# METHOD AND DEVICE FOR SENDING TRANSACTION INFORMATION AND FOR CONSENSUS VERIFICATION

## TECHNICAL FIELD

**[0001]** The present application relates to the field of blockchain technologies, and in particular, to a method and a device for sending transaction information and for consensus verification.

## BACKGROUND

**[0002]** In the field of blockchain technologies, a blockchain node responsible for performing consensus verification on transactions is referred to as a consensus node.

**[0003]** In a consensus verification phase, a consensus node that initiates consensus verification needs to pack transaction information of transactions generated within a time period into a preprocessed block and send the preprocessed block to another consensus node for consensus verification, and the another consensus node verifies whether a transaction pool of the another consensus node includes all transaction information in the preprocessed block. A verification result is that the consensus verification succeeds if the transaction pool includes all the transaction information in the preprocessed block. A verification result is that the consensus verification fails if the transaction pool does not include all the transaction information in the preprocessed block. Subsequently, consensus nodes reach a consensus that the transaction information in the preprocessed block is valid or invalid based on verification results of the consensus nodes on the preprocessed block, and use the consensus as a result of performing consensus verification by the consensus nodes on the preprocessed block. Therefore, it needs to be ensured that transaction information stored in transaction pools of the consensus nodes is consistent, to make the consensus verification result of the consensus nodes as accurate as possible.

**[0004]** In a transaction handling phase, for different transactions, each consensus node can serve as a transaction handling node (which is referred to as a handling node blow) to obtain transaction information of a transaction. For a transaction, a handling

node that corresponds to the transaction needs to send transaction information to each other consensus node, and another consensus node that receives the transaction information stores the transaction information in a transaction pool of the another consensus node. Consensus nodes maintain consistent transaction information that is stored in transaction pools of the consensus nodes by using such a method.

**[0005]** However, because network disturbances are always inevitable, the network instability often leads to the instability of information transmission between consensus nodes, and some consensus nodes may fail to receive sent information. For example, a handling node sends transaction information to other consensus nodes. If a network disturbance occurs, the transaction information sent by the handling node to the other consensus nodes may not be received by all the other consensus nodes. Consequently, transaction information stored in transaction pools of the consensus nodes cannot maintain consistent, thereby reducing the accuracy of a consensus verification result of the consensus nodes.

## SUMMARY

**[0006]** Implementations of the present application provide a method and a device for sending transaction information and for consensus verification, to alleviate the problem that the accuracy of a consensus verification result of consensus nodes is reduced by using an existing method for sending transaction information and performing consensus verification.

**[0007]** To alleviate the previous technical problem, the implementations of the present application are implemented as follows:

**[0008]** An implementation of the present application provides a method for sending transaction information, where the method includes the following: obtaining, by a consensus node, transaction information; sending the transaction information to another consensus node; and resending the transaction information based on a consensus verification failure notification when receiving the consensus verification failure notification that is sent by the another consensus node and that includes an information identifier of the transaction information.

**[0009]** An implementation of the present application provides a method for consensus verification, where the method includes the following: receiving, by a consensus node, a preprocessed block sent by another consensus node; performing

consensus verification on the preprocessed block based on transaction information stored in a transaction pool and transaction information included in the preprocessed block; and in a consensus verification process, if it is determined that at least one piece of transaction information included in the preprocessed block does not exist in the transaction pool, determining an information identifier of the transaction information missing from the transaction pool in the transaction information included in the preprocessed block, and sending a consensus verification failure notification that includes the information identifier to the other consensus nodes.

**[0010]** An implementation of the present application provides a device for sending transaction information, where the device includes the following: an acquisition module, configured to obtain transaction information; a first sending module, configured to send the transaction information to another consensus node; and a second sending module, configured to resend the transaction information based on a consensus verification failure notification when the consensus verification failure notification that is sent by the another consensus node and that includes an information identifier of the transaction information is received.

**[0011]** An implementation of the present application provides a device for consensus verification, where the device includes the following: a receiving module, configured to receive a preprocessed block sent by another consensus node; a consensus verification module, configured to perform consensus verification on the preprocessed block based on transaction information stored in a transaction pool and transaction information included in the preprocessed block; and a sending module, configured to: in a consensus verification process, if it is determined that at least one piece of transaction information included in the preprocessed block does not exist in the transaction pool, determine an information identifier of the transaction information missing from the transaction pool in the transaction information included in the preprocessed block, and send a consensus verification failure notification that includes the information identifier to the other consensus nodes.

**[0012]** It can be seen from the technical solutions provided in the previous implementations of the present application that, in the implementations of the present application, if another consensus node fails to receive transaction information sent by a handling node in a transaction handling phase, the another consensus node can send a consensus verification failure notification that includes an information identifier of the transaction information to the handling node in a consensus verification phase if

the another consensus node determines that the transaction information included in a preprocessed block does not exist in a transaction pool of the another consensus node, so that the handling node resends the transaction information to the another consensus node. According to the implementations of the present application, it can be ensured as much as possible that transaction information stored in transaction pools of consensus nodes is consistent, and thus the accuracy of a consensus verification result of the consensus nodes is not reduced.

## BRIEF DESCRIPTION OF DRAWINGS

**[0013]** To describe the technical solutions in the implementations of the present application or in the existing technology more clearly, the following briefly describes the accompanying drawings required for describing the implementations or the existing technology. Apparently, the accompanying drawings in the following description merely show some implementations of the present application, and a person of ordinary skill in the art can still derive other drawings from these accompanying drawings without creative efforts.

**[0014]** FIG. 1 is a flowchart illustrating a method for sending transaction information, according to an implementation of the present application;

**[0015]** FIG. 2 is a schematic diagram illustrating a reliable mechanism in a transaction handling phase, according to an implementation of the present application;

**[0016]** FIG. 3 is a flowchart illustrating a method for consensus verification, according to an implementation of the present application;

**[0017]** FIG. 4 is a schematic diagram illustrating a device for sending transaction information, according to an implementation of the present application; and

**[0018]** FIG. 5 is a schematic diagram illustrating a device for consensus verification, according to an implementation of the present application.

## DESCRIPTION OF IMPLEMENTATIONS

**[0019]** Implementations of the present application provide a method and a device for sending transaction information and for consensus verification.

**[0020]** To make a person skilled in the art understand the technical solutions in the present application better, the following clearly and comprehensively describes the

technical solutions in the implementations of the present application with reference to the accompanying drawings in the implementations of the present application. Apparently, the described implementations are merely some but not 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.

**[0021]** The technical solutions provided in the implementations of the present application are described in detail below with reference to the accompanying drawings.

**[0022]** FIG. 1 is a flowchart illustrating a method for sending transaction information, according to an implementation of the present application. The method includes the following steps.

**[0023]** S101. A consensus node obtains transaction information.

**[0024]** In the present implementation of the present application, the transaction information can be all details involved in a transaction, such as an account address, a transaction amount, and a transaction type.

**[0025]** In a transaction handling phase, the consensus node serves as a handling node, and can receive transaction information sent by a client device. The client device can be a client device of a blockchain node that participates in the transaction. Certainly, the consensus node can initiate a transaction by itself and generate transaction information of the transaction.

**[0026]** For ease of description, the handling node described below is a consensus node that handles a transaction in a handling phase, and is an execution body of the present implementation of the present application.

**[0027]** S102. Send the transaction information to another consensus node.

**[0028]** After obtaining the transaction information, the handling node stores the transaction information in a transaction pool of the handling node, and sends the transaction information to the another consensus node, so that the another consensus node stores the transaction information in a transaction pool of the another consensus node after receiving the transaction information. Then, in a consensus verification phase, each consensus node can verify a preprocessed block by determining whether transaction information stored in a transaction pool of the consensus node includes all transaction information in the preprocessed block.

**[0029]** The another consensus node is a consensus node other than the handling node. The transaction pool is a database used to store transaction information. Each consensus node has its own transaction pool. The transaction pool can be constructed in a memory of the consensus node or an external memory of the consensus node.

**[0030]** It is worthwhile to emphasize that each consensus node can carry out multi-process work, that is, simultaneously handle transaction A and participate in consensus verification on a preprocessed block (which does not include transaction A) that includes a batch of transaction information. As long as the computing capability is enough, the consensus node can simultaneously handle different transactions or perform consensus verification through a plurality of processes.

**[0031]** In the present implementation of the present application, to describe in detail how to ensure as much as possible that each other consensus node receives transaction information of a transaction, the transaction handling phase and the consensus verification phase are described relative to transaction information of the same transaction.

**[0032]** In the present implementation of the present application, the consensus node can create a thread for each other consensus node, and send the transaction information to the another consensus node through the thread. The consensus node can send the transaction information through the thread by using an asynchronous invocation technology. After the transaction information is sent, the thread is revoked regardless of whether the another consensus node can receive the transaction information. Alternatively, the consensus node can send the transaction information by using a synchronous invocation technology. After sending the transaction information, the consensus node continues to use the thread to wait for receiving a response signal returned by the another consensus node after the another consensus node receives the transaction information.

**[0033]** S103. Resend the transaction information based on a consensus verification failure notification when receiving the consensus verification failure notification that is sent by the another consensus node and that includes an information identifier of the transaction information.

**[0034]** In the present implementation of the present application, the consensus verification failure notification is sent by the another consensus node in the subsequent consensus verification phase. In the consensus verification phase, a consensus node (which is referred to as a leader node blow) that initiates a consensus

packs transaction information of a batch of transactions in a transaction pool of the consensus node into a preprocessed block, and sends the preprocessed block to a consensus node (which is referred to as a replica node) other than the leader node, so that consensus nodes (the leader node and replica nodes) perform consensus verification on the preprocessed block. The leader node can be elected by the consensus nodes, or can be randomly specified.

**[0035]** It is worthwhile to emphasize that the present implementation of the present application is performed by the handling node in the transaction handling phase. In the consensus verification phase, the handling node can be a leader node or a replica node. Implementations are not limited in the present application. In addition, the another consensus node is a consensus node other than the handling node in the handling phase, and the another consensus node usually can be a replica node or a leader node.

**[0036]** In brief, an identity of each consensus node may change accordingly after the transition from the transaction handling phase to the consensus verification phase. The previous handling node may become a replica node, and another previous consensus node may become a leader node.

**[0037]** In the present implementation of the present application, when verifying the preprocessed block, a replica node compares the transaction information in the preprocessed block with transaction information stored in a transaction pool of the replica node. If the transaction information stored in the transaction pool of the replica node includes all transaction information in the preprocessed block, it indicates that the replica node successfully receives the transaction information in the previous transaction handling phase.

**[0038]** If the transaction information stored in the transaction pool of the replica node does not include all transaction information in the preprocessed block, in other words, if some transaction information in the preprocessed block is missing from the transaction information stored in the transaction pool of the replica node, it indicates that the replica node has not successfully received the transaction information in the previous transaction handling phase. In this case, verification performed by the replica node on the preprocessed block fails, and the replica node broadcasts a consensus verification failure notification that includes an information identifier of the missing transaction information to other replica nodes and the leader node. Each consensus node that receives the consensus verification failure notification can check a

transaction pool of the consensus node, and send the transaction information that corresponds to the information identifier to the replica node that sends the consensus verification failure notification, if the transaction pool of the consensus node stores the transaction information that corresponds to the information identifier included in the consensus verification failure notification (the consensus node successfully receives the transaction information in the transaction handling phase or the consensus node is the handling node).

**[0039]** It is worthwhile to emphasize that there are usually a very large number of pieces of transaction information included in the preprocessed block, and the transaction information is usually broadcast by different handling nodes. Therefore, in the consensus verification phase, each consensus node may lack some transaction information included in the preprocessed block. The transaction information that each consensus node lacks is certainly stored in a transaction pool of a handling node that originally broadcasts the transaction information. In addition, the transaction information that each consensus node lacks can be stored in a transaction pool of more than one other consensus node. In other words, in the consensus verification phase, the transaction information that each consensus node lacks can be compensated for through joint assistance of other consensus nodes.

**[0040]** For example, there are five consensus nodes A, B, C, D, and E in a blockchain network. In a consensus verification phase, A that serves as a leader node generates a preprocessed block, and transaction information included in the preprocessed block is L, M, and N. L is broadcast by E when E serves as a handling node, M is broadcast by A when A serves as a handling node, and N is broadcast by D when D serves as a handling node. Assume that B and C fail to receive L in a handling phase of L, B and E fails to receive M in a handling phase of M, B fails to receive N in a handling phase of N. In this case, in a consensus verification phase, after A sends the preprocessed block to B, C, D, and E, B sends a consensus verification failure notification that includes information identifiers of L, M, and N to A, C, D and E because L, M, and N are missing from a transaction pool of B. As such, B separately receives L, M, and N sent by A; M and N sent by C; L, M, and N sent by D; and L and N sent by E. In other words, the transaction information missing from the transaction pool of B can be compensated for through consensus assistance of A, C, D, and E.

**[0041]** In addition, because of network disturbances, all consensus nodes that receive the consensus verification failure notification send the transaction information

to the replica node through compensation. In fact, it can be ensured that the replica node receives the transaction information in a large probability, thereby alleviating the network disturbances and ensuring the consistency in transaction pools of the consensus nodes.

**[0042]** In conclusion, in the present implementation of the present application, after sending the transaction information to the another consensus node, regardless of whether each other consensus node receives the transaction information, the consensus node can wait until the consensus verification phase to resend, through compensation, the transaction information to the another consensus node that sends the consensus verification failure notification based on the received consensus verification failure notification.

**[0043]** It can be seen that, according to the method for sending information shown in FIG. 1, if another consensus node fails to receive transaction information sent by a handling node in a transaction handling phase, the another consensus node can send a consensus verification failure notification that includes an information identifier of the transaction information to the handling node in a consensus verification phase if the another consensus node determines that the transaction information included in a preprocessed block does not exist in a transaction pool of the another consensus node, so that the handling node resends the transaction information to the another consensus node. According to the present implementation of the present application, it can be ensured as much as possible that transaction information stored in transaction pools of consensus nodes is consistent, and thus the accuracy of a consensus verification result of the consensus nodes is not reduced.

**[0044]** Further, in the present implementation of the present application, some reliable mechanisms can be used in the transaction handling phase to enhance the reliability of sending the transaction information by the consensus node to the another consensus node.

**[0045]** In step S102, the consensus node creates a thread for each other consensus node, and sends the transaction information to the another consensus node through the thread; and resends the transaction information to the another consensus node through the thread if it is determined that the another consensus node fails to receive the transaction information, until it is determined that the another consensus node receives the transaction information or a predetermined sending stop condition is satisfied.

**[0046]** It is determined that the another consensus node receives the transaction information if a response signal returned by the another consensus node is received through the thread within a specified time period. It is determined that the another consensus node fails to receive the transaction information if no response signal returned by the another consensus node is received through the thread within the specified time period.

**[0047]** The predetermined sending stop condition can be as follows: The number of times of sending the transaction information to the another consensus node reaches the predetermined number of times, or a time that elapsed since the first time the transaction information was sent to the another consensus node exceeds predetermined duration. The predetermined number of times and the predetermined duration are not limited in the present application.

**[0048]** It is worthwhile to note that the consensus node can resend the transaction information to the another consensus node through the thread each time by using the following method: The consensus node can resend the transaction information once it is determined that the previous sending fails, or can wait for a particular time interval to delay sending. The particular time interval can be configured, and the particular time interval for waiting for each resending can be the same or different. For example, the particular time interval gradually increases or decreases.

**[0049]** For example, the consensus node creates a thread for each other consensus node, and first sends the transaction information to the another consensus node through the thread. The consensus node can resend the transaction information after a delay of 5S if the another consensus node returns no response signal, can resend the transaction information after a delay of 15S if still no response signal is received, and can resend the transaction information after a delay of a longer time if still no corresponding signal is received. Resending of the transaction information continues like this way until the number of sending times reaches the predetermined number of times or the time that elapsed since the first time the transaction information was sent to the another consensus node exceeds the predetermined duration.

**[0050]** Particularly, in the previous example, if the number of times the consensus node sends the transaction information reaches the predetermined number of times, the consensus node can further add the transaction information to a back queue, and create a thread Recovery specially used for scanning the back queue. The Recovery scans the back queue every particular time interval to identify the transaction

information, and the transaction information is sent through the Recovery until the time that elapsed since the first time the transaction information was sent to the another consensus node exceeds the predetermined duration. Certainly, the particular time interval that the Recovery scans the back queue can also be configured. That is, the thread whose number of times of sending the transaction information reaches the predetermined number of times is revoked to save resources of the consensus node, the transaction information is added to the back queue, and the standing Recovery thread is responsible for scanning the back queue and sending the transaction information every particular time interval. As such, the resource waste caused by maintaining a relatively large number of threads by the consensus node can be alleviated.

**[0051]** In the previous example, the consensus node actually sends the transaction information to the another consensus node by using a weak synchronization invocation technology. After asynchronously invoking a thread for each other consensus node, the consensus node sends the transaction information to the another consensus node through the thread by using synchronization invocation. The thread occupies a computing resource of the consensus node, and the consensus node waits, through the thread, for the another consensus node to return the response signal. In addition, if the another consensus node fails to return the response signal within a specified time period, it indicates that the another consensus node fails to receive the transaction information, and the consensus node resends the transaction information to the another consensus node through the thread. As such, using the weak synchronization invocation technology in the transaction handling phase can make the consensus node do not need to put aside more important work processes to specially wait for the response signal returned by the another consensus node, and the consensus node can also use the thread that corresponds to each other consensus node to wait for the response signal and to repeatedly send the transaction information, thereby ensuring that the another consensus node successfully receives the transaction information.

**[0052]** In addition, in the present implementation of the present application, when the predetermined sending stop condition is satisfied, the transaction information can be added to the predetermined queue if the another consensus node still returns no response signal. When receiving the consensus verification failure notification, the queue is searched for the transaction information that corresponds to the information

identifier based on the information identifier included in the consensus verification failure notification, and the transaction information is sent.

**[0053]** The queue is storage space in the memory of the consensus node, and is configured to store transaction information that the consensus node repeatedly attempts to send in the handling phase but that is still not received by each other consensus node. The transaction information that the consensus node repeatedly attempts to send but that still fails to be received is stored in the queue when the transaction pool of the consensus node is not constructed in the memory, so that the consensus node can send the transaction information more quickly to the another consensus node that lacks the transaction information in the consensus verification phase.

**[0054]** Particularly, when the transaction pool is not constructed in the memory, in the consensus verification phase, when receiving a consensus verification failure notification sent by a replica node, the consensus node should check whether the queue of the consensus node includes transaction information that the replica node lacks, instead of checking the transaction pool of the consensus node; and can send the transaction information to the replica node if the queue of the consensus node includes the transaction information that the replica node lacks; or does not need to send the transaction information to the replica node if the queue of the consensus node does not include the transaction information that the replica node lacks (even if the transaction pool of the consensus node includes a transaction that the replica node lacks). As such, if the replica node lacks transaction information X, it can always be ensured that a handling node that broadcasts transaction information X obtains transaction information X from a queue of the handling node and sends transaction information X. Therefore, the replica node that lacks the transaction information can always be compensated quickly.

**[0055]** FIG. 2 is a schematic diagram illustrating a reliable mechanism in a transaction handling phase, according to an implementation of the present application. As shown in FIG. 2, in a transaction handling phase, a handling node attempts to send transaction information to another consensus node by using a method such as a delay of 5S or a delay of 10 min, and the method for attempting to send the transaction information is restrained. That is, the transaction information is sent by using a longer delay when the number of sending times is greater than 3. The transaction information is added to a predetermined queue and sending stops when an active period of the

transaction information (a time that elapsed since the first time the transaction information was sent to the another consensus node) exceeds predetermined duration (for example, one hour). In a consensus verification phase, the consensus node activates the transaction information in the queue only when receiving a consensus verification failure notification.

**[0056]** It is worthwhile to emphasize that the predetermined duration can be determined based on a period in which each consensus node performs consensus verification, or can be determined based on a troubleshooting period of a common network fault.

**[0057]** According to the reliable mechanism shown in FIG. 2, the consensus node delays attempting to send the transaction information, and after the consensus node repeatedly sends the transaction information after a delay of 5S two times, it indicates that there is a transmission problem that cannot be resolved temporarily (which may be that another consensus node that has not successfully received the transaction information goes down or a large fault occurs in a network). Therefore, the consensus node temporarily stops sending the transaction information to the another consensus node. After waiting for a relatively long time (for example, 10 min), the transmission problem may have been alleviated, and then the consensus node continues to send the transaction information. If the active period of the transaction information is too long, the transaction information can be added to the predetermined queue to wait for subsequently resending, through compensation, the transaction information to the another consensus node that lacks the transaction information.

**[0058]** In the present implementation of the present application, in the consensus verification phase, consensus nodes jointly compensate each consensus node that lacks the transaction information, so that the problem of the inconsistency in transaction pools of the consensus nodes that is caused by network disturbances can be well alleviated. On this basis, according to the reliable mechanism shown in FIG. 2, measures can be used in the transaction handling phase to increase a success rate of receiving the transaction information by the another consensus node.

**[0059]** FIG. 3 is a method for consensus verification, according to an implementation of the present application. The method includes the following steps:

**[0060]** S301. A consensus node receives a preprocessed block sent by another consensus node.

**[0061]** S302. Perform consensus verification on the preprocessed block based on

transaction information stored in a transaction pool and transaction information included in the preprocessed block.

**[0062]** S303. In a consensus verification process, if it is determined that at least one piece of transaction information included in the preprocessed block does not exist in the transaction pool, determine an information identifier of the transaction information missing from the transaction pool in the transaction information included in the preprocessed block.

**[0063]** S304. Send a consensus verification failure notification that includes the information identifier to the other consensus nodes.

**[0064]** In the present implementation of the present application, the consensus node is a consensus node that verifies the preprocessed block in the consensus verification phase, namely, a replica node. The another consensus node is a node that initiates a consensus, namely, a leader node.

**[0065]** Detailed description of the method for consensus verification shown in FIG. 3 has been recorded in the description of the method for sending transaction information shown in FIG. 1. Details are omitted here for simplicity.

**[0066]** Based on the method for sending transaction information shown in FIG. 1, an implementation of the present application further correspondingly provides a device for sending transaction information. As shown in FIG. 4, the device includes the following: an acquisition module 401, configured to obtain transaction information; a first sending module 402, configured to send the transaction information to another consensus node; and a second sending module 403, configured to resend the transaction information based on a consensus verification failure notification when the consensus verification failure notification that is sent by the another consensus node and that includes an information identifier of the transaction information is received.

**[0067]** The acquisition module 401 is configured to receive transaction information sent by a client device.

**[0068]** The first sending module 402 is configured to create a thread for each other consensus node, and send the transaction information to the another consensus node through the thread.

**[0069]** The first sending module 402 is configured to resend the transaction information to the another consensus node through the thread if it is determined that the another consensus node fails to receive the transaction information, until it is

determined that the another consensus node receives the transaction information or a predetermined sending stop condition is satisfied.

**[0070]** The first sending module 402 is configured to determine that the another consensus node receives the transaction information if a response signal returned by the another consensus node is received through the thread within a specified time period; or determine that the another consensus node fails to receive the transaction information if no response signal returned by the another consensus node is received through the thread within the specified time period.

**[0071]** The predetermined sending stop condition can include the following: The number of times of sending the transaction information to the another consensus node reaches the predetermined number of times, or a time that elapsed since the first time the transaction information was sent to the another consensus node exceeds predetermined duration.

**[0072]** The device further includes an adding module 404, configured to add the transaction information to a predetermined queue when the predetermined sending stop condition is satisfied.

**[0073]** The second sending module 403 is configured to search the queue for the transaction information that corresponds to the information identifier based on the information identifier included in the consensus verification failure notification, and send the transaction information.

**[0074]** Based on the method for consensus verification shown in FIG. 3, an implementation of the present application further correspondingly provides a device for consensus verification. As shown in FIG. 5, the device includes the following: a receiving module 501, configured to receive a preprocessed block sent by another consensus node; a consensus verification module 502, configured to perform consensus verification on the preprocessed block based on transaction information stored in a transaction pool and transaction information included in the preprocessed block; and a sending module 503, configured to: in a consensus verification process, if it is determined that at least one piece of transaction information included in the preprocessed block does not exist in the transaction pool, determine an information identifier of the transaction information missing from the transaction pool in the transaction information included in the preprocessed block, and send a consensus verification failure notification that includes the information identifier to the other consensus nodes.

**[0075]** In the 1990s, whether a technical improvement is a hardware improvement (for example, an improvement to a circuit structure, such as a diode, a transistor, or a switch) or a software improvement (an improvement to a method procedure) can be clearly distinguished. However, as technologies develop, current improvements to many method procedures can be considered as direct improvements to hardware circuit structures. A designer usually programs an improved method procedure into a hardware circuit to obtain a corresponding hardware circuit structure. Therefore, a method procedure can be improved 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, and a logical function of the PLD is determined by a user through device programming. The designer performs programming to "integrate" a digital system to a PLD without requesting a chip manufacturer to design and produce an application-specific integrated circuit chip. In addition, at present, instead of manually manufacturing an integrated circuit chip, such programming is mostly implemented by using "logic compiler" software. The logic compiler software is similar to a software compiler used to develop and write a program. Original code needs to be written in a particular programming language for compilation. The language is referred to as a hardware description language (HDL). There are many HDLs, such as Advanced Boolean Expression Language (ABEL), Altera Hardware Description Language (AHDL), Confluence, Cornell University Programming Language (CUPL), HDCal, Java Hardware Description Language (JHDL), Lava, Lola, MyHDL, PALASM, and Ruby Hardware Description Language (RHDL). The very-high-speed integrated circuit hardware description language (VHDL) and Verilog are most commonly used. A person skilled in the art should also understand that a hardware circuit that implements a logical method procedure can be readily obtained once the method procedure is logically programmed by using the several described hardware description languages and is programmed into an integrated circuit.

**[0076]** A controller can be implemented by using any appropriate method. For example, the controller can be a microprocessor or a processor and a computer readable medium storing computer readable program code (such as software or firmware) that can be executed by the microprocessor or the processor, a logic gate, a switch, an application-specific integrated circuit (ASIC), a programmable logic controller, or a built-in microcontroller. 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, in addition to implementing the controller by using the computer readable program code, logic programming can be performed on method steps to allow the controller to implement the same function in forms of the logic gate, the switch, the application-specific integrated circuit, the programmable logic controller, and the built-in microcontroller. Therefore, the controller can be considered as a hardware component, and a device included in the controller and configured to implement various functions can also be considered as a structure in the hardware component. Or the device configured to implement various functions can even be considered as both a software module implementing the method and a structure in the hardware component.

**[0077]** The system, device, module, or unit illustrated in the previous implementations can be implemented by using a computer chip or an entity, or can be implemented by using a product having 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, a wearable device, or a combination of any of these devices.

**[0078]** For ease of description, the device above 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.

**[0079]** A person skilled in the art should understand that an implementation 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, an optical memory, etc.) that include computer-usable program code.

**[0080]** 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, a built-in processor, or a processor of another programmable data processing device to generate a machine, so that the instructions executed by the computer or the processor of the another programmable data processing device generate a device for implementing a specified function in one or more processes in the flowcharts and/or in one or more blocks in the block diagrams.

**[0081]** These computer program instructions can be stored in a computer readable memory that can instruct the computer or the another programmable data processing device to work in a specific way, so that the instructions stored in the computer readable memory generate an artifact that includes an instruction device. The instruction device implements a specified function in one or more processes in the flowcharts and/or in one or more blocks in the block diagrams.

**[0082]** These computer program instructions can be loaded onto the computer or another programmable data processing device, so that a series of operations and operations and steps are performed on the computer or the another programmable device, thereby generating computer-implemented processing. Therefore, the instructions executed on the computer or the another programmable device provide steps for implementing a specified function in one or more processes in the flowcharts and/or in one or more blocks in the block diagrams.

**[0083]** In a typical configuration, a computing device includes one or more processors (CPU), one or more input/output interfaces, one or more network interfaces, and one or more memories.

**[0084]** The memory can include a non-persistent memory, a random access memory (RAM), a nonvolatile memory, and/or another form that are in a computer readable medium, for example, a read-only memory (ROM) or a flash memory (flash RAM). The memory is an example of the computer readable medium.

**[0085]** The computer readable medium includes persistent, non-persistent, movable, and unmovable media that can store messages by using any method or technology. The message can be a computer readable instruction, a data structure, a program module, or other data. Examples of a computer storage medium include but

are not limited to: a parameter random access memory (PRAM), a static random access memory (SRAM), a dynamic random access memory (DRAM) or another type of random access memory (RAM), 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 cassette, a magnetic tape, a magnetic tape/magnetic disk memory or another magnetic storage device, or any other non-transmission medium that can be used to store messages that can be accessed by a computing device. Based on the definition in the present specification, the computer readable medium does not include transitory computer readable media (transitory media) such as a modulated data signal and carrier.

**[0086]** It is worthwhile to further note that, the terms "include", "comprise", or their any other variants are intended to cover a non-exclusive inclusion, so a process, a method, a product 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, product, or device. Without more constraints, an element preceded by "includes a …" does not preclude the existence of additional identical elements in the process, method, product, or device that includes the element.

**[0087]** A person skilled in the art should understand that an implementation 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, an optical memory, etc.) that include computer-usable program code.

**[0088]** The present application can be described in the general context of computer executable instructions executed by a computer, for example, a program module. Generally, the program module includes a routine, a program, an object, a component, a data structure, etc. executing a specific task or implementing a specific abstract data type. The present application can also be practiced in distributed computing environments. In the distributed computing environments, tasks are performed by remote processing devices connected through a communications

network. In a distributed computing environment, the program module can be located in both local and remote computer storage media including storage devices.

**[0089]** The implementations in the present specification are described in a progressive way. For the same or similar parts of the implementations, references can be made to the implementations. Each implementation focuses on a difference from other implementations. Particularly, a system implementation is basically similar to a method implementation, and therefore is described briefly. For related parts, references can be made to related descriptions in the method implementation.

**[0090]** The previous implementations are 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, or improvement made without departing from the spirit and principle of the present application shall fall within the scope of the claims in the present application.

## CLAIMS

What is claimed is:

1. A method for sending transaction information, comprising:

obtaining, by a consensus node, transaction information;

sending the transaction information to another consensus node; and

resending the transaction information based on a consensus verification failure notification in response to receiving the consensus verification failure notification that is sent by the another consensus node and that comprises an information identifier of the transaction information.

2. The method according to claim 1, wherein the obtaining, by a consensus node, transaction information comprises:

receiving, by the consensus node, transaction information sent by a client device.

3. The method according to claim 1, wherein the sending the transaction information to another consensus node comprises:

creating a thread for each other consensus node; and

sending the transaction information to the another consensus node through the thread.

4. The method according to claim 3, wherein the sending the transaction information to the another consensus node through the thread comprises:

resending the transaction information to the another consensus node through the thread in response to determining that the another consensus node fails to receive the transaction information, until it is determined that the another consensus node receives the transaction information or a predetermined sending stop condition is satisfied.

5. The method according to claim 4, wherein the determining that the another consensus node receives the transaction information comprises:

determining that the another consensus node receives the transaction information in response to determining that a response signal returned by the another consensus

node is received through the thread within a specified time period; or

wherein the determining that the another consensus node fails to receive the transaction information comprises:

determining that the another consensus node fails to receive the transaction information in response to determining that no response signal returned by the another consensus node is received through the thread within the specified time period.

6. The method according to claim 4, wherein the predetermined sending stop condition comprises:

the number of times of sending the transaction information to the another consensus node reaches the predetermined number of times; or

a time that elapsed since the first time the transaction information was sent to the another consensus node exceeds predetermined duration.

7. The method according to claim 4, wherein when the predetermined sending stop condition is satisfied, wherein the method further comprises:

adding the transaction information to a predetermined queue; and

wherein the resending the transaction information based on a consensus verification failure notification comprises:

searching the queue for the transaction information that corresponds to the information identifier based on the information identifier comprised in the consensus verification failure notification; and

sending the transaction information.

8. A method for consensus verification, comprising:

receiving, by a consensus node, a preprocessed block sent by another consensus node;

performing consensus verification on the preprocessed block based on transaction information stored in a transaction pool and transaction information comprised in the preprocessed block;

in a consensus verification process, in response to determining that at least one piece of transaction information comprised in the preprocessed block does not exist in the transaction pool, determining an information identifier of the transaction information missing from the transaction pool in the transaction information

comprised in the preprocessed block; and

sending a consensus verification failure notification that comprises the information identifier to the other consensus nodes.

9. A device for sending transaction information, comprising:

an acquisition module, configured to obtain transaction information;

a first sending module, configured to send the transaction information to another consensus node; and

a second sending module, configured to resend the transaction information based on a consensus verification failure notification in response to receiving the consensus verification failure notification that is sent by the another consensus node and that comprises an information identifier of the transaction information.

10. The device according to claim 9, wherein the acquisition module is configured to receive transaction information sent by a client device.

11. The device according to claim 9, wherein the first sending module is configured to create a thread for each other consensus node, and send the transaction information to the another consensus node through the thread.

12. The device according to claim 11, wherein the first sending module is configured to resend the transaction information to the another consensus node through the thread in response to determining that the another consensus node fails to receive the transaction information, until it is determined that the another consensus node receives the transaction information or a predetermined sending stop condition is satisfied.

13. The device according to claim 12, wherein the first sending module is configured to determine that the another consensus node receives the transaction information in response to determining that a response signal returned by the another consensus node is received through the thread within a specified time period; or determine that the another consensus node fails to receive the transaction information in response to determining that no response signal returned by the another consensus node is received through the thread within the specified time period.

14. The device according to claim 12, wherein the predetermined sending stop condition comprises:

the number of times of sending the transaction information to the another consensus node reaches the predetermined number of times; or

a time that elapsed since the first time the transaction information was sent to the another consensus node exceeds predetermined duration.

15. The device according to claim 12, wherein the device further comprises:

an adding module, configured to add the transaction information to a predetermined queue when the predetermined sending stop condition is satisfied; and

wherein the second sending module is configured to search the queue for the transaction information that corresponds to the information identifier based on the information identifier comprised in the consensus verification failure notification, and send the transaction information.

16. A device for consensus verification, comprising:

a receiving module, configured to receive a preprocessed block sent by another consensus node;

a consensus verification module, configured to perform consensus verification on the preprocessed block based on transaction information stored in a transaction pool and transaction information comprised in the preprocessed block; and

a sending module, configured to:

in a consensus verification process, in response to determining that at least one piece of transaction information comprised in the preprocessed block does not exist in the transaction pool, determine an information identifier of the transaction information missing from the transaction pool in the transaction information comprised in the preprocessed block; and

send a consensus verification failure notification that comprises the information identifier to the other consensus nodes.