(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
03 January 2019 (03.01.2019)

WIPO | PCT

(10) International Publication Number
WO 2019/002004 A1

(51) International Patent Classification:
*G06F 9/38* (2018.01) *H04L 9/32* (2006.01)

(21) International Application Number:
PCT/EP2018/066146

(22) International Filing Date:
18 June 2018 (18.06.2018)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
15/633,357 26 June 2017 (26.06.2017) US

(71) Applicant: **INTERNATIONAL BUSINESS MACHINES CORPORATION** [US/US]; New Orchard Road, Armonk, New York 10504 (US).

(71) Applicant *(for MG only)*: **IBM UNITED KINGDOM LIMITED** [GB/GB]; PO Box 41, North Harbour, Portsmouth Hampshire PO6 3AU (GB).

(72) Inventors: **ARDASHEV, Ruslan**; IBM CORPORATION, INTELLECTUAL PROPERTY LAW DEPARTMENT, 2455 SOUTH ROAD, MS P386, POUGHKEEPSIE, New York 12601 (US). **GAUR, Nitin**; IBM CORPORATION, 11501 BURNET RD, AUSTIN, Texas 78758-3400 (US).

(74) Agent: **FOURNIER, Kevin**; IBM United Kingdom Limited, Intellectual Property Law, Hursley Park, Winchester Hampshire SO21 2JN (GB).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

(54) Title: BLOCKCHAIN TRANSACTION COMMITMENT ORDERING

(57) Abstract: A blockchain function may include one or more of identifying a plurality of new transactions to be committed to a blockchain, determining whether any of the plurality of new transactions comprises a data dependency with any other of the plurality of new transactions, and transmitting a plurality of messages providing validation of the plurality of the new transactions, responsive to the determining.

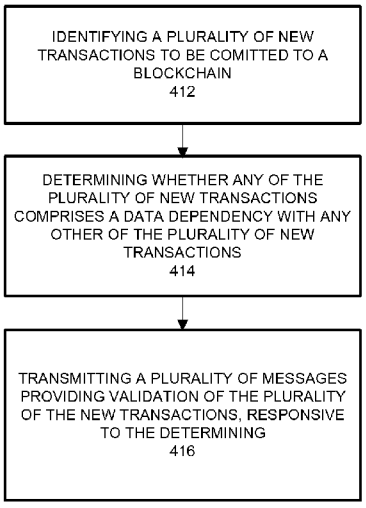


FIG. 4A

WO 2019/002004 A1

[Continued on next page]

**Published:**
— *with international search report (Art. 21(3))*

# BLOCKCHAIN TRANSACTION COMMITMENT ORDERING

## Technical Field

[0001] This application generally relates to transaction ordering, and more particularly, blockchain transaction commitment ordering.

## Background

[0002] A blockchain may be used as a public ledger to store any type of information. Although, primarily used for financial transactions, a blockchain can store any type of information, including assets (i.e., products, packages, services, status, etc.) in its immutable ledger. A decentralized scheme transfers authority and trusts to a decentralized network and enables its nodes to continuously and sequentially record their transactions on a public "block," creating a unique "chain" referred to as the blockchain. Cryptography, via hash codes, is used to secure an authentication of a transaction source and removes the need for a central intermediary.

[0003] The combination of cryptography and blockchain technology together ensures there is never a duplicate recording of the same transaction. Modern processors may execute instructions out of order (OoO), however this concept and what processors do are two distinct topics since processors must always maintain the architectural illusion of in-order execution, even though all that needs to be performed is the reduction of data dependences.

## Summary

[0004] One example embodiment may include one or more of identifying a plurality of new transactions to be committed to a blockchain, determining whether any of the plurality of new transactions comprises a data dependency with any other of the plurality of new transactions, and transmitting a plurality of messages providing validation of the plurality of the new transactions, responsive to the determining.

[0005] Another example embodiment may include an apparatus that includes a processor configured to perform one or more of identify a plurality of new transactions to be committed to a blockchain, determine whether any of the plurality of new transactions comprises a data dependency with any other of the plurality of new transactions, and a transmitter configured to transmit a plurality of messages indicating to proceed with validation of the plurality of the new transactions, responsive to the determining.

[0006] Still another example embodiment may include a non-transitory computer readable storage medium configured to store instructions that when executed cause a processor to perform one or more of identifying a

plurality of new transactions to be committed to a blockchain, determining whether any of the plurality of new transactions comprises a data dependency with any other of the plurality of new transactions, and transmitting a plurality of messages indicating to proceed with validation of the plurality of the new transactions, responsive to the determining.

**Brief Description of the Drawings**

[0007] FIG. 1 illustrates a transaction processing configuration illustrating the differences between out of order instruction timing and non-out of order instruction timing for transactions in a blockchain according to example embodiments.

[0008] FIG. 2 illustrates a transaction ordering logic configuration according to example embodiments.

[0009] FIG. 3 illustrates a system signaling diagram of the interactions between a transaction ordering computing device, an ordering module, and a blockchain according to example embodiments.

[0010] FIG. 4A illustrates a flow diagram of an example method of managing transaction ordering in the blockchain according to example embodiments.

[0011] FIG. 4B illustrates another flow diagram of an example method of managing transaction ordering in the blockchain according to example embodiments.

[0012] FIG. 5 illustrates an example network entity configured to support one or more of the example embodiments.

**Detailed Description**

[0013] It will be readily understood that the instant components, as generally described and illustrated in the figures herein, may be arranged and designed in a wide variety of different configurations. Thus, the following detailed description of the embodiments of at least one of a method, apparatus, non-transitory computer readable medium and system, as represented in the attached figures, is not intended to limit the scope of the application as claimed, but is merely representative of selected embodiments.

[0014] The instant features, structures, or characteristics as described throughout this specification may be combined in any suitable manner in one or more embodiments. For example, the usage of the phrases “example embodiments”, “some embodiments”, or other similar language, throughout this specification refers to the fact that a particular feature, structure, or characteristic described in connection with the embodiment may be included in at least one embodiment. Thus, appearances of the phrases “example embodiments”, “in some embodiments”, “in other embodiments”, or other similar language, throughout this specification do not necessarily all refer to the same

group of embodiments, and the described features, structures, or characteristics may be combined in any suitable manner in one or more embodiments.

[0015] In addition, while the term "message" may have been used in the description of embodiments, the application may be applied to many types of network data, such as, packet, frame, datagram, etc. The term "message" also includes packet, frame, datagram, and any equivalents thereof. Furthermore, while certain types of messages and signaling may be depicted in exemplary embodiments they are not limited to a certain type of message, and the application is not limited to a certain type of signaling.

[0016] The instant application in one embodiment relates to management of blockchain transaction ordering, and in another embodiment relates to identifying pending transactions and determining relationships between the transactions to optimize transaction commitment ordering.

[0017] Example embodiments provide a new module for blockchain network architectures that will enable blockchain networks operate efficiently regardless of data dependences between pending transactions to optimize network speed, processing efficiency and other computing resources. According to one example, an ordering module may be part of a transaction server which receives transactions and buffers the transactions and pending instructions used to analyze data dependences between those transactions. If no data dependences are identified, then instructions proceed to simultaneously forward instructions for validation and commitment of the transactions may be performed out-of-order. This approach optimizes instruction processing times.

[0018] FIG. 1 illustrates a transaction processing configuration illustrating the differences between out of order instruction timing and non-out of order instruction timing for transactions in a blockchain according to example embodiments. Referring to FIG. 1, the configuration 100 includes examples of in-order (without out-of-order) and out-of-order transaction processing. Referring to the example with transactions 112-124 (A-F), the out-of-order instructions permit bypassing of the waiting for previous instructions to first commit a transaction before proceeding to process further instructions. In the diagram 100, the size of the instructions does not reflect the amount of instructions, as the size is merely an indicator to illustrate time spent on that particular transaction/instruction. The 'in-order' instructions are about the same size as they are all sequentially processed in a similar amount of time whereas the out-of-order instructions are overlapping and asynchronous and may vary in time spent to perform the processing. In the 'out-of-order' model, the instructions for F 124 need to wait for instructions from transaction B 114 to complete, since a pending order module used to process the transactions (see FIG. 1). Further, data dependences are analyzed, and if a data dependency exists between instructions for D 118 and F 124, a signal is used to wait for all previous instructions to commit. In this case, instructions for B 114 are the last instruction to wait before proceeding to commitment. In the in-order approach or without out-of-order instruction timing, the instructions for transactions A 132, B 134 and C 136 are processed in-order without any type of parallel processing,

asynchronous processing and/or simultaneous processing of transactions for commitment. The various transactions may be identified according to their instructions that relate to data and make modifications. For example, Instructions for transactions A through E related to certain data but only D and F are detected to have data collisions by the pending order module.

[0019] Without OoO processing, the transactions are committed in the following order: A, B, C, D, E, F. Applying OoO transaction management support, the transactions are committed in the following order: A, C, E, D, B, F. Transactions using OoO may follow a procedure where a peer node publishes the beginning of a transaction to fellow peers as well as data surrounding which data is affected by that transaction. A transaction undergoes consensus by its peers and once approved, it is appended locally to the blockchain. Hash implementations or how the transactions are specifically linked out of order, with the "overall" results of a blockchain being the same, will vary by implementation.

[0020] FIG. 2 illustrates a transaction ordering logic configuration according to example embodiments. Referring to FIG. 2, the example illustration 200 provides for an order module 212 which provides committed instructions for committing transactions 214 to a pending order module 222. The committed instructions 224 of the pending order module 222 may be different from the original transaction order module 212. All other nodes 250 are also able to communicate the instructions to the pending order module 222. The pending order module 222 tracks all pending instructions and their states (e.g., pending sending out for consensus, pending consensus, pending commitment to the blockchain). Pending sending is the most premature status, pending consensus is second, and pending commitment is what has reached consensus and is awaiting finalization.

[0021] The order module 212 tracks the state of all previously committed instructions which already exist. Instructions begin a lifecycle based on user input to perform some predefined action(s) (i.e., request shares, transfer car, purchase house, enter mortgage information, etc.). These commands are then converted to blockchain instructions via smart contracts and the instructions include read and/or write data locations with clearly defined data locations, which are then inputted into the pending order module 222. In the pending order module 222, data locations of instructions are analyzed for data dependencies. If instructions are cleared to proceed, they are sent out for consensus in order, returned out-of-order, and are committed to the blockchain in no particular order. If data dependences exist instructions are marked to wait for all prior instructions to commit first before the latter instructions.

[0022] There are two categories of data dependencies including, write-after-read and write-after-write. Write-after-read instructions end up reading old data since a newer instruction has been dispatched to other nodes but not yet returned through consensus mechanisms. As a result, the newer instruction falsely reads an old value, and after that older instruction is committed, the newer instruction may proceed. In the write-after-write dependency,

two instructions are dispatched in a correct order, but the younger/newer instruction commits and writes first, followed by the older instruction. In this case, data irregularities will occur if both instructions write to the same data location. To deal with data dependences, younger instructions may be commanded to wait for the older instructions, with which they conflict, to commit first, which includes all the following instructions. As illustrated in FIG. 1, a data dependency exists between instructions B 114 and F 124, or, more specifically, instruction F 124 reads from the data location to which instruction B 114 writes. As a result, instruction F 124 is forced to wait for instruction B 114, and all following instructions, in order to commit.

[0023] In general, when data dependences exist, instructions are marked to wait for all prior instructions to commit prior to being committed. If conflicts exist between which data instructions are related to or interact with, such as in the example between D and F above, then F does not enter the "pending sending out for consensus" phase until D has been committed. This way we potential consensus issues are averted. When a transaction is submitted by a client to a peer node, the peer node enables the pending order module to scan for data dependences, and if no data dependences exist between any of the transactions that are currently pending consensus or commitment status, then the transaction is also sent out of consensus and commitment which can be committed out-of-order. If a new transaction does conflict with any of the "in processing" transactions/instructions (the terms are used interchangeably as the transactions includes discernable instructions), that transaction is then added to a queue to await all "in processing" instructions to complete. Once all "in processing" transactions are committed, the queued transactions are retrieved for commitment and the entire process will restart. One example of ordering may provide if A writes to location 1234 and B reads from location 1234, A must be 'waited for' in order to fully accept consensus and commitment before starting on B, since B will read from 1234, and consensus will be based on the results of what is read from 1234. This OoO consideration is one of many possibilities which may occur when there is more than one transaction pending for commitment.

[0024] FIG. 3 illustrates a system messaging diagram 300 of the interactions between a transaction ordering computing device, an ordering module, and a blockchain according to example embodiments. Referring to Fig. 3, the system 300 may include a number of components or modules which may include software, hardware or a combination of both. The components may include a first component, such as a transaction server 310, which may be a machine responsible for outputting transactions, instructions or other communication signals. A second component, such as an ordering module 320, may be one or both of the modules of FIG. 2 and may be part of the transaction server 310. The purpose of the ordering module 320 is to manage instructions for transaction management and commitment. In operation, new transactions 312 may be identified and queued for processing. The transactions may be buffered along with instructions for seeking consensus, ordering the transactions, identifying dependencies and committing the transactions to a third component such as a blockchain 330. The transactions are initially buffered based on an order they are received 314. The ordering and instructions may dictate various re-ordering procedures. The next operation is to identify data dependencies 316 between the

transactions. Actions are identified from each of the transactions 318 and those actions which are dependent on one another are identified as having dependencies. The ordering or reordering instructions 324 are then created and forwarded 324 to the ordering module 320 so the transactions can be committed 326 and forwarded 328 to the blockchain 330.

[0025] In one embodiment, the first component, the second component and the third component may be separate devices such as servers, computers or other computational devices or may be a single device. In other embodiments, the first component and the second component may be enclosed as, or perform as, a single device, the first component and the third component may be enclosed as, or perform as, a single device, and the second component and the third component may be enclosed as, or perform as, a single device. The components or devices 310, 320 and 330 may be directly connected or communicably coupled to one another, in a wired or wireless manner, and may reside locally and/or remotely.

[0026] FIG. 4A illustrates a flow diagram 400 of an example may include managing transaction ordering in the blockchain according to example embodiments. Referring to FIG. 4A, the example may include one or more of: identifying a plurality of new transactions to be committed to a blockchain 412, determining whether any of the plurality of new transactions includes a data dependency with any other of the plurality of new transactions 414, transmitting a plurality of messages indicating to proceed with validation of the plurality of the new transactions 416, responsive to determining that there are no data dependencies between any of the new transactions. The plurality of messages may be transmitted simultaneously. The method may include buffering the plurality of new transactions according to a chronological order in an ordering module including memory to store the plurality of new transactions, and committing one or more of the new transactions out of the chronological order with one or more other of the plurality of transactions, responsive to determining that there are no data dependencies between any of the new transactions. The method may also include identifying one or more actions identified from the plurality of blockchain transactions, and creating one or more instructions corresponding to the one or more actions. The method may further provide forwarding the one or more instructions to the ordering module, such that the one or more instructions includes read and write data locations, and committing the plurality of transactions based on the one or more instructions. The method may also include identifying one or more data dependencies among the plurality of transactions, and marking the one or more instructions to wait until a time when all previously created instructions are associated with committed transactions. The data dependencies include one or more of a same product identified in two or more transactions and a common party to a subsequent transaction.

[0027] FIG. 4B illustrates another flow diagram 450 of an example may include managing transaction ordering in the blockchain according to example embodiments. The example may include one or more of: identifying a plurality of new transactions to be committed to a blockchain 452, determining whether any of the plurality of new transactions comprises a data dependency with any other of the plurality of new transactions 454, responsive to

determining that at least one data dependency exists between two or more of the plurality of transactions, committing an earlier time stamped transaction of the two or more transactions having the at least one data dependency 456, and rejecting the later time stamped transaction of the two or more transactions having the at least one data dependency 458.

[0028] In this example, the data dependencies are identified and the chronological order of the transactions are also identified. As a result, when two or more transactions have a data dependency and do not have a common time stamp (i.e. different creation times). The earlier queued or received transaction will be permitted to be committed and any other transactions sharing the same data dependency will be suspended or rejected pending evidence of additional resources which have become available to fulfill the transaction requirements.

[0029] The above embodiments may be implemented in hardware, in a computer program executed by a processor, in firmware, or in a combination of the above. A computer program may be embodied on a computer readable medium, such as a storage medium. For example, a computer program may reside in random access memory ("RAM"), flash memory, read-only memory ("ROM"), erasable programmable read-only memory ("EPROM"), electrically erasable programmable read-only memory ("EEPROM"), registers, hard disk, a removable disk, a compact disk read-only memory ("CD-ROM"), or any other form of storage medium known in the art.

[0030] An exemplary storage medium may be coupled to the processor such that the processor may read information from, and write information to, the storage medium. In the alternative, the storage medium may be integral to the processor. The processor and the storage medium may reside in an application specific integrated circuit ("ASIC"). In the alternative, the processor and the storage medium may reside as discrete components. For example, FIG. 5 illustrates an example network element 500, which may represent or be integrated in any of the above-described components, etc.

[0031] As illustrated in FIG. 5, a memory 510 and a processor 520 may be discrete components of a network entity 500 that are used to execute an application or set of operations as described and depicted herein. The application may be coded in software in a computer language understood by the processor 520, and stored in a computer readable medium, such as, a memory 510. The computer readable medium may be a non-transitory computer readable medium that includes tangible hardware components, such as memory, that can store software. Furthermore, a software module 530 may be another discrete entity that is part of the network entity 500, and which contains software instructions that may be executed by the processor 520 to effectuate one or more of the functions described herein. In addition to the above noted components of the network entity 500, the network entity 500 may also have a transmitter and receiver pair configured to receive and transmit communication signals (not shown).

[0032] Although an exemplary embodiment of at least one of a system, method, and non-transitory computer readable medium has been illustrated in the accompanied drawings and described in the foregoing detailed description, it will be understood that the application is not limited to the embodiments disclosed, but is capable of numerous rearrangements, modifications, and substitutions as set forth and defined by the following claims. For example, the capabilities of the system of the various figures can be performed by one or more of the modules or components described herein or in a distributed architecture and may include a transmitter, receiver or pair of both. For example, all or part of the functionality performed by the individual modules, may be performed by one or more of these modules. Further, the functionality described herein may be performed at various times and in relation to various events, internal or external to the modules or components. Also, the information sent between various modules can be sent between the modules via at least one of: a data network, the Internet, a voice network, an Internet Protocol network, a wireless device, a wired device and/or via plurality of protocols. Also, the messages sent or received by any of the modules may be sent or received directly and/or via one or more of the other modules.

[0033] One skilled in the art will appreciate that the "system" or one or more components or elements of the instant application could be embodied as a personal computer, a server, a console, a personal digital assistant (PDA), a cell phone, a tablet computing device, a smartphone or any other suitable computing device, or combination of devices. Presenting the above-described functions as being performed by a "system" is not intended to limit the scope of the present application in any way, but is intended to provide one example of many embodiments. Indeed, methods, systems and apparatuses disclosed herein may be implemented in localized and distributed forms consistent with computing technology.

[0034] It should be noted that some of the system features described in this specification have been presented as modules, in order to more particularly emphasize their implementation independence. For example, a module may be implemented as a hardware circuit comprising custom very large scale integration (VLSI) circuits or gate arrays, off-the-shelf semiconductors such as logic chips, transistors, or other discrete components. A module may also be implemented in programmable hardware devices such as field programmable gate arrays, programmable array logic, programmable logic devices, graphics processing units, or the like.

[0035] A module may also be at least partially implemented in software for execution by various types of processors. An identified unit of executable code may, for instance, comprise one or more physical or logical blocks of computer instructions that may, for instance, be organized as an object, procedure, or function. Nevertheless, the executables of an identified module need not be physically located together, but may comprise disparate instructions stored in different locations which, when joined logically together, comprise the module and achieve the stated purpose for the module. Further, modules may be stored on a computer-readable medium,

which may be, for instance, a hard disk drive, flash device, random access memory (RAM), tape, or any other medium used to store data.

[0036] Indeed, a module of executable code could be a single instruction, or many instructions, and may even be distributed over several different code segments, among different programs, and across several memory devices. Similarly, operational data may be identified and illustrated herein within modules, and may be embodied in any suitable form and organized within any suitable type of data structure. The operational data may be collected as a single data set, or may be distributed over different locations including over different storage devices, and may exist, at least partially, merely as electronic signals on a system or network.

[0037] It will be readily understood that the components of the application, as generally described and illustrated in the figures herein, may be arranged and designed in a wide variety of different configurations. Thus, the detailed description of the embodiments is not intended to limit the scope of the application as claimed, but is merely representative of selected embodiments of the application.

[0038] One having ordinary skill in the art will readily understand that the above may be practiced with steps in a different order, with steps that are combined, with steps that are omitted, and/or with hardware elements in configurations that are different than those which are disclosed. Therefore, although the application has been described based upon these preferred embodiments, it would be apparent to those of skill in the art that certain modifications, variations, and alternative constructions would be apparent.

[0039] While preferred embodiments of the present application have been described, it is to be understood that the embodiments described are illustrative only and the scope of the application, in one embodiment, should be defined by the appended claims when considered with a full range of equivalents and modifications (e.g., protocols, hardware devices, software platforms etc.) thereto.

## CLAIMS

1. A method, comprising:

identifying a plurality of new transactions to be committed to a blockchain;

determining whether any of the plurality of new transactions comprises a data dependency with any other of the plurality of new transactions; and

transmitting a plurality of messages providing validation of the plurality of the new transactions, responsive to the determining.

2. The method of claim 1, wherein the plurality of messages are transmitted simultaneously.

3. The method of claim 1, further comprising:

buffering the plurality of new transactions according to a chronological order in an ordering module comprising memory to store the plurality of new transactions; and

committing one or more of the new transactions out of the chronological order with one or more other of the plurality of new transactions, responsive to the determining.

4. The method of claim 3, further comprising:

identifying one or more actions identified from the plurality of new transactions committed to the blockchain; and

creating one or more instructions corresponding to the one or more actions.

5. The method of claim 4, further comprising:

forwarding the one or more instructions to the ordering module, wherein the one or more instructions comprise read and write data locations; and

committing the plurality of new transactions based on the one or more instructions.

6. The method of claim 5, further comprising:

identifying one or more data dependencies among the plurality of new transactions; and

marking the one or more instructions to wait until a time when all previously created instructions are associated with committed transactions.

7. The method of claim 6, wherein the on or more data dependencies comprise one or more a same product identified in two or more transactions and a common party to a subsequent transaction.

8. A apparatus comprising means adapted for carrying out all the steps of the method according to any preceding method claim.

9. A computer program comprising instructions for carrying out all the steps of the method according to any preceding method claim, when said computer program is executed on a computer system.

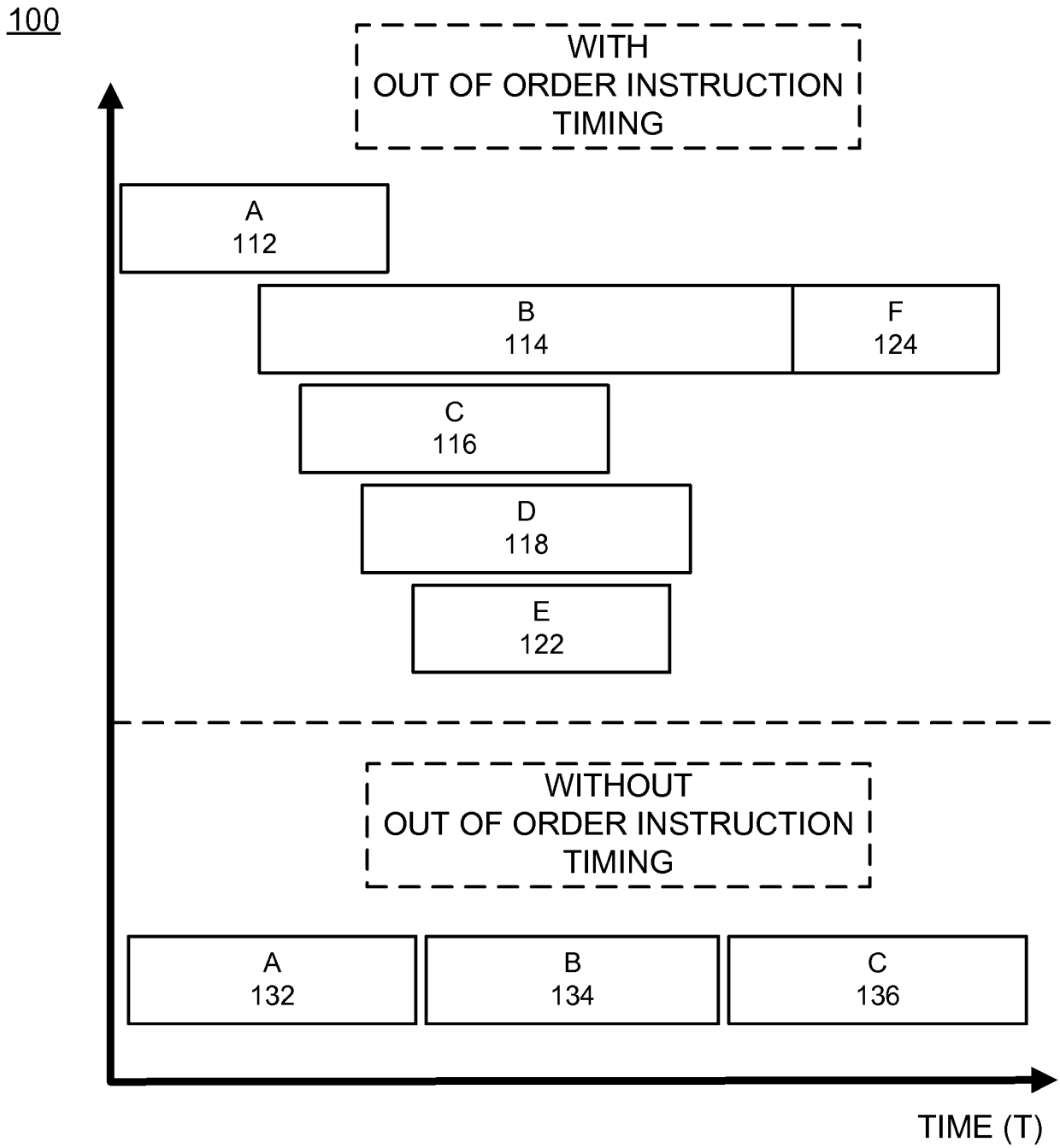
100
WITH
OUT OF ORDER INSTRUCTION
TIMING
A
112
B
114
F
124
C
116
D
118
E
122
WITHOUT
OUT OF ORDER INSTRUCTION
TIMING
A
132
B
134
C
136
TIME (T)

FIG. 1

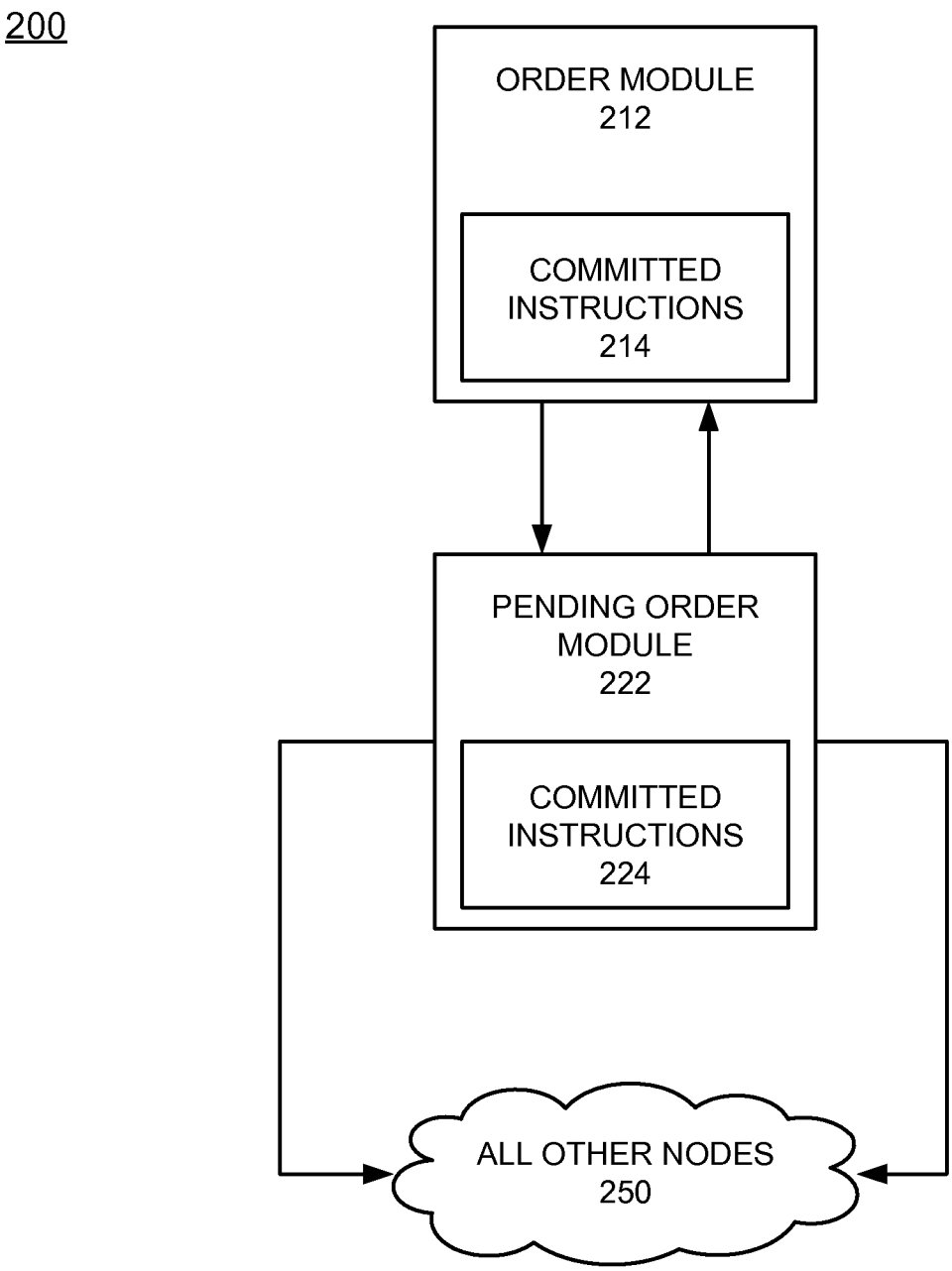
200
ORDER MODULE
212
COMMITTED
INSTRUCTIONS
214
PENDING ORDER
MODULE
222
COMMITTED
INSTRUCTIONS
224
ALL OTHER NODES
250

FIG. 2

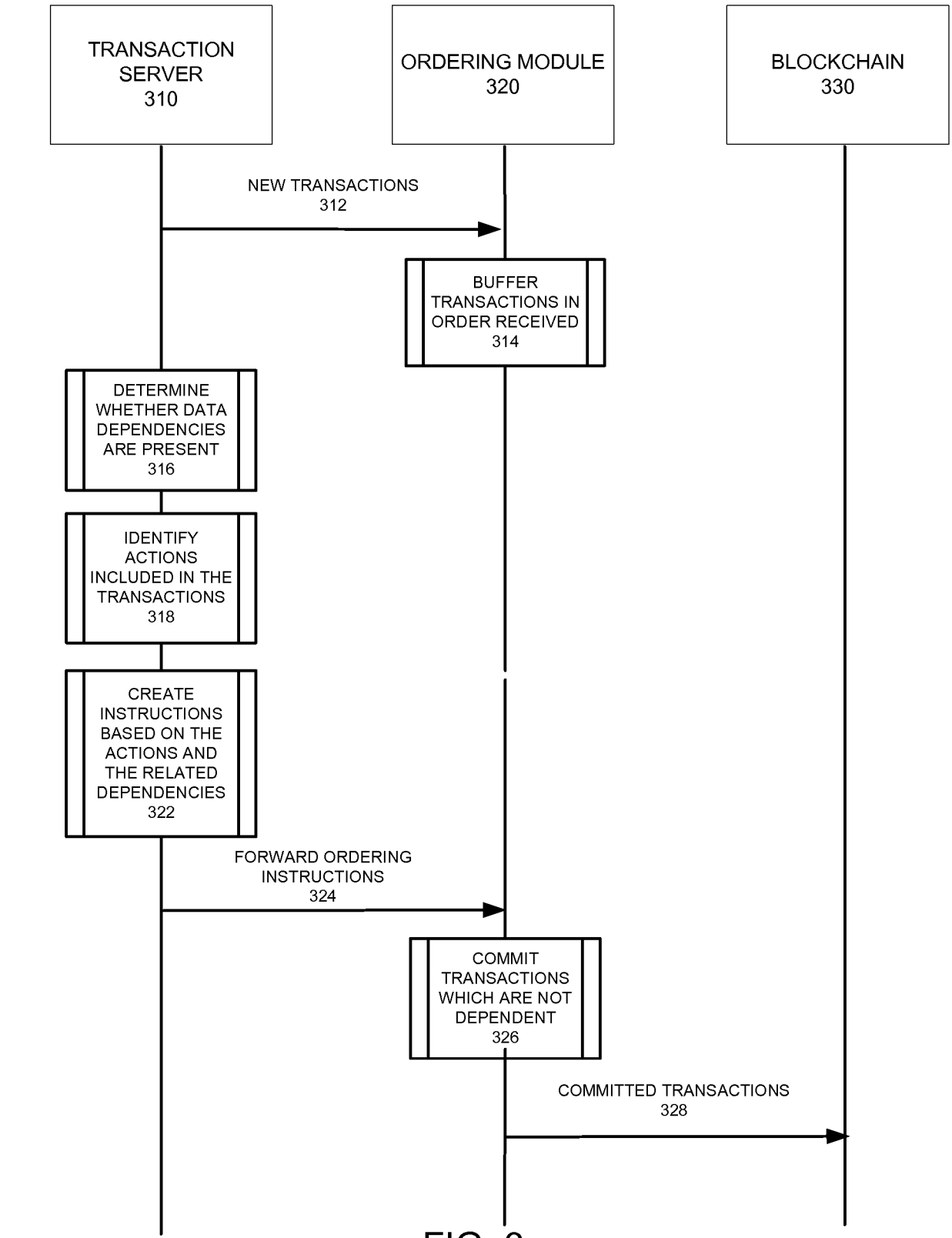
300
TRANSACTION SERVER 310
ORDERING MODULE 320
BLOCKCHAIN 330
NEW TRANSACTIONS 312
BUFFER TRANSACTIONS IN ORDER RECEIVED 314
DETERMINE WHETHER DATA DEPENDENCIES ARE PRESENT 316
IDENTIFY ACTIONS INCLUDED IN THE TRANSACTIONS 318
CREATE INSTRUCTIONS BASED ON THE ACTIONS AND THE RELATED DEPENDENCIES 322
FORWARD ORDERING INSTRUCTIONS 324
COMMIT TRANSACTIONS WHICH ARE NOT DEPENDENT 326
COMMITTED TRANSACTIONS 328

FIG. 3

400

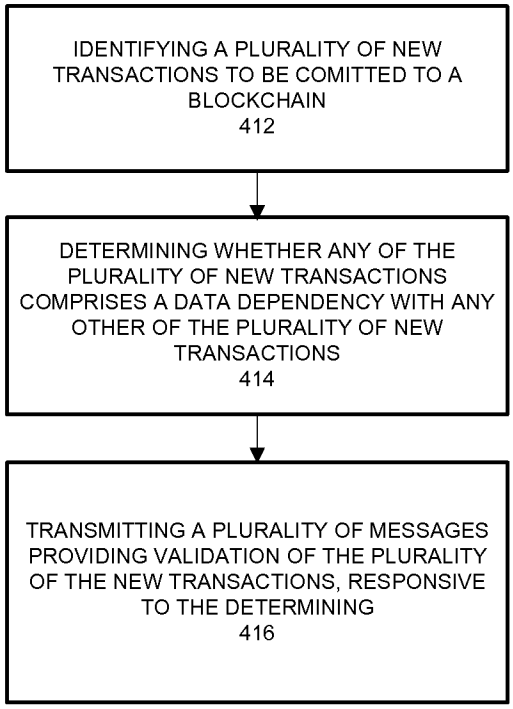
IDENTIFYING A PLURALITY OF NEW TRANSACTIONS TO BE COMITTED TO A BLOCKCHAIN
412
DETERMINING WHETHER ANY OF THE PLURALITY OF NEW TRANSACTIONS COMPRISES A DATA DEPENDENCY WITH ANY OTHER OF THE PLURALITY OF NEW TRANSACTIONS
414
TRANSMITTING A PLURALITY OF MESSAGES PROVIDING VALIDATION OF THE PLURALITY OF THE NEW TRANSACTIONS, RESPONSIVE TO THE DETERMINING
416

FIG. 4A

450

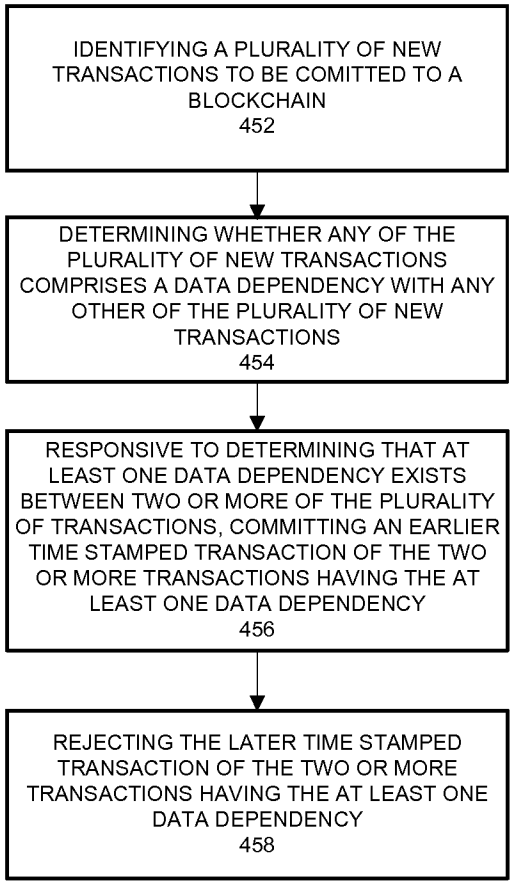
IDENTIFYING A PLURALITY OF NEW TRANSACTIONS TO BE COMITTED TO A BLOCKCHAIN
452
DETERMINING WHETHER ANY OF THE PLURALITY OF NEW TRANSACTIONS COMPRISES A DATA DEPENDENCY WITH ANY OTHER OF THE PLURALITY OF NEW TRANSACTIONS
454
RESPONSIVE TO DETERMINING THAT AT LEAST ONE DATA DEPENDENCY EXISTS BETWEEN TWO OR MORE OF THE PLURALITY OF TRANSACTIONS, COMMITTING AN EARLIER TIME STAMPED TRANSACTION OF THE TWO OR MORE TRANSACTIONS HAVING THE AT LEAST ONE DATA DEPENDENCY
456
REJECTING THE LATER TIME STAMPED TRANSACTION OF THE TWO OR MORE TRANSACTIONS HAVING THE AT LEAST ONE DATA DEPENDENCY
458

FIG. 4B

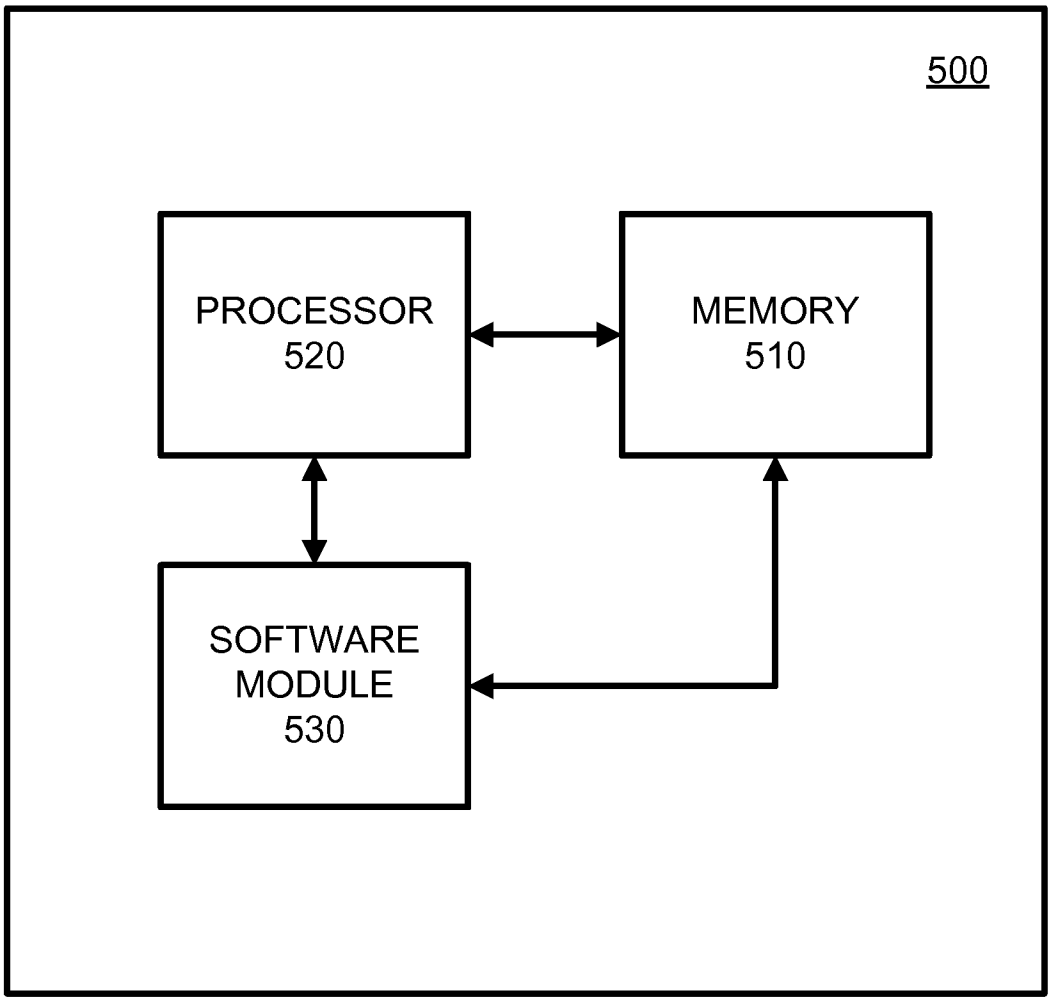


FIG. 5

# INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2018/066146

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F9/38 H04L9/32
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

| Category* | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| X | Andreas M. Antonopoulos: "Mastering Bitcoin - Unlocking Digital Cryptocurrencies" In: "Mastering bitcoin : [unlocking digital cryptocurrencies]", 20 December 2014 (2014-12-20), O'Reilly Media, Beijing Cambridge Farnham Köln Sebastopol Tokyo, XP055306939, ISBN: 978-1-4493-7404-4 pages 158, 175 | 1-9 |
| A | WO 2017/027648 A1 (STOLLMAN JEFF [US]) 16 February 2017 (2017-02-16) pages 11-12; figure 2 | 1-9 |

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

| Date of the actual completion of the international search | Date of mailing of the international search report |
|---|---|
| 15 August 2018 | 23/08/2018 |

| Name and mailing address of the ISA/ | Authorized officer |
|---|---|
| European Patent Office, P.B. 5818 Patentlaan 2<br>NL - 2280 HV Rijswijk<br>Tel. (+31-70) 340-2040,<br>Fax: (+31-70) 340-3016 | Horbach, Christian |

1

# INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/EP2018/066146

| Patent document cited in search report | Publication date | Patent family member(s) | Publication date |
|---|---|---|---|
| WO 2017027648 A1 | 16-02-2017 | EP 3335367 A1 | 20-06-2018 |
| | | WO 2017027648 A1 | 16-02-2017 |