METHOD AND APPARATUS FOR VERIFYING BLOCK DATA IN A BLOCKCHAIN

TECHNICAL FIELD

[0001] The present application relates to the field of computer communications technologies, and in particular, to a method and an apparatus for verifying block data in a blockchain.

BACKGROUND

[0002] A blockchain technology is also referred to as a distributed ledger technology, and is a distributed Internet database technology. The blockchain technology is decentralized, transparent, and trustable, and cannot be tampered with. Each piece of data stored in a blockchain is broadcast and stored in each block in the blockchain. Therefore, full and consistent data is stored in each block in the blockchain.

[0003] In practice, data stored in all blocks needs to keep consistent in a blockchain. A data verification method is usually used to verify whether data stored in all blocks is consistent.

[0004] Specifically, during block data verification, verification values of all blocks need to be determined, and the verification values of all the blocks need to be compared, to determine whether data stored in all the blocks in the blockchain is consistent.

[0005] Researches show that when a verification value of each block is to be determined, because an amount of data stored in the block is relatively large, when the verification value of the block is to be determined, all data stored in the block needs to be calculated, to calculate the verification value of the block. However, as an amount of data stored in the block increases, an amount of verification value calculation increases, and consequently data verification efficiency decreases.

[0006] Therefore, how to improve block data verification efficiency becomes a problem that needs to be alleviated in the art urgently.

[0007] The information disclosed in the background is merely intended to enhance an understanding of the general background of the present application, and should not be considered as acknowledging or conceding that the information constitutes the existing technology known to a person skilled in the art.

SUMMARY

[0008] In view of this, implementations of the present application provide a method and an apparatus for verifying block data, so as to alleviate a problem of relatively low block data verification efficiency in the existing technology.

[0009] An implementation of the present application provides a method for verifying block data, including: selecting, from a block including a plurality of data nodes, a data node configured to store newly added data; determining a verification value of the newly added data for the data node, and calculating an update verification value of the data node based on the verification value of the newly added data; and obtaining a verification value of the block based on the update verification value of the data node and original verification values of other data nodes included in the block; where the verification value of the block is used to verify whether data stored in the block is consistent with data stored in other blocks in a blockchain.

[0010] An implementation of the present application further provides an apparatus for verifying block data, including: a selection module, configured to select, from a block including a plurality of data nodes, a data node configured to store newly added data; a determining module, configured to determine a verification value of the newly added data for the data node, and calculate an update verification value of the data node based on the verification value of the newly added data; and a processing module, configured to obtain a verification value of the block based on the update verification value of the data node and original verification values of other data nodes included in the block; where the verification value of the block is used to verify whether data stored in the block is consistent with data stored in other blocks in a blockchain.

[0011] The previously described at least one technical solution used in the implementations of the present application can achieve the following beneficial effects:

[0012] When the new data is added to the block, the update verification value of

the data node configured to store the newly added data is determined by using the newly added data, so as to alleviate a problem in the existing technology that much time is consumed because a verification value is calculated by using all data in a block when new data is added to the block. As such, when the verification value of the block that the new data is added to is calculated, consumed time is irrelevant to an amount of data in the block, and time for calculating the verification value is effectively shortened, improving block data verification efficiency.

BRIEF DESCRIPTION OF DRAWINGS

[0013] The accompanying drawings described here are intended to provide a further understanding of the present application, and constitute a part of the present application. The illustrative implementations of the present application and descriptions of the implementations of the present application are intended to describe the present application, and do not constitute limitations on the present application. In the accompanying drawings:

[0014] FIG. 1 is a schematic flowchart illustrating a method for verifying block data, according to an implementation of the present application;

[0015] FIG. 2 is a schematic structural diagram illustrating a Merkle tree, according to an implementation of the present application;

[0016] FIG. 3 is a schematic flowchart of determining a verification value of newly added data, according to an implementation of the present application;

[0017] FIG. 4 is a schematic flowchart of determining a verification value of newly added data, according to an implementation of the present application; and

[0018] FIG. 5 is a schematic structural diagram illustrating an apparatus for verifying block data, according to an implementation of the present application.

DESCRIPTION OF IMPLEMENTATIONS

[0019] To achieve objectives of the present application, the implementations of the present application provide a method and an apparatus for verifying block data. A data node configured to store newly added data is selected from a block including a plurality of data nodes; a verification value of the newly added data is determined for the data node, and an update verification value of the data node is calculated based on

the verification value of the newly added data; and a verification value of the block is obtained based on the update verification value of the data node and original verification values of other data nodes included in the block. The verification value of the block is used to verify whether data stored in the block is consistent with data stored in other blocks in a blockchain. Compared with the existing technology for processing all historical data, in the implementations of the present application, the newly added data of the block is mainly considered, and for the block that the new data is added to, the verification value is calculated by using an incremental method, so as to alleviate a problem in the existing technology that much time is consumed because a verification value is calculated by using all data in a block when new data is added to the block. As such, when the verification value of the block that the new data is added to is calculated, consumed time is irrelevant to an amount of data in the block, and time for calculating the verification value is effectively shortened, improving block data verification efficiency.

[0020] It is worthwhile to note that a predetermined algorithm used to calculate a verification value that is described in the implementations of the present application can be various verification algorithms, or can be various irreversible algorithms, and a specific algorithm type is not specifically limited in the implementations of the present application.

[0021] For example, a needed verification value can be calculated by using a hash algorithm. The hash algorithm can also be referred to as a hash function. The hash function is a unidirectional cryptographic system, in other words, the hash function is an irreversible mapping from plaintext to ciphertext, in other words, the function includes only an encryption process and does not include a decryption process. In addition, the hash function can obtain an output of a fixed length after changing an input of any length. A typical hash algorithm includes MD2, MD4, MD5, SHA-1, etc.

[0022] A person skilled in the art should understand that the hash algorithm mentioned in the specification of the implementations of the present application is merely an example, and does not constitute a limitation on the present disclosure. A person skilled in the art can use other algorithms used for verification based on actual applications and needs. This is not specifically limited here.

[0023] To make the objectives, technical solutions, and advantages of the present application clearer, the following clearly and comprehensively describes the technical solutions of the present application with reference to specific implementations of the

present application and corresponding accompanying drawings. Apparently, the described implementations are only some rather than all of the implementations of the present application. All other implementations obtained by a person of ordinary skill in the art based on the implementations of the present application without creative efforts shall fall within the protection scope of the present application.

[0024] The following describes in detail the technical solutions provided in the implementations of the present application with reference to the accompanying drawings.

[0025] FIG. 1 is a schematic flowchart illustrating a method for verifying block data, according to an implementation of the present application. The method for verifying block data includes the following steps.

[0026] Step S110: Select, from a block including a plurality of data nodes, a data node configured to store newly added data.

[0027] A blockchain is obtained by connecting several blocks by using a chain data structure. Each block includes a plurality of data nodes. The data nodes included in the block can be constructed in a form of a Merkle tree, in other words, each include a root node and a plurality of leaf nodes.

[0028] In the Merkle tree structure, the data nodes included in the block are used as bottom leaf nodes in the Merkle tree structure, which are shown in FIG. 2.

[0029] It can be seen from FIG. 2 that in the Merkle tree structure, A, B, C, D, and E correspond to the data nodes in the block.

[0030] A person skilled in the art should understand that the Merkle tree is constructed based on the data nodes in the block, and the depth of the Merkle tree can be determined as needed.

[0031] It is worthwhile to note that to ensure consistency between data stored in all blocks in the blockchain, Merkle trees constructed in all the blocks in the blockchain have a same depth, and quantities of data nodes included in all the blocks are also the same. In addition, it should be further understood that the Merkle tree listed here is merely an example for description, and a person skilled in the art can use other appropriate data construction forms. This is not specifically limited here.

[0032] For data to be stored (the data here can be referred to as newly added data), a block used to store the newly added data is first determined, and then a data node in the block that is configured to store the newly added data is determined.

[0033] Therefore, when the newly added data is received, block information used

to store the newly added data is first determined. The block information includes a block identifier and a data node identifier.

[0034] Specifically, data content corresponding to the newly added data is calculated, to obtain the determined block information.

[0035] In this implementation of the present application, the data content corresponding to the newly added data can be determined based on a type of the newly added data, or can be determined based on data included in the newly added data. This is not specifically limited here.

[0036] For example, if the newly added data is text data, the data content corresponding to the newly added data can be understood as a content digest of the text data, or can be understood as the text data, and the data content (for example, the content digest or the text data) corresponding to the newly added data can be calculated based on a predetermined algorithm, to obtain a calculation result. In this case, the block information is determined based on the calculation result.

[0037] For another example, if the newly added data is one piece of transaction data, the transaction data can be calculated based on a predetermined algorithm, to obtain a calculation result. In this case, the block information is determined based on the calculation result.

[0038] In the predetermined algorithm described in this implementation of the present application, for example, in a remainder calculation method, the data content is converted into integer data, a quantity of blocks included in the blockchain is used as a divisor, and the integer data is calculated by using the divisor, to obtain a quotient and a remainder. The quotient here can be considered as a block identifier, and the remainder can be considered as a data node identifier in the block. Alternatively, the quotient can be considered as the data node identifier, and the remainder can be considered as the block identifier.

[0039] Certainly, to assign the newly added data to an appropriate block and data node as needed, a person skilled in the art can also use other methods for calculating the data content, for example, performing random calculation on the data content. This is not limited here.

[0040] Then, the blockchain is searched based on the block identifier for a block corresponding to the block identifier, and a data node corresponding to the data node identifier is determined from the found block.

[0041] In a process of obtaining the blockchain, one block identifier is determined

for each block. Therefore, when the block information is obtained, the blockchain can be searched based on the block identifier in the block information for the block corresponding to the block identifier.

[0042] Each block includes a plurality of data nodes, and one data node identifier can be determined for each data node. Therefore, when the block information is obtained, the data node corresponding to the data node identifier can be determined from the found blockchain based on the data node identifier in the block information.

[0043] Finally, the found data node is used as the data node configured to store the newly added data.

[0044] Step S120: Determine a verification value of the newly added data for the data node.

[0045] In this implementation of the present application, because data possibly has been stored in the data node before the newly added data is stored in the data node, and some of these data is possibly associated with the newly added data, the data associated with the newly added data can be referred to as associated data. Alternatively, the data stored in the data node is possibly not associated with the newly added data, in other words, the data node does not include associated data associated with the newly added data.

[0046] It is worthwhile to note that the association relationship described in this implementation of the present application can be a relationship generated because of a user account, a payment activity, a transaction time, a transaction location, etc., for example, two pieces of transaction data of one user. In this case, the two pieces of transaction data are associated because of a same user account. Alternatively, the association relationship can be a relationship generated because of a transaction behavior. For example, content recorded in one piece of transaction data is that a user A transfers 100 yuan to a user B, and content recorded in the other piece of transaction data is that the user B transfers 100 yuan to the user A. In this case, the two pieces of transaction data are associated because objects and money included in the transfer behavior are the same. Specific content of the association relationship is not specifically limited here.

[0047] In addition, no data possibly has been stored in the data node before the newly added data is stored in the data node. In this case, it can be considered that the data node does not include associated data associated with the newly added data.

[0048] Therefore, determining the verification value of the newly added data can

include but is not limited to the following two cases:

[0049] Case 1: The data node does not include associated data associated with the newly added data.

[0050] For determining the verification value of the newly added data, reference can be made to FIG. 3. FIG. 3 is a schematic flowchart of determining a verification value of newly added data, according to an implementation of the present application.

[0051] S310. Determine a character string corresponding to the newly added data.

[0052] In this implementation of the present application, the character string corresponding to the newly added data can be determined by using any appropriate technical methods of converting data into a character string, for example, by converting the newly added data into the character string by using a corresponding transfer function. This is not specifically limited here.

[0053] S320. Calculate the character string based on a predetermined algorithm, to obtain the verification value of the newly added data.

[0054] Specifically, the character string is calculated by using a hash algorithm, to obtain a hash value of the newly added data, and use the hash value as the verification value of the newly added data.

[0055] A person skilled in the art should understand that the hash algorithm mentioned here is merely an example, and does not constitute a limitation on the present disclosure.

[0056] Case 2: The data node includes associated data associated with the newly added data.

[0057] For determining the verification value of the newly added data, reference can be made to FIG. 4. FIG. 4 is a schematic flowchart of determining a verification value of newly added data, according to an implementation of the present application.

[0058] S410. Separately determine a first character string corresponding to the associated data and a second character string corresponding to the newly added data.

[0059] Likewise, the first character string corresponding to the associated data and the second character string corresponding to the newly added data can be determined by using any appropriate technical methods of converting data into a character string, for example, by converting the newly added data into the character string by using a corresponding transfer function. This is not specifically limited here.

[0060] S420. Sort the first character string and the second character string by using a predetermined method.

[0061] It is worthwhile to note that the predetermined method can be determined based on an actual need, or can be determined based on an experimental result. A sorting method used in this implementation of the present application is not limited. For example, the first character string and the second character string are sequentially sorted, to combine the first character string and the second character string to obtain a character string group used for calculation. Whether the first character string is before the second character string or the second character string is before the first character string is not limited in this implementation of the present application. A person skilled in the art can also use other sorting methods.

[0062] S430. Calculate a sorted character string based on a predetermined algorithm, to obtain the verification value of the newly added data.

[0063] Specifically, the sorted character string is calculated by using a hash algorithm, to obtain a hash value of the newly added data, and use the hash value as the verification value of the newly added data.

[0064] Step S130: Calculate an update verification value of the data node based on the verification value of the newly added data.

[0065] Specifically, an original verification value of the data node obtained before the newly added data is stored; and the update verification value of the data node is calculated based on the original verification value and the verification value of the newly added data.

[0066] In this implementation of the present application, if other data has been stored before the newly added data is stored in the data node, a verification value is calculated for the data node when the other data is stored. The verification value can be referred to as an original verification value for the newly added data.

[0067] For example, for a data node A, one verification value is calculated each time one piece of data is stored, in other words, the verification value of the data node changes with data in the data node. Assume that data 11 and data 12 have been stored in the data node A before the newly added data is stored, a verification value a is calculated when the data 11 is stored in the data node, and a can be referred to as an original verification value when the data 12 is stored in the data node. In this case, a verification value b is calculated by using the original verification value, and b can be referred to as an update verification value.

[0068] Assume that the newly added data is data 13, a verification value c of the data 13 is obtained according to step S120. In this case, a verification value, namely,

the verification value b, of the data node obtained before the data 13 is stored needs to be determined, and the verification value b is referred to as an original verification value. The update verification value of the data node is calculated based on a predetermined algorithm by using the verification value c and the verification value b.

[0069] It is worthwhile to note that if no data has been stored in the data node, and the newly added data is used as first data of the data node, the verification value of the newly added data can be considered as the verification value of the data node in this implementation of the present application.

[0070] Step S140: Obtain a verification value of the block based on the update verification value of the data node and original verification values of other data nodes included in the block.

[0071] In this implementation of the present application, a tree structure corresponding to the plurality of data nodes included in the block is determined, and the original verification value of the other data nodes included in the block is determined; and the tree structure is traversed based on the update verification value of the data node and the original verification values of the other data nodes included in the block, to obtain a verification value of a root node of the tree structure, and use the verification value of the root node as the verification value of the block.

[0072] Content shown in FIG. 2 is used as an example. Assume that no newly added data has been stored in the block shown in FIG. 2, a verification value corresponding to the data node A shown in FIG. 2 is a, a verification value corresponding to the data node B is b, a verification value corresponding to the data node C is c, a verification value corresponding to the data node D is d, and a verification value corresponding to the data node E is e.

[0073] In this case, the newly added data needs to be stored in the data node A, and therefore the verification value a, the verification value b, the verification value c, the verification value d, and the verification value e are referred to as original verification values.

[0074] After the newly added data is stored in the data node A, the verification value of the data node A changes. Assume that the verification value of the data node A changes from the verification value a to a verification value g, the verification value g can be referred to as an update verification value of the data node A.

[0075] In this case, a verification value of the block is calculated based on a predetermined algorithm by using the verification value g of the data node A, the

verification value b of the data node B, the verification value c of the data node C, the verification value d of the data node D, and the verification value e of the data node E.

[0076] For example, the verification value of the block is calculated based on the tree structure that is corresponding to the data node A, the data node B, the data node C, the data node D, and the data node E and that is shown in FIG. 2 and by using the verification value g of the data node A, the verification value b of the data node B, the verification value c of the data node C, the verification value d of the data node D, and the verification value e of the data node E.

[0077] After the verification value of the block is obtained, the verification value can be compared with verification values of other blocks in a blockchain. If the two verification values are consistent, it indicates that data stored in the block is consistent with data stored in the other blocks, and the verification succeeds. If the two verification values are inconsistent, it indicates that data stored in the block is inconsistent with data stored in the other blocks. For example, the data in the block possibly has been lost or tampered with, and the verification cannot succeed.

[0078] In this implementation of the present application, the verification value of the block that changes with the newly added data is quickly determined by using an incremental calculation method, so as to shorten time for calculating the verification value, and improves system computing performance.

[0079] According to the technical solutions provided in the implementations of the present application, a data node configured to store newly added data is selected from a block including a plurality of data nodes; a verification value of the newly added data is determined for the data node, and an update verification value of the data node is calculated based on the verification value of the newly added data; and a verification value of the block is obtained based on the update verification value of the data node and original verification values of other data nodes included in the block. The verification value of the block is used to verify whether data stored in the block is consistent with data stored in other blocks in a blockchain. Compared with the existing technology for processing all historical data, in the implementations of the present application, the newly added data of the block is mainly considered, and for the block that the new data is added to, the verification value is calculated by using an incremental method, so as to alleviate a problem in the existing technology that much time is consumed because a verification value is calculated by using all data in a block when new data is added to the block. As such, when the verification value of the

block that the new data is added to is calculated, consumed time is irrelevant to an amount of data in the block, and time for calculating the verification value is effectively shortened, improving block data verification efficiency.

[0080] In addition, if original data that is not processed is tampered with, although the tampering is not found when the verification value of the newly added data is calculated in the implementations of the present application, the data-tampering can be surely found when data update is performed in each block of the blockchain because of structure and properties of the blockchain. Therefore, final block data verification is not affected.

[0081] FIG. 5 is a schematic structural diagram illustrating an apparatus for verifying block data, according to an implementation of the present application. The apparatus for verifying block data includes a selection module 510, a determining module 520, and a processing module 530.

[0082] The selection module 510 is configured to select, from a block including a plurality of data nodes, a data node configured to store newly added data.

[0083] The determining module 520 is configured to determine a verification value of the newly added data for the data node, and calculate an update verification value of the data node based on the verification value of the newly added data.

[0084] The processing module 530 is configured to obtain a verification value of the block based on the update verification value of the data node and original verification values of other data nodes included in the block.

[0085] The verification value of the block is used to verify whether data stored in the block is consistent with data stored in other blocks in a blockchain.

[0086] In another implementation of the present application, that the selection module 510 selects, from a block including a plurality of data nodes, a data node configured to store newly added data includes the following: when the newly added data is received, determining block information used to store the newly added data, where the block information includes a block identifier and a data node identifier; searching, based on the block identifier, the blockchain for a block corresponding to the block identifier, and determining, from the found block, a data node corresponding to the data node identifier; and using the found data node as the data node configured to store the newly added data.

[0087] In another implementation of the present application, that the selection module 510 determines block information used to store the newly added data includes

the following: calculating data content corresponding to the newly added data, to obtain the determined block information.

[0088] In another implementation of the present application, that the determining module 520 determines a verification value of the newly added data includes the following: when it is determined that the data node does not include associated data of the newly added data, determining a character string corresponding to the newly added data; and calculating the character string based on a predetermined algorithm, to obtain the verification value of the newly added data.

[0089] In another implementation of the present application, that the determining module 520 determines a verification value of the newly added data includes the following: when it is determined that the data node includes associated data of the newly added data, separately determining a first character string corresponding to the associated data and a second character string corresponding to the newly added data; sorting the first character string and the second character string by using a predetermined method; and calculating a sorted character string based on a predetermined algorithm, to obtain the verification value of the newly added data.

[0090] In another implementation of the present application, that the determining module 520 calculates an update verification value of the data node based on the verification value of the newly added data includes the following: determining an original verification value of the data node obtained before the newly added data is stored; and calculating the update verification value of the data node based on the original verification value and the verification value of the newly added data.

[0091] In another implementation of the present application, that the processing module 530 obtains a verification value of the block based on the update verification value of the data node and original verification values of other data nodes included in the block includes the following: determining a tree structure corresponding to the plurality of data nodes included in the block, and determining the original verification values of the other data nodes included in the block; and traversing the tree structure based on the update verification value of the data node and the original verification values of the other data nodes included in the block, to obtain a verification value of a root node of the tree structure, and use the verification value of the root node as the verification value of the block.

[0092] It is worthwhile to note that the apparatus for verifying block data described in this implementation of the present application can be implemented by

using software, or can be implemented by using hardware. This is not specifically limited here. According to the apparatus for verifying block data, for the block that the new data is added to, the verification value is calculated by using an incremental method, so as to alleviate a problem in the existing technology that much time is consumed because a verification value is calculated by using all data in a block when new data is added to the block. As such, when the verification value of the block that the new data is added to is calculated, consumed time is irrelevant to an amount of data in the block, and time for calculating the verification value is effectively shortened, improving block data verification efficiency.

[0093] In the 1990s, improvement of a technology can be clearly distinguished between hardware improvement (for example, improvement on a circuit structure such as a diode, a transistor, or a switch) and software improvement (improvement on a method procedure). However, with the development of technologies, improvement of many method procedures can be considered as direct improvement of a hardware circuit structure. Designers almost all program an improved method procedure to a hardware circuit, to obtain a corresponding hardware circuit structure. Therefore, it cannot be said that improvement of a method procedure cannot be implemented by using a hardware entity module. For example, a programmable logic device (PLD) (for example, a field programmable gate array (FPGA)) is such an integrated circuit. A logical function of the programmable logic device is determined by component programming executed by a user. The designers perform voluntary programming to "integrate" a digital system into a single PLD without requiring a chip manufacturer to design and produce a dedicated integrated circuit chip. In addition, instead of manually producing an integrated circuit chip, the programming is mostly implemented by "logic compiler" software, which is similar to a software compiler used during program development. Original code before compiling is also written in a specific programming language, which is referred to as a hardware description language (HDL), and there is more than one type of HDL, such as ABEL (Advanced Boolean Expression Language), AHDL (Altera Hardware Description Language), Confluence, CUPL (Cornell University Programming Language), HDCal, JHDL (Java Hardware Description Language), Lava, Lola, MyHDL, PALASM, and RHDL (Ruby Hardware Description Language), etc. Currently, VHDL (Very-High-Speed Integrated Circuit Hardware Description Language) and Verilog are most commonly used. A person skilled in the art should also understand

that a method procedure only needs to be logically programmed, and programmed to the integrated circuit by using the previous hardware description languages so that a hardware circuit that implements the logical method procedure can be easily obtained.

[0094] A controller can be implemented by using any appropriate method. For example, the controller can be a microprocessor or a processor, or a computer-readable medium, a logic gate, a switch, an application-specific integrated circuit (ASIC), a programmable logic controller, or an embedded microprocessor that stores computer readable program code (such as software or firmware) that can be executed by the microprocessor or the processor. Examples of the controller include but are not limited to the following microprocessors: ARC 625D, Atmel AT91SAM, Microchip PIC18F26K20, and Silicone Labs C8051F320. The memory controller can also be implemented as a part of the control logic of the memory. A person skilled in the art also knows that a controller can be implemented by using pure computer-readable program code, and the steps in the method can be logically programmed to enable the controller to further implement same functions in forms of a logic gate, a switch, an application-specific integrated circuit, a programmable logic controller, an embedded microcontroller, etc. Therefore, the controller can be considered as a hardware component, and an apparatus that is included in the controller and that is configured to implement various functions can also be considered as a structure in the hardware component. Alternatively, an apparatus configured to implement various functions can be considered as both a software module for implementing the method and a structure in the hardware component.

[0095] The system, apparatus, module, or unit described in the described implementations can be implemented by a computer chip or an entity, or implemented by a product with a certain function. A typical implementation device is a computer. The computer can be, for example, a personal computer, a laptop computer, a cellular phone, a camera phone, a smartphone, a personal digital assistant, a media player, a navigation device, an email device, a game console, a tablet computer, or a wearable device, or a combination of any of these devices.

[0096] For ease of description, the described apparatus is described by dividing functions into various units. Certainly, when the present application is implemented, the functions of the units can be implemented in one or more pieces of software and/or hardware.

[0097] A person skilled in the art should understand that the implementations of

the present disclosure can be provided as a method, a system, or a computer program product. Therefore, the present disclosure can use a form of hardware only implementations, software only implementations, or implementations with a combination of software and hardware. In addition, the present disclosure can use a form of a computer program product that is implemented on one or more computer-usable storage media (including but not limited to a disk memory, a CD-ROM, and an optical memory) that include computer-usable program code.

[0098] The present disclosure is described with reference to the flowcharts and/or block diagrams of the method, the device (system), and the computer program product according to the implementations of the present disclosure. It should be understood that computer program instructions can be used to implement each process and/or each block in the flowcharts and/or the block diagrams and a combination of a process and/or a block in the flowcharts and/or the block diagrams. These computer program instructions can be provided for a general-purpose computer, a dedicated computer, an embedded processor, or a processor of any other programmable data processing device to generate a machine so that the instructions executed by a computer or a processor of any other programmable data processing device generate an apparatus for implementing a specific function in one or more processes in the flowcharts or in one or more blocks in the block diagrams.

[0099] These computer program instructions can be stored in a computer readable memory that can instruct the computer or any other programmable data processing device to work in a specific method, so that the instructions stored in the computer readable memory generate an artifact that includes an instruction apparatus. The instruction apparatus implements a specific function in one or more processes in the flowcharts and/or in one or more blocks in the block diagrams.

[0100] These computer program instructions can be loaded to a computer or other programmable data processing devices, so that a series of operations and steps are performed on the computer or the other programmable devices, generating computer-implemented processing. Therefore, the instructions executed on the computer or the other programmable devices provide steps for implementing a specific function in one or more processes in the flowcharts or in one or more blocks in the block diagrams.

[0101] In a typical configuration, the computing device includes one or more processors (CPU), one or more input/output interfaces, one or more network

interfaces, and one or more memories.

[0102] The memory can include a non-persistent memory, a random access memory (RAM) and/or a nonvolatile memory in a computer readable medium, for example, a read-only memory (ROM) or a flash memory (flash RAM).

[0103] The computer readable medium includes persistent, non-persistent, movable, and unmovable media that can implement information storage by using any method or technology. Information can be a computer readable instruction, a data structure, a program module, or other data. A computer storage medium includes but is not limited to a phase-change random access memory (PRAM), a static random access memory (SRAM), a dynamic random access memory (DRAM), a random access memory (RAM) of another type, a read-only memory (ROM), an electrically erasable programmable read-only memory (EEPROM), a flash memory or another memory technology, a compact disc read-only memory (CD-ROM), a digital versatile disc (DVD) or another optical storage, a magnetic tape, a magnetic disk storage, another magnetic storage device, or any other non-transmission media that can be used to store information that can be accessed by the computing device. Based on the definition in the present specification, the computer readable medium does not include transitory computer readable medium(transitory media), for example, a modulated data signal and carrier.

[0104] It is worthwhile to further note that the term "include", "contain", or their any other variant is intended to cover a non-exclusive inclusion, so that a process, a method, merchandise, or a device that includes a list of elements not only includes those elements but also includes other elements which are not expressly listed, or further includes elements inherent to such process, method, merchandise, or device. An element preceded by "includes a ..." does not, without more constraints, preclude the existence of additional identical elements in the process, method, merchandise, or device that includes the element.

[0105] A person skilled in the art should understand that the implementations of the present application can be provided as a method, a system, or a computer program product. Therefore, the present application can use a form of hardware only implementations, software only implementations, or implementations with a combination of software and hardware. In addition, the present application can use a form of a computer program product that is implemented on one or more computer-usable storage media (including but not limited to a disk memory, a

CD-ROM, and an optical memory) that include computer-usable program code.

[0106] The previous descriptions are merely implementations of the present application, and are not intended to limit the present application. A person skilled in the art can make various modifications and changes to the present application. Any modification, equivalent replacement, improvement, etc. made within the spirit and principle of the present application shall fall within the scope of the claims of the present application.

CLAIMS

What is claimed is:

1. A method for verifying block data, comprising:

selecting, from a block comprising a plurality of data nodes, a data node configured to store newly added data;

determining a verification value of the newly added data for the data node;

calculating an update verification value of the data node based on the verification value of the newly added data; and

obtaining a verification value of the block based on the update verification value of the data node and original verification values of other data nodes comprised in the block;

wherein the verification value of the block is used to verify whether data stored in the block is consistent with data stored in other blocks in a blockchain.

2. The method for verifying block data according to claim 1, wherein the selecting, from a block comprising a plurality of data nodes, a data node configured to store newly added data comprises the following:

in response to receiving the newly added data, determining block information used to store the newly added data, wherein the block information comprises a block identifier and a data node identifier;

searching, based on the block identifier, the blockchain for a block corresponding to the block identifier;

determining, from the found block, a data node corresponding to the data node identifier; and

using the found data node as the data node configured to store the newly added data.

3. The method for verifying block data according to claim 2, wherein the determining block information used to store the newly added data comprises the following:

calculating data content corresponding to the newly added data, to obtain the determined block information.

4. The method for verifying block data according to claim 1, wherein the determining a verification value of the newly added data comprises the following:

in response to determining that the data node does not comprise associated data of the newly added data, determining a character string corresponding to the newly added data; and

calculating the character string based on a predetermined algorithm, to obtain the verification value of the newly added data.

5. The method for verifying block data according to claim 1, wherein the determining a verification value of the newly added data comprises the following:

in response to determining that the data node comprises associated data of the newly added data, separately determining a first character string corresponding to the associated data and a second character string corresponding to the newly added data;

sorting the first character string and the second character string by using a predetermined method; and

calculating a sorted character string based on a predetermined algorithm, to obtain the verification value of the newly added data.

6. The method for verifying block data according to claim 1, wherein the calculating an update verification value of the data node based on the verification value of the newly added data comprises the following:

determining an original verification value of the data node obtained before the newly added data is stored; and

calculating the update verification value of the data node based on the original verification value and the verification value of the newly added data.

7. The method for verifying block data according to claim 1, wherein the obtaining a verification value of the block based on the update verification value of the data node and original verification values of other data nodes comprised in the block comprises the following:

determining a tree structure corresponding to the plurality of data nodes comprised in the block;

determining the original verification values of the other data nodes comprised in the block;

traversing the tree structure based on the update verification value of the data node and the original verification values of the other data nodes comprised in the block, to obtain a verification value of a root node of the tree structure; and

determining the verification value of the root node as the verification value of the block.

8. An apparatus for verifying block data, comprising:

a selection module, configured to select, from a block comprising a plurality of data nodes, a data node configured to store newly added data;

a determining module, configured to:

determine a verification value of the newly added data for the data node; and

calculate an update verification value of the data node based on the verification value of the newly added data; and

a processing module, configured to obtain a verification value of the block based on the update verification value of the data node and original verification values of other data nodes comprised in the block;

wherein the verification value of the block is used to verify whether data stored in the block is consistent with data stored in other blocks in a blockchain.

9. The apparatus for verifying block data according to claim 8, wherein that a selection module selects, from a block comprising a plurality of data nodes, a data node configured to store newly added data comprises the following:

in response to receiving the newly added data , determining block information used to store the newly added data, wherein the block information comprises a block identifier and a data node identifier;

searching, based on the block identifier, the blockchain for a block corresponding to the block identifier, and determining, from the found block, a data node corresponding to the data node identifier; and

using the found data node as the data node configured to store the newly added data.

10. The apparatus for verifying block data according to claim 9, wherein that the selection module determines block information used to store the newly added data comprises the following:

calculating data content corresponding to the newly added data, to obtain the determined block information.

11. The apparatus for verifying block data according to claim 8, wherein that a determining module determines a verification value of the newly added data comprises the following:

in response to determining that the data node does not comprise associated data of the newly added data, determining a character string corresponding to the newly added data; and

calculating the character string based on a predetermined algorithm, to obtain the verification value of the newly added data.

12. The apparatus for verifying block data according to claim 8, wherein that a determining module determines a verification value of the newly added data comprises the following:

in response to determining that the data node comprises associated data of the newly added data, separately determining a first character string corresponding to the associated data and a second character string corresponding to the newly added data;

sorting the first character string and the second character string by using a predetermined method; and

calculating a sorted character string based on a predetermined algorithm, to obtain the verification value of the newly added data.

13. The apparatus for verifying block data according to claim 8, wherein that a determining module calculates an update verification value of the data node based on the verification value of the newly added data comprises the following:

determining an original verification value of the data node obtained before the newly added data is stored; and

calculating the update verification value of the data node based on the original verification value and the verification value of the newly added data.

14. The apparatus for verifying block data according to claim 8, wherein that a processing module obtains a verification value of the block based on the update verification value of the data node and original verification values of other data nodes

comprised in the block comprises the following:

determining a tree structure corresponding to the plurality of data nodes comprised in the block;

determining the original verification values of the other data nodes comprised in the block;

traversing the tree structure based on the update verification value of the data node and the original verification values of the other data nodes comprised in the block, to obtain a verification value of a root node of the tree structure; and

determining the verification value of the root node as the verification value of the block.