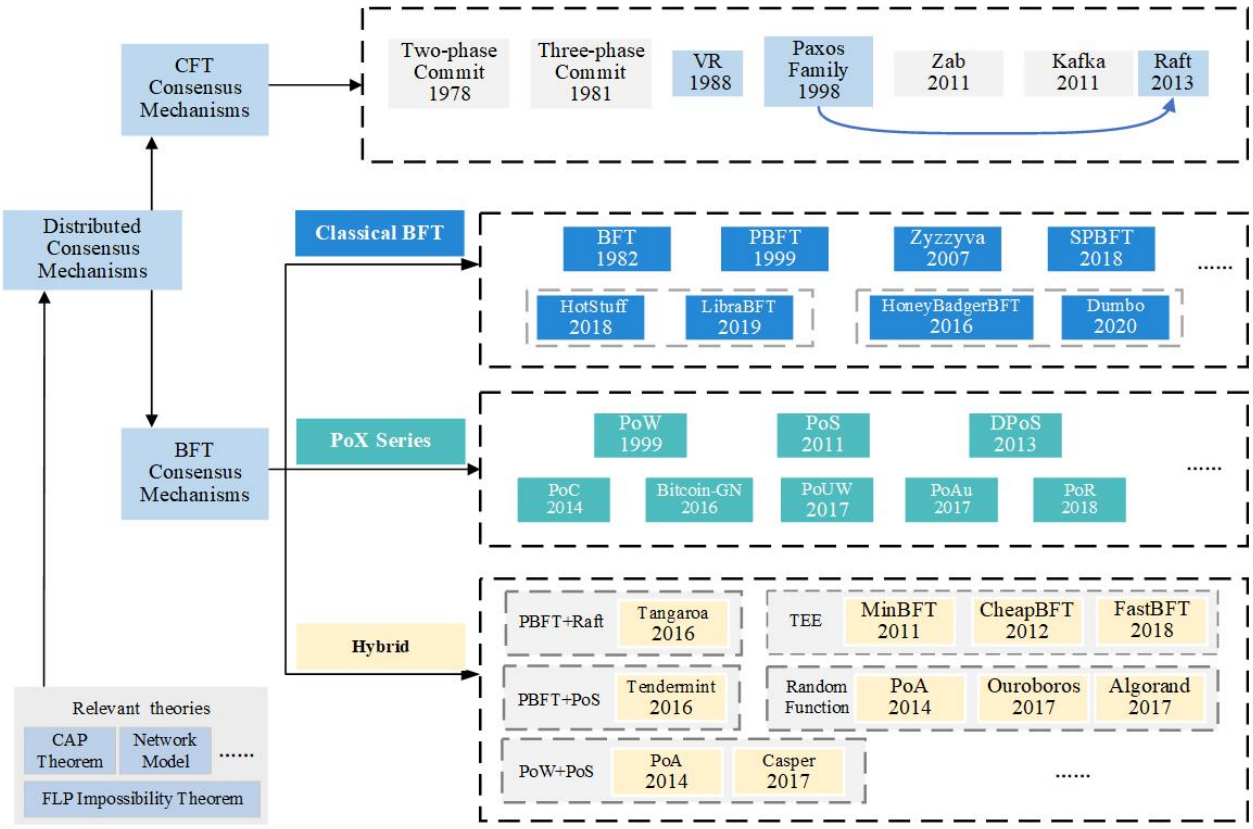

Figure 5. Some important blockchain consensus mechanisms.

The basis of the classification of different types of consensus mechanisms and their characteristics are shown in Table 2.

Table 2. The basis of classification of different types of consensus mechanisms and their characteristics.

Type of Consensus Mechanism	Basis of Classification	Characteristics
CFT consensus mechanisms	Only apply to non-Byzantine scenarios	High performance but poor fault tolerance
Classical BFT consensus mechanisms	Suitable for Byzantine scenarios and mainly used in permissioned blockchain	Deterministic consensus
PoX series consensus mechanisms	Suitable for Byzantine scenarios and mainly used in permissionless blockchain	Probabilistic consensus
Hybrid consensus mechanisms	Combine two or more kinds of consensus mechanisms	Can leverage the strengths of each mechanism while mitigating their weaknesses, resulting in a more robust and resilient system

3.2. CFT Consensus Mechanisms

CFT consensus mechanisms can only handle crash faults. A CFT mechanism cannot guarantee the system's reliability in blockchain scenarios. Thus, CFT consensus mechanisms are mainly used in closed environments.

3.2.1. VR

The VR mechanism was updated in 2010 [59] but was initially proposed in 1988 [57]. According to several famous concepts, theories, and theorems in distributed systems, such as FLP, CAP, and network models, some assumptions and constraints were put forward about its running environment to ensure that the consensus mechanism can run smoothly. These assumptions and constraints are briefly summarized below.

Liveness and Safety Environment Assumptions and Constraints of VR Mechanism:

- VR works in an asynchronous network;
- VR handles crash faults but does not handle Byzantine faults;
- VR satisfies $n = 2f + 1$ adversary model.

The VR mechanism is based on the primary copy technique [60]. Expressed in professional terms, it is a primary backup approach or a passive replication mode [61]. One feature of the VR mechanism is that it uses the primary, just one of the replicas, to order client requests; the other replicas are backups. The primary has the authority and responsibility to determine the order of client requests, and the backups need to accept the order selected by the primary. When the primary replica fails, the VR mechanism performs a process called view change to select a new primary for the system. The process of choosing the primary is realized by voting. If the failed one is not the primary one, there is no need to enter the phase of reselecting the primary node. When the replica is unable as either the primary or backup, VR provides a way to recover and continue; its consistency process in the normal case is shown in Figure 6.

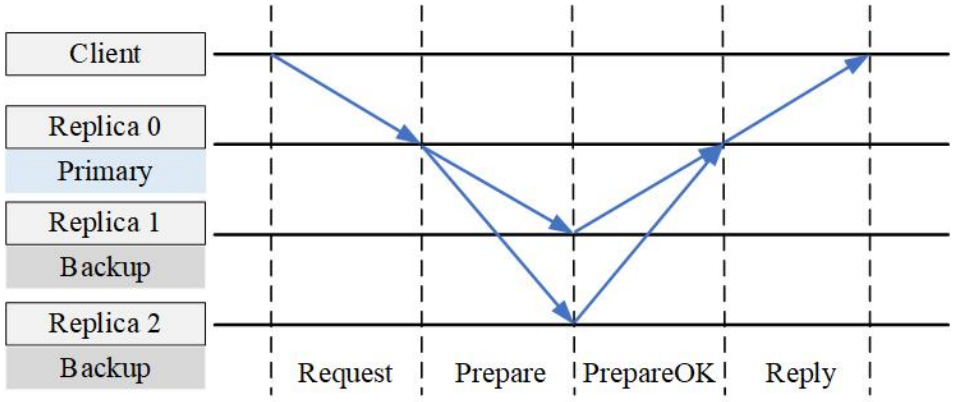

Figure 6. The consistency process of the VR mechanism in a normal case.

VR provides three subalgorithms to ensure correctness: normal operation, view changes, and recovery. The normal case protocol specifies that the operation of the VR mechanism works when the primary is not faulty. The view changes protocol is used to mask the failures of the primary. The recovery protocol enables the failed replicas to return to a normal state and correctly rejoin the group. VR largely extends the idea of the earlier work on two-phase commit [59].

3.2.2. Paxos

With Lambert's publication on the part-time parliament in 1998, a Greek island named Paxos became famous, which is not only a fictitious place name but also the name of a consensus mechanism [29]. Paxos is a state machine replication protocol. The original presentation of the Paxos mechanism used Paxos as a metaphor to describe the process of passing resolutions on Paxos Island. As a mechanism for the fault tolerance of distributed systems, Paxos is difficult to understand. In 2001, Lamport republished a simple description in Paxos Made Simple [62]. Later, the Paxos mechanism was improved in multiple versions, forming a Paxos-style consensus mechanism family, including Basic Paxos, Multi Paxos [63], and Cheap Paxos [64], among others.

The Environment Assumptions and Constraints of Paxos are the same as those of VR:

- Paxos works in an asynchronous network;
- Paxos handles crash faults but does not handle Byzantine faults;
- Paxos satisfies the $n = 2f + 1$ adversary model.

The Paxos mechanism includes three roles: proposers, acceptors, and learners. The role of the proposer is to provide a proposal or request. The proposal information includes

a proposal ID and the value of the proposal. The acceptors' function is to participate in decision making and respond to proposers' proposals. If most acceptors accept the proposal, it is said to be approved. The learners do not participate in decision making, as they learn the latest agreed proposal from proposers or acceptors. One node can play multiple roles. In Paxos, there are three stages.

Stage 1: Prepare stage. The proposer sends a prepared request to each acceptor, and the acceptors promise the received prepared request. In this stage, the proposal ID is accepted.

Stage 2: Accept stage. After the proposer receives more than half of the total number of acceptors, the proposer issues a proposal request to acceptors, who accept the received proposal request. When more than half of the acceptors in the system have the same proposal value, a consensus is reached. In this stage, the value of the proposal is accepted.

Stage 3: Learn stage. The proposer sends the formed resolution to all learners.

The flow of basic Paxos is shown in Figure 7.

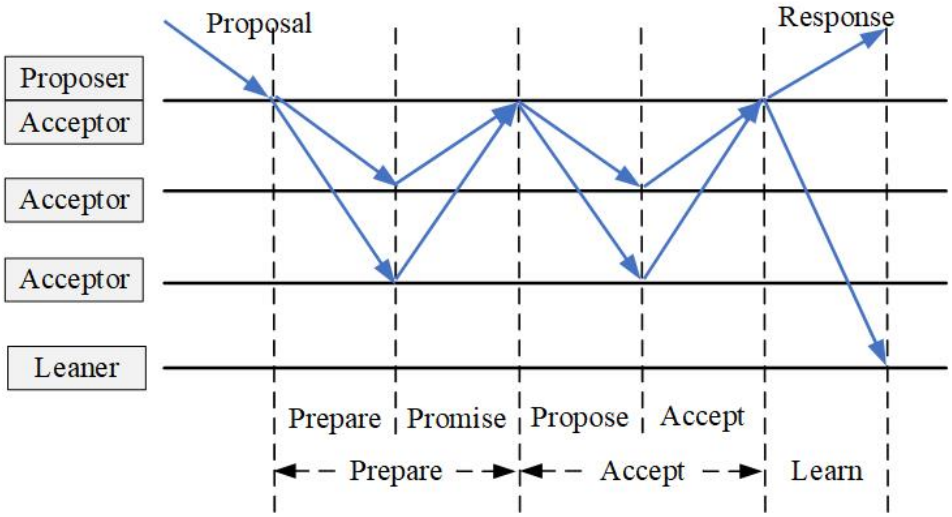

Figure 7. The flow of the Basic Paxos mechanism.

Basic Paxos mechanism can only form a decision on one request. Nevertheless, it is necessary to continuously determine multiple requests, and the processing should be efficient in practical applications. To solve this problem, Multi Paxos provides two improvements based on Basic Paxos. First, to determine each value, an instance of the Paxos mechanism is run to form a resolution. A unique instance ID identifies each Paxos instance. Second, a leader is elected from all proposers, and the leader uniquely submits the proposal to the acceptors for voting. In this way, the proposer has no competition, and the live lock problem is resolved. When there is only one leader in the system to submit the value, the preparation stage can be skipped; thus, system efficiency is improved. The flow of Multi Paxos is shown in Figure 8.

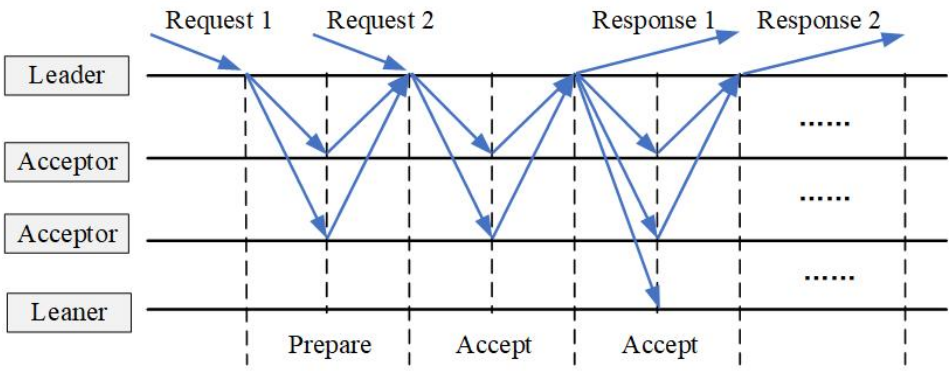

Figure 8. The flow of the Multi Paxos mechanism.

Multi Paxos needs to select a leader, and the determination of a leader is also the formation of a resolution. So, a Basic Paxos can be executed to elect a leader. After selecting a leader, only the leader can submit a proposal. If the leader cdrashes, the service is temporarily unavailable, so the leader needs to be reselected to continue the service. If only one leader in the system submits a proposal, the preparation stage can be skipped.

Multi Paxos changes the scope of the preparation stage to all instances submitted by the leader, so it only needs to execute the prepare stage once for continuous submission and perform the accept stage later. It changes the two stages into a single stage, improving efficiency. Each instance is identified by an instance ID that is locally and incrementally generated by the leader to distinguish multiple instances submitted consecutively.

3.2.3. Raft

Raft is a consensus mechanism proposed by Ongaro et al. in 2013 [30], which is similar to VR and Paxos. Compared with the Paxos algorithm, the mechanism structures are different. In terms of understandability, Raft is more friendly. It has many open-source implementations and is usually used for private networks such as IPFS Private Cluster [35].

Each server in the raft cluster has three states: leader, follower, or candidate. Like the primary in VR, the leader handles all client requests, while the follower passively responds, like the backup in VR. The system can only have one leader at any time. During normal operations, there are only a leader and followers. The candidate is used to select a new leader, and it is a temporary role.

Raft breaks down the problem of achieving consensus into three smaller and more manageable subproblems: leader election, log replication, and safety.

The concepts of term and heartbeat mechanisms are introduced in the leader election stage. Term acts as a logical clock in Raft. Each term begins with an election. Raft ensures that there is only one leader in a given term. The heartbeat mechanism is used to trigger the leader's election. If a follower receives communication within a period of time, the follower considers that there is no feasible leader and starts to elect a new leader. The leader's election process in raft is shown in Figure 9.

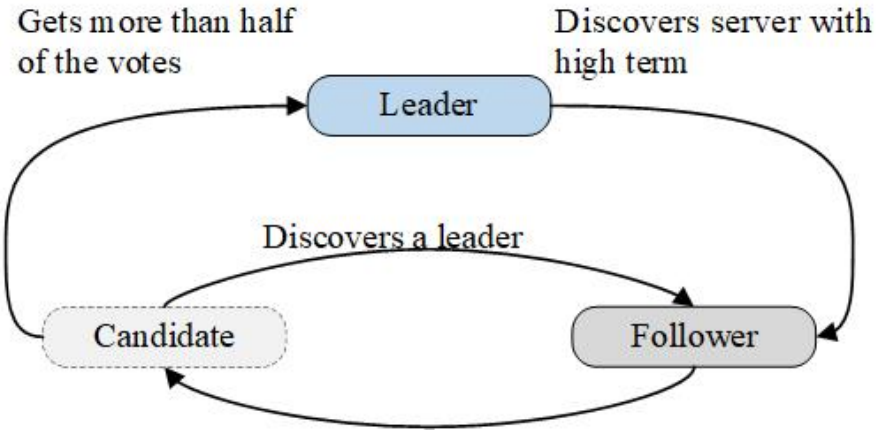

Figure 9. The leader's election process in Raft.

When a leader is elected, it enters the log replication stage, and all requests from the client requests are sent to the leader, which schedules the order of the concurrent requests. The leader informs the followers of these requests and the execution order. Then, the leader and followers execute these requests in the same order. To ensure consistent status between the leader and the followers, Raft uses a replication state machine, which can be understood as: the same starting state and inputs will produce the same output end in the same state. The process of Raft log replication is shown in Figure 10.

After receiving the request from the client, the leader adds the appeal to its log as log entries. Then, the leader sends replication log entries to followers in parallel, reaching a consensus when the leader receives more than half of the followers' feedback. The new

leader continues to work as a leader. If the system fails, the leader cannot access most of the followers, and the system automatically enters a leadership election.

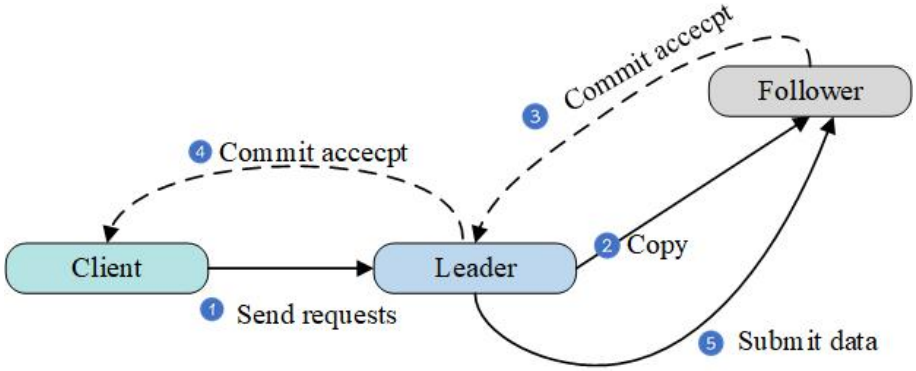

Figure 10. The process of Raft log replication.

3.3. Classical BFT Consensus Mechanisms

Byzantine faults are closer to real life, and there are three ways to deal with such faults. In this section, we introduce and analyze several representative classical BFT consensus mechanisms. VR, Paxos, and Raft are CFT consensus mechanisms. Because most of the classic BFT consensus mechanisms are affected by PBFT, the PBFT algorithm is reviewed first.

3.3.1. PBFT

Castro and Liskov first proposed the practical Byzantine fault tolerance (PBFT) algorithm in 1999 [55]. It is an extension of VR that allows the system to survive Byzantine faults based on cryptographic techniques, such as public-key signatures, message authentication codes, and message digests produced by collision-resistant hash functions [59]. It uses state machine replication techniques.

Liveness and Safety Environment Assumptions and Constraints of PBFT:

- PBFT works in asynchronous environments [55];
- PBFT handles Byzantine faults;
- PBFT satisfies the $n = 3f + 1$ adversary model.

In PBFT, nodes are divided into two categories: primary node and normal node. Like VR, PBFT uses a primary node to order client requests. However, unlike VR, the primary of the PBFT is elected in turn. Indeed, because the primary in PBFT might be lying, PBFT adds an extra phase before the preparation phase based on VR. The PBFT can be seen as a three-phase protocol. The three phases are pre-prepare, prepare, and commit. The phases of the PBFT are illustrated in Figure 11.

First, the client sends a request to the primary. Then the system starts the well-known three-phase commit consensus submission process. In the pre-prepare phase, the primary sends a pre-prepared message to other nodes called replicas. After receiving the pre-prepared message, the replica verifies it. If the message passes the verification, the replica accepts it; otherwise, it rejects the request. In the preparation phase, after the replica agrees to the request, it messages all other replicas in the network. This process is carried out in parallel by all the replicas. If a replica receives more than $2f + 1$ identical prepared messages within the specified time range, it enters the commit phase. In the commit phase, each replica sends commit messages to other replicas in the network. This process is also carried out in parallel by all replicas. When a replica receives more than $2f + 1$ identical commit messages, most replicas in the network have entered the commit phase. Now that consensus is reached, the replica executes the request and returns the result to the client.

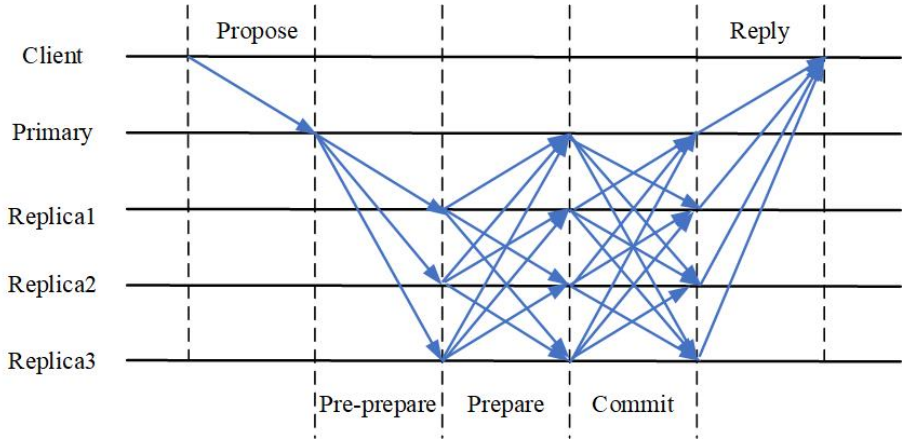

Figure 11. The phases of the PBFT.

In the pre-prepare and prepare phases, the PBFT ensures the order requests can be sent in the same view, even when the primary is faulty. In the prepare and commit phases, PBFT uses all-to-all communication to ensure that requests that commit are ordered across views. In each step of the PBFT, we trust the group, not the individuals.

3.3.2. Optimization Based on PBFT

The most significant disadvantage of PBFT is the high complexity of view change. With the rapid development of blockchain technology, various blockchain consensus mechanisms are constantly being proposed. These mechanisms are either variations of the original algorithm, improvements of PBFT and other mechanisms, or microinnovations made to improve performance in some aspects, combined with VRF, or combined with sharding technology, or mechanisms that make significant improvements to meet the needs of new scenarios.

In 2007, Zyzzyva [65] was proposed. Compared with PBFT, Zyzzyva adopts a more effective strategy when the primary node is honest, the network is in good condition, and the consensus is fast. Conversely, suppose the situation is the opposite. In that case, the consensus is slower than that of PBFT because it needs to run the algorithm according to the assumption that the situation is good before it can deal with negative situations.

MinBFT [66], CheapBFT [67], and FastBFT [68] were proposed in 2011, 2012, and 2018, respectively. These three mechanisms work in a partially synchronous network. They try to use the trusted execution environment (TEE) to solve the Byzantine problem. The consensus mechanism based on TEE uses TEE to support message processing. This method can simplify the BFT problem to a CFT problem. To prevent ambiguity by faulty replicas, MinBFT improves PBFT using a trusted counter service. In FastBFT, a tree structure communication mode is proposed, which reduces the node communication complexity and improves the algorithm's scalability. However, because Byzantine nodes can deliberately cause the replacement of members in tree structure communication mode, FastBFT depends on a relatively stable cluster environment, and the correctness of consensus mechanisms based on TEE depends on trusted hardware, so the application scenarios are limited.

In 2016, inspired by the PBFT and Raft mechanisms, Tangaroa [69] was proposed. Tangaroa has the Byzantine fault-tolerant capability and maintains Raft's safety, activity, simplicity, and understandability. In 2018, SBFT [70] was proposed. SBFT uses a linear communication mode using a collector and includes a fast path. This means that when all replicas are normal and synchronous, SBFT can enter the fast consensus mechanism. Again, SBFT reduces client communication overhead and adds redundant servers to improve resilience and performance.

In 2018, HotStuff [71,72] was proposed by VMware Research. It is a leader-based Byzantine fault-tolerant mechanism. It is similar to PBFT but a new stage is added to solve the hidden lock problem. Hotstuff adopts a threshold signature scheme. All nodes

do not directly broadcast the signed voting message but send the vote to the leader first. Then, the leader aggregates all signatures before posting to other nodes. This aggregation can reduce the time for signature verification. In 2019, a robust and efficient consensus mechanism based on HotStuff, called LibraBFT, i.e., the core of Libra Blockchain, was proposed by Facebook [73].

Asynchronous BFT consensus mechanisms have been designed to achieve consensus in a more complex network environment. The partially synchronous BFT consensus mechanisms perform well among the three network models. In 2016, HoneyBadgerBFT [74], the first practical asynchronous BFT consensus mechanism, was proposed. It introduces the concept of multi-ound random numbers so that every node can obtain a consensus result with high probability after repeating the consensus of multiple rounds. In HoneyBadgerBFT, there is no master node, and each node can initiate a proposal in their consensus. The content submitted in each consensus round comprises the final selected bid. Because the consensus of the asynchronous BFT consensus mechanism is repeated in multiple rounds, it has higher latency, especially in WANs. In 2020, Dumbo [75] was proposed. It showed that the leading cause of HoenyBadgerBFT's limited performance is the call to many randomized submodules. To overcome this limitation, Dumbo proposes a new, proven reliable broadcast primitive. More precisely, Dumbo ensures the correct completion of transaction broadcast through cryptographic facilities and provides an efficient construction method based on threshold signature schemes. In this way, the delay is significantly reduced, and the asynchronous BFT mechanism has a higher practical value.

3.4. PoX Series Consensus Mechanisms

This section mainly describes the PoX series consensus mechanisms used in the blockchain. The idea underlying these mechanisms is to use some scarce resources X so malicious attackers cannot retrieve X quickly. In this way, the system can remain safe in a decentralized and permissionless manner.

3.4.1. PoW

To solve the issue of consensus within a permissionless blockchain, Nakamoto employed an economic protocol known as PoW. The concept of PoW can be traced back to a paper published in 1992 [76], where solutions to the problem of spam emails were proposed. Its main idea is to require the e-mail sender to complete some computing tasks before sending e-mails. Subsequently, Adam Back proposed the concept of Hashcash in 1997 and updated it in 2002 [77], presenting it as a mechanism to stem the abuse of Internet resources. Satoshi Nakamoto cited Hashcash in his Bitcoin White Paper in 2008. Moreover, with the popularity and development of Bitcoin, to overcome the defects and shortcomings of PoW, researchers have proposed many mechanisms based on PoW. In this section, we introduce the PoW consensus mechanism and analyze its characteristics in combination with Bitcoin, which is the main application of blockchain and the most typical application of the PoW.

To record the blockchain transaction data on the blockchain and reach a consensus within a specific time, PoW provides an idea: all nodes in the blockchain network perform competitive accounting. This idea means that if nodes want to generate and write a new block into the blockchain, they must solve a complex puzzle that is easy to verify but difficult to find. Whichever node first solves the answer obtains the accounting rights. After that, the node obtaining the accounting rights broadcasts the solved answer and transaction record to other nodes for verification and begins the next mining round. If other node participants validate the transaction of the block and the answer to the puzzle is correct, it means that the answer is credible. The new block written into the verifier's blockchain, and the verifier enters the next round of competitive mining.

To further discuss this puzzle, we should focus on three elements: the workload proof function of mining, block, and difficulty value. The function is the calculation method of this puzzle. The block determines the input data of this problem, and the difficulty value

determines the required calculation amount for this puzzle. The process of random number search in the Bitcoin PoW consensus mechanism is shown in Figure 12.

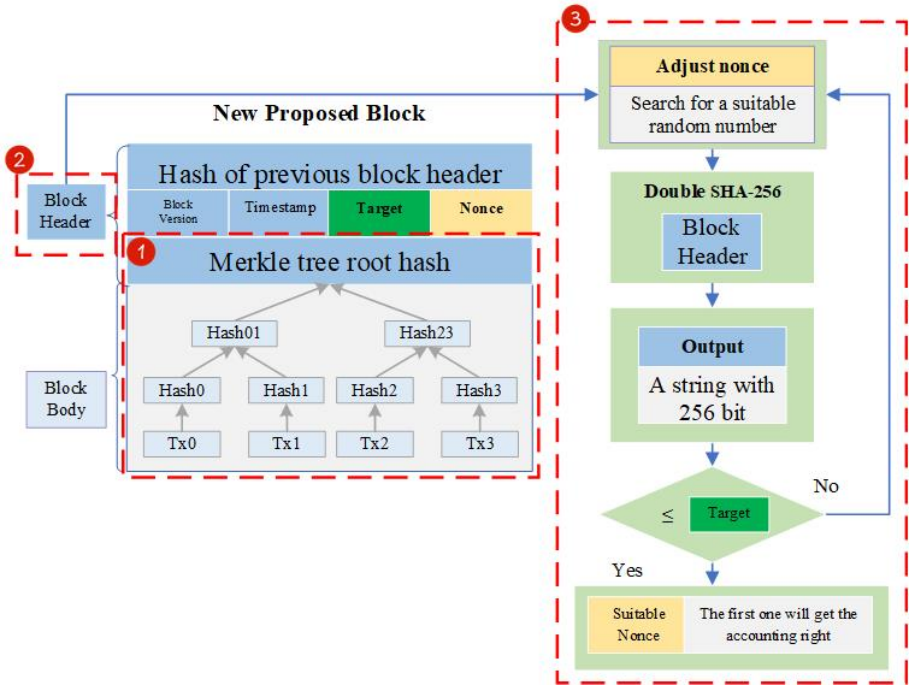

Figure 12. The process of random number search in Bitcoin PoW.

- Step 1: Generate Merkle root hash. The node generates a transaction and a Merkle root hash through the Merkle tree algorithm, with all other transactions to be packaged.
- Step 2: Construct the block header. The block header serves as an input parameter for solving the puzzle. The Merkle root hash calculated in the first step and other components of the block header are combined to form a complete block header.
- Step 3: Calculate the output of the puzzle. Bitcoin uses the SHA-256 algorithm, which belongs to the SHA-2 hash function series. Solving the puzzle requires finding a suitable nonce so that the hash value of the block header after double SHA-256 calculations is less than a certain number, called the target.

The above is the workload certification process of a single node. The whole process of the accounting stage of PoW consensus is as follows:

- The client generates a new transaction and broadcasts it to the whole network;
- Each node receives the request and enters the transaction into the block;
- Each node completes its workload proof;
- When a node finds a certificate, it broadcasts it to the whole network;
- Other nodes recognize the block's validity only when the block's transaction is valid and did not exist before;
- Accept the block and create a new block at the end of the block.

In short, PoW is a guarantee of value that is used to prove that the selected transaction record block is reliable and that the proof of work mechanism is trustworthy.

The PoW in Bitcoin follows the most extended chain principle: the longer chain is considered legal when there is a fork. We must always determine whether the current short or illegal chain will become the longest or legal chain in the future. Nevertheless, researchers think that if a block receive enough successor blocks, the certainty approaches 100%.

Except for the most extended chain principle, the PoW mechanism can be combined with many different principles. For example, Conflux is a fast, scalable, decentralized blockchain system. It uses blocks and edges to form a directed acyclic graph (DAG) instead

of a linear chain [78]. Conflux selects the legal chain based on the GHOST rule, which determines for each fork in the chain the heaviest subtree rooted at the fork [79].

3.4.2. Optimization Based on PoW

With the advancement of blockchain research, people pay attention to the shortcomings of PoW, such as performance, resource waste problems, and computing power centralization problems.

In terms of performance efficiency, in the Bitcoin PoW mechanism, miners compete to update the blockchain. Blocks are generated every 10 min in Bitcoin, and their size is limited to 1 MB. The transaction throughput is about seven transactions per second. Some consensus algorithms have been proposed to improve the performance of PoW, such as Litecoin [2], Ethereum [6], and Bitcoin-NG [80], among others. In Litecoin, blocks are generated every 2.5 min. In Ethereum, blocks are generated every 15 s, while in Ripple and Bitcoin-NG, blocks are divided into key and micro blocks. Ripple [5], a real-time gross settlement system and currency exchange network, primarily processes up to 1500 transactions per second (TPS) on its network by sacrificing a certain degree of decentralization.

Regarding resource waste, the existing improvement measures can be divided into two types: using valuable services to replace mining or using other capability certificates. One of the examples of proof of useful work (PoUW) [81], using valuable services to replace mining is PrimeCoin [4]. It is a cryptocurrency that originated from Bitcoin. It distinguishes itself from other cryptocurrencies by utilizing a novel proof-of-work system based on prime numbers and is the first one designed with scientific computing as its work. Examples of using other capability certificates are PoS [37], proof of capacity (PoC) [82], proof of authority (PoAu) [83], and proof of reputation (PoR) [84]. PoS uses the concept of virtual mining instead of mining. PoC is very similar to PoW. The main difference is that PoC uses storage instead of computing in PoW. PoAu and PoR select nodes with high authority and credibility as outgoing nodes. Because the block has the node's signature, if the node performs some malicious activities, it loses the qualification to update the block.

In terms of computing power centralization, with the rise of the price of Bitcoin, the equipment used for mining in the consensus has been upgraded from the early personal computers to GPUs and then further evolved into application-specific integrated circuit (ASIC) mining machines. This special mining equipment dramatically increases the cost of mining. At the same time, the mining environment has also evolved from single mining to cluster mining, and many large mining pools have also emerged. The increasingly centralized computing power has produced security problems, such as double-spend attacks and selfish mining. Memory-intensive functions have replaced the original functions because of the intensive computing characteristics of SHA-256 functions, which can reduce mining dependence on ASIC mining machines to a certain extent, such as Litecoin [2] and Ethereum [6]. In this way, we can avoid the problem of computing power centralization.

3.4.3. PoS

PoS is a consensus mechanism proposed to solve the enormous computing power consumption problem of PoW. In PoS, coin age is critical, which is defined as currency amount times holding period. For example, if you have 10 coins and keep them for 10 days, it means your coin age is 100. The first cryptocurrency that adopted PoS is PPcoin, proposed in 2012 by S. King and S. Nadal [37]. They introduced a timestamp field into each transaction to facilitate the computation of coin age. One day, if you spend all your coins, your coin age is cleared. That is, the coin age determines the difficulty of mining. Specifically, each node in Peercoin solves a PoW puzzle with its difficulty. The more consumed your coin age, the more the difficulty reduces. The blocks in PPcoin are separated into two different types. One is PoW blocks, and another is PoS blocks called coinstake. They represent special transaction paid to the stake owner. Through these special transactions, the node can generate a block and mint for PoS. The process of random number search in the PPcoin PoS consensus mechanism is shown in Figure 13.

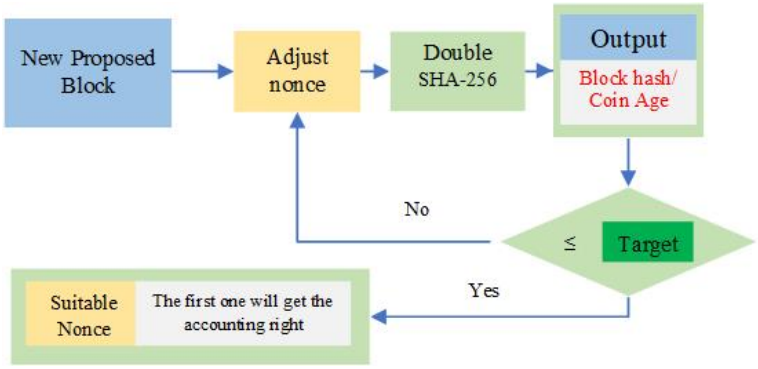

Figure 13. The process of random number search in PPcoin PoS consensus mechanism.

Compared with Bitcoin's PoW, PoS introduces a new minting process based on the consumed coin age in the coinstake transaction for PoS blocks. The main chain in PPcoin is the blockchain with the highest total consumed coin age. Once a node successfully obtains the right to add a block to the blockchain, its stake is cleared, and a new round of stake accumulation begins.

3.4.4. DPoS

The DPoS consensus mechanism was proposed in 2014 to solve the problems of PoW and PoS. A typical application of DPoS is Bitshare [38]. It adopts a decentralized democratic approach. In this approach, each coin is equivalent to one vote, and the holder can cast some of their votes to their trusted delegates. The system selects the top n nodes that receive the most significant number of votes as the system delegates. Their job is to produce blocks, and before each block is signed, it must verify that the trusted node has signed the previous block.

DPoS is divided into two parts:

(1) Stakeholders vote to elect a group of block producers;

(2) Block producers schedule production by turns.

Like PoW and PoS, DPoS is also a longest-chain-wins algorithm [85]. Whenever an honest node sees an effective longest chain, it switches from the current fork to the longest chain, making the longest chain longer and longer. However, unlike PoW and PoS, DPoS can still stably operate under most network conditions. The improvement from PoW to DPoS is shown in Figure 14.

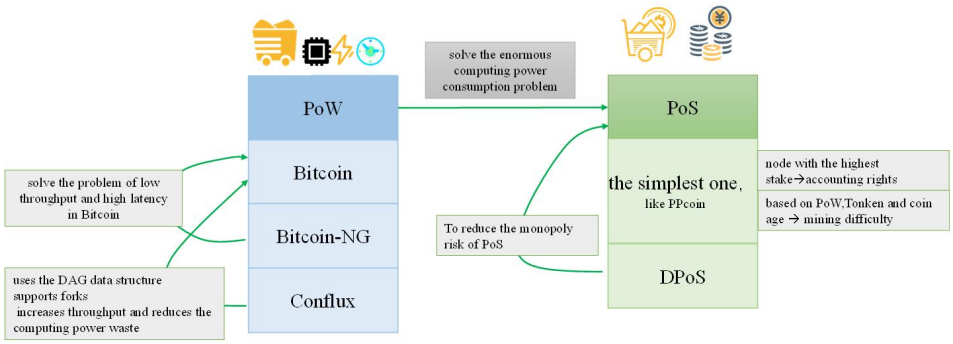

Figure 14. The improvement from PoW to DPoS.

3.4.5. Optimization Based on PoS

PoS alleviates the energy waste problem experienced with PoW. However, it is still a competitive consensus mechanism based on puzzle solving and can easily fork, including stake grinding, nothing-at-stake problem, and long-range attacks. To further solve the computational power waste, the PoS based on random function is proposed, such as Proof of Activity (PoA) [86], Ouroboros [87], and Algorand [88], among others. PoA was proposed in 2014. It is a blockchain consensus that combines the capabilities of the PoW

and PoS mechanisms. It begins with PoW and ends with PoS. However, while mining for a block, PoA introduces a follow-the-satoshi subroutine. Ouroboros was proposed in 2017. The purpose of Ouroboros is to select an accountant according to the stake, and this process is unpredictable. Therefore, no malicious attacks occur. Algorand is a blockchain network and project founded in 2017. Its most notable feature is the pure proof of stake (PPoS) consensus mechanism, randomly selecting validators weighted by their staked ALGO coin. The Byzantine protocol is used in Algorand's PPoS architecture but is optimized for high performance and large scales.

In addition to consensus mechanisms based on random functions, some hybrid consensus mechanisms have been proposed, such as Tendermint and Casper. Tendermint was proposed in 2016 [39]. Its purpose is to provide a consensus algorithm that is more effective and secure than Bitcoin's workload proof. Tendermint was built as a simple currency to participate in consensus in the early days. Users must bind a certain amount of currency to a margin account. If they misbehave, the money is recovered, which makes Tendermint a PoS algorithm. It is optimized for the traditional PBFT algorithm and only needs two voting rounds to reach a consensus. Tendermint includes two main technical components: Tendermint Core and Application BlockChain Interface (ABCI). Tendermint Core is a blockchain consensus engine that ensures that each machine records the same transaction in the same order, while ABCI ensures that any programming language can process transactions. Casper was proposed in 2017 [89]. It is a PoW and PoS mix algorithm. Its purpose is to enable Ethereum to smoothly convert to pure PoS. The Casper FFG algorithm consists of two penalty conditions, a fork choice rule, and a dynamic set of verification nodes.

4. Comparison and Selection

From Paxos and Raft to various variants of Paxos and Raft and BFT consensus mechanisms, blockchain consensus mechanisms have been developing, improving, and evolving. Based on the introduction of each type of consensus mechanism in Section 3, a summary of the advantages and disadvantages of the representative consensus mechanisms is shown in Table 3, and the comparison of consensus mechanisms is shown in Table 4.

Table 3. The advantages and disadvantages of representative consensus mechanisms.

Consensus Mechanism	Advantages	Disadvantages
Paxos	Low resource consumption. No tokens. Deterministic consensus.	Non-Byzantine fault tolerance. More difficult to understand and implement than Raft.
Raft	Low resource consumption. Raft's processes and descriptions are clear. Easier to understand and implement than Paxos. No tokens. Deterministic consensus.	Non-Byzantine fault tolerance.
PBFT	Low resource consumption. High throughput. High consensus efficiency. No tokens. Deterministic consensus.	High network complexity and low scalability. Limited number of nodes.
PoW	Decentralized. High security and high attack difficulty. High scalability and unlimited number of nodes.	High resource consumption. High hardware dependency. Long time to generate new blocks. Deterministic consensus and prone to fork.
PoS	Higher block generate efficiency and better than PoW. Lower resource consumption than PoW.	Centralization trend. Deterministic consensus.
DPoS	Lower resource consumption. High consensus efficiency.	Fairness lower than PoS. Dependent on tokens.

Table 4. The comparison of consensus mechanisms.

Mechanism	Energy Consumption	Blockchain Type	Mechanism Type	Network Model	Adversary Model
VR	Low	Private/Consortium	CFT	Asynchronous	$n = 2f + 1$
Paxos	Low	Private/Consortium	CFT	Asynchronous	$n = 2f + 1$
Raft	Low	Private/Consortium	CFT	Asynchronous	$n = 2f + 1$
PBFT	Low	Private/Consortium	Classical BFT	Asynchronous	$n = 3f + 1$
Zyzzyva	Low	Private/Consortium	Classical BFT	Partially Synchronous	$n = 3f + 1$
MinBFT	Low	Private/Consortium	Classical BFT	Partially Synchronous	$n = 2f + 1$
CheapBFT	Low	Private/Consortium	Classical BFT	Partially Synchronous	$n = f + 1$
FastBFT	Low	Private/Consortium	Classical BFT	Partially Synchronous	$n = f + 1$
Tangaroa	Low	Private/Consortium	PBFT + Raft	Partially Synchronous	$n = 3f + 1$
SBFT	Low	Private/Consortium	Classical BFT	Partially Synchronous	$n = 3f + 1$
HotStuff	Low	Private/Consortium	Classical BFT	Partially Synchronous	$n = 3f + 1$
LibraBFT	Low	Private/Consortium	Classical BFT	Partially Synchronous	$n = 3f + 1$
HoneyBadgerBFT	Low	Private/Consortium	Classical BFT	Asynchronous	$n = 3f + 1$
Dumbo	Low	Private/Consortium	Classical BFT	Asynchronous	$n = 3f + 1$
PoW	Hight	Public	PoX Series	Partially Synchronous	$n = 2f + 1$
PoS	Medium	Public	PoX Series	Partially Synchronous	$n = 2f + 1$
DPoS	Low	Public	PoX Series	Partially Synchronous	$n = 2f + 1$
Proof of Capacity	Low	Public	PoX Series	Partially Synchronous	$n = 2f + 1$
Proof of Authority	Low	Public	PoX Series	Partially Synchronous	$n = 2f + 1$
Proof of Reputation	Low	Public	PoX Series	Partially Synchronous	$n = 2f + 1$
Proof of Activity	Low	Private/Consortium	Pow + PoS	Partially Synchronous	$n = 2f + 1$
Ouroboros	Low	Private/Consortium	PoS	Partially Synchronous	$n = 2f + 1$
Algorand	Low	Private/Consortium	BFT + PoS	Partially Synchronous	$n = 3f + 1$
Tendermint	Low	Private/Consortium	PBFT + PoS	Partially Synchronous	$n = 3f + 1$
Casper	Low	Private/Consortium	Pow + PoS	Partially Synchronous	$n = 3f + 1$

According to the classification in this work, the application scenarios of different types of consensus mechanisms are shown in Table 5.

Table 5. The application scenarios for different types of consensus mechanisms.

Type	Public	Private	Consortium	Hybrid
CFT	Poor performance	Suitable	Suitable	Suitable
Classical BFT	Sybil attack, poor performance	Suitable	Suitable	Suitable
PoX series	Suitable	Poor performance	Poor performance	Suitable
Hybrid	Suitable	Suitable	Suitable	Suitable

Consensus mechanisms based on CFT, such as Paxos and Raft, are designed to handle failures caused by crashed or offline nodes in a distributed system. That is, CFT consensus mechanisms can only run in very reliable network environments and are currently mainly used in closed distributed systems within enterprises where access is restricted and strictly controlled. In this way, the consensus mechanism can effectively eliminate the threat of malicious nodes and ensure that the consensus process runs smoothly and efficiently.

Classical BFT consensus mechanisms have stringent requirements on the network environment. Even in Byzantine scenarios, the entire system is secure only if the system satisfies the $n = 3f + 1$ adversary model. Additionally, the exact number of malicious nodes in a permissioned network is relatively easy to solve. For example, in a private network, the number of nodes that may act maliciously can be estimated. If some nodes behave maliciously, their real identities can be quickly located, improving the network's safety. Suppose this kind of consensus mechanism is applied to a permissionless blockchain. In that case, a Sybil attack [90] would be easy because attackers can forge identities to control many nodes and the entire distributed network. In addition, the unlimited number of nodes in the permissionless blockchain affects the system's performance. Therefore, the classical BFT consensus mechanisms generally apply to reliable networks that require permission control and have few nodes.

Unlike classical BFT consensus mechanisms, PoX series is mainly used for public blockchains.

In general, the choice of consensus mechanisms is strongly related to the application scenario. Paxos or Raft can be used in a trusted environment, PBFT can be used in a permissioned blockchain, and PoX series consensus mechanisms can be used in a permissionless blockchain. These traditional consensus mechanisms have also gained new vigor in the age of blockchain.

Many big companies have also developed distributed consensus mechanisms or have chosen existing ones to meet their business needs. The consensus mechanisms selected by mainstream blockchain platforms are shown in Table 6.

Table 6. The consensus mechanisms selected by mainstream blockchain platforms.

Blockchain Platform	Year	Underlying Consensus	Ledger Type	Smart Contract	Governance
Bitcoin [1]	2008	PoW	Permissionless	No	The Bitcoin Core developers
Ethereum1.0 [6]	2013	PoW	Permissionless	Yes	Ethereum Developers
Dragonchain [42]	2014	Context-based verification with five levels of consensus	Public, private, and hybrid	Yes	Dragonchain Foundation
Tezos [91]	2014	PoS	Public, private, and hybrid	Yes	Dynamic Ledger Solutions, Inc.
Quorum [8]	2015	Raft/IBFT	Permissioned	Yes	J.P. Morgan
Hyperledger Fabric [7]	2017	Pluggable	Permissioned	Yes	Linux Foundation
Fisco Bcos [92]	2017	PBFT/Raft	Permissioned	Yes	FISCO open-source working group
Hyperledger Sawtooth [93]	2018	PoET/PBFT/RAFT	Permissioned	Yes	Hyperledger and the Linux Foundation
Zilliqa [94]	2018	A scalable BFT protocol for consensus	Permissionless	Yes	The Zilliqa Team
R3 Corda [9]	2018	Pluggable	Permissioned	Yes	R3 Consortium
Libra [73]	2019	LibraBFT (based on Hotstuff)	Permissioned	Yes	Facebook
EOS.IO [95]	2018	DPoS+ABFT	Permissioned	Yes	EOSIO Core Arbitration Forum
Ethereum2.0 [96]	2022	PoS	Permissionless	Yes	Ethereum Developers

Different blockchain platforms use different consensus mechanisms referring to their unique application scenario. For example, platforms such as Bitcoin and Ethereum V1.0 have chosen the PoW consensus mechanism because PoW has high security and decentralization, but its energy consumption is high, and processing speed is relatively slow, making it suitable for commercial projects with small transaction volumes but high-security requirements.

On the other hand, some platforms, such as Tezos and Ethereum V2.0, have chosen the PoS consensus mechanism because PoS is more efficient in terms of energy consumption and processing speed than PoW, is suitable for commercial projects with large transaction volumes, but has relatively low-security requirements.

In addition, some platforms, such as EOS, have chosen the DPoS consensus mechanism because DPoS can provide higher throughput and lower latency, so are suitable for commercial projects with very high transaction volumes that require high throughput and low latency.

Some guidelines to follow when selecting consensus mechanisms, especially in the era of artificial intelligence, are listed as follows:

- Security: Security is a critical consideration when selecting a consensus mechanism. The mechanism should be designed to prevent attacks, such as double spending and 51% attacks, and should be resistant to collusion and other forms of manipulation.
- Scalability: Scalability is another important consideration, particularly for AI applications that require high throughput and low latency. The consensus mechanism should be able to handle a large number of transactions per second and should be able to scale as the network grows.
- Energy efficiency: Energy efficiency is becoming an increasingly important consideration, as the energy consumption of blockchain networks can be significant. The consensus mechanism should be designed to minimize energy consumption while providing high security and scalability.
- Decentralization: Decentralization is a core principle of blockchain technology, and the consensus mechanism should be designed to promote decentralization and prevent the concentration of power in a small number of nodes.
- Consensus finality: Consensus finality refers to the degree to which a consensus mechanism can guarantee that a transaction has been successfully processed and recorded on the blockchain. For AI applications requiring high reliability and accuracy levels, consensus finality is a critical consideration.
- Consensus speed: Consensus speed refers to the time it takes for the network to reach consensus on a transaction. For AI applications that require real-time processing, consensus speed is an important consideration.

Overall, the selection of a consensus mechanism should be based on a careful evaluation of the specific requirements in the era of artificial intelligence. Developers and businesses should consider factors such as security, scalability, energy efficiency, decentralization, consensus finality, and consensus speed when selecting a consensus mechanism for their blockchain-based AI application.

For specific scenarios, such as finance, logistics, and healthcare, the ideas of consensus mechanism selection for specific scenarios are listed as follows:

- In the finance industry, where transactions must be quickly and securely processed, consensus protocols that prioritize transaction speed and security, such as PoS, can be more appropriate.
- In the logistics industry, where transparency and traceability are key, consensus protocols that prioritize decentralization and transparency, such as PoA or DPoS, can be more appropriate.
- In the healthcare industry, where data privacy and security are of utmost importance, consensus protocols that prioritize security and privacy (e.g., PBFT combines privacy

protection technologies such as zero-knowledge proof and federated learning) can be more appropriate.

Furthermore, take the business intelligence field, for example. Blockchain technology has become an effective way to innovate and improve business processes [97]. Before analyzing how to choose consensus for the business process, we should determine the relationship between blockchain and business process. It should perfectly match the enterprise's business process to achieve twice the result with half the effort.

Using the business process characteristics presented in [97], the relationship between blockchain and business process is shown in Table 7.

Table 7. The relationship between blockchain and the business process.

Business Process Characteristics	Role of Blockchain
Transient vs. persistent	Offers a trusted repository for quality of service (QoS) information
Dynamic vs. static	Offers smart contract transactions and permission mechanisms to solve trust problems in cross-organizational business processes
Formation vs. enactment	Offers a trusted repository for quality of service (QoS) information
Centralization vs. decentralization	Establishes trust among untrusted partners

The blockchain is a trusted infrastructure to guarantee the trust of collaborations among multiple partners in trustless business process environments. The diagram integrating blockchain technology into the business process is shown in Figure 15.

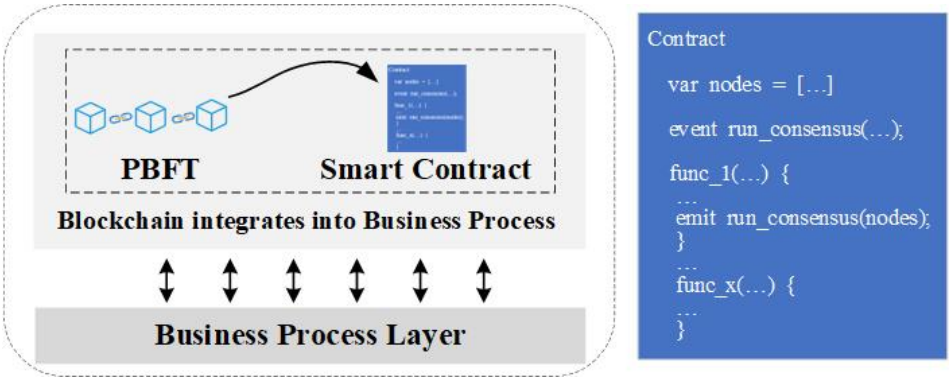

Figure 15. The relationship of blockchain and business process.

The architecture is divided into the business process layer, and the blockchain integrates into the business process layer. Business processes usually involve multiple organizations, and the time and trust of the nodes that perform consensus processes are the focus. Smart contracts help the system in selecting nodes to implement consensus.

Because it combines the characteristics of various consensus mechanisms, PBFT is chosen for its high efficiency, which is practical for business process operations and especially for time-critical ones. Conversely, PoW is impractical because it is unreliable and has high latency.

Specifically, in the blockchain integrated into the business process layer, business processes are encoded into smart contracts. The introduction of the smart contract in Figure 15 is as follows:

- The variable named nodes in the smart contract refers to the nodes to execute the consensus.
- Any function "func_x()" with the keyword consensus tells the blockchain system about the nodes' information, and the system should ensure that only authorized nodes can execute the consensus.

- The keywords "emit" and "event" are responsible for recording consensus operations in the blockchain, such as recording the leaders of nodes in each round, as part of the PBFT. Other nodes besides the leaders of nodes are responsible for performing validation and chaining up the verified new block without errors.

5. Conclusions and Future Work

Bitcoin has made blockchain technology rapidly develop and become a technology enabling several applications in many scenarios. Meanwhile, the PoW in Bitcoin pushed consensus mechanisms onto a new track. Although consensus mechanisms have been constantly developed and improved, they still have room for further improvement and exploration. This study focused on distributed consensus mechanisms' development, analyzing the traditional and blockchain consensus mechanisms, and comparing them from different perspectives. The future research directions of the blockchain consensus mechanisms can be outlined as follows:

- Different scenarios require blockchains with other characteristics, but different blockchains also need to be connected. Focusing on cross-chain technology and providing users with more comprehensive and efficient services could be the directions for further study.
- The traditional BFT consensus mechanisms cannot dynamically manage nodes. Although BFT consensus mechanisms have had many improvements and many variants have been created, such mechanisms mostly focus on guaranteeing consensus consistency and enhancing availability. There needs to be more research on how to dynamically manage nodes. Take the PBFT algorithm as an example. For adding or deleting nodes, this algorithm needs to shut down all nodes, update the configuration file, and restart all nodes. However, such behavior is unacceptable in real production environments. The way to solve the dynamic node management problem of BFT consensus mechanisms represents a challenge to be solved before applying blockchain technology in practical scenarios.
- In the future, the development of quantum computers will weaken the blockchain's cryptographic foundations. From a cryptographic perspective, we can design a system that matches the security requirements arising from quantum computing. From the consensus perspective, we can develop consensus mechanisms that do not require competition.

In addition to the field of blockchain consensus, the next step also expands the focus to other key technologies of blockchain, such as smart contracts, privacy protection, scalability, and so on. The specific description is as follows:

- In terms of smart contracts, the optimization, and expansion of smart contracts, such as how to improve the execution efficiency and security of smart contracts and how to support more programming languages, could be the focus of future study.
- In terms of privacy protection, how to utilize encryption and other privacy protection technologies to enhance the privacy protection capabilities of blockchain could be studied.
- Regarding scalability, some new technologies and solutions have been proposed, such as sharding technology, side chains, lightning networks, etc. How to further optimize these technologies to improve the performance and scalability of blockchain networks could be analyzed.

Funding: This work was partially supported by the National Natural Science Foundation of China under grant 62072170, the Science and Technology Project of the Department of Communications of Hunan Provincial under grant 202101, the Key Research and Development Program of Hunan Province under grant 2022GK2015, the Hunan Provincial Natural Science Foundation of China under grant 2021JJ30141, and SERICS project grant PE00000014 under the NRRP MUR program funded by the EU-NGEU.

Data Availability Statement: Data supporting this systematic review are contained within the paper. In addition, the data are available in the original publications, reports, and preprints that were cited in the reference section.

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

References

1. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System. Decentralized Business Review. 2008. p. 21260. Available online: https://bitcoin.org/en/bitcoin-paper (accessed on 27 October 2022).
2. Tu, Z.; Xue, C. Effect of bifurcation on the interaction between Bitcoin and Litecoin. *Financ. Res. Lett.* **2019**, *31*, 382–385. [CrossRef]
3. Gupta, A.; Chaudhary, B.; Dwivedi, P. *A Comprehensive Study on Namecoin*; Technical report; EasyChair: Greater Manchester, UK, 2022.
4. King, S. Primecoin: Cryptocurrency with Prime Number Proof-of-Work. 2013, Volume 1. Available online: https://primecoin.io/primecoin-paper.pdf (accessed on 27 October 2022).
5. Schwartz, D.; Youngs, N.; Britto, A. The ripple protocol consensus algorithm. *Ripple Labs Inc White Pap.* **2014**, *5*, 151. Available online: https://ripple.com/files/ripple_consensus_whitepaper.pdf (accessed on 27 October 2022).
6. Wood, G. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Proj. Yellow Pap.* **2014**, *151*, 1–32. Available online: https://ethereum.github.io/yellowpaper/paper.pdf (accessed on 27 October 2022).
7. Androulaki, E.; Barger, A.; Bortnikov, V.; Cachin, C.; Christidis, K.; De Caro, A.; Enyeart, D.; Ferris, C.; Laventman, G.; Manevich, Y.; et al. Hyperledger fabric: A distributed operating system for permissioned blockchains. In Proceedings of the Thirteenth EuroSys Conference, Porto, Portugal, 23–26 April 2018; pp. 1–15.
8. Baliga, A.; Subhod, I.; Kamat, P.; Chatterjee, S. Performance evaluation of the quorum blockchain platform. *arXiv* **2018**, arXiv:1809.03421.
9. Brown, R.G. The corda platform: An introduction. *Retrieved* **2018**, *27*, 2018. Available online: https://corda.net/content/corda-platform-whitepaper.pdf (accessed on 27 October 2022).
10. Sunarya, P.A.; Williams, A.; Khoirunisa, A.; Bein, A.S.; Sari, D.M. A blockchain based online business intelligence learning system. *Blockchain Front. Technol.* **2021**, *1*, 87–103. [CrossRef]
11. Yang, R.; Wakefield, R.; Lyu, S.; Jayasuriya, S.; Han, F.; Yi, X.; Yang, X.; Amarasinghe, G.; Chen, S. Public and private blockchain in construction business process and information integration. *Autom. Constr.* **2020**, *118*, 103276. [CrossRef]
12. Liang, W.; Zhang, D.; Lei, X.; Tang, M.; Li, K.C.; Zomaya, A.Y. Circuit copyright blockchain: Blockchain-based homomorphic encryption for IP circuit protection. *IEEE Trans. Emerg. Top. Comput.* **2020**, *9*, 1410–1420. [CrossRef]
13. Chen, L.; Yu, Q.; Liang, W.; Cai, J.; Zhu, H.; Xie, S. Overview of Medical Data Privacy Protection based on Blockchain Technology. In Proceedings of the 2022 IEEE 7th International Conference on Smart Cloud (SmartCloud), Shanghai, China, 8–10 October 2022; pp. 200–205.
14. Liang, W.; Xiao, L.; Zhang, K.; Tang, M.; He, D.; Li, K.C. Data fusion approach for collaborative anomaly intrusion detection in blockchain-based systems. *IEEE Internet Things J.* **2021**, *9*, 14741–14751. [CrossRef]
15. Liang, W.; Tang, M.; Long, J.; Peng, X.; Xu, J.; Li, K.C. A secure fabric blockchain-based data transmission technique for industrial Internet-of-Things. *IEEE Trans. Ind. Inform.* **2019**, *15*, 3582–3592. [CrossRef]
16. Meng, X.; Xu, J.; Liang, W.; Xu, Z.; Li, K.C. A lightweight anonymous cross-regional mutual authentication scheme using blockchain technology for internet of vehicles. *Comput. Electr. Eng.* **2021**, *95*, 107431. [CrossRef]
17. Xu, Z.; Liang, W.; Li, K.C.; Xu, J.; Jin, H. A blockchain-based roadside unit-assisted authentication and key agreement protocol for internet of vehicles. *J. Parallel Distrib. Comput.* **2021**, *149*, 29–39. [CrossRef]
18. Gadekallu, T.R.; Huynh-The, T.; Wang, W.; Yenduri, G.; Ranaweera, P.; Pham, Q.V.; da Costa, D.B.; Liyanage, M. Blockchain for the metaverse: A review. *arXiv* **2022**, arXiv:2203.09738.
19. Das, D.; Bose, P.; Ruaro, N.; Kruegel, C.; Vigna, G. Understanding security issues in the NFT ecosystem. In Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, Los Angeles, CA, USA, 7–11 November 2022; pp. 667–681.
20. Zhang, S.; Yao, T.; Arthur Sandor, V.K.; Weng, T.H.; Liang, W.; Su, J. A novel blockchain-based privacy-preserving framework for online social networks. *Connect. Sci.* **2021**, *33*, 555–575. [CrossRef]
21. Zheng, Z.; Xie, S.; Dai, H.N.; Chen, X.; Wang, H. Blockchain challenges and opportunities: A survey. *Int. J. Web Grid Serv.* **2018**, *14*, 352–375. [CrossRef]
22. Gao, N.; Han, D.; Weng, T.H.; Xia, B.; Li, D.; Castiglione, A.; Li, K.C. Modeling and analysis of port supply chain system based on Fabric blockchain. *Comput. Ind. Eng.* **2022**, *172*, 108527. [CrossRef]
23. Vukolić, M. The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication. In Proceedings of the Open Problems in Network Security: IFIP WG 11.4 International Workshop, iNetSec 2015, Zurich, Switzerland, 29 October 2015; Revised Selected Papers; pp. 112–125.
24. Zheng, Z.; Xie, S.; Dai, H.; Chen, X.; Wang, H. An overview of blockchain technology: Architecture, consensus, and future trends. In Proceedings of the 2017 IEEE International Congress on Big Data (BigData Congress), Honolulu, HI, USA, 25–30 June 2017; pp. 557–564.

25. Viriyasitavat, W.; Hoonsopon, D. Blockchain characteristics and consensus in modern business processes. *J. Ind. Inf. Integr.* **2019**, *13*, 32–39. [CrossRef]
26. Biswas, S.; Sharif, K.; Li, F.; Maharjan, S.; Mohanty, S.P.; Wang, Y. PoBT: A lightweight consensus algorithm for scalable IoT business blockchain. *IEEE Internet Things J.* **2019**, *7*, 2343–2355. [CrossRef]
27. Xu, J.; Lin, J.; Liang, W.; Li, K.C. Privacy preserving personalized blockchain reliability prediction via federated learning in IoT environments. *Clust. Comput.* **2022**, *25*, 2515–2526. [CrossRef]
28. Wang, Y.; Peng, H.; Su, Z.; Luan, T.H.; Benslimane, A.; Wu, Y. A platform-free proof of federated learning consensus mechanism for sustainable blockchains. *IEEE J. Sel. Areas Commun.* **2022**, *40*, 3305–3324. [CrossRef]
29. Lamport, L. The part-time parliament. In *Concurrency: The Works of Leslie Lamport*; ACM Transactions on Computer Systems; ACM: New York, NY, USA, 2019; pp. 277–317. Available online: https://dl.acm.org/doi/pdf/10.1145/279227.279229 (accessed on 27 October 2022).
30. Ongaro, D.; Ousterhout, J. In search of an understandable consensus algorithm. In Proceedings of the 2014 USENIX Annual Technical Conference (USENIX ATC'14), Philadelphia, PA, USA, 19–20 June 2014; pp. 305–319.
31. Castro, M.; Liskov, B. Practical byzantine fault tolerance. In Proceedings of the OsDI, New Orleans, LO, USA, 22–25 February 1999; Volume 99, pp. 173–186.
32. Nguyen, G.T.; Kim, K. A survey about consensus algorithms used in blockchain. *J. Inf. Process. Syst.* **2018**, *14*, 101–128.
33. Fu, X.; Wang, H.; Shi, P. A survey of Blockchain consensus algorithms: Mechanism, design and applications. *Sci. China Inf. Sci.* **2021**, *64*, 1–15. [CrossRef]
34. Merrad, Y.; Habaebi, M.H.; Elsheikh, E.A.; Suliman, F.E.M.; Islam, M.R.; Gunawan, T.S.; Mesri, M. Blockchain: Consensus Algorithm Key Performance Indicators, Trade-Offs, Current Trends, Common Drawbacks, and Novel Solution Proposals. *Mathematics* **2022**, *10*, 2754. [CrossRef]
35. Liang, W.; Yang, Y.; Yang, C.; Hu, Y.; Xie, S.; Li, K.C.; Cao, J. PDPChain: A consortium blockchain-based privacy protection scheme for personal data. *IEEE Trans. Reliab.* **2022**, 1–13. [CrossRef]
36. Liang, W.; Fan, Y.; Li, K.C.; Zhang, D.; Gaudiot, J.L. Secure data storage and recovery in industrial blockchain network environments. *IEEE Trans. Ind. Inform.* **2020**, *16*, 6543–6552. [CrossRef]
37. King, S.; Nadal, S. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. *Self-Publ. Pap. August* **2012**, *19*. Available online: https://www.peercoin.net/read/papers/peercoin-paper.pdf (accessed on 27 October 2022).
38. Larimer, D. Delegated Proof-of-Stake (dpos), Bitshare Whitepaper, 3. 2019. Available online: https://whitepaper.io/document/388/bitshares-whitepaper (accessed on 27 October 2022).
39. Buchman, E. Tendermint: Byzantine Fault Tolerance in the Age of Blockchains. Ph.D. Thesis, University of Guelph, Guelph, ON, Canada, 2016.
40. Bronski, P.; Creyts, J.; Gao, S.; Hambridge, S.; Hartnett, S.; Hesse, E.; Morris, J.; Nanavatty, J.; Pennington, N. The Decentralized Autonomous Area Agent (D3A) Market Model; Energy Web Foundation: Berlin, Germany, 2018.
41. Howson, P. Building trust and equity in marine conservation and fisheries supply chain management with blockchain. *Mar. Policy* **2020**, *115*, 103873. [CrossRef]
42. Laha, T.; Bandyopadhyay, A.; Deb, K.; Koley, S. A Methodical Study on Blockchain Technology. In *Proceedings of the International Conference on Network Security and Blockchain Technology: ICNSBT 2021*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 391–403.
43. Dolgui, A.; Ivanov, D.; Sokolov, B. Ripple effect in the supply chain: An analysis and recent literature. *Int. J. Prod. Res.* **2018**, *56*, 414–430. [CrossRef]
44. Hurwicz, L. The design of mechanisms for resource allocation. *Am. Econ. Rev.* **1973**, *63*, 1–30.
45. Yokoyama, Y. From money to culture: The practical indeterminacy of Bitcoin's values and temporalities. *Econ. Anthropol.* **2023**, *10*, 32–43. [CrossRef]
46. Diffie, W.; Hellman, M.E. New directions in cryptography. In *Democratizing Cryptography: The Work of Whitfield Diffie and Martin Hellman*; Association for Computing Machinery: New York, NY, USA, 2022; pp. 365–390.
47. Rivest, R.L.; Shamir, A.; Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **1978**, *21*, 120–126. [CrossRef]
48. Koblitz, N. Elliptic curve cryptosystems. *Math. Comput.* **1987**, *48*, 203–209. Available online: https://www.ams.org/journals/mcom/1987-48-177/S0025-5718-1987-0866109-5/S0025-5718-1987-0866109-5.pdf (accessed on 27 October 2022).
49. Johnson, D.; Menezes, A.; Vanstone, S. The elliptic curve digital signature algorithm (ECDSA). *Int. J. Inf. Secur.* **2001**, *1*, 36–63. [CrossRef]
50. Debnath, S.; Chattopadhyay, A.; Dutta, S. Brief review on journey of secured hash algorithms. In Proceedings of the IEEE 2017 4th International Conference on Opto-Electronics and Applied Optics (Optronix), Kolkata, India, 2–3 November 2017; pp. 1–5.
51. de Ocáriz Borde, H.S. An Overview of Trees in Blockchain Technology: Merkle Trees and Merkle Patricia Tries. 2022. Available online: https://www.researchgate.net/publication/358740207_An_Overview_of_Trees_in_Blockchain_Technology_Merkle_Trees_and_Merkle_Patricia_Tries (accessed on 27 October 2022).
52. Akkoyunlu, E.A.; Ekanadham, K.; Huber, R.V. Some constraints and tradeoffs in the design of network communications. In Proceedings of the Fifth ACM Symposium on Operating Systems Principles, Austin, TX, USA, 19–21 November 1975; pp. 67–74.
53. Gray, J.; Lamport, L. Consensus on transaction commit. *ACM Trans. Database Syst. (TODS)* **2006**, *31*, 133–160. [CrossRef]

54. Singh, P.; Yadav, P.; Shukla, A.; Lohia, S. An extended three phase commit protocol for concurrency control in distributed systems. *Int. J. Comput. Appl.* **2011**, *975*, 8887. [CrossRef]
55. Lamport, L.; Shostak, R.; Pease, M. The Byzantine generals problem. In *Concurrency: The Works of Leslie Lamport*; Transactions on Programming Languages and Systems; ACM: New York, NY, USA, 2019; pp. 203–226.
56. Fischer, M.J.; Lynch, N.A.; Paterson, M.S. Impossibility of distributed consensus with one faulty process. *J. ACM (JACM)* **1985**, *32*, 374–382. [CrossRef]
57. Oki, B.M.; Liskov, B.H. Viewstamped replication: A new primary copy method to support highly-available distributed systems. In Proceedings of the Seventh Annual ACM Symposium on Principles of Distributed Computing, Toronto, ON, Canada, 15–17 August 1988; pp. 8–17.
58. 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]
59. Liskov, B. From viewstamped replication to Byzantine fault tolerance. In *Replication: Theory and Practice*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 121–149.
60. Budhiraja, N.; Marzullo, K.; Schneider, F.B.; Toueg, S. The primary-backup approach. *Distrib. Syst.* **1993**, *2*, 199–216.
61. Van Renesse, R.; Schiper, N.; Schneider, F.B. Vive la différence: Paxos vs. viewstamped replication vs. zab. *IEEE Trans. Dependable Secur. Comput.* **2014**, *12*, 472–484. [CrossRef]
62. Lamport, L. Paxos made simple. In *ACM SIGACT News (Distributed Computing Column) 32, 4 (Whole Number 121, December 2001)*; ACM: New York, NY, USA, 2001; pp. 51–58.
63. Chandra, T.D.; Griesemer, R.; Redstone, J. Paxos made live: An engineering perspective. In Proceedings of the Twenty-Sixth Annual ACM Symposium on Principles of Distributed Computing, Portland, OR, USA, 12–15 August 2007; pp. 398–407.
64. Lamport, L.; Massa, M. Cheap paxos. In Proceedings of the IEEE International Conference on Dependable Systems and Networks, Florence, Italy, 28 June–1 July 2004; pp. 307–314.
65. Kotla, R.; Alvisi, L.; Dahlin, M.; Clement, A.; Wong, E. Zyzzyva: Speculative byzantine fault tolerance. In *Proceedings of the Twenty-First ACM SIGOPS Symposium on Operating Systems Principles*; ACM: New York, NY, USA, 2007; pp. 45–58.
66. Veronese, G.S.; Correia, M.; Bessani, A.N.; Lung, L.C.; Verissimo, P. Efficient byzantine fault-tolerance. *IEEE Trans. Comput.* **2011**, *62*, 16–30. [CrossRef]
67. Kapitza, R.; Behl, J.; Cachin, C.; Distler, T.; Kuhnle, S.; Mohammadi, S.V.; Schröder-Preikschat, W.; Stengel, K. CheapBFT: Resource-efficient Byzantine fault tolerance. In Proceedings of the 7th ACM European Conference on Computer Systems, Bern, Switzerland, 10–13 April 2012; pp. 295–308.
68. Liu, J.; Li, W.; Karame, G.O.; Asokan, N. Scalable byzantine consensus via hardware-assisted secret sharing. *IEEE Trans. Comput.* **2018**, *68*, 139–151. [CrossRef]
69. Copeland, C.; Zhong, H. Tangaroa: A Byzantine Fault Tolerant Raft. 2016. Available online: https://www.scs.stanford.edu/14au-cs244b/labs/projects/copeland_zhong.pdf (accessed on 27 October 2022).
70. Gueta, G.G.; Abraham, I.; Grossman, S.; Malkhi, D.; Pinkas, B.; Reiter, M.; Seredinschi, D.A.; Tamir, O.; Tomescu, A. Sbft: A scalable and decentralized trust infrastructure. In Proceedings of the 2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Portland, OR, USA, 24–27 June 2019; pp. 568–580.
71. Yin, M.; Malkhi, D.; Reiter, M.K.; Gueta, G.G.; Abraham, I. HotStuff: BFT consensus in the lens of blockchain. *arXiv* **2018**, arXiv:1803.05069.
72. Yin, M.; Malkhi, D.; Reiter, M.K.; Gueta, G.G.; Abraham, I. HotStuff: BFT consensus with linearity and responsiveness. In Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, Toronto, ON, Canada, 29 July–2 August 2019; pp. 347–356.
73. Baudet, M.; Ching, A.; Chursin, A.; Danezis, G.; Garillot, F.; Li, Z.; Malkhi, D.; Naor, O.; Perelman, D.; Sonnino, A. State machine replication in the libra blockchain. *Libra Assn. Tech. Rep.* **2019**, *7*, 1–21.
74. Miller, A.; Xia, Y.; Croman, K.; Shi, E.; Song, D. The honey badger of BFT protocols. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016; pp. 31–42.
75. Guo, B.; Lu, Z.; Tang, Q.; Xu, J.; Zhang, Z. Dumbo: Faster asynchronous bft protocols. In Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual, 9–13 November 2020; pp. 803–818.
76. Dwork, C.; Naor, M. Pricing via processing or combatting junk mail. In Proceedings of the Advances in Cryptology—CRYPTO'92: 12th Annual International Cryptology Conference, Santa Barbara, CA, USA, 16–20 August 1992; Proceedings 12; pp. 139–147.
77. Back, A. Hashcash-a Denial of Service Counter-Measure. 2002. Available online: http://www.hashcash.org/hashcash.pdf (accessed on 27 October 2022).
78. Li, C.; Li, P.; Zhou, D.; Xu, W.; Long, F.; Yao, A. Scaling nakamoto consensus to thousands of transactions per second. *arXiv* **2018**, arXiv:1805.03870.
79. Sompolinsky, Y.; Zohar, A. Secure high-rate transaction processing in bitcoin. In Proceedings of the Financial Cryptography and Data Security: 19th International Conference, FC 2015, San Juan, PR, USA, 26–30 January 2015; Revised Selected Papers 19; pp. 507–527.
80. 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 ({NSDI} 16), Santa Clara, CA, USA, 16–18 March 2016; pp. 45–59.

81. Hoffmann, F. Challenges of Proof-of-Useful-Work (PoUW). *arXiv* **2022**, arXiv:2209.03865.
82. Dziembowski, S.; Faust, S.; Kolmogorov, V.; Pietrzak, K. Proofs of space. In Proceedings of the Advances in Cryptology—CRYPTO 2015: 35th Annual Cryptology Conference, Santa Barbara, CA, USA, 16–20 August 2015; Proceedings, Part II; pp. 585–605.
83. Islam, M.M.; Merlec, M.M.; In, H.P. A comparative analysis of proof-of-authority consensus algorithms: Aura vs Clique. In Proceedings of the 2022 IEEE International Conference on Services Computing (SCC), Barcelona, Spain, 10–16 July 2022; pp. 327–332.
84. Gai, F.; Wang, B.; Deng, W.; Peng, W. Proof of reputation: A reputation-based consensus protocol for peer-to-peer network. In Proceedings of the Database Systems for Advanced Applications: 23rd International Conference, DASFAA 2018, Gold Coast, QLD, Australia, 21–24 May 2018; Proceedings, Part II 23; pp. 666–681.
85. Snider, M.; Samani, K.; Jain, T. Delegated proof of stake: Features & tradeoffs. *Multicoin Cap.* **2018**, *19*, 1–19.
86. Bentov, I.; Lee, C.; Mizrahi, A.; Rosenfeld, M. Proof of activity: Extending bitcoin's proof of work via proof of stake [extended abstract] y. *ACM Sigmetrics Perform. Eval. Rev.* **2014**, *42*, 34–37. [CrossRef]
87. Kiayias, A.; Russell, A.; David, B.; Oliynykov, R. Ouroboros: A provably secure proof-of-stake blockchain protocol. In Proceedings of the Advances in Cryptology—CRYPTO 2017: 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, 20–24 August 2017; Proceedings, Part I; pp. 357–388.
88. 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, Shanghai, China, 28 October 2017; pp. 51–68.
89. Buterin, V.; Griffith, V. Casper the friendly finality gadget. *arXiv* **2017**, arXiv:1710.09437.
90. Aggarwal, S.; Kumar, N. Attacks on blockchain. In *Advances in Computers*; Elsevier: Amsterdam, The Netherlands, 2021; Volume 121, pp. 399–410.
91. Doan, T.T.H.; Thiemann, P. Towards Contract Modules for the Tezos Blockchain (Short Paper). In Proceedings of the 3rd International Workshop on Formal Methods for Blockchains (FMBC 2021), Los Angeles, CA, USA, 18–19 July 2021; Schloss Dagstuhl-Leibniz-Zentrum für Informatik.
92. Zhang, J.; Yang, Y.; Zhao, D.; Wang, Y. A node selection algorithm with a genetic method based on PBFT in consortium blockchains. *Complex Intell. Syst.* **2022,** 1–21. Available online: https://link.springer.com/article/10.1007/s40747-022-00907-2 (accessed on 27 October 2022). [CrossRef]
93. Ampel, B.; Patton, M.; Chen, H. Performance modeling of hyperledger sawtooth blockchain. In Proceedings of the 2019 IEEE International Conference on Intelligence and Security Informatics (ISI), Shenzhen, China, 1–3 July 2019; pp. 59–61.
94. Secure, A. The Zilliqa Project: A Secure, Scalable Blockchain Platform. 2018. Available online: https://docs.zilliqa.com/positionpaper.pdf (accessed on 27 October 2022).
95. Dernayka, I.; Chehab, A. Blockchain development platforms: Performance comparison. In Proceedings of the IEEE 2021 11th IFIP International Conference on New Technologies, Mobility and Security (NTMS), Paris, France, 19–21 April 2021; pp. 1–6.
96. Cassez, F.; Fuller, J.; Asgaonkar, A. Formal verification of the ethereum 2.0 beacon chain. In Proceedings of the Tools and Algorithms for the Construction and Analysis of Systems: 28th International Conference, TACAS 2022, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2022, Munich, Germany, 2–7 April 2022; Proceedings, Part I; pp. 167–182.
97. Viriyasitavat, W.; Da Xu, L.; Niyato, D.; Bi, Z.; Hoonsopon, D. Applications of blockchain in business processes: A comprehensive review. *IEEE Access* **2022**, *10*, 118900–118925. [CrossRef]