

BLOCKCHAIN CONSENSUS METHOD AND APPARATUS, MEDIUM, ELECTRONIC DEVICE, AND PROGRAM PRODUCT

RELATED APPLICATION

[0001] This application claims priority to Chinese Patent Application No. 202210622140.7, entitled "BLOCKCHAIN CONSENSUS METHOD AND APPARATUS, MEDIUM, AND ELECTRONIC DEVICE" and filed with the China National Intellectual Property Administration on June 2, 2022, which is incorporated herein by reference in its entirety.

FIELD OF THE TECHNOLOGY

[0002] This application relates to the field of blockchain technologies, and specifically, to a blockchain consensus.

BACKGROUND OF THE DISCLOSURE

[0003] In a blockchain network including a plurality of blockchain nodes, to achieve a decentralized and trusted environment, all blockchain nodes generally need to execute a unified consensus algorithm to achieve consistent data synchronization between the nodes. However, it is difficult for the unified consensus algorithm to meet diverse system service requirements, and there is a problem of poor flexibility.

SUMMARY

[0004] This application provides a blockchain consensus method, a blockchain consensus apparatus, a computer-readable medium, an electronic device, and a computer program product, aiming to improve flexibility of a blockchain consensus.

[0005] Other characteristics and advantages of this application become apparent from the following detailed description, or may be learned in part through the practice of this application.

[0006] According to an aspect of embodiments of this application, a blockchain consensus method is provided. The method is performed by a current blockchain node in a blockchain network, and the method includes:

[0007] broadcasting consensus requests corresponding to at least two consensus stages in the blockchain network, the blockchain network including at least two blockchain nodes executing consensus algorithms of different types, the consensus algorithms of different types including a first consensus algorithm and a second consensus algorithm;

[0008] separately obtaining response messages broadcast by the blockchain network in the consensus stages, the response messages being messages from the blockchain nodes in response to the consensus requests;

[0009] counting a node quantity of blockchain nodes transmitting the response messages in a same consensus stage, the node quantity including a quantity of the at least two blockchain nodes executing the consensus algorithms of different types; and

[0010] in a case that the node quantity satisfies a consensus condition of the first consensus algorithm, reaching a consensus on a blockchain node executing the first consensus algorithm, and synchronizing a consensus result of the first consensus algorithm with a blockchain node executing the second consensus algorithm.

[0011] According to an aspect of the embodiments of this application, a blockchain consensus apparatus is provided, the apparatus including:

[0012] a request module, configured to broadcast consensus requests corresponding to at least two consensus stages in the blockchain network, the blockchain network including at least two blockchain nodes executing consensus algorithms of different types, the consensus algorithms of different types including a first consensus algorithm and a second consensus algorithm;

[0013] a response module, configured to separately obtain response messages broadcast by the blockchain network in the consensus stages, the response messages being messages from the blockchain nodes in response to the consensus requests;

[0014] a statistics module, configured to count a node quantity of blockchain nodes transmitting the response messages in a same consensus stage, the node quantity including a quantity of the at least two blockchain nodes executing the consensus algorithms of different types; and

[0015] a synchronization module, configured to: in a case that the node quantity satisfies a consensus condition of the first consensus algorithm, reach a consensus on a blockchain node executing the first consensus algorithm, and synchronize a consensus result of the first consensus algorithm with a blockchain node executing the second consensus algorithm.

[0016] According to an aspect of the embodiments of this application, a computer-readable medium is provided, storing a computer program, the computer program, when executed by a processor, implementing the blockchain consensus method according to the foregoing technical solution.

[0017] According to an aspect of the embodiments of this application, an electronic device is provided, the electronic device including: a processor; and a memory, configured to store executable instructions of the processor, the processor being configured to execute the executable instructions to perform the blockchain consensus method according to the foregoing technical solution.

[0018] According to an aspect of the embodiments of this application, a computer program product is provided, including a computer program, the computer program, when executed by a processor, implementing the blockchain consensus method according to the foregoing technical solution.

[0019] According to some embodiments of this application, the computer program product or the computer program includes a computer program, the computer program being stored in a computer-readable storage medium. A processor of a computer device reads the computer program from the computer-readable storage medium and executes the computer program to cause the computer device to perform the blockchain consensus method according to the technical solution.

[0020] In the technical solution provided in the embodiments of this application, by configuring at least two blockchain nodes executing consensus algorithms of different types in a blockchain network, a node quantity of blockchain nodes transmitting consensus responses may be counted in a process of performing a node consensus, and when the node quantity satisfies a consensus condition of any one of the consensus algorithms, a consensus result may be synchronized by a blockchain node that satisfies the consensus condition with a blockchain node that does not satisfy the consensus condition, thereby improving flexibility of a blockchain consensus.

[0021] It is to be understood that the foregoing general description and the following detailed description are merely for exemplary and explanatory purposes, and cannot limit this application.

BRIEF DESCRIPTION OF THE DRAWINGS

[0022] FIG. 1 is a schematic diagram of a composition of a blockchain system according to an embodiment of this application.

[0023] FIG. 2 shows a composition structure of a blockchain maintained in a blockchain network.

[0024] FIG. 3 exemplarily shows a network architecture of a blockchain network to which a technical solution of this application is applied.

[0025] FIG. 4 is a flowchart of steps of a blockchain consensus method according to an embodiment of this application.

[0026] FIG. 5 shows a data structure of a response message according to an embodiment of this application.

[0027] FIG. 6 shows a blockchain consensus method in which a node quantity is counted based on a consensus stage according to an embodiment of this application.

[0028] FIG. 7 shows a consensus process of a low fault tolerance consensus algorithm according to an embodiment of this application.

[0029] FIG. 8 shows a consensus process of a high fault tolerance consensus algorithm according to an embodiment of this application.

[0030] FIG. 9 shows a scenario example in a process of performing algorithm switching according to an embodiment of this application.

[0031] FIG. 10 is a flowchart of method steps of performing consensus authentication in an application scenario according to an embodiment of this application.

[0032] FIG. 11 is a schematic block diagram of a structure of a blockchain consensus apparatus according to an embodiment of this application.

[0033] FIG. 12 is a schematic block diagram of a structure of a computer system of an electronic device adapted to implement an embodiment of this application.

DESCRIPTION OF EMBODIMENTS

[0034] Now, exemplary implementations are described comprehensively with reference to the accompanying drawings. However, the examples of implementations may be implemented in various forms, and it should not be understood as being limited to the examples described herein. Conversely, the implementations are provided to make this application more comprehensive and complete, and comprehensively convey the idea of the examples of the implementations to a person skilled in the art.

[0035] In addition, the described features, structures, or characteristics may be combined in one or more embodiments in any appropriate manner. In the following description, a lot of specific details are provided to give a full understanding of the embodiments of this application. However, a person skilled in the art is to be aware of that, the technical solution in this application may be implemented without one or more of the particular details, or other methods, unit, apparatus, or step

may be adopted. In other cases, well-known methods, apparatuses, implementations, or operations are not shown or described in detail, to avoid obscuring the aspects of this application.

[0036] The block diagrams shown in the accompanying drawings is merely a functional entity and does not necessarily correspond to a physically independent entity. To be specific, such functional entities may be implemented in the form of software, or implemented in one or more hardware modules or integrated circuits, or implemented in different networks and/or processor apparatuses and/or microcontroller apparatuses.

[0037] The flowcharts shown in the accompanying drawings are merely examples for descriptions, do not necessarily include all content and operations/steps, and are not necessarily performed in the described orders. For example, some operations/steps may be further divided, while some operations/steps may be combined or partially combined. Therefore, an actual execution order may vary depending on an actual situation.

[0038] In a specific implementation of this application, related data such as a request, a response, and the like that are generated when a user uses a blockchain product is involved. When each embodiment of this application is applied to a specific product or technology, user permission or consent is required to be obtained, and relevant collection, use, and processing of data are required to comply with relevant laws, regulations, and standards of relevant countries and regions.

[0039] A blockchain is an anti-counterfeiting, tamper-proof, and traceable shared digital ledger with a chained-block data structure and built through a transparent and trustworthy rule in a peer-to-peer network environment. The chained-block data structure is a data structure that stores a transaction that occurs within a period of time in a unit of a block, and uses a cryptographic algorithm to connect blocks into a chain in chronological order. A ledger is distributed to all member nodes in a network, permanently recording a history of an asset transaction that have occurred between peer nodes in the network in a sequential chain of blocks linked by using a hash cryptographic algorithm. All confirmed and verified transactions are linked from a beginning of a chain all the way to a latest block, hence comes the name blockchain. The blockchain may act as a single source of truth, and members of a blockchain network can only view transactions relevant to the members.

[0040] FIG. 1 is a schematic diagram of a composition of a blockchain system according to an embodiment of this application. A blockchain system 100 may include at least one client 110 and a blockchain network 120, and the blockchain network 120 includes at least one blockchain node 121. The client 110 may be an electronic device such as a smartphone, a tablet computer, a notebook computer, a desktop computer, a smart wearable device, a smart in-vehicle device, a smart

payment terminal, or a facial recognition terminal. By installing a corresponding client application, a blockchain data service may be provided for a user. The blockchain node 121 may be a terminal device or a server. For example, the blockchain node 121 may be an independent physical server, or may be a server cluster including a plurality of physical servers, or may be a cloud server providing basic cloud computing services, such as a cloud service, a cloud database, cloud computing, a cloud function, cloud storage, a network service, cloud communication, a middleware service, a domain name service, a security service, a content delivery network (CDN), big data, and an artificial intelligence platform.

[0041] In the blockchain network 120, each blockchain node 121 may receive input information during normal operating, and maintain shared data in the blockchain network based on the received input information. To ensure information exchange, there may be information connections among all the blockchain nodes 121, and the blockchain nodes 121 may transmit information through the information connections. For example, when any blockchain node 121 in the blockchain network 120 receives the input information and broadcasts the input information in the blockchain network 120, another node in the blockchain network 120 may obtain the input information according to a consensus algorithm, and store the input information as shared data.

[0042] The blockchain nodes 121 in the blockchain network 120 all have corresponding node identifiers, and the blockchain nodes 121 in the blockchain network 120 all may store node identifiers of other blockchain nodes in the same blockchain network, to subsequently broadcast a generated block to the other nodes in the blockchain network 120 according to the node identifiers of the other blockchain nodes. Each blockchain node 121 may maintain a node identifier list shown in table 1, and a node name and a node identifier are correspondingly stored in the node identifier list. The node identifier may be an internet protocol (IP) address and any other type of information that can identify the node. Table 1 is a node identifier list using the IP address as an example.

Table 1

Node name	Node identifier
Node 1	117. 114. 151. 174
Node 2	117. 116. 189. 145
...	...
Node N	119. 123. 145. 223

[0043] FIG. 2 shows a composition structure of a blockchain maintained in a blockchain network. As shown in FIG. 2, the blockchain includes a plurality of sequentially connected blocks. Whenever there is new data that needs to be written to the blockchain, the data is to be summarized

into a newly generated block, and the newly generated block is to be linked to an end of the blockchain. Through a consensus algorithm, that blocks newly added on each node device 121 are exactly the same may be ensured. Data of a current block is recorded in a block body of each block, and a hash of a previous block connected to the block is saved in a block header of the block. If transaction data in the previous block changes, a hash of the current block also changes accordingly. Therefore, data uploaded to the blockchain network is difficult to be tampered with, which may improve reliability of shared data.

[0044] FIG. 3 exemplarily shows a network architecture of a blockchain network to which a technical solution of this application is applied. As shown in FIG. 3, in an application scenario of achieving a network transaction and a network payment, a transaction subject node 310, a transaction platform node 320, and a resource allocation node 330 that are located in the blockchain network jointly maintain one or more blockchains 340.

[0045] The transaction subject node 310 is a blockchain node that provides a transaction service for a user, and may be, for example, a merchant or an individual seller who performs a network transaction with the user.

[0046] The transaction platform node 320 is a blockchain node that provides a transaction scenario for a network transaction, and may be, for example, an e-commerce website, an online mall, or a service platform that provides a network transaction service.

[0047] The resource allocation node 330 is a blockchain node that provides a fund management and allocation service for a network transaction, and may be, for example, a third-party payment institution.

[0048] When performing the network transaction with the transaction subject node 310, the user may pay a transaction fund to the transaction subject node 310 through an account opened on the resource allocation node 330. The transaction platform node 320, as a provider of the transaction scenario, needs to extract a certain amount of the transaction fund to be used as a platform service fee. To implement credible fund allocation between the transaction subject node 310 and the transaction platform node 320, a corresponding smart contract (for example, a ledger contract) for fund allocation may be concluded based on the blockchain network. When the network transaction occurs, the resource allocation node 330 may allocate a fund to each object in the transaction scenario according to the smart contract. Transaction data generated by the network transaction on each blockchain node may be uploaded and saved through the blockchain 340, and decentralized data saving may effectively avoid a crisis of trust.

[0049] Due to an error such as node downtime, a network failure, a software error, or a malicious node, data saved between the blockchain nodes may be inconsistent. The consensus algorithm defines a set of distributed algorithms for network interaction between nodes, ensuring that in a system environment where the foregoing error occurs, consistency of data between the blockchain nodes may still be ensured. In a related technology of this application, all blockchain nodes in a blockchain network need to use a unified consensus algorithm for consensus authentication, to ensure consistency of a consensus process and a consensus result. However, in an actual blockchain service processing process, an upper-layer service system using a blockchain service continues to change with a user requirement, and has different demands in terms of performance, availability, fault tolerance, scalability, and the like. However, in a current blockchain network, a single consensus algorithm is used and a requirement of the upper-layer service system cannot be matched. If the consensus algorithm of the blockchain node needs to be changed on the blockchain network, the entire blockchain network must be controlled to stop a service. After algorithms of all the blockchain nodes are changed, a blockchain network configured with a new consensus algorithm is re-enabled. This manner of switching the consensus algorithm may easily lead to a long-term and high-frequency service interruption. Not only efficiency of consensus algorithm switching is low, but service continuity and stability reliability of the blockchain network is seriously affected.

[0050] For the foregoing technical problem existing in the related technology, this application provides a solution in which a plurality of consensus algorithms of different types can be flexibly switched online, allowing the blockchain system to change, without stopping the service, the consensus algorithm in real time to meet a requirement of a service system.

[0051] Technical solutions such as a blockchain consensus method, a blockchain consensus apparatus, a computer-readable medium, an electronic device, and a computer program product that are provided in this application are described in detail below with reference to specific implementations.

[0052] FIG. 4 is a flowchart of steps of a blockchain consensus method according to an embodiment of this application. The blockchain consensus method may be performed by the client or the blockchain node shown in FIG. 1. For ease of description, any blockchain node in a blockchain network is used as a current blockchain node, and the current blockchain node is used as an execution subject for executing this embodiment of this application. The current blockchain node may be configured with the client shown in FIG. 1. In other words, in this embodiment of this application, the blockchain consensus method performed on the current blockchain node is used as

an example for description. As shown in FIG. 4, the blockchain consensus method in this embodiment of this application includes the following S410 to S440.

[0053] S410: Broadcast consensus requests corresponding to at least two consensus stages in a blockchain network, the blockchain network including at least two blockchain nodes executing consensus algorithms of different types.

[0054] The blockchain network in this embodiment of this application may include any one of a public block chain, a private block chain, or a consortium block chain. A most decentralized one is the public block chain. A node/participant joining the public block chain may read data on the chain, publish a transaction, and compete for an accounting right of a new block, and the like. In addition, each node/participant may freely join and exit the public block chain. The private block chain is on the contrary. An accounting right of the private block chain is controlled by a specified organization or institution, and a data reading right is also controlled by the organization or institution. There are few participants and the participants cannot join the private block chain at will, and need to be reviewed by the organization or institution. The consortium block chain, also referred to as a community blockchain, is a blockchain whose consensus process is controlled by a pre-selected node, is a mixture of the public block chain and the private block chain, and may implement "partial decentralization". Generally, there is a corresponding entity or organization for each node on the chain. Participants join the network through authorization and form a consortium of stakeholders to jointly maintain operation of the blockchain. Through the consortium block chain, new participants may join an already formed blockchain and share data without having to build the blockchain from scratch. Whether it is the public block chain, the private block chain, or the consortium block chain, a function of a smart contract may be provided. A smart contract on the blockchain is a contract that may be triggered by a transaction on a blockchain system. The smart contract may be defined in a form of code.

[0055] The smart contract, also referred to as chaincode or application code, is a computer protocol designed to spread, verify, or execute a contract in an information-based manner, is a program deployed in a node of the blockchain network, carries service logic for executing a transaction, and runs in an isolated operating environment (for example, a container or a virtual machine). A contract program automatically executed by each node in the blockchain system according to a specific condition may operate data stored on the chain. This is an important way for a service entity to interact with the blockchain and use the blockchain to implement service logic. An objective of the smart contract is to provide a security method superior to a conventional contract, and reduce other transaction costs related to the contract. The smart contract allows trusted

transactions to be performed without a third party, and these transactions are traceable and irreversible.

[0056] Consensus authentication needs to be performed, between different blockchain nodes through the consensus algorithm, on transaction data generated based on the smart contract, to ensure that consistent transaction data may be saved on each blockchain node. In an embodiment of this application, after a user conducts a network transaction through a client running the blockchain system and generates transaction data, the client may transmit a consensus request to a blockchain node that communicates with the client. The blockchain node that receives the consensus request may broadcast the consensus request on the blockchain network so that the consensus request can be transmitted to all blockchain nodes in the blockchain network.

[0057] In this embodiment of this application, at least two consensus algorithms of different types may be simultaneously run on the blockchain network. For example, a part of blockchain nodes in the blockchain network execute a first consensus algorithm, and another part of the blockchain nodes execute a second consensus algorithm different from the first consensus algorithm. The first consensus algorithm and the second consensus algorithm may have a same consensus condition, or may have different consensus conditions. The first consensus algorithm and the second consensus algorithm may include consensus stages of a same quantity or consensus stages of different quantities.

[0058] In an embodiment of this application, one or more blockchain nodes may be selected from the blockchain network to be used as master nodes for processing a consensus request, and another blockchain node other than the master node in the blockchain network is used as a slave node. When the client generates the consensus request, the consensus request may be first transmitted to the master node, and the master node delivers the consensus request to each slave node. The master node may be generated through voting, rotational designation, or random selection.

[0059] S420: Separately obtain response messages broadcast by the blockchain network in the consensus stages, the response messages being messages from the blockchain nodes in response to the consensus requests.

[0060] After receiving the consensus requests, the blockchain nodes in the blockchain network may respond to the consensus requests, generate the response messages, and broadcast the response messages on the blockchain network.

[0061] In an embodiment of this application, the blockchain nodes may perform consensus verification on the received consensus requests, and broadcast and transmit the response messages on the blockchain network after verification is passed. The consensus verification performed by the blockchain node on the consensus request may include two aspects: signature verification and data verification. The signature verification is for verifying reliability of a source of the consensus request, and the data verification is for verifying authenticity of content of the consensus request.

[0062] In an embodiment of this application, the consensus request broadcast in the blockchain network carries a digital signature of a blockchain node that transmits the consensus request, and the response message broadcast in the blockchain network also carries a digital signature of a blockchain node that transmits the response message. Each blockchain node in the blockchain network may hold a public key and a private key that form an asymmetric key pair. When digital signature is performed on the consensus request or the response message, content digest may be extracted from data content by using a digest extraction algorithm. The content digest is then encrypted by using the private key to form a digital signature, and the digital signature may be verified by using the public key.

[0063] A content digest is a fixed-length string that uniquely corresponds to a piece of data content, and is generated by applying a one-way hash encryption function to the data content. If the data content is tampered with during network transmission, whether the data content has changed may be determined by comparing content digest of the data content before transmission and content digest of the data content after transmission. Therefore, integrity of the data content can be verified based on the content digest. For the content digest, plaintext of to-be-transmitted data content is mapped into a string of ciphertext by using a one-way hash function. The string of ciphertext is also referred to as a digital fingerprint. The content digest has a fixed length, and results of summarizing different plaintext into ciphertext are always different, while digests of same plaintext must be consistent. The digest extraction algorithm in this embodiment of this application may include, for example, a message digest (MD) algorithm, a secure hash algorithm (SHA), a message authentication code (MAC) algorithm, or the like.

[0064] The digital signature is a message digest algorithm with a key. The key includes a public key and a private key. The key is configured to verify data integrity, authenticate a data source, and resist repudiation, and follows an OSI reference model, private key signature, and public key verification. The digital signature is also a combination of an asymmetric encryption algorithm and a message digest algorithm. Common digital signature algorithms mainly include a

rivest-shamir-adleman (RSA) algorithm, a digital signature algorithm (DSA), and an elliptic curve digital signature algorithm (ECDSA).

[0065] Based on the RSA algorithm, a pair of RSA keys may be generated first, one of which is a secret key saved by a service entity; and the other is a public key, which may be disclosed to an outside world and may even be registered in a network server. To improve confidentiality, an RSA key must be at least 500 bits long, and 1024 bits is generally recommended. This makes a huge computation amount for encryption. To reduce the computation amount, when transmitting information, a combination of conventional encryption method and a public key encryption method is often used. In other words, the information is encrypted by using an improved DES or IDEA session key, and then the session key and information digest are encrypted by using an RSA key. After the other party receives the information, decryption is performed by using different keys and the information digest may be checked.

[0066] An important feature of the DSA is disclosure of two prime numbers. In this way, when other subjects p and q are used, even if a private key is not known, whether the subjects were generated randomly or were generated through forgery or modification can be confirmed.

[0067] The ECDSA is a combination of elliptic curves cryptography (ECC) and the DSA. An entire signature process is similar to that of the DSA. A difference is that an algorithm used in signature is the ECC. A final signed value is also divided into two parameters: r and s .

[0068] In an embodiment of this application, after the signature verification performed on the consensus request is completed, data verification may be performed on the data content carried in the consensus request. A method of the data verification may include, for example, performing consistency check between the data content carried in the consensus request and data content locally cached by the current blockchain node. If the verification passes, it indicates that the blockchain node that transmits the consensus request caches the same data content as the current blockchain node, and the data content is not modified during network transmission.

[0069] S430: Count a node quantity of blockchain nodes transmitting the response messages in a same consensus stage, the node quantity including a quantity of the at least two blockchain nodes executing the consensus algorithms of different types.

[0070] After receiving the response message broadcast and transmitted in the blockchain network, the blockchain node may determine a message source of each response message by parsing the response message. By counting the response messages with different message sources, the node quantity of the blockchain nodes transmitting the response messages may be obtained. The

node quantity represents a quantity of blockchain nodes executing consensus algorithms of different types in response to the consensus requests in the blockchain network.

[0071] In an embodiment of this application, each blockchain node may receive and count response messages transmitted from other blockchain nodes. For example, if the blockchain network includes n blockchain nodes, each blockchain node may receive at most $n-1$ response messages transmitted by other blockchain nodes. In addition, each blockchain node itself also transmits a response message to the outside world. Therefore, a maximum node quantity of the blockchain nodes transmitting the response messages and counted by a blockchain node is a total quantity of blockchain nodes in the blockchain network. Message source statistics is performed by each blockchain node, which may improve credibility of the consensus process.

[0072] In an embodiment of this application, one or more master nodes in the blockchain network may perform source statistics on the response messages, while other slave nodes do not need to perform quantity statistics. For example, in a consensus process, the master node is generated through voting, rotational designation, or random selection. After transmitting a consensus request to the blockchain network, the master node may monitor a response message returned from the slave node in the blockchain network, and count a node quantity of slave nodes transmitting response messages. An advantage of performing message source statistics through the master node is that computing cost of other slave nodes may be reduced and unnecessary data overhead is avoided.

[0073] In an embodiment of this application, to improve consensus efficiency, the node quantity of the blockchain nodes transmitting response messages may be counted within a preset time range. If a new response message is received beyond the time range, the new response message may be discarded without being counted.

[0074] In an embodiment of this application, the time range for counting the node quantity may be a fixed time window with a specified time length based on a first time point when the current blockchain node receives the consensus request as a start point of the range, or may be a fixed time window with a specified time length based on a second time point when the current blockchain node transmits the response message as a start point of the range.

[0075] In an embodiment of this application, the time range for counting the node quantity may be a dynamic time window with a varying length in which an end point of the range is determined based on a third time point when a latest response message is received and a preset time length. For example, if a last time a blockchain node receives a response message is a time point A , based on a time length B , an end point of the time range may be determined to be $A+B$. If the

blockchain node does not receive a new response message between time points A to A+B, source statistics performed on the response message may be stopped. If the blockchain node receives a new response message between time points A to A+B, for example, if a new response message is received at a time point C, the end point of the time range may be updated to C+B. By analogy, until the blockchain node does not receive a new response message for a long time, source statistics performed on the response message may be stopped.

[0076] In an embodiment of this application, the time range for performing source statistics on the response message may be a time range with a static start point and a dynamic end point. For example, any time point when a consensus request is received, a response message is transmitted, or a response message is received for the first time is used as the static start point, and the dynamic end point is determined based on a latest new response message received and a preset time length. Using a dynamically changing time range for source statistics can improve statistical efficiency while ensuring accuracy of data statistics, avoiding affecting a consensus progress due to a too long time range, and avoiding inaccurate data statistics caused by a too short time range.

[0077] S440: In a case that the node quantity satisfies a consensus condition of the first consensus algorithm, reach a consensus on a blockchain node executing the first consensus algorithm, and synchronize a consensus result of the first consensus algorithm with a blockchain node executing the second consensus algorithm, the first consensus algorithm and the second consensus algorithm being consensus algorithms of different types.

[0078] The consensus condition of the consensus algorithm is a determination condition for determining whether the consensus result can be obtained. For example, the consensus condition of a consensus algorithm may be that a counted node quantity of blockchain nodes transmitting response messages exceeds a preset ratio of a total quantity of blockchain nodes. The consensus algorithms of different types configured in the blockchain network may have a same consensus condition, or may have different consensus conditions.

[0079] In an embodiment of this application, the blockchain node in the blockchain network may determine whether the counted node quantity satisfies the consensus condition based on the consensus algorithm executed by the blockchain node. For example, if a consensus algorithm executed by a current blockchain node is the first consensus algorithm, whether the consensus condition of the first consensus algorithm is satisfied may be determined based on the counted node quantity; and if the consensus algorithm executed by the current blockchain node is the second consensus algorithm, whether a consensus condition of the second consensus algorithm is satisfied may be determined based on the counted node quantity.

[0080] In an embodiment of this application, a part of blockchain nodes in the blockchain network execute the first consensus algorithm, and another part of the blockchain nodes execute the second consensus algorithm. The consensus condition of the first consensus algorithm is that a ratio of a counted node quantity of the blockchain nodes transmitting the response messages to a total quantity of blockchain nodes is greater than a first ratio threshold, the consensus condition of the second consensus algorithm is that the ratio of the counted node quantity of the blockchain nodes transmitting the response messages to the total quantity of blockchain nodes is greater than a second ratio threshold, and the second ratio threshold is greater than the first ratio threshold. If the ratio of the counted node quantity to the total quantity of blockchain nodes is greater than the first ratio threshold, but less than the second ratio threshold, it may be determined that the node quantity satisfies the consensus condition of the first consensus algorithm and does not satisfy the consensus condition of the second consensus algorithm. Based on this, the blockchain node executing the first consensus algorithm can obtain a consensus result of a successful consensus. Although the blockchain node executing the second consensus algorithm fails to achieve a successful consensus, the blockchain node executing the first consensus algorithm may perform data synchronization with the blockchain node executing the second consensus algorithm, so that the blockchain node executing the second consensus algorithm can also complete a consensus.

[0081] In the blockchain consensus method provided in this embodiment of this application, by configuring at least two blockchain nodes executing consensus algorithms of different types in a blockchain network, a node quantity of blockchain nodes transmitting consensus response messages may be counted in a process of performing a node consensus, and when the node quantity satisfies a consensus condition of any one of the consensus algorithms, a consensus result may be synchronized by a blockchain node that satisfies the consensus condition with a blockchain node that does not satisfy the consensus condition, thereby improving flexibility of a blockchain consensus.

[0082] In an embodiment of this application, the consensus algorithms of different types include consensus stages of different quantities, each response message carries an algorithm type and a node signature that are of the blockchain node, the algorithm type indicates a consensus algorithm executed by the blockchain node, and the node signature includes a digital signature of the blockchain node in each consensus stage.

[0083] FIG. 5 shows a data structure of a response message according to an embodiment of this application. As shown in FIG. 5, a response message generated based on a specified consensus algorithm may include a type field 501, a signature field 502, and a data field 503.

[0084] The type field 501 indicates an algorithm type of a consensus algorithm executed by a blockchain node that transmits the response message, which may include, for example, a CFT consensus algorithm or a BFT consensus algorithm.

[0085] The CFT consensus algorithm, namely a crash fault tolerance (CFT) algorithm, is a type of consensus algorithm that can ensure consistency of node data when facing a fault-type error. The fault-type error includes a benign error such as node downtime or a network error.

[0086] The BFT consensus algorithm, namely a Byzantine fault tolerance (BFT) algorithm, is a type of consensus algorithm that can ensure consistency of node data when facing a Byzantine-type error. The Byzantine-type error includes the foregoing fault-type error, and a malignant error such as a software error or a malicious node.

[0087] The signature field 502 is for saving a digital signature of the blockchain node transmitting the response message in each consensus stage. The digital signature may include, for example, a pre-preparation stage signature, a preparation stage signature, and a submission stage signature.

[0088] When generating the response message, the blockchain node executing the CFT consensus algorithm may use the pre-preparation stage signature and the submission stage signature in the data structure; and when generating the response message, the blockchain node executing the BFT consensus algorithm may use signatures of three stages.

[0089] In an embodiment of this application, a response message of a stage carries digital signatures of all previous stages. For example, a response message of a first stage carries a digital signature of the first stage, a response message of a second stage simultaneously carries digital signatures that are of the first stage and the second stage, and a response message of a third stage simultaneously carries digital signatures that are of the first stage, the second stage, and the third stage.

[0090] The data field 503 is for saving data of requesting consensus authentication, and may be, for example, transaction data of a network transaction.

[0091] The data structure of the response message provided in this embodiment of this application is applicable to a plurality of consensus algorithms of different types, so that consensus algorithms with consensus stages of different quantities can perform consensus authentication through a same response message, and a matching diversity of the blockchain network for different consensus algorithms is improved.

[0092] For the consensus algorithms with consensus stages of different quantities, because each blockchain node in the blockchain network has a specified difference in computing power and network transmission speed, in a process of performing consensus authentication, a consensus stage of each blockchain node may also be different. Therefore, when a previous blockchain node receives a response message transmitted by another blockchain node, a corresponding consensus stage needs to be identified based on the response message, to effectively count a node quantity of blockchain nodes in different consensus stages. FIG. 6 shows a blockchain consensus method in which a node quantity is counted based on a consensus stage according to an embodiment of this application. As shown in FIG. 6, the method includes the following S610 to S670.

[0093] S610: Broadcast consensus requests corresponding to at least two consensus stages in a blockchain network, the blockchain network including at least two blockchain nodes executing consensus algorithms of different types.

[0094] In this embodiment of this application, at least two consensus algorithms of different types may be simultaneously run on the blockchain network. For example, a part of blockchain nodes in the blockchain network execute a first consensus algorithm, and another part of the blockchain nodes execute a second consensus algorithm different from the first consensus algorithm. The first consensus algorithm and the second consensus algorithm may have a same consensus condition, or may have different consensus conditions.

[0095] In an embodiment of this application, one or more blockchain nodes may be selected from the blockchain network to be used as master nodes for processing a consensus request, and another blockchain node other than the master node in the blockchain network is used as a slave node. When a client generates the consensus request, the consensus request may be first transmitted to the master node, and the master node delivers the consensus request to each slave node. The master node may be generated through voting, rotational designation, or random selection.

[0096] S620: Separately obtain response messages broadcast by the blockchain network in the consensus stages, the response messages being messages from the blockchain nodes in response to the consensus requests.

[0097] After receiving the consensus requests, the blockchain nodes in the blockchain network may respond to the consensus requests, generate the response messages, and broadcast the response messages on the blockchain network.

[0098] In an embodiment of this application, the blockchain nodes may perform consensus verification on the received consensus requests, and broadcast and transmit the response messages

on the blockchain network after verification is passed. The consensus verification performed by the blockchain node on the consensus request may include two aspects: signature verification and data verification. The signature verification is for verifying reliability of a source of the consensus request, and the data verification is for verifying authenticity of content of the consensus request.

[0099] In an embodiment of this application, the consensus request broadcast in the blockchain network carries a digital signature of a blockchain node that transmits the consensus request, and the response message broadcast in the blockchain network also carries a digital signature of a blockchain node that transmits the response message. Each blockchain node in the blockchain network may hold a public key and a private key that form an asymmetric key pair. When digital signature is performed on the consensus request or the response message, content digest may be extracted from data content by using a digest extraction algorithm. The content digest is then encrypted by using the private key to form a digital signature, and the digital signature may be verified by using the public key.

[00100] In an embodiment of this application, after the signature verification performed on the consensus request is completed, data verification may be performed on the data content carried in the consensus request. A method of the data verification may include, for example, performing consistency check between the data content carried in the consensus request and data content locally cached by the current blockchain node. If the verification passes, it indicates that the blockchain node that transmits the consensus request caches the same data content as the current blockchain node, and the data content is not modified during network transmission.

[00101] S630: Parse each response message, to obtain an algorithm type and a node signature that are carried in the response message.

[00102] Each response message carries an algorithm type and a node signature that are of the blockchain node, the algorithm type indicates a consensus algorithm executed by the blockchain node, and the node signature includes a digital signature of the blockchain node in each consensus stage.

[00103] The data structure shown in FIG. 6 may be obtained by parsing the response message, the algorithm type may be obtained based on the type field, and the node signature corresponding to each consensus stage may be obtained based on the signature field.

[00104] S640: Identify, according to the algorithm type, a consensus algorithm executed by a message transmitting node, where the message transmitting node is the blockchain node transmitting the response message

[00105] The blockchain nodes in the blockchain network all have corresponding node identifiers, and the blockchain nodes in the blockchain network all may store node identifiers of other blockchain nodes in the blockchain network, to subsequently broadcast data such as a generated consensus request, a response message, and a block to be uploaded to the chain to the other blockchain nodes in the blockchain network according to the node identifiers of the other blockchain nodes. Each blockchain node may maintain a node identifier list, and a node name and a node identifier are correspondingly stored in the node identifier list. The node identifier may be an internet protocol (IP) address and any other information that can identify the blockchain node.

[00106] When a blockchain node broadcasts and transmits a response message in the blockchain network, a node identifier of the blockchain node may be encapsulated in the response message. When a current blockchain node receives the response message broadcast on the blockchain network, a message transmitting node transmitting the response message may be determined based on the node identifier parsed in the response message. A consensus algorithm executed by the message transmitting node may be determined based on an algorithm type parsed in the response message.

[00107] S650: Identify a consensus stage of the message transmitting node according to the node signature.

[00108] By parsing the response message, the signature field thereof may be read to obtain the node signature corresponding to each consensus stage. Because each blockchain node may have differences in a consensus progress, the response messages broadcast on the blockchain network may also carry node signatures corresponding to different consensus stages. For example, if a specified blockchain node is in a pre-preparation stage, a response message transmitted by the blockchain node is to carry a node signature of the pre-preparation stage. For another example, if a specified blockchain node has completed consensus authentication of a pre-preparation stage and is currently in a preparation stage, a response message transmitted by the blockchain node is to carry node signatures of two parts: the pre-preparation stage and the preparation stage.

[00109] In an embodiment of this application, by parsing the response message, one or more consensus stages in which the node signature exists may be filtered. According to an execution order of the consensus stages, a current consensus stage of the message transmitting node may be determined.

[00110] If there is a node signature in only one consensus stage in the signature field, the message transmitting node may be determined to be in the consensus stage. For example, if there is a node signature in only a pre-preparation stage in a signature field of a response message, and a

signature field corresponding to another consensus stage is empty, a message transmitting node transmitting the response message may be determined to be currently in the pre-preparation stage.

[00111] If there are node signatures in at least two consensus stages in the signature field, a consensus stage with a later execution order may be determined as a current consensus stage of the message transmitting node based on an execution order of the at least two consensus stages. For example, if there are node signatures in both a pre-preparation stage and a preparation stage in a signature field of a response message, and a signature field corresponding to another consensus stage is empty, because the preparation stage in a consensus algorithm is executed after the pre-preparation stage, the message transmitting node of the response message may be determined to be currently in the preparation stage.

[00112] S660: Count, according to the consensus algorithm executed by the message transmitting node and the consensus stage of the message transmitting node, a node quantity of message transmitting nodes being in a same consensus stage with a current blockchain node.

[00113] Based on the consensus algorithm executed by the message transmitting node, whether the message transmitting node and the current blockchain node execute consensus algorithms of a same type may be determined. In a case that the message transmitting node and the current blockchain node execute consensus algorithms of a same type, whether the message transmitting node and the current blockchain node are in a same consensus stage may be determined directly based on a consensus stage of the message transmitting node. In a case that the message transmitting node and the current blockchain node execute consensus algorithms of different types, whether a consensus stage of the message transmitting node is the same as that of the current blockchain node needs to be determined according to a pre-configured matching relationship.

[00114] Before counting a node quantity of message transmitting nodes being in a same consensus stage with the current blockchain node, the current blockchain node may first obtain a matching relationship of consensus stages between the consensus algorithms of different types. In a case that the message transmitting node and the current blockchain node execute the consensus algorithms of different types, the message transmitting node being in the same consensus stage with the current blockchain node is determined according to the matching relationship.

[00115] For any two consensus algorithms of different types, whether the blockchain nodes are in a same consensus stage may be determined based on the matching relationship. For example, the message transmitting node executes one of the first consensus algorithm and the second consensus algorithm, and the current blockchain node executes the other one of the first consensus algorithm and the second consensus algorithm. The first consensus algorithm includes N consensus

stages, and the second consensus algorithm includes M consensus stages; and the matching relationship between the consensus stages of the consensus algorithms of different types includes: first P consensus stages of the first consensus algorithm and first P consensus stages of the second consensus algorithm match each other, and P is a smaller value of $N-1$ and $M-1$; and last $N-P$ consensus stages of the first consensus algorithm and last $M-P$ consensus stages of the second consensus algorithm match each other.

[00116] For example, the first consensus algorithm is a CFT consensus algorithm, and includes two consensus stages, namely, a preparation stage and a submission stage; and the second consensus algorithm is a BFT consensus algorithm, and includes three consensus stages, namely, a pre-preparation stage, a preparation stage, and a submission stage.

[00117] Based on the foregoing matching relationship, it may be learned that a previous consensus stage of the first consensus algorithm and a previous consensus stage of the second consensus algorithm match each other. In other words, the preparation stage of the CFT consensus algorithm and the pre-preparation stage of the BFT consensus algorithm match each other. A latter consensus stage of the first consensus algorithm and latter two consensus stages of the second consensus algorithm match each other. In other words, the submission stage of the CFT consensus algorithm and the preparation stage and the submission stage that are of the BFT consensus algorithm match each other.

[00118] Based on this, if a blockchain node is in the preparation stage of the CFT consensus algorithm, and another blockchain node is in the pre-preparation stage of the BFT consensus algorithm, the two blockchain nodes may be determined to be in a same consensus stage. Similarly, if a blockchain node is in the submission stage of the CFT consensus algorithm, and another blockchain node is in the preparation stage or the submission stage of the BFT consensus algorithm, the two blockchain nodes may be determined to be in a same consensus stage.

[00119] In an embodiment of this application, after counting the node quantity of the message transmitting nodes being in the same consensus stage with the current blockchain node, the current blockchain node may determine a subsequent algorithm execution action based on a consensus stage of the current blockchain node. In a case that the consensus stage of the current blockchain node is a last consensus stage of an executed consensus algorithm, a consensus result of the consensus algorithm is determined based on whether the node quantity satisfies a consensus condition of the consensus algorithm; and in a case that the consensus stage of the current blockchain node is not a last consensus stage of an executed consensus algorithm, based on whether

the node quantity satisfies a consensus condition of the consensus algorithm, whether to execute a next consensus stage of the consensus algorithm is determined.

[00120] For example, the consensus stage of the current blockchain node is the preparation stage of the CFT consensus algorithm. Through counting a node quantity of blockchain nodes being in a same consensus stage (including the preparation stage of the CFT consensus algorithm or the pre-preparation stage of the BFT consensus algorithm) with the current blockchain node, whether the node quantity satisfies the consensus condition of the CFT consensus algorithm may be determined. If the consensus condition of the CFT consensus algorithm is satisfied, the submission stage may be entered; and if the consensus condition of the CFT consensus algorithm is not satisfied, a response message transmitted from another blockchain node needs to be waited continuously, and a counted node quantity is updated in real time.

[00121] In an embodiment of this application, each blockchain node may receive and count response messages transmitted from other blockchain nodes. For example, if the blockchain network includes n blockchain nodes, each blockchain node may receive at most $n-1$ response messages transmitted by other blockchain nodes. In addition, each blockchain node itself also transmits a response message to the outside world. Therefore, a maximum node quantity of the blockchain nodes transmitting the response messages and counted by a blockchain node is a total quantity of blockchain nodes in the blockchain network. Message source statistics is performed by each blockchain node, which may improve credibility of the consensus process.

[00122] In an embodiment of this application, one or more master nodes in the blockchain network may perform source statistics on the response messages, while other slave nodes do not need to perform quantity statistics. For example, in a consensus process, the master node is generated through voting, rotational designation, or random selection. After transmitting a consensus request to the blockchain network, the master node may monitor a response message returned from the slave node in the blockchain network, and count a node quantity of slave nodes transmitting response messages. An advantage of performing message source statistics through the master node is that computing cost of other slave nodes may be reduced and unnecessary data overhead is avoided.

[00123] In an embodiment of this application, to improve consensus efficiency, the node quantity of the blockchain nodes transmitting response messages may be counted within a preset time range. If a new response message is received beyond the time range, the new response message may be discarded without being counted.

[00124] In an embodiment of this application, the time range for counting the node quantity may be a fixed time window with a specified time length based on a first time point when the current blockchain node receives the consensus request as a start point of the range, or may be a fixed time window with a specified time length based on a second time point when the current blockchain node transmits the response message as a start point of the range.

[00125] In an embodiment of this application, the time range for counting the node quantity may be a dynamic time window with a varying length in which an end point of the range is determined based on a third time point when a latest response message is received and a preset time length. For example, if a last time a blockchain node receives a response message is a time point A, based on a time length B, an end point of the time range may be determined to be A+B. If the blockchain node does not receive a new response message between time points A to A+B, source statistics performed on the response message may be stopped. If the blockchain node receives a new response message between time points A to A+B, for example, if a new response message is received at a time point C, the end point of the time range may be updated to C+B. By analogy, until the blockchain node does not receive a new response message for a long time, source statistics performed on the response message may be stopped.

[00126] In an embodiment of this application, the time range for performing source statistics on the response message may be a time range with a static start point and a dynamic end point. For example, any time point when a consensus request is received, a response message is transmitted, or a response message is received for the first time is used as the static start point, and the dynamic end point is determined based on a latest new response message received and a preset time length. Using a dynamically changing time range for source statistics can improve statistical efficiency while ensuring accuracy of data statistics, avoiding affecting a consensus progress due to a too long time range, and avoiding inaccurate data statistics caused by a too short time range.

[00127] S670: In a case that the node quantity satisfies a consensus condition of the first consensus algorithm, reach a consensus on a blockchain node executing the first consensus algorithm, and synchronize a consensus result of the first consensus algorithm with a blockchain node executing the second consensus algorithm, the first consensus algorithm and the second consensus algorithm being consensus algorithms of different types.

[00128] The consensus condition of the consensus algorithm is a determination condition for determining whether the consensus result can be obtained. For example, the consensus condition of a consensus algorithm may be that a counted node quantity of blockchain nodes transmitting response messages exceeds a preset ratio of a total quantity of blockchain nodes. The consensus

algorithms of different types configured in the blockchain network may have a same consensus condition, or may have different consensus conditions.

[00129] In an embodiment of this application, the blockchain node in the blockchain network may determine whether the counted node quantity satisfies the consensus condition based on the consensus algorithm executed by the blockchain node. For example, if a consensus algorithm executed by a current blockchain node is the first consensus algorithm, whether the consensus condition of the first consensus algorithm is satisfied may be determined based on the counted node quantity; and if the consensus algorithm executed by the current blockchain node is the second consensus algorithm, whether a consensus condition of the second consensus algorithm is satisfied may be determined based on the counted node quantity.

[00130] In an embodiment of this application, a part of blockchain nodes in the blockchain network execute the first consensus algorithm, and another part of the blockchain nodes execute the second consensus algorithm. The consensus condition of the first consensus algorithm is that a ratio of a counted node quantity of the blockchain nodes transmitting the response messages to a total quantity of blockchain nodes is greater than a first ratio threshold, the consensus condition of the second consensus algorithm is that the ratio of the counted node quantity of the blockchain nodes transmitting the response messages to the total quantity of blockchain nodes is greater than a second ratio threshold, and the second ratio threshold is greater than the first ratio threshold. If the ratio of the counted node quantity to the total quantity of blockchain nodes is greater than the first ratio threshold, but less than the second ratio threshold, it may be determined that the node quantity satisfies the consensus condition of the first consensus algorithm and does not satisfy the consensus condition of the second consensus algorithm. Based on this, the blockchain node executing the first consensus algorithm can obtain a consensus result of a successful consensus. Although the blockchain node executing the second consensus algorithm fails to achieve a successful consensus, the blockchain node executing the first consensus algorithm may perform data synchronization with the blockchain node executing the second consensus algorithm, so that the blockchain node executing the second consensus algorithm can also complete a consensus.

[00131] In an embodiment of this application, the first consensus algorithm is one of a low fault tolerance consensus algorithm and a high fault tolerance consensus algorithm, and the second consensus algorithm is one of the low fault tolerance consensus algorithm and the high fault tolerance consensus algorithm that is different from the first consensus algorithm. A consensus condition of the low fault tolerance consensus algorithm is that a quantity ratio of the blockchain nodes transmitting the response messages to all blockchain nodes is greater than a first ratio

threshold, a consensus condition of the high fault tolerance consensus algorithm is that the quantity ratio of the blockchain nodes transmitting the response messages to all blockchain nodes is greater than a second ratio threshold, and the first ratio threshold is less than the second ratio threshold. Each of the first ratio threshold and the second ratio threshold is a constant greater than 0 and less than 1. For example, the first ratio threshold is $1/2$, and the second ratio threshold is $2/3$.

[00132] If the quantity ratio of the blockchain nodes transmitting the response messages to all blockchain nodes is less than $1/2$, the node quantity neither satisfies the consensus condition of the low fault tolerance consensus algorithm nor the consensus condition of the high fault tolerance consensus algorithm, and all blockchain nodes executing the two consensus algorithms cannot complete consensus authentication.

[00133] If the quantity ratio of the blockchain nodes transmitting the response messages to all blockchain nodes is greater than $1/2$ but less than $2/3$, the node quantity satisfies the consensus condition of the low fault tolerance consensus algorithm but does not satisfy the consensus condition of the high fault tolerance consensus algorithm, and a blockchain node executing the low fault tolerance consensus algorithm can complete consensus authentication, while a blockchain node executing the high fault tolerance consensus algorithm cannot complete consensus authentication. On this basis, the blockchain node executing the low fault tolerance consensus algorithm may synchronize a consensus result with the blockchain node executing the high fault tolerance consensus algorithm, thereby enabling all blockchain nodes in the blockchain network to complete consensus authentication.

[00134] If the quantity ratio of the blockchain nodes transmitting the response messages to all blockchain nodes is greater than $2/3$, the node quantity satisfies both the consensus condition of the low fault tolerance consensus algorithm and the consensus condition of the high fault tolerance consensus algorithm, and all blockchain nodes executing the two consensus algorithms can complete consensus authentication.

[00135] In an embodiment of this application, the low fault tolerance consensus algorithm includes consensus stages of a first quantity, the high fault tolerance consensus algorithm includes consensus stages of a second quantity, and the first quantity is less than the second quantity. Fault tolerance of the consensus algorithm may be improved by increasing a quantity of consensus stages.

[00136] In an embodiment of this application, the consensus stage of the low fault tolerance consensus algorithm includes a preparation stage and a submission stage that are executed in sequence, the consensus stage of the high fault tolerance consensus algorithm includes a pre-preparation stage, a preparation stage, and a submission stage that are executed in sequence, the

preparation stage of the low fault tolerance consensus algorithm matches the pre-preparation stage of the high fault tolerance consensus algorithm, and the submission stage of the low fault tolerance consensus algorithm matches the preparation stage and the submission stage that are of the high fault tolerance consensus algorithm.

[00137] FIG. 7 shows a consensus process of a low fault tolerance consensus algorithm according to an embodiment of this application. The low fault tolerance consensus algorithm may be, for example, a Paxos algorithm in a CFT consensus algorithm. The low fault tolerance consensus algorithm in this embodiment of this application is based on interaction between nodes of two stages. An advantage is that the interaction between nodes is simple, so that implementation is easy and performance is high.

[00138] As shown in FIG. 7, a process of performing consensus authentication based on the low fault tolerance consensus algorithm may include the following steps.

[00139] S701: A client transmits a consensus request to a master node. The consensus request may be, for example, a write-data request for writing data to a blockchain.

[00140] S702: The master node performs first-stage node interaction (preparation stage) and distributes an authentication request to another slave node. After receiving the authentication request and performing consensus processing, the slave node may return a response message to the master node.

[00141] In a preparation stage, the authentication request transmitted by the master node to another slave node carries data that needs to be written to the blockchain, and a digital signature that is of the master node and is obtained after the data is digested and encrypted by using a private key of the master node. By using the data structure shown in FIG. 5 as an example, the authentication request transmitted by the master node to the slave node carries the digital signature of the master node written in a field "pre-preparation stage signature".

[00142] When the slave node receives the authentication request transmitted by the master node, signature verification may be performed, by using a public key of the master node, on the digital signature carried in the authentication request. After the signature verification is passed, the slave node may digest and encrypt, by using a private key of the slave node, the data that needs to be written to the blockchain to obtain a digital signature of the slave node. By using the data structure shown in FIG. 5 as an example, the response message returned by the slave node to the master node in the preparation stage carries the digital signature of the master node written in the field "pre-preparation stage signature" and the digital signature of the slave node.

[00143] S703: When the master node receives response messages that exceed $1/2$ of a total quantity of nodes (including the master node itself), the master node executes a second stage (a submission stage), and the master node submits data and notifies another slave node to submit data. After receiving the notification, the slave node may write data to a blockchain saved by the slave node itself, and return a response message to the master node.

[00144] When receiving the response message returned by the slave node in the preparation stage, the master node may collect and aggregate a digital signature of the slave node carried in each response message.

[00145] After entering the submission stage, the master node may transmit a submission notification carrying an aggregated digital signature to each slave node. By using the data structure shown in FIG. 5 as an example, the submission notification transmitted by the master node to the slave node carries the digital signature of the master node written in the field "pre-preparation stage signature" and a plurality of digital signatures of the slave nodes, and additionally carries a digital signature of the master node written in a field "submission stage signature". A quantity of digital signatures of the master nodes and the digital signatures of the slave nodes in the field "pre-pre-preparation stage signature" should exceed $1/2$ of the total quantity of nodes. In other words, a consensus condition of a low fault tolerance consensus algorithm in the preparation stage is satisfied.

[00146] When receiving the submission notification transmitted by the master node, the slave node may perform signature verification in a manner the same as that in a previous stage, and write the digital signature of the slave node itself in the response message after the verification is passed. By using the data structure shown in FIG. 5 as an example, the response message returned by the slave node to the master node in the submission stage carries the digital signature of the master node written in the field "pre-preparation stage signature" and the plurality of digital signatures of the slave nodes, and additionally carries the digital signature of the master node written in the field "submission stage signature" and the digital signature of the slave node itself.

[00147] S704: The master node returns a response to the client indicating that the data has been written successfully. In this case, the client may determine that the data has been successfully written into an entire blockchain system.

[00148] When receiving the response message returned by the slave node in the submission stage, the master node may collect and aggregate a digital signature of the slave node carried in each response message. By using the data structure shown in FIG. 5 as an example, the response message returned by the master node to the client carries the digital signature of the master node

written in a field "pre-preparation stage signature" and a plurality of digital signatures of the slave nodes, and additionally carries a digital signature of the master node written in a field "submission stage signature" and a plurality of digital signatures of the slave nodes. A quantity of digital signatures of the master nodes and the digital signatures of the slave nodes in the field "pre-preparation stage signature" should exceed $1/2$ of the total quantity of nodes, and a quantity of digital signatures of the master nodes and the digital signatures of the slave nodes in the field "submission stage signature" should exceed $1/2$ of the total quantity of nodes. In other words, a consensus condition of a low fault tolerance consensus algorithm in the preparation stage and the submission stage is satisfied.

[00149] FIG. 8 shows a consensus process of a high fault tolerance consensus algorithm according to an embodiment of this application. The high fault tolerance consensus algorithm may be, for example, a PBFT algorithm in a BFT consensus algorithm. Compared with a CFT consensus algorithm, the BFT consensus algorithm needs to ensure consistency of a system even in a case of a software error or a node misbehavior. Therefore, node interaction of the BFT algorithm is more complex and generally requires three stages of interaction; and a quantity of fault tolerance nodes is smaller, and generally $1/3$ of faulty nodes may be tolerated.

[00150] As shown in FIG. 8, a process of performing consensus authentication based on the high fault tolerance consensus algorithm may include the following steps.

[00151] S801: A client transmits a consensus request to a master node. The consensus request may be, for example, a write-data request for writing data to a blockchain.

[00152] S802: The master node performs first-stage node interaction (pre-preparation stage) and distributes an authentication request to another slave node. After receiving the authentication request and performing consensus processing, the slave node may return a response message to the master node. An objective of this stage is to confirm that requests received by a plurality of slave nodes are the same, to prevent the master node from having malicious behavior and transmitting different messages to different slave nodes.

[00153] In the pre-preparation stage, the authentication request transmitted by the master node to another slave node carries data that needs to be written to the blockchain, and a digital signature that is of the master node and is obtained after the data is digested and encrypted by using a private key of the master node. By using the data structure shown in FIG. 5 as an example, the authentication request transmitted by the master node to the slave node carries the digital signature of the master node written in a field "pre-preparation stage signature".

[00154] After the slave node receives the authentication request transmitted by the master node, signature verification may be performed, by using a public key of the master node, on the digital signature carried in the authentication request. After the signature verification is passed, the slave node may digest and encrypt, by using a private key of the slave node, the data that needs to be written to the blockchain to obtain a digital signature of the slave node. By using the data structure shown in FIG. 5 as an example, the response message returned by the slave node to the master node in the pre-preparation stage carries the digital signature of the master node written in the field "pre-preparation stage signature" and the digital signature of the slave node.

[00155] S803: When the master node receives response messages, in the pre-preparation stage, that exceed $2/3$ of a total quantity of nodes (including the master node itself), the master node executes a second stage (a preparation stage), and the master node re-transmits a message signature to another slave node. The slave node is to verify whether a quantity of message signatures is greater than $2/3$ of the total quantity of nodes. If the quantity of message signatures is greater than $2/3$, the slave node returns a response message indicating successful voting to the master node.

[00156] When receiving the response message returned by the slave node in the pre-preparation stage, the master node may collect and aggregate a digital signature of the slave node carried in each response message.

[00157] After entering the preparation stage, the master node may transmit a preparation message carrying an aggregated digital signature to each slave node. By using the data structure shown in FIG. 5 as an example, the preparation message transmitted by the master node to the slave node carries the digital signature of the master node written in the field "pre-preparation stage signature" and a plurality of digital signatures of the slave nodes, and additionally carries a digital signature of the master node written in a field "preparation stage signature". A quantity of digital signatures of the master nodes and the digital signatures of the slave nodes in the field "pre-pre-preparation stage signature" should exceed $2/3$ of the total quantity of nodes. In other words, a consensus condition of a high fault tolerance consensus algorithm in the pre-preparation stage is satisfied.

[00158] After receiving the preparation message transmitted by the master node, the slave node may perform signature verification in a manner the same as that in a previous stage, and write the digital signature of the slave node itself in the response message after the verification is passed. By using the data structure shown in FIG. 5 as an example, the response message returned by the slave node to the master node in the preparation stage carries the digital signature of the master node written in the field "pre-preparation stage signature" and the plurality of digital signatures of

the slave nodes, and additionally carries the digital signature of the master node written in the field "submission stage signature" and the digital signature of the slave node itself.

[00159] S804: When the master node receives response messages that exceed $\frac{2}{3}$ of a total quantity of nodes (including the master node itself), the master node executes a third stage (a submission stage), and the master node submits data and notifies another slave node to submit data. After receiving the notification, the slave node may write data to a blockchain saved by the slave node itself, and return a response message to the master node.

[00160] When receiving the response message returned by the slave node in the preparation stage, the master node may collect and aggregate a digital signature of the slave node carried in each response message.

[00161] After entering the submission stage, the master node may transmit a submission notification carrying an aggregated digital signature to each slave node. By using the data structure shown in FIG. 5 as an example, the submission notification transmitted by the master node to the slave node carries the digital signature of the master node written in the field "pre-preparation stage signature" and a plurality of digital signatures of the slave nodes, and additionally carries a digital signature of the master node written in a field "preparation stage signature" and a plurality of digital signatures of the slave nodes, and a digital signature of the master node written in a field "submission stage signature". A quantity of digital signatures of the master nodes and the digital signatures of the slave nodes in the field "pre-preparation stage signature" should exceed $\frac{2}{3}$ of the total quantity of nodes, and a quantity of digital signatures of the master nodes and the digital signatures of the slave nodes in the field "preparation stage signature" should exceed $\frac{2}{3}$ of the total quantity of nodes. In other words, a consensus condition of a high fault tolerance consensus algorithm in the pre-preparation stage and the preparation stage is satisfied.

[00162] When receiving the preparation message transmitted by the master node, the slave node may perform signature verification in a manner the same as that in a previous stage, and write the digital signature of the slave node itself in the response message after the verification is passed. By using the data structure shown in FIG. 5 as an example, the response message returned by the slave node to the master node in the submission stage carries the digital signature of the master node written in the field "pre-preparation stage signature" and the plurality of digital signatures of the slave nodes, and additionally carries the digital signature of the master node written in the field "preparation stage signature" and the plurality of digital signatures of the slave nodes, and the digital signature of the master node written in the field "submission stage signature" and the digital signature of the slave node itself.

[00163] S805: The master node returns a response to the client indicating that the data has been written successfully. In this case, the client may determine that the data has been successfully written into an entire blockchain system.

[00164] When receiving the response message returned by the slave node in the submission stage, the master node may collect and aggregate a digital signature of the slave node carried in each response message. By using the data structure shown in FIG. 5 as an example, the response message returned by the master node to the client carries the digital signature of the master node written in the field "pre-preparation stage signature" and the plurality of digital signatures of the slave nodes, and additionally carries the digital signature of the master node written in the field "preparation stage signature" and the plurality of digital signatures of the slave nodes, and the digital signature of the master node written in the field "submission stage signature" and the plurality of the digital signatures of the slave nodes. A quantity of digital signatures of the master nodes and the digital signatures of the slave nodes in the field "pre-preparation stage signature" should exceed $2/3$ of the total quantity of nodes, a quantity of digital signatures of the master nodes and the digital signatures of the slave nodes in the field "preparation stage signature" also should exceed $2/3$ of the total quantity of nodes, and a quantity of digital signatures of the master nodes and the digital signatures of the slave nodes in the field "submission stage signature" also should exceed $2/3$ of the total quantity of nodes. In other words, a consensus condition of a high fault tolerance consensus algorithm in the pre-preparation stage, the preparation stage, and the submission stage is satisfied.

[00165] In a blockchain system in the related technology, one type of system pursues system performance and adopts a CFT-type consensus, such as Fabric; and the other type pursues wider fault tolerance and adopts a BFT-type consensus, such as Tendermint. In an actual enterprise-level application, an upper-layer service system using a blockchain service continues to change with a user requirement, and has different demands in terms of performance, availability, fault tolerance, scalability, and the like. However, in a current blockchain system, a single consensus algorithm is used and a requirement of the upper-layer service system cannot be matched. An embodiment of this application provides a solution in which a CFT algorithm and a BFT algorithm are flexibly switched online, allowing the blockchain system to change, without stopping the service, the consensus algorithm to meet a requirement of a service system.

[00166] In an embodiment of this application, in a process of broadcasting the consensus request in the blockchain network, algorithm switching may be performed on some or all

blockchain nodes in the blockchain network, to change consensus algorithms executed by the some or all blockchain nodes.

[00167] When some blockchain nodes complete algorithm switching and another part of the blockchain nodes do not complete algorithm switching, there are at least two blockchain nodes executing different consensus algorithms in the blockchain network. Based on this, consensus authentication may be performed by using the foregoing embodiments of this application.

[00168] In an embodiment of this application, the method of performing algorithm switching on some or all blockchain nodes in the blockchain network may include:

[00169] performing credibility prediction on the blockchain network, to determine whether an operating environment of the blockchain network is a trusted environment;

[00170] switching, in a case that the operating environment of the blockchain network is the trusted environment, some or all blockchain nodes in the blockchain network from executing a low fault tolerance consensus algorithm in the blockchain network to executing a high fault tolerance consensus algorithm; and

[00171] switching, in a case that the operating environment of the blockchain network is a non-trusted environment, some or all blockchain nodes in the blockchain network from executing a high fault tolerance consensus algorithm in the blockchain network to executing a low fault tolerance consensus algorithm.

[00172] In an embodiment of this application, the method of performing credibility prediction on the blockchain network may include:

[00173] obtaining operating duration of each blockchain node in the blockchain network;

[00174] determining the blockchain node as a non-trusted node in a case that the operating duration is less than a duration threshold;

[00175] determining the operating environment of the blockchain network as the non-trusted environment in a case that a quantity of non-trusted nodes in the blockchain network is greater than a quantity threshold; and

[00176] determining the operating environment of the blockchain network as the trusted environment in a case that the quantity of non-trusted nodes in the blockchain network is less than the quantity threshold.

[00177] For example, in a supply chain management scenario, the blockchain system is often initiated by an enterprise in a specified link in a supply chain process. In this case, the blockchain

system may be considered to be in a trusted environment, there is no need to consider a case of a malicious node, and a simple CFT algorithm may be used as a consensus algorithm. When another enterprise in the supply chain join the blockchain system, to prevent the another enterprise from intentionally having malicious behavior, the steps of algorithm switching in this embodiment of this application may be performed to switch the consensus algorithm to a BFT algorithm. When the blockchain system has been running stably for a period of time and each participating enterprise may be considered a trusted node, the steps of algorithm switching in this embodiment of this application may be performed again to switch the consensus algorithm to the CFT algorithm to improve performance of the blockchain system.

[00178] FIG. 9 shows a scenario example in a process of performing algorithm switching according to an embodiment of this application. By using algorithm switching of four nodes as an example, each blockchain node is to execute a different consensus algorithm at a different moment of algorithm switching.

[00179] When there are two consensus algorithms with different distribution ratios in a blockchain network, five scenarios shown in FIG. 8 may be formed. In a switch from a CFT consensus to a BFT consensus, the nodes are gradually switched from all CFT consensus algorithms to all BFT consensus algorithms. In other words, scenario 1 gradually changes to scenario 5. In a switch from the BFT consensus to the CFT consensus, the nodes are gradually switched from all BFT consensus algorithms to all CFT consensus algorithms. In other words, scenario 5 gradually changes to scenario 1.

[00180] Table 2 shows a ratio relationship of blockchain nodes executing different consensus algorithms to a total quantity of nodes in the five scenarios.

Table 2

Scenario	Quantity ratio of CFT nodes to all nodes	Quantity ratio of BFT nodes to all nodes
Scenario 1	$= 1$	$= 0$
Scenario 2	$> \frac{1}{2}$	$< \frac{1}{2}$
Scenario 3	$< \frac{1}{2}$ and $> \frac{1}{3}$	$> \frac{1}{2}$ and $< \frac{2}{3}$
Scenario 4	$< \frac{1}{3}$	$> \frac{2}{3}$
Scenario 5	$= 0$	$= 1$

[00181] The following describes consensus solutions in five application scenarios respectively.

[00182] Scenario 1:

[00183] In scenario 1, because a consensus algorithm of each node is CFT, this scenario is equivalent to a scenario of using a CFT algorithm alone. As long as all nodes follow a step of a local CFT algorithm, system data can be consistent.

[00184] Scenario 2:

[00185] In scenario 2, a consensus algorithm of only node 1 is a BFT algorithm, and consensus algorithms of the other three nodes are all CFT algorithms. In this case, a total quantity of CFT nodes (3) is greater than 1/2 of a total quantity of nodes (4). According to a consensus condition of the CFT algorithm, a consensus may be reached among the three CFT nodes. After the consensus is reached, node 1 synchronizes a consensus result to ensure that the system data is consistent.

[00186] Scenario 3:

[00187] In scenario 3, the quantity of CFT nodes cannot satisfy the consensus condition of being greater than 1/2 required by the CFT algorithm, and a quantity of BFT nodes cannot satisfy a consensus condition of being greater than 2/3 required by the BFT algorithm. In this case, according to the normal CFT algorithm or BFT algorithm, all nodes in the system cannot reach a consensus.

[00188] In this scenario, in this embodiment of this application, data interaction among the nodes may be performed by using a response message with the data structure shown in FIG. 5.

[00189] Refer to FIG. 5. The data structure includes three signature fields: a pre-preparation stage signature, a preparation stage signature, and a submission stage signature. A CFT consensus node uses the pre-preparation stage signature and the submission stage signature; and a BFT consensus node uses signatures of three stages.

[00190] FIG. 10 is a flowchart of method steps of performing consensus authentication in an application scenario according to an embodiment of this application. As shown in FIG. 10, in the foregoing scenario 3, the method of performing consensus authentication may include the following steps.

[00191] S1001: A master node receives a consensus request transmitted by a client.

[00192] S1002: The master node transmits a <pre-preparation> message to a slave node.

[00193] S1003: The slave node replies to the <pre-preparation> message to the master node.

[00194] A preparation stage of a CFT consensus algorithm and a pre-preparation stage of a BFT consensus algorithm match each other. Therefore, a node using the CFT consensus algorithm may treat the <pre-preparation> message as a <preparation> message. In this case, both a slave node using a CFT consensus and a slave node using a BFT consensus reply to the <pre-preparation> message to the master node.

[00195] S1004: The master node determines whether <pre-preparation> messages of more than $2/3$ of a total quantity of nodes are received.

[00196] If a quantity does not exceed $2/3$, return to S1003 and the master node continues to receive a <pre-preparation> message replied by another slave node. If the quantity exceeds $2/3$, consensus authentication in the pre-preparation stage is completed, the preparation stage is entered, and S1005 is performed.

[00197] S1005: The master node transmits a <preparation> message to the slave node.

[00198] S1006: Determine whether the slave node receiving the <preparation> message is a CFT node executing the CFT consensus algorithm. If a determining result is yes, step S1007 is performed. If a determining result is no, it indicates that the slave node is a BFT node executing the BFT consensus algorithm, and jump to S1010.

[00199] S1007: The slave node submits data.

[00200] When the slave node using the CFT consensus algorithm receives the <preparation> message from the master node, it may be said that the master node has received more than $2/3$ of the <preparation> messages. This is greater than the condition of $1/2$ required by the CFT algorithm, so that in this case, the slave node executing the CFT consensus algorithm may treat the <preparation> message as a <submission> message and directly submit the data. In this case, the node using the CFT consensus algorithm may reach a consensus, and the data is consistent.

[00201] S1008: The slave node replies to a <submission> message to the master node.

[00202] After completing submitting the data, the slave node executing the CFT consensus algorithm returns the <submission> message to the master node, so that another BFT node may achieve a submission condition that a total quantity of <preparation>+<submission> reaches $2/3$. S1010 is then performed.

[00203] S1009: The slave node replies to the <preparation> message to the master node.

[00204] S1010: The master node determines whether <preparation> messages of more than $2/3$ of the total quantity of nodes are received.

[00205] If it is determined that a quantity does not exceed $2/3$, S1011 is continued to be performed. If it is determined that the quantity exceeds $2/3$, consensus authentication in the preparation stage is completed, a submission stage is entered, and jump to perform S1012.

[00206] S1011: The master node determines whether <preparation> messages and <submission> messages that are of more than $2/3$ of the total quantity of nodes are received.

[00207] The submission stage of the CFT consensus algorithm and the preparation stage and the submission stage that are of the BFT consensus algorithm match each other. Therefore, when the master node receives a <submission> message response replied by the CFT node, the master node may treat the <submission> message response as a <preparation> message response.

[00208] Based on this, if it is determined that the master node receives <preparation> messages and <submission> messages that are of more than $2/3$ of the total quantity of nodes, consensus authentication in the preparation stage is completed, the submission stage is entered, and continue to perform S1012. If it is determined that the master node does not receive <preparation> messages and <submission> messages that are of more than $2/3$ of the total quantity of nodes, return to S1005, and the master node continues to transmit the <preparation> message to another slave node.

[00209] S1012: The master node transmits a <submission> message to the slave node.

[00210] S1013: Each slave node submits data and returns the <submission> message to the master node, and a system reaches a consensus.

[00211] A node using the BFT consensus may achieve a <preparation> message response condition of being more than $2/3$. In this case, the BFT consensus node may reach a consensus based on a step of the BFT algorithm, and consistency of the data with the CFT node may be ensured.

[00212] Scenario 4:

[00213] In scenario 4, node 4 executes the CFT consensus algorithm, and the other three nodes execute the BFT consensus algorithm. In this case, a total quantity of BFT nodes (3) is greater than $2/3$ of the total quantity of nodes (4). According to the BFT consensus algorithm, a consensus may be reached among the three BFT nodes. After the consensus is reached, node 4 synchronizes a consensus result to ensure consistency of the system data.

[00214] Scenario 5:

[00215] In scenario 5, all nodes execute the BFT consensus algorithm. Based on a step of the BFT consensus algorithm, the system may reach a consistent consensus.

[00216] Based on the description of each application scenario above, it may be learned that the online consensus algorithm switching solution provided in this embodiment of this application may support switching the consensus algorithm without stopping a service, making the blockchain system more flexibly adapt to a requirement of an upper-layer service, and making the blockchain system more widely used in different scenarios.

[00217] Although the steps of the method in this application are described in a specific order in the accompanying drawings, this does not require or imply that the steps have to be performed in the specific order, or all the steps shown have to be performed to achieve an expected result. Additionally or alternatively, some steps may be omitted, a plurality of steps may be combined into one step, and/or one step may be decomposed into a plurality of steps for execution, and the like.

[00218] The following describes apparatus embodiments of this application, and the apparatus embodiments may be used for performing the blockchain consensus method in the foregoing embodiment of this application. FIG. 11 is a schematic block diagram of a structure of a blockchain consensus apparatus according to an embodiment of this application. As shown in FIG. 11, a blockchain consensus apparatus 1100 includes:

[00219] a request module 1110, configured to broadcast consensus requests corresponding to at least two consensus stages in a blockchain network, the blockchain network including at least two blockchain nodes executing consensus algorithms of different types;

[00220] a response module 1120, configured to separately obtain response messages broadcast by the blockchain network in the consensus stages, the response messages being messages from the blockchain nodes in response to the consensus requests, and the consensus algorithms of different types including a first consensus algorithm and a second consensus algorithm;

[00221] a statistics module 1130, configured to count a node quantity of blockchain nodes transmitting the response messages in a same consensus stage, the node quantity including a quantity of the at least two blockchain nodes executing the consensus algorithms of different types; and

[00222] a synchronization module 1140, configured to: in a case that the node quantity satisfies a consensus condition of the first consensus algorithm, reach a consensus on a blockchain

node executing the first consensus algorithm, and synchronize a consensus result of the first consensus algorithm with a blockchain node executing the second consensus algorithm.

[00223] In an embodiment of this application, based on the foregoing embodiments, the consensus algorithms of different types include consensus stages of different quantities, each response message carries an algorithm type and a node signature that are of the blockchain node, the algorithm type indicates a consensus algorithm executed by the blockchain node, and the node signature includes a digital signature of the blockchain node in each consensus stage.

[00224] In an embodiment of this application, based on the foregoing embodiments, the statistics module 1030 may further include:

[00225] a parsing module, configured to parse each response message, to obtain the algorithm type and the node signature that are carried in the response message;

[00226] an algorithm identification module, configured to identify, according to the algorithm type, a consensus algorithm executed by a message transmitting node, where the message transmitting node is the blockchain node transmitting the response message;

[00227] a stage identification module, configured to identify a consensus stage of the message transmitting node according to the node signature; and

[00228] a quantity statistics module, configured to count, according to the consensus algorithm executed by the message transmitting node and the consensus stage of the message transmitting node, a node quantity of message transmitting nodes being in a same consensus stage with a current blockchain node.

[00229] In an embodiment of this application, based on the foregoing embodiments, the statistics module 1030 may further include:

[00230] a relationship obtaining module, configured to obtain a matching relationship between consensus stages of the consensus algorithms of different types; and

[00231] a node determination module, configured to determine, in a case that the message transmitting node and the current blockchain node execute consensus algorithms of different types, the message transmitting node being in the same consensus stage with the current blockchain node according to the matching relationship.

[00232] In an embodiment of this application, based on the foregoing embodiments, the first consensus algorithm includes N consensus stages, and the second consensus algorithm includes M

consensus stages; and the matching relationship between the consensus stages of the consensus algorithms of different types includes:

[00233] first P consensus stages of the first consensus algorithm and first P consensus stages of the second consensus algorithm match each other, and P is a smaller value of N-1 and M-1; and

[00234] last N-P consensus stages of the first consensus algorithm and last M-P consensus stages of the second consensus algorithm match each other.

[00235] In an embodiment of this application, based on the foregoing embodiments, after counting the node quantity of the message transmitting nodes being in the same consensus stage with the current blockchain node, the statistics module 1030 may further include:

[00236] a consensus result determination module, configured to determine, in a case that the consensus stage of the current blockchain node is a last consensus stage of an executed consensus algorithm, a consensus result of the consensus algorithm based on whether the node quantity satisfies a consensus condition of the executed consensus algorithm; and

[00237] a consensus stage execution module, configured to determine, in a case that the consensus stage of the current blockchain node is not a last consensus stage of an executed consensus algorithm, based on whether the node quantity satisfies a consensus condition of the executed consensus algorithm, whether to execute a next consensus stage of the consensus algorithm.

[00238] In an embodiment of this application, based on the foregoing embodiments, the first consensus algorithm is one of a low fault tolerance consensus algorithm and a high fault tolerance consensus algorithm, and the second consensus algorithm is one of the low fault tolerance consensus algorithm and the high fault tolerance consensus algorithm that is different from the first consensus algorithm; and

[00239] a consensus condition of the low fault tolerance consensus algorithm is that a quantity ratio of the blockchain nodes transmitting the response messages to all blockchain nodes is greater than a first ratio threshold, a consensus condition of the high fault tolerance consensus algorithm is that the quantity ratio of the blockchain nodes transmitting the response messages to all blockchain nodes is greater than a second ratio threshold, and the first ratio threshold is less than the second ratio threshold.

[00240] In an embodiment of this application, based on the foregoing embodiments, the low fault tolerance consensus algorithm includes consensus stages of a first quantity, the high fault

tolerance consensus algorithm includes consensus stages of a second quantity, and the first quantity is less than the second quantity.

[00241] In an embodiment of this application, based on the foregoing embodiments, each consensus stage of the low fault tolerance consensus algorithm includes a preparation stage and a submission stage that are executed in sequence, each consensus stage of the high fault tolerance consensus algorithm includes a pre-preparation stage, a preparation stage, and a submission stage that are executed in sequence, the preparation stage of the low fault tolerance consensus algorithm matches the pre-preparation stage of the high fault tolerance consensus algorithm, and the submission stage of the low fault tolerance consensus algorithm matches the preparation stage and the submission stage that are of the high fault tolerance consensus algorithm.

[00242] In an embodiment of this application, based on the foregoing embodiments, the blockchain consensus apparatus 1000 may further include:

[00243] an algorithm switching module, configured to perform algorithm switching on some or all blockchain nodes in the blockchain network, to change consensus algorithms executed by the some or all blockchain nodes.

[00244] In an embodiment of this application, based on the foregoing embodiments, the algorithm switching module may further include:

[00245] a credibility prediction module, configured to perform credibility prediction on the blockchain network, to determine whether an operating environment of the blockchain network is a trusted environment;

[00246] a first algorithm switching module, configured to switch, in a case that the operating environment of the blockchain network is a trusted environment, some or all blockchain nodes in the blockchain network from executing a low fault tolerance consensus algorithm in the blockchain network to executing a high fault tolerance consensus algorithm; and

[00247] a second algorithm switching module, configured to switch, in a case that the operating environment of the blockchain network is a non-trusted environment, some or all blockchain nodes in the blockchain network from executing a high fault tolerance consensus algorithm in the blockchain network to executing a low fault tolerance consensus algorithm.

[00248] In an embodiment of this application, based on the foregoing embodiments, for the performing credibility prediction on the blockchain network, and the credibility prediction module may further include:

[00249] a duration obtaining module, configured to obtain operating duration of each blockchain node in the blockchain network;

[00250] a non-trusted node determination module, configured to determine the blockchain node as a non-trusted node in a case that the operating duration is less than a duration threshold;

[00251] a non-trusted environment determination module, configured to determine the operating environment of the blockchain network as a non-trusted environment in a case that a quantity of non-trusted nodes in the blockchain network is greater than a quantity threshold; and

[00252] a trusted environment determination module, configured to determine the operating environment of the blockchain network as a trusted environment in a case that the quantity of non-trusted nodes in the blockchain network is less than the quantity threshold.

[00253] Specific details of the blockchain consensus apparatus provided in the embodiments of this application have been described in detail in the corresponding method embodiment. Details are not described herein again.

[00254] FIG. 12 is a schematic block diagram of a structure of a computer system of an electronic device for implementing an embodiment of this application.

[00255] A computer system 1200 of the electronic device shown in FIG. 12 is merely an example, and does not constitute any limitation on functions and use ranges of the embodiments of this application.

[00256] As shown in FIG. 12, the computer system 1200 includes a central processing unit (CPU) 1201. The CPU 1201 may perform various appropriate actions and processing according to a program stored in a read-only memory (ROM) 1202 or a program loaded from a storage portion 1208 into a random access memory (RAM) 1203. The random access memory 1203 further stores various programs and data required for system operations. The central processing unit 1201, the read-only memory 1202, and the random access memory 1203 are connected to each other through a bus 1204. An input/output (I/O) interface 1205 is also connected to the bus 1204.

[00257] The following components are connected to the input/output interface 1205: an input part 1206 including a keyboard, a mouse, and the like; an output part 1207 including a cathode ray tube (CRT), a liquid crystal display (LCD), a loudspeaker, and the like; a storage part 1208 including hard disk, and the like; and a communication part 1209 including a network interface card such as an LAN card or a modem. The communication part 1209 performs communication processing by using a network such as the Internet. A driver 1210 is also connected to the input/output interface 1205 as required. A removable medium 1211, such as a magnetic disk, an

optical disc, a magneto-optical disk, or a semiconductor memory, is mounted on the driver 1210 as required, so that a computer program read from the removable medium is installed into the storage part 1208 as required.

[00258] Particularly, according to an embodiment of this application, the processes described in the foregoing may be implemented as computer software programs. For example, an embodiment of this application includes a computer program product, the computer program product includes a computer program carried on a computer-readable medium, and the computer program includes program code used for performing the methods shown in the flowcharts. In such an embodiment, the computer program may be downloaded and installed from a network through the communication part 1209, and/or installed from the removable medium 1211. When the computer program is executed by the central processing unit 1201, the various functions defined in the system of this application are executed.

[00259] The computer-readable medium shown in the embodiments of this application may be a computer-readable signal medium or a computer-readable storage medium or any combination thereof. The computer-readable storage medium may be, for example, but is not limited to, an electric, magnetic, optical, electromagnetic, infrared, or semi-conductive system, apparatus, or device, or any combination thereof. A more specific example of the computer-readable storage medium may include but is not limited to: an electrical connection having one or more wires, a portable computer magnetic disk, a hard disk, a RAM, a ROM, an erasable programmable read-only memory (EPROM), a flash memory, an optical fiber, a compact disc read-only memory (CD-ROM), an optical storage device, a magnetic storage device, or any appropriate combination thereof. In this application, the computer-readable storage medium may be any tangible medium including or storing a program, and the program may be used by or in combination with an instruction execution system, apparatus, or device. In this application, the computer-readable signal medium may include a data signal being in a baseband or propagated as at least a part of a carrier wave, and carries computer-readable program code. The propagated data signal may be in a plurality of forms, including, but not limited to, an electromagnetic signal, an optical signal, or any appropriate combination thereof. The computer-readable signal medium may alternatively be any computer-readable medium other than the computer-readable medium. The computer-readable medium may send, propagate or transmit a program that is used by or used in combination with an instruction execution system, apparatus or device. The program code included in the computer-readable medium may be transmitted by using any suitable medium, including but not limited to: a wireless medium, a wired medium, or the like, or any suitable combination thereof.

[00260] The flowcharts and block diagrams in the accompanying drawings illustrate possible system architectures, functions, and operations that may be implemented by a system, a method, and a computer program product according to various embodiments of this application. In this regard, each box in the flowchart or the block diagram may represent a module, a program segment, or a part of code. The module, the program segment, or the part of code includes one or more executable instructions used for implementing specified logic functions. In some implementations used as substitutes, functions marked in boxes may alternatively occur in a sequence different from that marked in an accompanying drawing. For example, two boxes shown in succession may actually be performed basically in parallel, and sometimes the two boxes may be performed in a reverse sequence. This is determined by a related function. Each block in the block diagram or the flowchart, and a combination of blocks in the block diagram or the flowchart may be implemented by using a dedicated hardware-based system that performs a specified function or operation, or may be implemented by using a combination of dedicated hardware and computer instructions.

[00261] Although several modules or units of a device for action execution are mentioned in the foregoing detailed descriptions, the division is not mandatory. Actually, according to the implementations of this application, the features and functions of two or more modules or units described above may be specified in one module or unit. Conversely, the features and functions of one module or unit described above may be further divided into a plurality of modules or units to be embodied.

[00262] Through the description of the foregoing embodiments, a person skilled in the art can easily understand that the exemplary implementations described herein may be implemented by software, or may be implemented by combining software with necessary hardware. Therefore, the technical solutions of the embodiments of this application may be implemented in a form of a software product. The software product may be stored in a non-volatile storage medium (which may be a CD-ROM, a USB flash drive, a removable hard disk, or the like) or on a network, including several instructions for instructing a computing device (which may be a PC, a server, a touch terminal, a network device, or the like) to perform the methods according to the embodiments of this application.

[00263] Other implementation solutions of this application will be apparent to a person skilled in the art from consideration of the specification and practice of the disclosure here. This application is intended to cover any variation, use, or adaptive change of this application. These variations, uses, or adaptive changes follow the general principles of this application and include

common general knowledge or common technical means in the art that are not disclosed in this application.

[00264] It should be understood that this application is not limited to the precise structures described above and shown in the accompanying drawings, and various modifications and changes can be made without departing from the scope of this application. The scope of this application is subject only to the appended claims.

CLAIMS

1. A blockchain consensus method, performed by a current blockchain node in a blockchain network, the method comprising:

broadcasting consensus requests corresponding to at least two consensus stages in the blockchain network, the blockchain network comprising at least two blockchain nodes executing consensus algorithms of different types, the consensus algorithms of different types comprising a first consensus algorithm and a second consensus algorithm;

separately obtaining response messages broadcast by the blockchain network in the consensus stages, the response messages being messages from the blockchain nodes in response to the consensus requests;

counting a node quantity of blockchain nodes transmitting the response messages in a same consensus stage, the node quantity comprising a quantity of the at least two blockchain nodes executing the consensus algorithms of different types; and

in a case that the node quantity satisfies a consensus condition of the first consensus algorithm, reaching a consensus on a blockchain node executing the first consensus algorithm, and synchronizing a consensus result of the first consensus algorithm with a blockchain node executing the second consensus algorithm.

2. The blockchain consensus method according to claim 1, wherein the consensus algorithms of different types comprise consensus stages of different quantities, each response message carries an algorithm type and a node signature that are of the blockchain node, the algorithm type indicates a consensus algorithm executed by the blockchain node, and the node signature comprises a digital signature of the blockchain node in each consensus stage.

3. The blockchain consensus method according to claim 2, wherein the counting a node quantity of blockchain nodes transmitting the response messages in a same consensus stage comprises:

parsing each response message, to obtain the algorithm type and the node signature that are carried in the response message;

identifying, according to the algorithm type, a consensus algorithm executed by a message transmitting node, wherein the message transmitting node is the blockchain node transmitting the response message;

identifying a consensus stage of the message transmitting node according to the node signature;
and

counting, according to the consensus algorithm executed by the message transmitting node and the consensus stage of the message transmitting node, a node quantity of message transmitting nodes being in a same consensus stage with the current blockchain node.

4. The blockchain consensus method according to claim 3, wherein before the counting a node quantity of message transmitting nodes being in a same consensus stage with the current blockchain node, the method further comprises:

obtaining a matching relationship between consensus stages of the consensus algorithms of different types; and

determining, in a case that the message transmitting node and the current blockchain node execute the consensus algorithms of different types, the message transmitting node being in the same consensus stage with the current blockchain node according to the matching relationship.

5. The blockchain consensus method according to claim 4, wherein the first consensus algorithm comprises N consensus stages, and the second consensus algorithm comprises M consensus stages; and the matching relationship between the consensus stages of the consensus algorithms of different types comprises:

first P consensus stages of the first consensus algorithm and first P consensus stages of the second consensus algorithm match each other, and P is a smaller value of N-1 and M-1; and

last N-P consensus stages of the first consensus algorithm and last M-P consensus stages of the second consensus algorithm match each other.

6. The blockchain consensus method according to claim 3, wherein after the counting a node quantity of message transmitting nodes being in a same consensus stage with the current blockchain node, the method further comprises:

determining, in a case that the consensus stage of the current blockchain node is a last consensus stage of an executed consensus algorithm, a consensus result of the consensus algorithm based on whether the node quantity satisfies a consensus condition of the executed consensus algorithm; and

determining, in a case that the consensus stage of the current blockchain node is not a last consensus stage of an executed consensus algorithm, based on whether the node quantity satisfies the consensus condition of the executed consensus algorithm, whether to execute a next consensus

stage of the consensus algorithm.

7. The blockchain consensus method according to claim 1, wherein the first consensus algorithm is one of a low fault tolerance consensus algorithm and a high fault tolerance consensus algorithm, and the second consensus algorithm is one of the low fault tolerance consensus algorithm and the high fault tolerance consensus algorithm that is different from the first consensus algorithm; and

a consensus condition of the low fault tolerance consensus algorithm is that a quantity ratio of the blockchain nodes transmitting the response messages to all blockchain nodes is greater than a first ratio threshold, a consensus condition of the high fault tolerance consensus algorithm is that the quantity ratio of the blockchain nodes transmitting the response messages to all blockchain nodes is greater than a second ratio threshold, and the first ratio threshold is less than the second ratio threshold.

8. The blockchain consensus method according to claim 7, wherein the low fault tolerance consensus algorithm comprises consensus stages of a first quantity, the high fault tolerance consensus algorithm comprises consensus stages of a second quantity, and the first quantity is less than the second quantity.

9. The blockchain consensus method according to claim 8, wherein each consensus stage of the low fault tolerance consensus algorithm comprises a preparation stage and a submission stage that are executed in sequence, each consensus stage of the high fault tolerance consensus algorithm comprises a pre-preparation stage, a preparation stage, and a submission stage that are executed in sequence, the preparation stage of the low fault tolerance consensus algorithm matches the pre-preparation stage of the high fault tolerance consensus algorithm, and the submission stage of the low fault tolerance consensus algorithm matches the preparation stage and the submission stage that are of the high fault tolerance consensus algorithm.

10. The blockchain consensus method according to claim 1, wherein in a process of broadcasting consensus requests in the blockchain network, the method further comprises:

performing algorithm switching on some or all blockchain nodes in the blockchain network, to change consensus algorithms executed by the some or all blockchain nodes.

11. The blockchain consensus method according to claim 10, wherein the performing algorithm switching on some or all blockchain nodes in the blockchain network comprises:

performing credibility prediction on the blockchain network, to determine whether an operating environment of the blockchain network is a trusted environment;

switching, in a case that the operating environment of the blockchain network is a trusted environment, some or all blockchain nodes in the blockchain network from executing a low fault tolerance consensus algorithm in the blockchain network to executing a high fault tolerance consensus algorithm; and

switching, in a case that the operating environment of the blockchain network is a non-trusted environment, some or all blockchain nodes in the blockchain network from executing a high fault tolerance consensus algorithm in the blockchain network to executing a low fault tolerance consensus algorithm.

12. The blockchain consensus method according to claim 11, wherein the performing credibility prediction on the blockchain network comprises:

obtaining operating duration of each blockchain node in the blockchain network;

determining the blockchain node as a non-trusted node in a case that the operating duration is less than a duration threshold;

determining the operating environment of the blockchain network as a non-trusted environment in a case that a quantity of non-trusted nodes in the blockchain network is greater than a quantity threshold; and

determining the operating environment of the blockchain network as a trusted environment in a case that the quantity of non-trusted nodes in the blockchain network is less than the quantity threshold.

13. A blockchain consensus apparatus, comprising:

a request module, configured to broadcast consensus requests corresponding to at least two consensus stages in the blockchain network, the blockchain network comprising at least two blockchain nodes executing consensus algorithms of different types, the consensus algorithms of different types comprising a first consensus algorithm and a second consensus algorithm;

a response module, configured to separately obtain response messages broadcast by the blockchain network in the consensus stages, the response messages being messages from the blockchain nodes in response to the consensus requests;

a statistics module, configured to count a node quantity of blockchain nodes transmitting the response messages in a same consensus stage, the node quantity comprising a quantity of the at least two blockchain nodes executing the consensus algorithms of different types; and

a synchronization module, configured to: in a case that the node quantity satisfies a consensus

condition of the first consensus algorithm, reach a consensus on a blockchain node executing the first consensus algorithm, and synchronize a consensus result of the first consensus algorithm with a blockchain node executing the second consensus algorithm.

14. A computer-readable medium, storing a computer program, the computer program, when executed by a processor, implementing the blockchain consensus method according to any one of claims 1 to 12.

15. An electronic device, comprising:

a processor; and

a memory, configured to store executable instructions of the processor,

the processor being configured to enable, by executing the executable instructions, the electronic device to perform the blockchain consensus method according to any one of claims 1 to 12.

16. A computer program product, comprising a computer program, the computer program, when executed by a processor, implementing the blockchain consensus method according to any one of claims 1 to 12.