(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 2019/006454 A1

(51) International Patent Classification:
G06F 11/14 (2006.01) *G06F 17/20* (2006.01)
G06F 11/16 (2006.01)

(21) International Application Number:
PCT/US2018/040590

(22) International Filing Date:
02 July 2018 (02.07.2018)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
15/639,965 30 June 2017 (30.06.2017) US

(72) Inventors; and
(71) Applicants: **SHOOLMAN, Yiftach** [IL/IL]; 681 Har-aminadav St., Maccabim, 71799 Modi'in (IL). **STEINBERG, Yoav** [US/IL]; 3 Tavas Street, 45353 Hod Hasharon (IL). **GOTTLIEB, Yossi** [US/IL]; 3 Tavas Street, 45353 Hod Hasharon (IL). **ARGA, Oran** [IL/IL]; Kibbutz Sa'ar, 22101 Nahariyya (IL).

(74) Agent: **BYRNE, Matthew T.** et al.; Byrne Poh LLP, 11 Broadway, Suite 760, New York, NY 10004 (US).

(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).

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

(54) **Title:** METHODS, SYSTEMS, AND MEDIA FOR CONTROLLING APPEND-ONLY FILE REWRITES

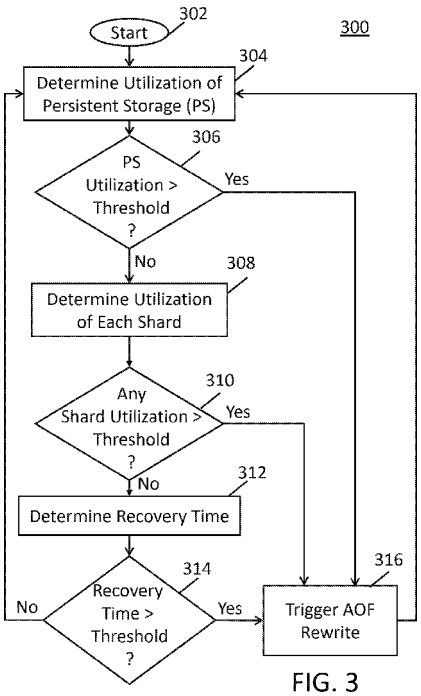

FIG. 3

(57) **Abstract:** Methods, systems, and media for controlling append-only file rewrites are provided. In accordance with some embodiments, the methods comprising: determining a utilization of a shard of a database; determining whether the utilization exceeds a persistent storage utilization threshold; determining a recovery time to recover the shard from an append-only file; determining whether the recovery time exceeds a recovery time threshold; and when the utilization is determined to exceed the utilization threshold or when the recovery time is determined to exceed the recovery time threshold, causing an append-only-file rewrite to be performed.

WO 2019/006454 A1

METHODS, SYSTEMS, AND MEDIA FOR CONTROLLING APPEND-ONLY FILE REWRITES

Technical Field

[0001] This application claims the benefit of United States Patent Application No. 15/639,965, filed June 30, 2017, which is hereby incorporated by reference herein in its entirety.

Technical Field

[0002] The disclosed subject matter relates to methods, systems, and media for controlling append-only file rewrites.

Background

[0003] As database access speed requirements continue to increase and the costs for random access memory (RAM) continue to decrease, the popularity of in-memory database systems continues to grow. In such database systems, rather than storing data in non-volatile storage devices like computer hard disks, data is stored in RAM, which is significantly faster.

[0004] A problem with in-memory databases that use volatile storage (such as RAM) is that the data stored therein can be easily lost in various circumstances, such as a power failure.

[0005] A popular approach to managing data-persistence in an in-memory database system is to write data to an append-only file (AOF) which is stored in non-volatile storage (e.g., such as a computer hard disk, a Storage Area Network (SAN), or a Network Attached Storage (NAS)) (which is also referred to herein as persistent storage). Typically, in such an approach, every "write" command received by the in-memory database is also written to the AOF stored on a computer non-volatile storage. Modern in-memory database systems like Redis provide multiple ways to control the "writes" of commands to the AOF, for example: a new line is written in the AOF for every "write" command; or all new "write" commands are written after N seconds to the AOF, where N has any suitable value.

[0006] In an event in which an in-memory database node fails and the data that was previously hosted in the node's RAM is lost, the AOF can be loaded to database memory to recover from the data loss.

[0007] The problem with the AOF approach is that the AOF tends to grow very quickly. This is the case because every "write" command that arrives at the database system is written as a new line of text in the AOF. Thus, multiple "write" commands for the same database object will result in multiple lines in the AOF, rather than just one.

[0008] To control the size of the AOF, an AOF rewrite event can be triggered in which the existing in-memory dataset is rewritten to a new AOF so that there is only one line for each object. Alternatively, a rewrite operation may create a new file in which its first part contains a snapshot of the in-memory dataset in a serialized and compressed way (in the Redis term this is called RDB format), and every new 'write' operation will be added to the file using AOF format. This event is usually triggered when the size of the AOF on the disk is N times larger (where N can have any suitable value) than the size of the dataset in RAM.

[0009] One way to implement AOF rewrite for in-memory database like Redis is to take a snapshot of the in-memory dataset using the Linux Copy On Write (COW) process. This guarantees that the new rewrite AOF represents a point in time state of the database. During the rewrite process, the in-memory database maintains two copies of the AOF: (1) the one that writes to the current AOF; and (2) the one that writes to the new rewrite AOF. In addition, the in-memory database maintains an internal in-memory buffer that includes all the changes that were made to the dataset from the time the snapshot was taken through the end of the AOF rewrite process.

[0010] An AOF rewrite process is completed when the entire dataset snapshot is written to the new AOF and all the changes that were made to the dataset since the snapshot was taken are also written to the new AOF.

[0011] A problem with AOF rewrite events is that they are disk-intensive operations. When the rewrite event happens, it can block updates to the in-memory database from being written to the current AOF, which can significantly delay the entire database execution time. Such characteristics can cause an in-memory database to violate standard terms of a service level agreement between a database service provider and its customer.

[0012] Accordingly, it is desirable to provide new methods, systems, and media for controlling append-only file rewrites.

Summary

[0013] In accordance with various embodiments of the disclosed subject matter, methods, systems, and media for controlling append-only file rewrites are provided.

[0014] In accordance with some embodiments of the disclosed subject matter, methods for controlling append-only file rewrites are provided, the methods comprising: determining a utilization of a shard of a database; determining whether the utilization exceeds a persistent storage utilization threshold; determining a recovery time to recover the shard from an append-only file; determining whether the recovery time exceeds a recovery time threshold; and when the utilization is determined to exceed the utilization threshold or when the recovery time is determined to exceed the recovery time threshold, causing an append-only-file rewrite to be performed.

[0015] In accordance with some embodiments of the disclosed subject matter, systems for controlling append-only file rewrites are provided, the systems comprising: at least one hardware processor configured to: determine a utilization of a shard of a database; determine whether the utilization exceeds a persistent storage utilization threshold; determine a recovery time to recover the shard from an append-only file; determine whether the recovery time exceeds a recovery time threshold; and when the utilization is determined to exceed the utilization threshold or when the recovery time is determined to exceed the recovery time threshold, cause an append-only-file rewrite to be performed.

[0016] In accordance with some embodiments, non-transitory computer-readable media containing computer executable instructions that, when executed by a processor, cause the processor to perform a method for controlling append-only-file rewrites are provided, the method comprising: determining a utilization of a shard of a database; determining whether the utilization exceeds a persistent storage utilization threshold; determining a recovery time to recover the shard from an append-only file; determining whether the recovery time exceeds a recovery time threshold; and when the utilization is determined to exceed the utilization threshold or when the recovery time is determined to exceed the recovery time threshold, causing an append-only-file rewrite to be performed.

Brief Description of the Drawings

[0017] Various objects, features, and advantages of the disclosed subject matter can be more fully appreciated with reference to the following detailed description of the disclosed subject matter when considered in connection with the following drawings, in which like reference numerals identify like elements.

[0018] FIG. 1 is an example of a diagram of a cluster architecture in accordance with some embodiments of the disclosed subject matter.

[0019] FIG. 2 is an example of a diagram of a node architecture in accordance with some embodiments of the disclosed subject matter.

[0020] FIG. 3 is an example of a flow diagram of a process for determining whether to trigger an AOF rewrite event in accordance with some embodiments of the disclosed subject matter.

[0021] FIG. 4 is an example of a flow diagram of a process for determining a recovery time in accordance with some embodiments of the disclosed subject matter.

[0022] FIG. 5 is an example of a diagram of hardware that can be used to implement one or more of servers that can be used to implement any of the components depicted in FIGS. 1 and/or 2 in accordance with some embodiments.

Detailed Description

[0023] In accordance with various embodiments, mechanisms (which can include methods, systems, and/or media) for controlling append-only file (AOF) rewrites are provided.

[0024] Generally speaking, these mechanisms can control AOF rewrite events by triggering an AOF rewrite event when total persistent storage utilization meets or exceeds a threshold, when an AOF's utilization of an AOF quota meets or exceeds a threshold, when and/or when the time to recover the portion of the database from the AOF meets or exceeds a threshold. These mechanisms can work on a shard level, where a shard represents a subset of the database's dataset and is usually managed by a different process.

[0025] In accordance with some embodiments, an example of in-memory non-relational database can be implemented as a system 100 as illustrated in FIG. 1. As shown, system 100 can include one or more clusters 102 and one or more applications 104.

[0026] Clusters 102 can include one or more clusters of nodes 110 for providing in-memory data storage as well as related functionality as described further below. Clusters 102 can also include any suitable persistent storage 112 that is coupled to nodes 110 in some embodiments.

[0027] Applications 104 can be one or more applications that use data and related information stored in nodes 110. As shown in FIG. 1, each application 104 can be executed using one or more servers. Any suitable servers can be used in some embodiments.

[0028] Turning to FIG. 2, an example of a node 200 that can be used as a node 110 in a cluster 102 in accordance with some embodiments is illustrated. As shown, node 200 can include a proxy 202, a cluster node manager (CNM) 204, zero or more shards 206, and common cluster storage (CCS) 208, in some embodiments. Also, also shown in FIG. 2, node 200 can be coupled to any suitable persistent storage device 210 in some embodiments.

[0029] In accordance with some embodiments, proxy 202 can be used to manage the control flow of node 200, to manage the control flow between node 200 and one or more other nodes in the same cluster, and to manage the control flow between node 200 and one or more nodes in another cluster. Proxy 202 can also be used to perform client authentication in some embodiments, and request forwarding once a request is authenticated. Any suitable one or more client authentication mechanisms can be used. For example, Secured Socket Layer (SSL) Authentication, Simple Authentication and Security Layer (SASL) authentication, password authentication, source IP authentication, Amazon Web Service Security Group, and/or any other suitable authentication mechanisms can be used in some embodiments.

[0030] In accordance with some embodiments, cluster node manager (CNM) 204 can be used to perform node management functions and cluster management functions. For example, such functions can include provisioning/deprovisioning of a new database, shard migration (e.g., moving a shard from one node to another), re-sharding (e.g., adding more shard(s) to a database), auto-scaling (e.g., adding/removing nodes from the cluster, re-balancing (e.g., optimal re-ordering of the shards on cluster nodes), resource management (e.g., determining if a given shard has reached its maximum processing capacity, or is about to exceed its memory limit), and/or any other suitable function related to managing a node and/or a cluster.

[0031] In accordance with some embodiments, CNM 204 as a node manager can also provide a process for determining when to trigger an AOF rewrite event, performed by shard 206, such as the process described below in connection with FIG. 3.

[0032] In some embodiments, shard(s) 206 can be used to provide in-memory non-relational database functionality and any other suitable shard process(es). In some embodiments, the shard(s) can be based on the open-source Redis server with enhanced functionality. In some embodiments, the shard(s) can represent one of the following options: (1) a Redis database (DB); (2) a partition of a Redis DB; (3) a Memcached Bucket; or (4) a partition of a Memcached Bucket. In some embodiments, each cluster's node manages N shards 206, and there can be any suitable number of shards, including zero, in some embodiments.

[0033] In some embodiments, shard(s) 206 use persistent storage 210 to write its AOFs.

[0034] In accordance with some embodiments, common cluster store (CCS) 208 is an internal cluster repository service (which can be based on the Redis architecture). In some embodiments, this service can include a per shard, a per database, a per-node and a per-cluster configuration, statistics, and alert information. All the nodes in a cluster can be synchronized with the per-cluster configuration. The proxy and CNM can be registered to the CCS to receive configuration change events which are relevant to their operations.

[0035] In some embodiments, a node can store one or more shards of the database, which can include one or more database partitions. Each shard can be either a master of a shard or a slave of a shard such that a master of a shard can serve both read and write requests, and a slave of a shard can only serve read requests. In some embodiments, a single node can store multiple shards in any suitable combination of masters of shards and slaves of shards.

[0036] Turning to FIG. 3, an example 300 of a process for determining when to trigger an AOF rewrite operation for a node of a database is shown. In some embodiments, this process can be performed by a CNM 204.

[0037] As illustrated, after process 300 begins at 302, the process can determine the current total utilization of the persistent storage containing the AOFs for the shards of the node at 304. This determination can be made in any suitable manner. For example, in some embodiments, the total utilization of the persistent storage can be determined by dividing the sum of the sizes of all AOFs by the persistent storage size.

[0038] Next, at 306, process 300 can determine whether the current total utilization of the persistent storage is greater than (or greater than or equal to) a threshold for triggering an AOF rewrite. Any suitable threshold (or thresholds) can be used in some embodiments. For example,

in some embodiments, the threshold can be 90%. As another example, in some embodiments, the threshold can be 80%.

[0039] Alternatively to determining current total utilization of the persistent storage as a percentage and comparing that current total utilization of the persistent storage to a threshold percentage at 304 and 306, absolute values of persistent storage space can be used for the current total utilization and threshold. More particularly, for example, in some embodiments, the current total utilization of the persistent storage can be equal to the sum of the sizes of the AOFs, and the threshold can be equal to the size of the persistent storage (or some percentage thereof).

[0040] If the current total utilization of the persistent storage is determined to be greater than (or greater than or equal to) the threshold for triggering an AOF rewrite at 306, process 300 can branch to 316 at which an AOF rewrite is triggered. Once the AOF rewrite is triggered at 316, process 300 can loop back to 304.

[0041] If process 300 determines that the current total utilization of the persistent storage is determined to be not greater than (or greater than or equal to) the threshold for triggering an AOF rewrite at 306, the process can determine the current utilization of the AOF with respect to the AOF's quota for each shard of the node at 308. This determination can be made in any suitable manner. For example, in some embodiments, the current utilization of an AOF for a shard can be determined by dividing the current AOF size for the shard by the maximum AOF size for the shard. The maximum AOF size for the shard can have any suitable value. For example, in some embodiments, the maximum size for the AOF can be equal to the RAM utilization of the shard on the database node (when expressed as a percentage) multiplied by the total usable persistent storage space. Thus, if the RAM utilization of the shard on the database node corresponds to 10% of the database node's usable RAM and the total usable persistent storage space is 100 GB, then the maximum file size can be calculated to be equal to 10 GB (10% * 100 GB).

[0042] Next, at 310, process 300 can determine whether the AOF persistent storage utilization of any shard is greater than (or greater than or equal to) a threshold for triggering an AOF rewrite based on the persistent storage utilization. Any suitable threshold (or thresholds) can be used in some embodiments. For example, in some embodiments, the threshold can be 90%. As another example, in some embodiments, the threshold can be 80%.

[0043] Alternatively to determining persistent storage utilization as a percentage and comparing that persistent storage utilization to a threshold percentage at 308 and 310, absolute values of persistent storage sizes can be used for the AOF persistent storage utilization and threshold. More particularly, for example, in some embodiments, the AOF persistent storage utilization for a shard can be equal to the size of the AOF for the shard, and the threshold can be equal to the maximum AOF size (or some percentage thereof) for the shard (e.g., which can be calculated as described above).

[0044] If the persistent storage utilization of any shard is determined to be greater than (or greater than or equal to) the threshold for triggering an AOF rewrite at 310, process 300 can branch to 316 at which an AOF rewrite is triggered. Once the AOF rewrite is triggered at 316, process 300 can loop back to 304.

[0045] If the persistent storage utilization of any shard is determined to be not greater than (or greater than or equal to) the threshold for triggering an AOF rewrite at 310, process 300 can branch to 312 at which it can determine the recovery time for a recovery of the node from the AOFs for the node (i.e., the AOFs for all of the shards of the node). Process 300 can determine this recovery time in any suitable manner. For example, in some embodiments, this recovery time can be determined using the process of FIG. 4, which is described below.

[0046] Next, at 314, process 300 can determine whether the recovery time is greater than (or greater than or equal to) a threshold for triggering an AOF rewrite based on the recovery time. Any suitable threshold (or thresholds) can be used in some embodiments. For example, in some embodiments, the threshold can be equal to a maximum downtime specified in one or more service-level agreements to which the database must comply. More particularly, for example, this threshold can be one hour.

[0047] If the recovery time is determined to be greater than (or greater than or equal to) the threshold for triggering an AOF rewrite at 314, process 300 can branch to 316 at which an AOF rewrite is triggered. Once the AOF rewrite is triggered at 316, or if the recovery time of any shard is determined to be not greater than (or greater than or equal to) the threshold for triggering an AOF rewrite at 314, process 300 can loop back to 304.

[0048] Turning to FIG. 4, an example 400 of a process for determining the recovery time for a recovery of the node from the AOFs for the node is shown. In some embodiments, this process can be performed by a combination of Shard 206 (FIG. 2) that accumulates the time it takes to

execute each command in the AOF file and CNM 204 (FIG. 2) that compares the accumulated value to what is written in the CCS as recovery threshold and trigger an AOF rewrite operation when the threshold has reached. As illustrated, once process 400 begins at 402, the process can initialize the rewrite time for the shard to an estimated value at 404 that represents the estimated time it takes for the initial part of the AOF file to be loaded in-memory. This estimation can be based the number of keys in the shard and the read throughput provided by the persistent storage volume. Next, the process can wait for a write operation to the shard at 406.

[0049] Once a write operation is completed, process 400 can determine the time to complete the write operation. This determination can be made in any suitable manner. For example, in some embodiments, this determination can be made by measuring the execution time of any write operation that is written to the AOF.

[0050] Next, at 410, process 400 can add the time determined at 408 to a total rewrite time for the shard. Then, at 412, process can determine whether a rewrite operation has been triggered (e.g., as described above in connection with 312 of FIG. 3). If a rewrite operation has be determined to have been triggered, process 400 can loop back to 404. Otherwise, process 400 can loop back to 406.

[0051] As described above in connection with FIGS. 1 and 2, servers can be used to implement one or more application servers, clusters, and/or persistent storage. Any one or more of these servers can be any suitable general purpose device or special purpose device. As described further below, any of these general or special purpose devices can include any suitable components such as a hardware processor (which can be a microprocessor, a digital signal processor, a controller, etc.), memory, communication interfaces, display controllers, input devices, etc.

[0052] In some embodiments, communications between any two or more of the components described in connection with FIGS. 1 and 2 can be implemented using any suitable computer network or combination of such networks including the Internet, an intranet, a wide-area network (WAN), a local-area network (LAN), a wireless network, a Wi-Fi network, a digital subscriber line (DSL) network, a frame relay network, an asynchronous transfer mode (ATM) network, a virtual private network (VPN), an intranet, etc.

[0053] FIG. 5 illustrates an example 500 of hardware that can be used to implement one or more of the servers in accordance with some embodiments. As shown, server 500 can include a

hardware processor 502, memory 504, a communication interface 506, input/output interface(s) 508, and/or any other suitable components. Hardware processor 502 can be implemented using any suitable microprocessor, microcontroller, digital signal processor, etc. Memory 504 can be implemented using any suitable random access memory (RAM), read only memory (ROM), magnetic media, optical media, etc. Communication interface 506 can include any suitable one-way or two-way interface(s) for communicating over wires, optical fiber, airwaves, and/or any other suitable media. Input/output interface(s) 508 can include any suitable interface for receiving user input and/or providing output including keyboards, touch screens, pointer controls (mouse, mice, track ball, stylus, track pad, etc.), displays, audio outputs, etc.

[0054] In some embodiments, any suitable computer readable media can be used for storing instructions for performing the functions and/or processes described herein. This media can be part of or separate from memory 504 in some embodiments. For example, in some embodiments, computer readable media can be transitory or non-transitory. For example, non-transitory computer readable media can include media such as magnetic media (such as hard disks, floppy disks, etc.), optical media (such as compact discs, digital video discs, Blu-ray discs, etc.), semiconductor media (such as random access memory (RAM), flash memory, electrically programmable read only memory (EPROM), electrically erasable programmable read only memory (EEPROM), etc.), any suitable media that is not fleeting or devoid of any semblance of permanence during transmission, and/or any suitable tangible media. As another example, transitory computer readable media can include signals on networks, in wires, conductors, optical fibers, circuits, any suitable media that is fleeting and devoid of any semblance of permanence during transmission, and/or any suitable intangible media.

[0055] In some embodiments of the disclosed subject matter, the above described steps of the processes of FIGS. 3 and 4 can be executed or performed in any order or sequence not limited to the order and sequence shown and described in the figures. Also, some of the above steps of the processes of FIGS. 3 and 4 can be executed or performed substantially simultaneously where appropriate or in parallel to reduce latency and processing times. Furthermore, it should be noted that FIGS. 3 and 4 are provided as examples only. At least some of the steps shown in these figures may be performed in a different order than represented, performed concurrently, or omitted.

[0056] The provision of the examples described herein (as well as clauses phrased as "such as," "e.g.," "including," and the like) should not be interpreted as limiting the claimed subject matter to the specific examples; rather, the examples are intended to illustrate only some of many possible aspects. It should also be noted that, as used herein, the term mechanism can encompass hardware, software, firmware, or any suitable combination thereof.

[0057] Although the invention has been described and illustrated in the foregoing illustrative embodiments, it is understood that the present disclosure has been made only by way of example, and that numerous changes in the details of implementation of the invention can be made without departing from the spirit and scope of the invention, which is limited only by the claims that follow. Features of the disclosed embodiments can be combined and rearranged in various ways.

What is claimed is:

1. A method for controlling append-only-file (AOF) rewrites, comprising:

determining an AOF utilization of a shard of a database;

determining whether the AOF utilization exceeds an AOF utilization threshold;

determining a recovery time to recover the shard from an AOF;

determining whether the recovery time exceeds a recovery time threshold; and

when the AOF utilization is determined to exceed the AOF utilization threshold or when the recovery time is determined to exceed the recovery time threshold, causing an AOF rewrite to be performed.

2. The method of claim 1, wherein determining the utilization comprises dividing an AOF size for the shard by a maximum AOF size for the shard.

3. The method of claim 1, wherein determining the recovery time is based on a maximum downtime specified in a service level agreement.

4. The method of claim 1, wherein determining the recovery time includes:

determining an initial estimated loading time of a snapshot part of the AOF;

for each write operation of a plurality of write operations to the AOF, determining an amount of time required to complete the write operation; and

summing the determined amounts of time for the plurality of write operations with the initial estimate loading time.

5. The method of claim 1, further comprising:

determining a current total persistent storage utilization;

determining whether the current total persistent storage utilization exceeds a total persistent storage utilization threshold; and

when the current total persistent storage utilization exceeds the total persistent storage utilization threshold, causing an AOF rewrite to be performed.

6. A system for controlling append-only-file (AOF) rewrites, comprising:

at least one hardware processor configured to:

determine an AOF utilization of a shard of a database;

determine whether the AOF utilization exceeds an AOF utilization threshold;

determine a recovery time to recover the shard from an AOF;

determine whether the recovery time exceeds a recovery time threshold; and

when the AOF utilization is determined to exceed the AOF utilization threshold or when the recovery time is determined to exceed the recovery time threshold, cause an AOF rewrite to be performed.

7. The system of claim 6, wherein the at least one hardware processor determines the utilization by at least dividing an AOF size for the shard by a maximum AOF size for the shard.

8. The system of claim 6, wherein the at least one hardware processor determines the recovery time based at least on a maximum downtime specified in a service level agreement.

9. The system of claim 6, wherein the at least one hardware processor determines the recovery time by at least:

determining an initial estimated loading time of a snapshot part of the AOF;

for each write operation of a plurality of write operations to the AOF, determining an amount of time required to complete the write operation; and

summing the determined amounts of time for the plurality of write operations with the initial estimated loading time.

10. The system of claim 6, wherein the at least one hardware processor is also configured to:

determine a current total persistent storage utilization;

determine whether the current total persistent storage utilization exceeds a total persistent storage utilization threshold; and

when the current total persistent storage utilization exceeds the total persistent storage utilization threshold, cause an AOF rewrite to be performed.

11. A non-transitory computer-readable medium containing computer executable instructions that, when executed by a processor, cause the processor to perform a method for controlling append-only-file (AOF) rewrites, the method comprising:

determining an AOF utilization of a shard of a database;

determining whether the AOF utilization exceeds an AOF utilization threshold;

determining a recovery time to recover the shard from an AOF;

determining whether the recovery time exceeds a recovery time threshold; and

when the AOF utilization is determined to exceed the AOF utilization threshold or when the recovery time is determined to exceed the recovery time threshold, causing an AOF rewrite to be performed.

12. The non-transitory computer-readable medium of claim 11, wherein determining the utilization comprises dividing an AOF size for the shard by a maximum AOF size for the shard.

13. The non-transitory computer-readable medium of claim 11, wherein determining the recovery time is based on a maximum downtime specified in a service level agreement.

14. The non-transitory computer-readable medium of claim 11, wherein determining the recovery time includes:

determining an initial estimated loading time of a snapshot part of the AOF;

for each write operation of a plurality of write operations to the AOF, determining an amount of time required to complete the write operation; and

summing the determined amounts of time for the plurality of write operations with the initial estimated loading time.

15. The non-transitory computer-readable medium of claim 11, wherein the method further comprises:

determining a current total persistent storage utilization;

determining whether the current total persistent storage utilization exceeds a total persistent storage utilization threshold; and

when the current total persistent storage utilization exceeds the total persistent storage utilization threshold, causing an AOF rewrite to be performed.

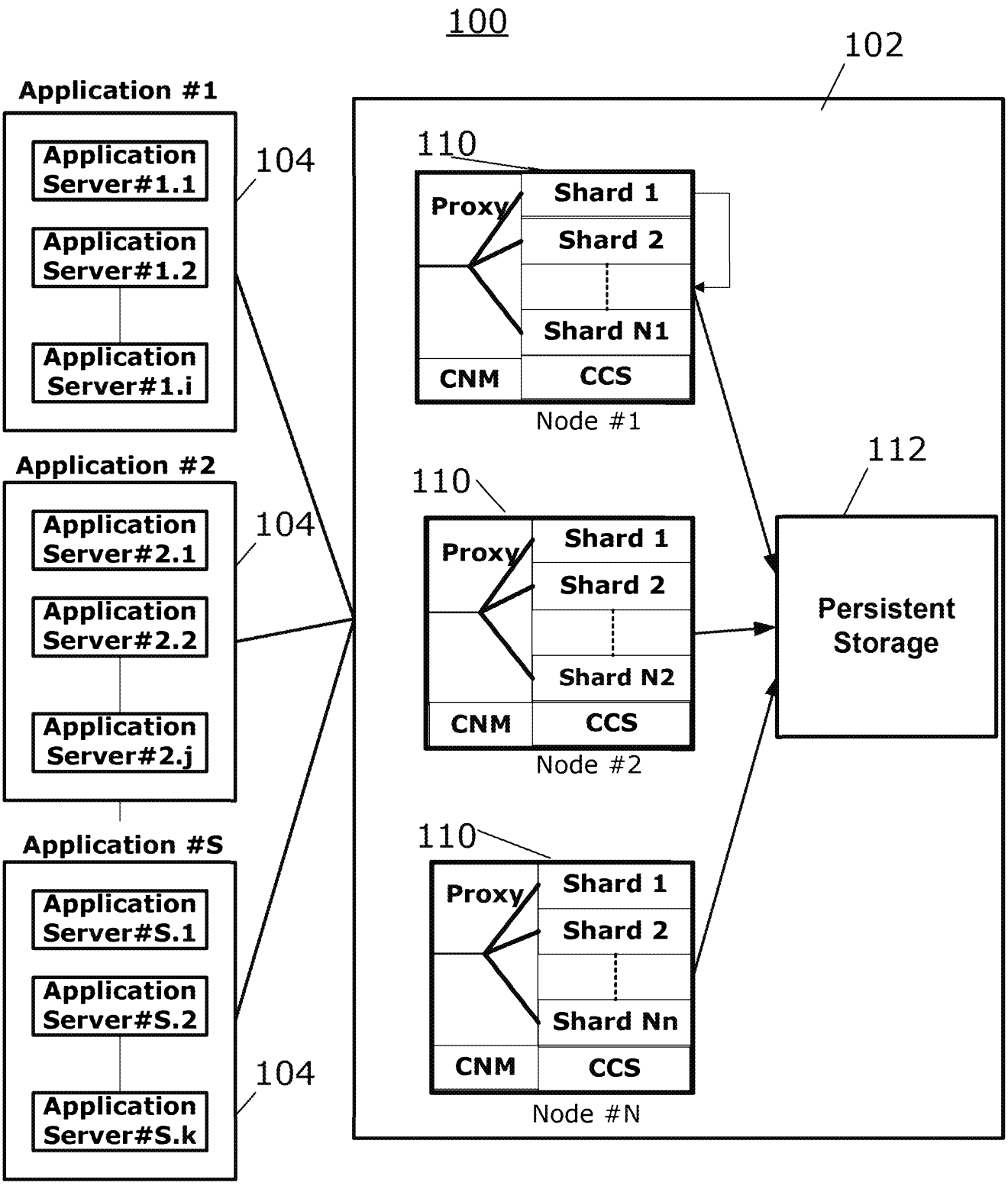
100
102
Application #1
Application Server#1.1
Application Server#1.2
Application Server#1.i
104
Application #2
Application Server#2.1
Application Server#2.2
Application Server#2.j
104
Application #S
Application Server#S.1
Application Server#S.2
Application Server#S.k
104
110
Proxy
Shard 1
Shard 2
Shard N1
CNM
CCS
Node #1
110
Proxy
Shard 1
Shard 2
Shard N2
CNM
CCS
Node #2
110
Proxy
Shard 1
Shard 2
Shard Nn
CNM
CCS
Node #N
112
Persistent Storage

FIG. 1

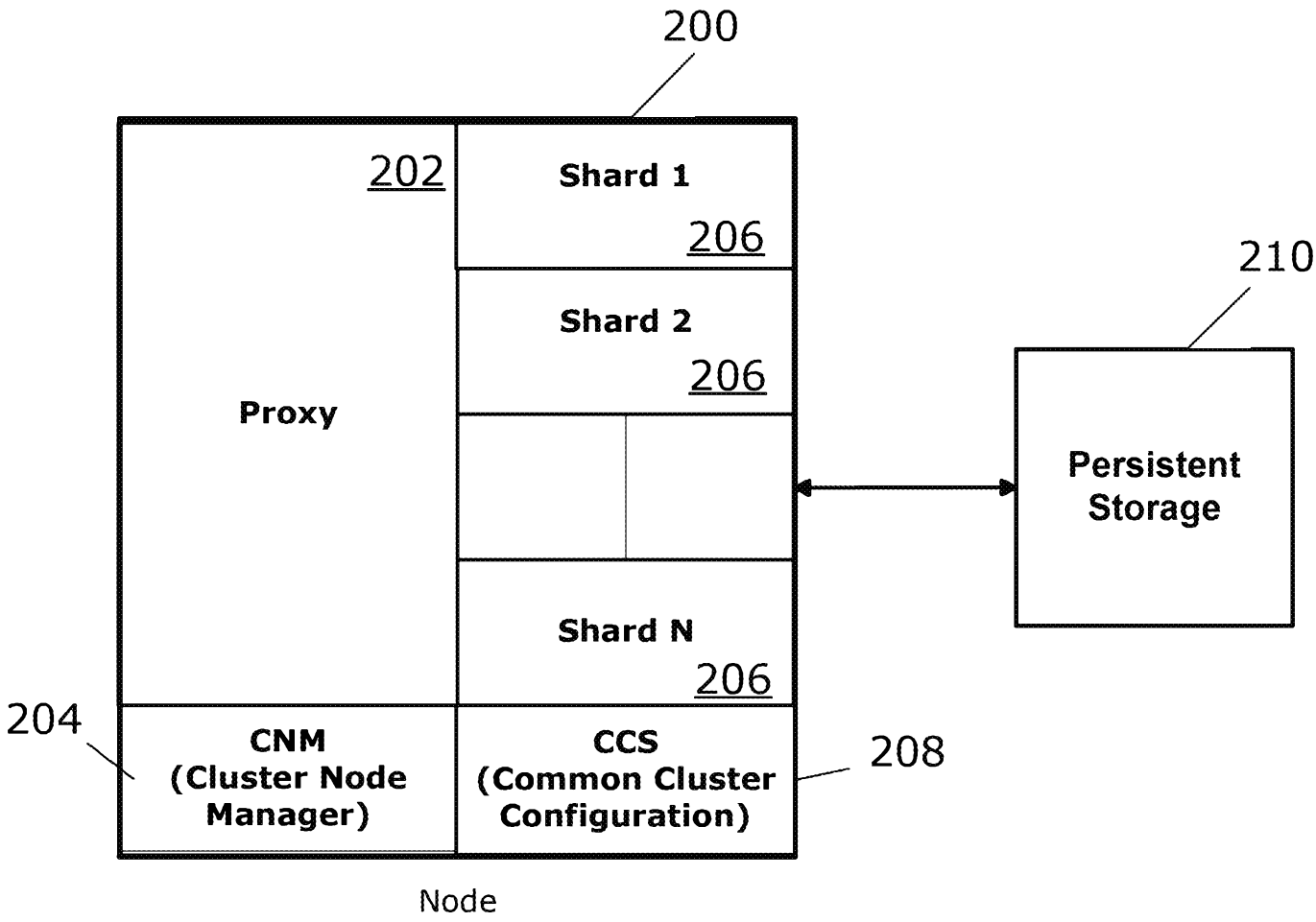
200
202
Shard 1
206
Shard 2
206
Proxy
Shard N
206
210
Persistent Storage
204
CNM (Cluster Node Manager)
CCS (Common Cluster Configuration)
208
Node

FIG. 2

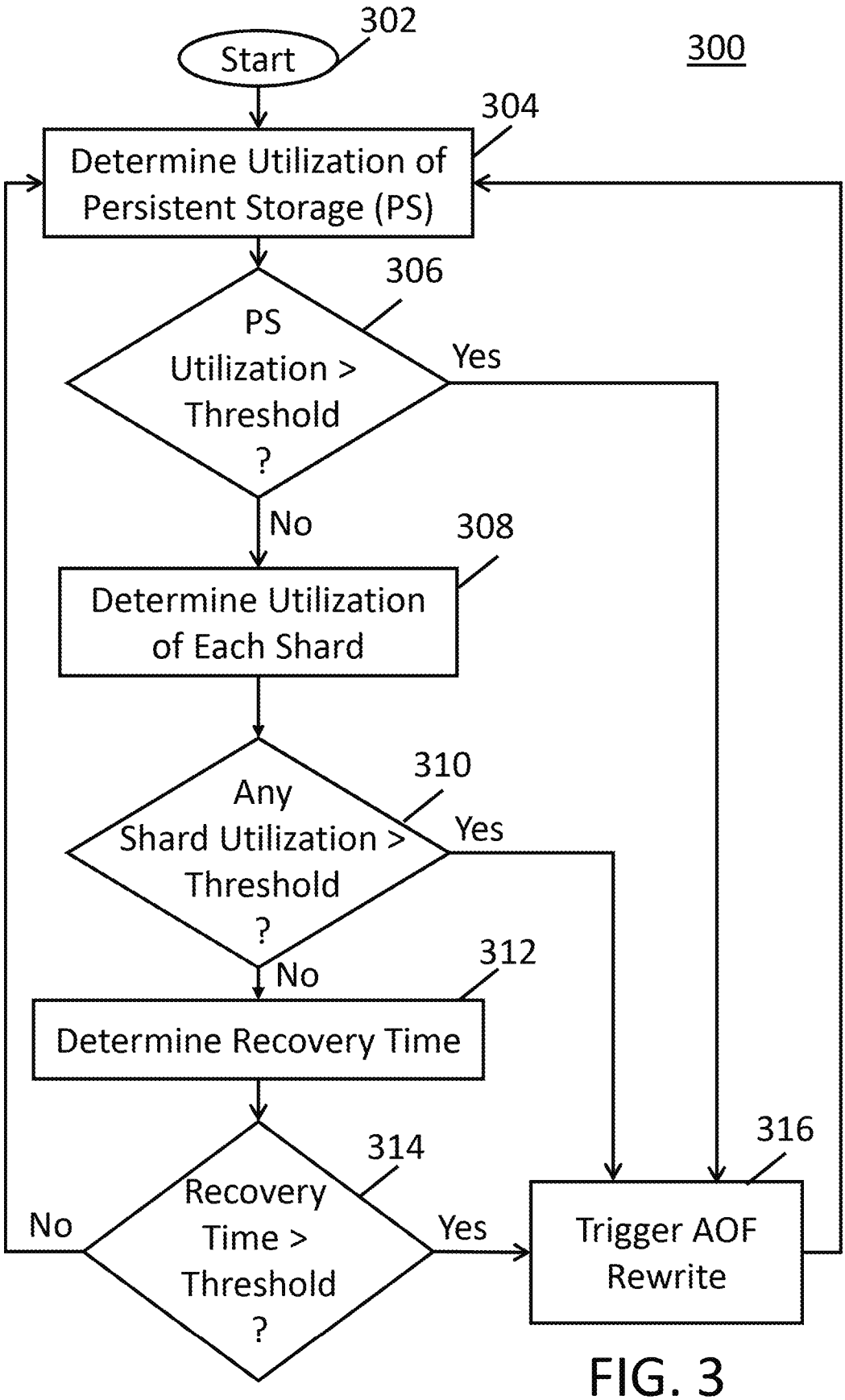
300
302
Start
304
Determine Utilization of Persistent Storage (PS)
306
PS Utilization > Threshold ?
Yes
No
308
Determine Utilization of Each Shard
310
Any Shard Utilization > Threshold ?
Yes
No
312
Determine Recovery Time
314
Recovery Time > Threshold ?
No
Yes
316
Trigger AOF Rewrite

FIG. 3

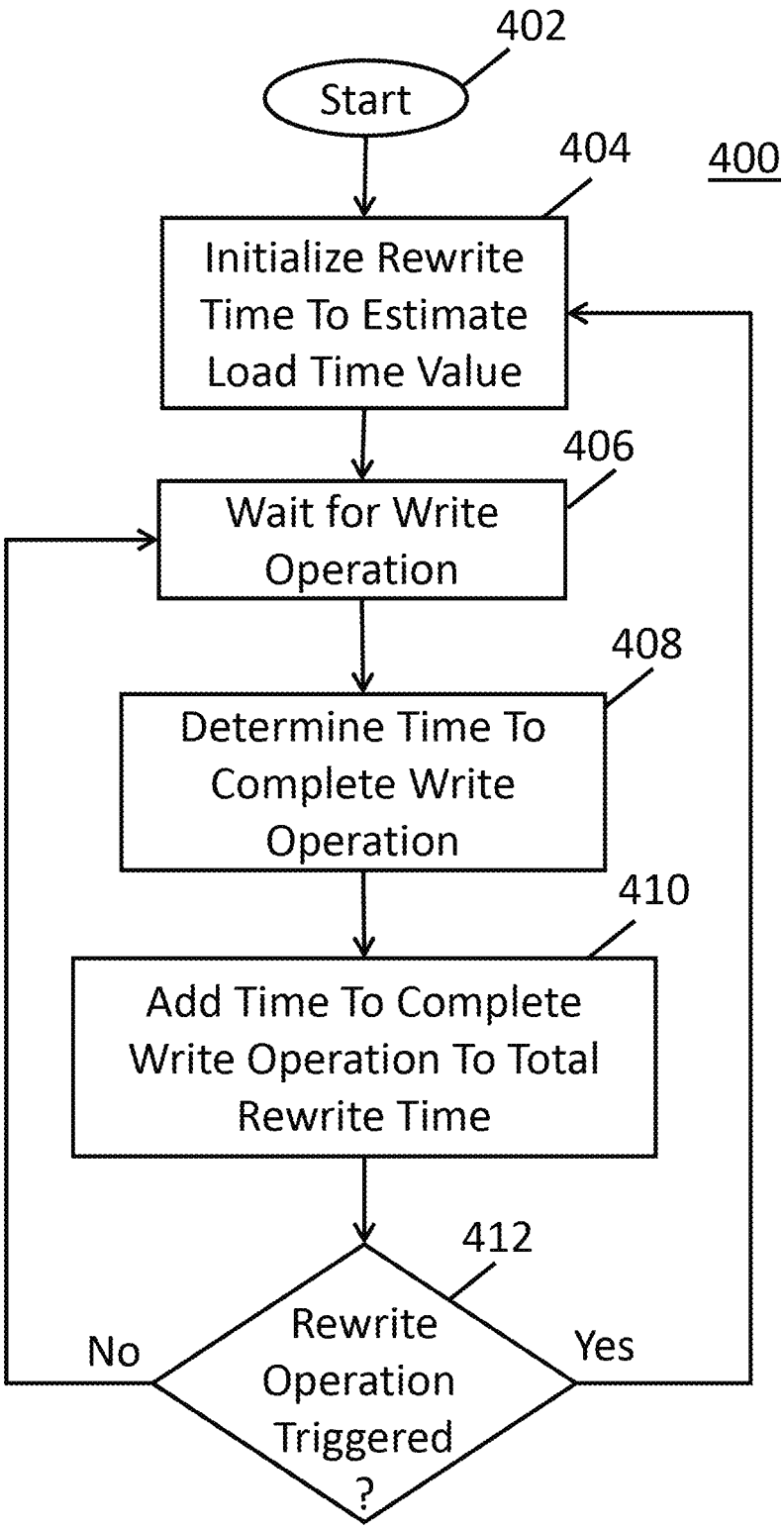
402
Start
404
400
Initialize Rewrite Time To Estimate Load Time Value
406
Wait for Write Operation
408
Determine Time To Complete Write Operation
410
Add Time To Complete Write Operation To Total Rewrite Time
412
Rewrite Operation Triggered ?
No
Yes

FIG. 4

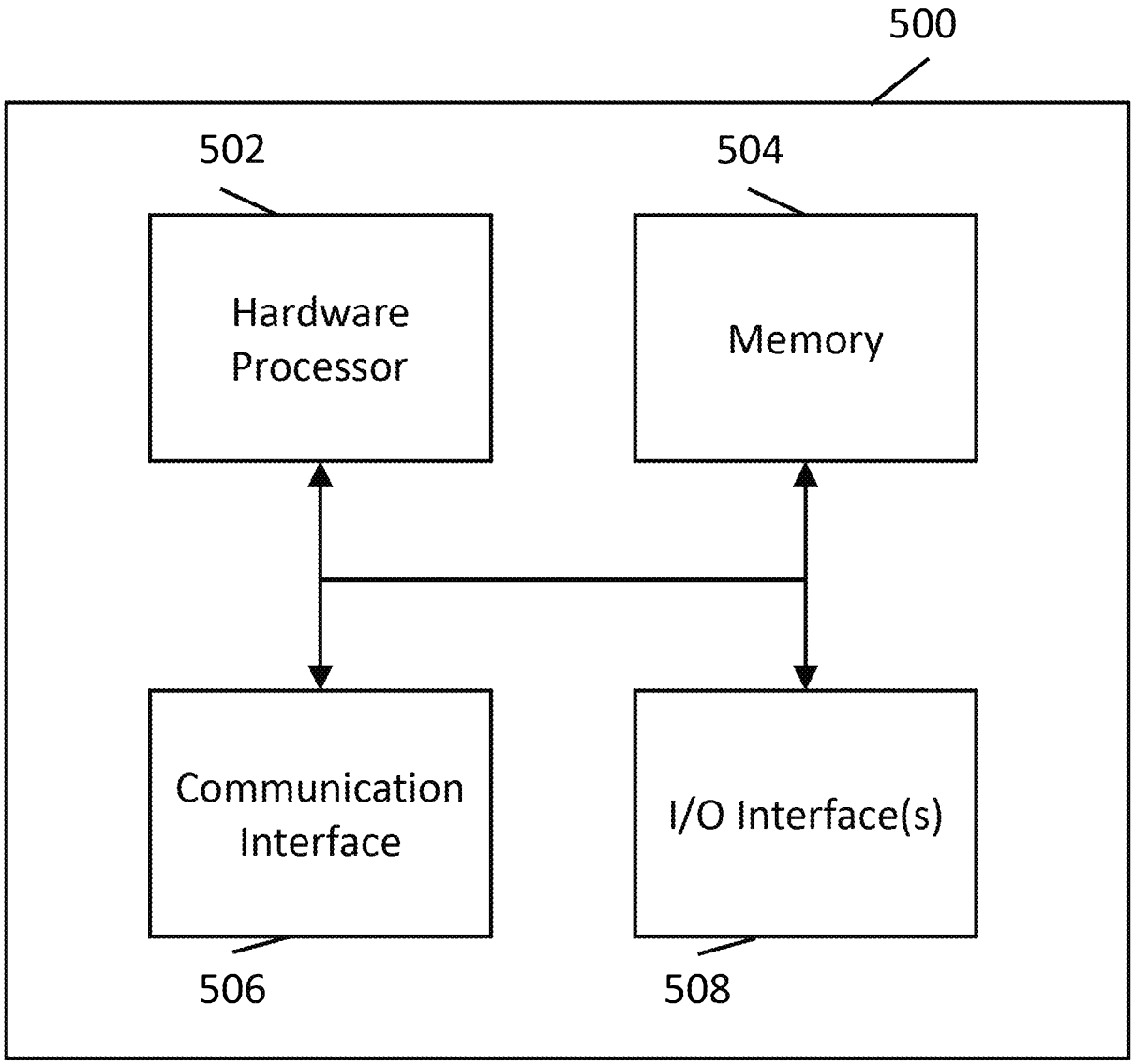
500
502
504
Hardware Processor
Memory
Communication Interface
I/O Interface(s)
506
508

FIG. 5