(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 3 889 785 A1

(12) EUROPEAN PATENT APPLICATION

published in accordance with Art. 153(4) EPC

(43) Date of publication:
06.10.2021 Bulletin 2021/40

(21) Application number: **19901639.5**

(22) Date of filing: **29.08.2019**

(51) Int Cl.:
G06F 12/00 *(2006.01)*

(86) International application number:
PCT/CN2019/103281

(87) International publication number:
WO 2020/134143 (02.07.2020 Gazette 2020/27)

(84) Designated Contracting States:
AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO RS SE SI SK SM TR
Designated Extension States:
BA ME
Designated Validation States:
KH MA MD TN

(30) Priority: **25.12.2018 CN 201811589151**

(71) Applicant: **Huawei Technologies Co., Ltd.**
Shenzhen Guangdong 518129 (CN)

(72) Inventors:
- **ZHAN, Yun**
  Shenzhen, Guangdong 518129 (CN)
- **XIE, Huiyun**
  Shenzhen, Guangdong 518129 (CN)
- **WANG, Tonglei**
  Shenzhen, Guangdong 518129 (CN)

(74) Representative: **Pfenning, Meinig & Partner mbB**
Patent- und Rechtsanwälte
Theresienhöhe 11a
80339 München (DE)

(54) STRIPE RECONSTRUCTION METHOD IN STORAGE SYSTEM AND STRIPING SERVER

(57) A stripe reassembling method is provided, including: A stripe server selects stripes (701); uses data strips (D_{21}, D_{32} and D_{13}) including valid data in the stripes S_1, S_2, and S_3 as data strips in a new stripe S_4, and generates data of a parity strip P_{41} for data of a data strip in S_4 according to an EC algorithm the same as that of S_1, S_2, and S_3 (702); and stores the parity data of the parity strip P_{41} on a parity storage node N_1 (703). In a plurality of selected to-be-reassembled stripes, a maximum of one data strip includes valid data in data strips distributed on a same data storage node, a data strip that includes valid data and that is in the plurality of stripes is used as data of a data strip in a new stripe, parity data of a parity strip is generated for the data of the data strip in the new stripe, and the parity data of the parity strip is stored in a corresponding parity storage node.

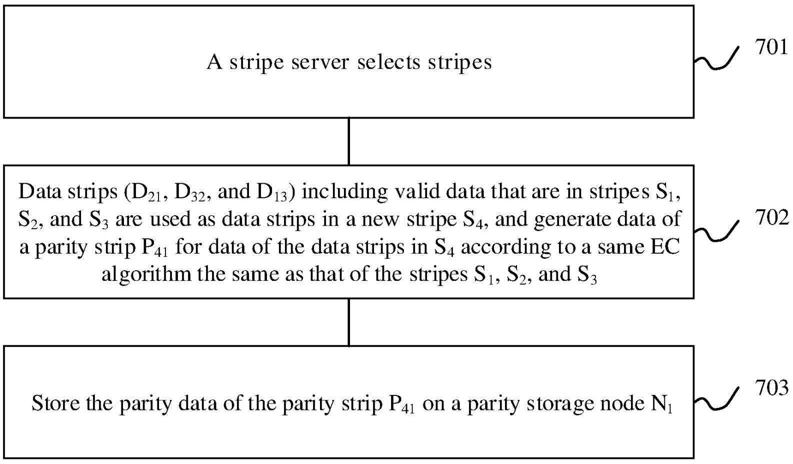

A stripe server selects stripes
701
Data strips (D21, D32, and D13) including valid data that are in stripes S1, S2, and S3 are used as data strips in a new stripe S4, and generate data of a parity strip P41 for data of the data strips in S4 according to a same EC algorithm the same as that of the stripes S1, S2, and S3
702
Store the parity data of the parity strip P41 on a parity storage node N1
703

FIG. 7

Printed by Jouve, 75001 PARIS (FR)

Description

TECHNICAL FIELD

[0001] The present invention relates to the field of information technologies, and in particular, to a stripe reassembling method in a storage system and a stripe server.

BACKGROUND

[0002] A distributed storage system includes a plurality of storage nodes. When a client receives a write request sent by a host and writes data into the distributed storage system, the data is stored in a corresponding storage node in a form of a stripe. For example, according to an erasure coding (Erasure Coding, EC) algorithm, a quantity of data strips in a stripe is N, a quantity of parity strips in the stripe is M, and a length of the stripe is N+M, where both N and M are positive integers. In the distributed storage system, a stripe with a length of N+M includes N+M strips with a same length, and each strip is distributed on one storage node in the distributed storage system. Therefore, one strip is distributed on N+M storage nodes.

[0003] The client receives a data write request, where the write request is used to modify data of a data strip that has been stored in a stripe. Generally, the client writes the data of the data write request to a new stripe, to mark data of a data strip in an original stripe as invalid. When a quantity of data strips that store invalid data in a stripe reaches a specific value, the distributed storage system reassembles the stripe, and reassembles valid data of data strips in a plurality of stripes into a new stripe. In this process, valid data needs to be migrated across storage nodes.

SUMMARY

[0004] This application provides a stripe reassembling method in a storage system and a computer program product, to reduce cross-node migration of valid data in a stripe reassembling process.

[0005] According to a first aspect, this application provides a stripe reassembling method in a storage system. The storage system includes N data storage nodes that store data strips and M parity storage nodes that store parity strips. R stripes are distributed on the N+M storage nodes, each stripe S_i includes N data strips and M parity strips, a data strip D_{ix} is distributed on an x^{th} storage node in the N data storage nodes, and a parity strip P_{iy} is distributed on a y^{th} storage node in the M parity storage nodes. N, M, and R are positive integers, R is not less than 2, a value of i is an integer ranging from 1 to R, a value of x is an integer ranging from 1 to N, and a value of y is an integer ranging from 1 to M. A stripe server selects the R stripes, where a maximum of one data strip includes valid data in data strips D_{ix} that are in the R stripes and that are distributed on a same data storage node. In this solution, the stripe server generates parity data of a parity strip P_{Ky} in a new stripe S_K for data of the data strip including valid data in the R stripes, where K is an integer different from 1 to R. The new stripe S_K includes the data strip including valid data in the R stripes and the parity strip P_{Ky}. The stripe server stores the parity data of the parity strip P_{Ky} on the y^{th} storage node in the M parity storage nodes. In the stripe reassembling method, a data strip including valid data in a plurality of stripes is used as a data strip in a new stripe. In addition, a maximum of one data strip includes valid data in data strips D_x that are in the plurality of stripes and that are on distributed a same data storage node. Therefore, data of the data strip does not need to be migrated across storage nodes during stripe reassembling.

[0006] With reference to the first aspect of this application, in a possible implementation, the method further includes: After the stripe server stores the parity data of the parity strip P_{Ky} on the y^{th} parity storage node in the M parity storage nodes, the stripe server indicates a data storage node on which a data strip that stores garbage data of the R stripes is distributed to perform garbage collection. In another implementation, the data storage node checks a status of stored data of a data strip, and starts garbage collection when the data of the data strip is determined as garbage data. In a specific implementation, when storing the data of the data strip, the data storage node creates a mapping between a host access address of the data of the data strip and an identifier of the data strip. When the data of the data strip is garbage data, the mapping between the host access address of the data of the data strip and the identifier of the data strip is marked as invalid, so that the data of the data strip is determined as garbage data. Further, the data storage node may perform garbage collection in the data strip after the stripe server releases a stripe including the data strip. In another implementation, the data storage node may determine, based on an amount of garbage data of the data strip stored in the data storage node, to start garbage collection. Alternatively, a parity storage node may determine, based on whether data of a parity strip is garbage data, whether to start garbage collection, or perform garbage collection based on an indication of the stripe server.

[0007] With reference to the first aspect of this application and the foregoing possible implementation of the first aspect, in another possible implementation, the stripe server records a mapping between the new stripe and a strip identifier, where the strip identifier includes an identifier of the data strip storing valid data in the R stripes and an identifier of the parity strip in the new stripe.

[0008] With reference to the first aspect of this application and the foregoing possible implementations of the first aspect, in another possible implementation, the stripe server releases the R stripes. After the stripe server releases the R stripes, the R stripes may be reallocated to newly written data.

[0009] With reference to the first aspect of this application and the foregoing possible implementations of the first aspect, in another possible implementation, the stripe server determines the stripe S_i, where a quantity of data strips including garbage data in the stripe Si meets a reassembly threshold. The stripe server determines, based on a quantity of data strips including garbage data in a stripe, whether the reassembly threshold is reached in the stripe. The stripe server records a stripe in which the reassembly threshold is reached. In a specific implementation, an identifier of the stripe in which the reassembly threshold is reached may be recorded by using a link.

[0010] According to a second aspect, this application provides a stripe server. The stripe server is used in a storage system, and the storage system includes N data storage nodes storing data strips and M parity storage nodes storing parity strips. R stripes are distributed on the N+M storage nodes, each stripe S_i includes N data strips and M parity strips, a data strip D_{ix} is distributed on an xth storage node in the N data storage nodes, and a parity strip P_{iy} is distributed on a yth storage node in the M parity storage nodes. N, M, and R are positive integers, R is not less than 2, a value of i is an integer ranging from 1 to R, a value of x is an integer ranging from 1 to N, and a value of y is an integer ranging from 1 to M. The stripe server includes an interface and a processor, the interface communicates with the processor, and the processor is configured to perform the first aspect of this application and the possible implementations of the first aspect.

[0011] According to a third aspect, this application provides a stripe server. The stripe server is used in a storage system, and the storage system includes N data storage nodes storing data strips and M parity storage nodes storing parity strips. R stripes are distributed on the N+M storage nodes, each stripe S_i includes N data strips and M parity strips, a data strip D_{ix} is distributed on an xth storage node in the N data storage nodes, and a parity strip P_{iy} is distributed on a yth storage node in the M parity storage nodes. N, M, and R are positive integers, R is not less than 2, a value of i is an integer ranging from 1 to R, a value of x is an integer ranging from 1 to N, and a value of y is an integer ranging from 1 to M. The stripe server includes corresponding units, and each unit is configured to perform corresponding operations of the first aspect of this application and the possible implementations of the first aspect.

[0012] According to a fourth aspect, this application provides a computer program product. The computer program product includes a computer instruction, the computer program product is used in a storage system, and the storage system includes N data storage nodes storing data strips and M parity storage nodes storing parity strips. R stripes are distributed on the N+M storage nodes, each stripe S_i includes N data strips and M parity strips, a data strip D_{ix} is distributed on an xth storage node in the N data storage nodes, and a parity strip P_{iy} is distributed on a yth storage node in the M parity storage nodes. N, M, and R are positive integers, R is not less than 2, a value of i is an integer ranging from 1 to R, a value of x is an integer ranging from 1 to N, and a value of y is an integer ranging from 1 to M. A stripe server that is used in the storage system executes the computer instruction, to perform the first aspect of this application and the possible implementations of the first aspect.

BRIEF DESCRIPTION OF DRAWINGS

[0013]

FIG. 1 is a schematic diagram of a distributed storage system according to an embodiment of the present invention;

FIG. 2 is a schematic structural diagram of a server in a distributed block storage system according to an embodiment of the present invention;

FIG. 3 is a schematic diagram of a partition view of a distributed block storage system according to an embodiment of the present invention;

FIG. 4 is a schematic diagram of a relationship between a strip and a storage node in a distributed block storage system according to an embodiment of the present invention;

FIG. 5 is a schematic diagram of distribution of stripes according to an embodiment of the present invention;

FIG. 6 is a schematic diagram of garbage data in stripes according to an embodiment of the present invention;

FIG. 7 is a flowchart of a stripe reassembling method according to an embodiment of the present invention;

FIG. 8 is a schematic diagram of stripe reassembling distribution according to an embodiment of the present invention; and

FIG. 9 is a schematic structural diagram of a stripe server according to an embodiment of the present invention.

EMBODIMENTS OF THE PRESENT INVENTION

[0014] A storage system in the embodiments of the present invention may be a distributed storage system, for example, FusionStorage® series or OceanStor® 9000 series of Huawei®. For example, as shown in FIG. 1, a distributed storage system includes a plurality of servers, such as a server 1, a server 2, a server 3, a server 4, a server 5, and a server 6. The servers communicate with each other through InfiniBand, the Ethernet, or the like. In the embodiments of the present invention, the server in the distributed storage system is also referred to as a storage node. In a practical application, a quantity of servers in the distributed storage system may be increased based on an actual requirement. This is not limited in the embodiments of the present invention. It should be noted that the storage node may alternatively be a hard disk (for example, a disk or a solid-state drive), a storage array, or a server chassis, provided that strips in a same stripe are located in different hard disks. The following uses an example in which the storage node is a server for description. In other cases, a principle is the same, and details are not described in the present invention.

[0015] The server in the distributed storage system is in a structure shown in FIG. 2. As shown in FIG. 2, each server in the distributed storage system includes a central processing unit (Central Processing Unit, CPU) 201, a memory 202, an interface 203, a hard disk 1, a hard disk 2, and a hard disk 3. The memory 202 stores a computer instruction, and the CPU 201 executes the computer instruction in the memory 202 to perform a corresponding operation. The interface 203 may be a hardware interface, for example, a network interface card (Network Interface Card, NIC) or a host bus adapter (Host Bus Adaptor, HBA), or may be a program interface module. A hard disk includes a solid-state drive (Solid State Disk, SSD), a hard disk drive, or a hybrid hard disk. In a hard disk drive such as an HDD (Hard Disk Drive), a hard disk interface may be a serial advanced technology attachment (Serial Advanced Technology Attachment, SATA) interface, a serial attached small computer system interface (Serial Attached Small Computer System Interface, SAS) interface, a fiber channel (Fiber Channel, FC) interface, a peripheral component interconnect-express (Peripheral Component Interconnect-Express, PCIe) interface, a non-volatile memory express (Non-Volatile Memory Express, NVMe) interface, or the like. The CPU 201 may be replaced by a field programmable gate array (Field Programmable Gate Array, FPGA) or other hardware, or an FPGA or other hardware and the CPU 201 jointly perform the foregoing corresponding operation. For ease of description, in the embodiments of the present invention, the CPU 201 and the memory 202 are referred to as a processor, or hardware that replaces the CPU 201 and the memory 202 is referred to as a processor, or a combination of the CPU 201, the memory 202, and other hardware is referred to as a processor.

[0016] A client in the distributed storage system writes data into the distributed storage system based on a write request of a host, or reads data from the distributed storage system based on a read request of the host. The server in the embodiments of the present invention may be used as a client. In addition, the client may also be a device independent of the server shown in FIG. 2. The host in the embodiments of the present invention may be a server, a virtual machine (Virtual Machine, VM), a terminal device, or the like. This is not limited in the embodiments of the present invention. The client in the distributed storage system provides a storage resource of the distributed storage system for the host. For example, in a distributed block storage system, a client provides a block resource such as a logical unit for a host, to provide a data access operation for the host. The logical unit is also referred to as a logical unit number (Logical Unit Number, LUN). In a distributed file storage system, a client provides a file resource for a host. In a distributed object storage system, a client provides an object resource for a host.

[0017] In the embodiments of the present invention, the distributed block storage system is used as an example. A client provides a block protocol access interface, so that the client provides a distributed block storage access point service. A host may access a storage resource in a storage resource pool in the distributed block storage system by using the client. Generally, the block protocol access interface is configured to provide a logical unit for the host. A server runs a program of the distributed block storage system, so that a server that includes a hard disk is used as a storage node to store data of the client. For example, one hard disk may be used as one storage node by default in the server. When the server includes a plurality of hard disks, one server may be used as a plurality of storage nodes. In another implementation, the server runs the program of the distributed block storage system program to serve as one storage node. This is not limited in the embodiments of the present invention. Therefore, for a structure of the storage node, refer to FIG. 3 and related description. During initialization of the distributed block storage system, hash space (for example, 0 to 2^32) is divided into N equal parts, each equal part is one partition (Partition), and the N equal parts are equally allocated based on a quantity of hard disks. For example, in the distributed block storage system, N is 3600 by default, to be specific, partitions are P1, P2, P3, ..., and P3600. Assuming that the distributed block storage system currently includes 18 disks (storage nodes), each storage node carries 200 partitions. A partition P includes M storage nodes N_j. A correspondence between a partition and a storage node, that is, a mapping between a partition and a storage node N_j included in the partition, is also referred to as a partition view. As shown in FIG. 3, in an example in which a partition includes four storage nodes N_j, a partition view is "P2-a storage node N_1 (a data storage node)-a storage node N_2 (a data storage node)-a storage node N_3 (a data storage node)-a storage node N_4 (a parity storage node)", where j is an integer ranging from 1 to M. The partition view may be allocated during the initialization of the distributed block storage system, and may be subsequently adjusted based on a change of the quantity of hard disks in the distributed block storage system. The client saves the partition view.

[0018] Based on a reliability requirement of the distributed block storage system, data reliability may be improved by using an erasure coding (Erasure Coding, EC) algorithm. For example, a 3+1 mode is used, to be specific, a stripe includes three data strips and one parity strip. In the embodiments of the present invention, data is stored in a partition in a form of stripe, and one partition includes R stripes S_i, where i is an integer ranging from 1 to R. In the embodiments of the present invention, P2 is used as an example for description.

[0019] It should be noted that a stripe (stripe) is logical storage space, the stripe (stripe) includes a plurality of strips, and a strip is also referred to as a strip unit (strip unit). "Data of a strip" is content stored in the strip.

[0020] Strips included in a stripe include a data strip (data strip) and a parity strip (parity strip). A strip used to store data (based on application scenarios, data may be user data, service data, and application data) is referred to as a data strip, and a strip used to store parity data is referred to as a parity strip. There is a parity relationship between data and parity data. According to the EC algorithm, when content in some strips is faulty or cannot be read, content in the other strips may be used to restore. In the embodiments of the present invention, a storage node that stores data of a data strip is referred to as a data storage node, a storage node that stores parity data of a parity strip is referred to as a parity storage node, and the data storage node and the parity storage node are collectively referred to as storage nodes.

[0021] In the distributed block storage system, slice management is performed on the hard disk by 8 KB (Kilobyte, KB), and allocation information of each 8 KB slice is recorded in a metadata management area of the hard disk. Slices of the hard disk form a storage resource pool. The distributed block storage system includes a stripe server. A specific implementation may be that a stripe management program is run on one or more servers in the distributed block storage system. The stripe server allocates a stripe to a partition. For a structure of the stripe server, refer to the server structure shown in FIG. 2. Still using the partition view shown in FIG. 4 as an example, the stripe server allocates a stripe with a length 4 based on the partition view shown in FIG 4. To be specific, the stripe includes three data strips and one parity strip. The stripe server allocates, to a stripe S_i of a partition P2, a storage address, that is, storage space, from a data storage node N_x that stores a data strip D_{ix} corresponding to the partition, and a storage address from a parity storage node D_{iy} of a parity strip P_y corresponding to the partition. This specifically includes: allocating, to D_{i1}, a storage address from a data storage node N_1 that stores data strips, allocating, to D_{i2}, a storage address from a data storage node N_2 that stores data strips, allocating, to D_{i3}, a storage address from a data storage node N_3 that stores data strips, and allocating, to P_{i1}, a storage address from a parity storage node N_1 (to be specific, a storage node N_4 in FIG. 4) that stores parity strips. In another implementation, the stripe server does not allocate, to the stripe S_i of the partition P_2, a storage address from the data storage node N_x that stores the data strip D_{ix} corresponding to the partition, or a storage address from the storage node D_{iy} of the parity strip P_y corresponding to the partition. To be specific, when the client writes data to a storage node, the data storage node N_x that stores data strips allocates a storage address to the data strip D_{ix}, and the parity storage node N_y that stores parity strips allocates a storage address to the parity strip P_{iy}, where a value of x is an integer ranging from 1 to 3, and a value of y is 1. In the embodiments of the present invention, a storage address allocated to a strip may be specifically a logical address of a hard disk in the storage node, for example, a logical block address (Logical Block Address, LBA) of the hard disk. In another implementation, in an SSD that supports an open channel (Open-channel), a storage address allocated to a strip may alternatively be a physical address of the SSD. In another implementation, when an LUN is mounted on a storage node, a storage address allocated to a strip is an LBA of the LUN. Stripe metadata records a mapping relationship between a stripe identifier and a strip identifier, to be specific, a correspondence between S_i, the data strip D_{ix}, and the parity strip P_{iy}. A strip included in the stripe S_i may be found based on the correspondence. The stripe server further records a correspondence between a strip and the stripe S_i. Based on the correspondence, the stripe S_i may be found by using the strip, so that stripe information, for example, all strips included in S_i, is queried.

[0022] To reduce a quantity of strip identifiers managed by the stripe server, the stripe server allocates a version number to a strip identifier in a stripe. After a stripe is released, a version number of a strip identifier of a strip in the released stripe is updated, so that a version number of the strip identifier is used as a strip identifier of a strip in a new stripe. The stripe server pre-allocates a strip to the stripe S_i, so that waiting time may be reduced when the client writes data, thereby improving write performance of the distributed block storage system. In the embodiments of the present invention, a strip in the stripe S_i has a unique identifier in the distributed block storage system.

[0023] In the embodiments of the present invention, an example in which a stripe length is 4 is used. To be specific, the stripe includes three data strips and one parity strip. Distribution of stripes on storage nodes is shown in FIG. 5. A stripe Si includes data strips D_{11}, D_{12}, and D_{13}, and a parity strip P_{11}, a stripe S_2 includes data strips D_{21}, D_{22}, and D_{23}, and a parity strip P_{21}, and a stripe S_3 includes data strips D_{31}, D_{32}, and D_{33} and a parity strip P_{31}. Data of the data strips D_{11}, D_{21}, and D_{31} is stored on a data storage node N_1 that stores data strips, data of the data strips D_{12}, D_{22}, and D_{32} is stored on a data storage node N_2 that stores data strips, data of the data strip D_{13}, D_{23}, and D_{33} is stored on a data storage node N_3 that stores data strips, and parity data of the parity strips P_{11}, P_{21}, and P_{31} is stored on a parity storage node N_1 that stores parity strips. A host modifies stored data, for example, modifies data of the data strips D_{11}, D_{12}, D_{22}, D_{23}, D_{31}, and D_{33}. The storage system provided in the embodiments of the present invention supports only appending, to be specific, when existing data is modified only, modified data is stored in a new data strip, and the modified data does not occupy a data strip used to store data before the modification. In an appending-based storage system, after data is modified in a data strip, data in a data strip that stores the data before the modification becomes garbage data, that is, invalid data. Therefore, as shown in FIG. 6, the data of the data strips D_{11}, D_{12}, D_{22}, D_{23}, D_{31}, and D_{33} is marked as garbage data. When an amount of garbage data in the stripe reaches a reassembly threshold, the stripe server starts a stripe reassembling operation. The stripe server determines, based on a quantity of data strips including garbage data in the stripe S_i, that the stripe S_i reaches the reassembly threshold. The stripe server determines, based on a quantity of data strips including garbage data in a stripe, whether the reassembly threshold is reached in the stripe. The stripe server records a stripe in which the reassembly threshold is reached. In a specific implementation, an identifier of the stripe in which the reassembly threshold is reached may be recorded by using a link.

[0024] In an embodiment of the present invention shown in FIG. 7, step 701: A stripe server selects stripes.

[0025] The selected stripes are to-be-reassembled stripes, and in the to-be- reassembled stripes, a maximum of one data strip includes valid data in data strips distributed on a same data storage node. For example, strip distribution of stripes S_1, S_2, and S_3 is the same, and there are three groups of data strips on three data storage nodes. A first group includes D_{11}, D_{21}, and D_{31}, the second group includes D_{12}, D_{22}, and D_{32}, and the third group includes D_{13}, D_{23}, and D_{33}. As shown in FIG. 6, in the stripes S_1, S_2, and S_3, only one data strip includes valid data in data strips distributed on a same data storage node. Therefore, the stripes S_1, S_2, and S_3 are selected for reassembling. In other words, S_1, S_2, and S_3 are selected for reassembling because: (1) the stripes participating in reassembling each have a data strip that stores valid data; and (2) the data strips including valid data in the stripes participating in reassembling are not distributed on a same data storage node, that is, y in D_{iy} is different.

[0026] In this embodiment of the present invention, valid data is data whose host access address undergoes no write operation after the data is written into a data storage node.

[0027] Step 702: Use the data strips (D_{21}, D_{32}, and D_{13}) including valid data in the stripes S_1, S_2, and S_3 as data strips in a new stripe S4, and generate data of a parity strip P_{41} for data of the data strips in S_4 according to an EC algorithm the same as that of S_1, S_2, and S_3.

[0028] Step 703: Store the parity data of the parity strip P_{41} on a parity storage node N_1.

[0029] The stripe server records a mapping between the stripe S_4 and the strips D_{21}, D_{32}, D_{13}, and P_{41}, and the strips D_{21}, D_{32}, D_{13}, and P_{41} are strips in the stripe S_4.

[0030] In the embodiment shown in FIG. 7, the data strips D_{21}, D_{32}, and D_{13} are reassembled into the stripe S_4 shown in FIG. 8. In the stripe S_4 shown in FIG. 8, data of the data strips D_{21}, D_{32}, and D_{13} does not need to be moved, and does not need to be migrated across storage nodes.

[0031] Further, the stripe server indicates a storage node on which a data strip that stores garbage data is distributed to perform garbage collection. In this embodiment of the present invention, the stripe server indicates data storage nodes that store data of the data strips D_{11}, D_{12}, D_{22}, D_{23}, D_{31}, and D_{33} to perform garbage collection. Further, parity data stored in P_{11}, P_{21}, and P_{31} is meaningless. Therefore, the stripe server further indicates the parity storage node that stores the parity data of the parity strips P_{11}, P_{21}, and P_{31} to perform garbage collection. After a storage node performs garbage collection, storage space occupied by garbage data on the storage node is released. In another implementation, a storage node performs garbage collection based on a status of stored data of a strip. For example, a data storage node checks a status of stored data of a data strip, and starts garbage collection when the data of the data strip is determined as garbage data. In a specific implementation, when storing the data of the data strip, the data storage node creates a mapping between a host access address of the data of the data strip and an identifier of the data strip. When the data of the data strip is garbage data, the mapping between the host access address of the data of the data strip and the identifier of the data strip is marked as invalid, so that the data of the data strip is determined as garbage data. Further, the data storage node may perform garbage collection in the data strip after the stripe server releases a stripe including the data strip. In another implementation, the data storage node may determine, based on an amount of garbage data in data strips stored in the data storage node, to start garbage collection. For example, a total quantity of data strips stored in the data storage node is 1000, and a quantity of data strips that stores garbage data is 600. If a threshold for starting garbage collection by the data storage node is 60%, to be specific, when a quantity of data strips that stores garbage data reaches 60%, the data storage node starts garbage collection. Therefore, each data storage node in a distributed system may independently perform garbage collection in a data strip. This embodiment of the present invention is also applicable to garbage collection by a parity storage node that stores parity strips. In this embodiment of the present invention, a stripe is reassembled with another stripe, data of a parity strip in the original stripe is garbage data, and garbage collection described in this embodiment of the present invention also needs to be performed. For a specific implementation, refer to garbage collection in a data strip. In this embodiment of the present invention, a plurality of storage nodes may independently perform garbage collection in a strip by stripe reassembling.

[0032] The stripe server releases the stripes S_1, S_2, and S_3. Specifically, that the stripe server releases the stripes S_1, S_2, and S_3 includes: The stripe server sets the stripes S_1, S_2, and S_3 to an idle state, so that new data may be subsequently allocated to the stripes S_1, S_2, and S_3.

[0033] According to stripe reassembling method provided in this embodiment of the present invention, valid data does not need to be migrated across storage nodes, thereby improving stripe reassembling performance of a storage system.

[0034] In this embodiment of the present invention, in the selected to-be-reassembled stripes, a maximum of one data strip includes valid data in data strips distributed on a same data storage node. A specific implementation may be that in the to-be-reassembled stripes, no data strip includes valid data in data strips distributed on a same data storage node. In this scenario, data strips in a reassembled stripe include an idle data strip, to be specific, no data strip includes valid data at a same location, and content of the data strips at the same location is empty in a new stripe. In a specific implementation of this embodiment of the present invention, two stripes may be further reassembled.

[0035] Based on the foregoing description, an embodiment of the present invention further provides a stripe server, used in a storage system in the embodiments of the present invention, for example, a distributed storage system. The storage system includes N data storage nodes that store data strips and M parity storage nodes that store parity strips. R stripes are distributed on the N+M storage nodes, each stripe S_i includes N data strips and M parity strips, a data strip D_{ix} is distributed on an x^th storage node in the N data storage nodes, and a parity strip P_{iy} is distributed on a y^th storage node in the M parity storage nodes. N, M, and R are positive integers, R is not less than 2, a value of i is an integer ranging from 1 to R, a value of x is an integer ranging from 1 to N, and a value of y is an integer ranging from 1 to M. As shown in FIG. 9, the stripe server includes a selection unit 901, a generation unit 902, and a storage unit 903. The selection unit 901 is configured to select the R stripes, where a maximum of one data strip includes valid data in data strips D_{ix} that are in the R stripes and that are distributed on a same data storage node. The generation unit 902 is configured to generate parity data of a parity strip P_{Ky} of a new stripe S_K for data of the data strip including valid data in the R stripes, where K is an integer different from 1 to R, and the new stripe S_K includes the data strip including valid data in the R stripes and the parity strip P_{Ky}. The storage unit 903 is configured to store the parity data of the parity strip P_{Ky} on the y^th storage node in the M parity storage nodes.

[0036] Further, the stripe server shown in FIG. 9 further includes an indication unit, configured to indicate a data storage node on which a data strip that stores garbage data of the R stripes is distributed to perform garbage collection. Further, the stripe server shown in FIG. 9 further includes a releasing unit, configured to release the R stripes. Further, the stripe server shown in FIG. 9 further includes a determining unit, configured to determine the stripe S_i, where a quantity of data strips including garbage data in the stripe S_i meets a reassembly threshold.

[0037] For an implementation of the stripe server shown in FIG. 9 in this embodiment of the present invention, refer to the foregoing described functions and structures of the stripe server in the embodiments of the present invention. Another implementation of the stripe server shown in FIG. 9 in this embodiment of the present invention may be implemented by software modules, or implemented by a combination of software and hardware.

[0038] Correspondingly, an embodiment of the present invention further provides a computer-readable storage medium and a computer program product. The computer-readable storage medium and the computer program product include a computer instruction, to implement various solutions described in the embodiments of the present invention.

[0039] In the embodiments of the present invention, identifiers used to describe stripes, data strips, parity strips, and storage nodes are merely used to describe the embodiments of the present invention more clearly. In actual product implementation, similar identifiers are not necessarily required. Therefore, in the embodiments of the present invention, the identifiers used to describe the stripes, the data strips, the parity strips, and the storage nodes are not intended to limit the present invention.

[0040] In this embodiment of the present invention, that the stripe S_i includes N data strips and M parity strips, a data strip D_{ix} is distributed on an x^th data storage node in the N data storage nodes, and a parity strip P_{iy} is distributed on a y^th parity storage node in the M parity storage nodes, means that the x^th data storage node in the N data storage nodes provides a storage address for the data strip D_{ix}, data of the data strip D_{ix} is stored at the storage address provided by the x^th data storage node in the N data storage nodes, the y^th parity storage node in the M parity storage nodes provides a storage address for the parity strip P_{iy}, and parity data of the parity strip P_{iy} is stored at the storage address provided by the y^th parity storage node in the M parity storage nodes. Therefore, this is also referred to as that the stripe S_i is distributed on the N+M storage nodes.

[0041] In the several embodiments provided in the present invention, it should be understood that the disclosed apparatus and method may be implemented in other manners. For example, division into the units in the described apparatus embodiment is merely logical function division and another division may be used in actual implementation. For example, a plurality of units or components may be combined or integrated into another system, or some features may be ignored or not performed. In addition, the displayed or discussed mutual couplings or direct couplings or communication connections may be implemented through some interfaces. The indirect couplings or communication connections between the apparatuses or units may be implemented in electronic, mechanical, or other forms.

[0042] The units described as separate parts may or may not be physically separate, and parts displayed as units may or may not be physical units, may be located in one position, or may be distributed on a plurality of network units. Some or all of the units may be selected according to actual requirements to achieve the objectives of the solutions of the embodiments.

[0043] In addition, functional units in the embodiments of the present invention may be integrated into one processing unit, or each of the units may exist alone physically, or two or more units are integrated into one unit.

Claims

1. A stripe reassembling method in a storage system, wherein the storage system comprises N data storage nodes that store data strips and M parity storage nodes that store parity strips; R stripes are distributed on the N+M storage nodes, each stripe Si comprises N data strips and M parity strips, a data strip D_{ix} is distributed on an xth data storage node in the N data storage nodes, and a parity strip P_{iy} is distributed on a yth parity storage node in the M parity storage nodes; and N, M, and R are positive integers, R is not less than 2, a value of i is an integer ranging from 1 to R, a value of x is an integer ranging from 1 to N, and a value of y is an integer ranging from 1 to M; and
   the method comprises:

   selecting, by a stripe server, the R stripes, wherein a maximum of one data strip comprises valid data in data strips D_{ix} that are in the R stripes and that are distributed on a same data storage node;
   generating, by the stripe server, parity data of a parity strip P_{Ky} of a new stripe S_K for data of the data strip comprising valid data in the R stripes, wherein the new stripe S_K comprises the data strip comprising valid data in the R stripes and the parity strip P_{Ky}, and K is an integer different from 1 to R; and
   storing, by the stripe server, the parity data of the parity strip P_{Ky} on the yth parity storage node in the M parity storage nodes.

2. The method according to claim 1, wherein after the storing, by the stripe server, the parity data of the parity strip P_{Ky} on the yth parity storage node in the M parity storage nodes, the method further comprises:
   indicating, by the stripe server, a data storage node on which a data strip that stores garbage data of the R stripes is distributed to perform garbage collection.

3. The method according to claim 2, wherein after the storing, by the stripe server, the parity data of the parity strip P_{Ky} on the yth parity storage node in the M parity storage nodes, the method further comprises:
   releasing, by the stripe server, the R stripes.

4. The method according to claim 1, wherein before the selecting, by a stripe server, the R stripes, the method further comprises:

   determining, by the stripe server, the stripe S_i, wherein a quantity of data strips comprising garbage data in the stripe S_i meets a reassembly threshold.

5. A stripe server, wherein the stripe server is used in a storage system, and the storage system comprises N data storage nodes that store data strips and M parity storage nodes that store parity strips; R stripes are distributed on the N+M storage nodes, each stripe S_i comprises N data strips and M parity strips, a data strip D_{ix} is distributed on an xth storage node in the N data storage nodes, and a parity strip P_{iy} is distributed on a yth storage node in the M parity storage nodes; N, M, and R are positive integers, R is not less than 2, a value of i is an integer ranging from 1 to R, a value of x is an integer ranging from 1 to N, and a value of y is an integer ranging from 1 to M; and the stripe server comprises an interface and a processor, the interface communicates with the processor, and the processor is configured to:

   select the R stripes, wherein a maximum of one data strip comprises valid data in data strips D_{ix} that are in the R stripes and that are distributed on a same data storage node;
   generate parity data of a parity strip P_{Ky} of a new stripe S_K for data of the data strip comprising valid data in the R stripes, wherein the new stripe S_K comprises the data strip comprising valid data in the R stripes and the parity strip P_{Ky}, and K is an integer different from 1 to R; and
   store the parity data of the parity strip P_{Ky} on the yth storage node in the M parity storage nodes.

6. The stripe server according to claim 5, wherein the processor is further configured to indicate a data storage node on which a data strip that stores garbage data of the R stripes is distributed to perform garbage collection.

7. The stripe server according to claim 6, wherein the processor is further configured to release the R stripes.

8. The stripe server according to claim 5, wherein the processor is further configured to determine the stripe S_i, wherein a quantity of data strips comprising garbage data in the stripe Si meets a reassembly threshold.

9. A stripe server, wherein the stripe server is used in a storage system, and the storage system comprises N data storage nodes that store data strips and M parity storage nodes that store parity strips; R stripes

are distributed on the N+M storage nodes, each stripe S_i comprises N data strips and M parity strips, a data strip D_{ix} is distributed on an x^{th} storage node in the N data storage nodes, and a parity strip P_{iy} is distributed on a y^{th} storage node in the M parity storage nodes; N, M, and R are positive integers, R is not less than 2, a value of i is an integer ranging from 1 to R, a value of x is an integer ranging from 1 to N, and a value of y is an integer ranging from 1 to M; and the stripe server comprises a selection unit, a generation unit, and a storage unit;

the selection unit is configured to select the R stripes, wherein a maximum of one data strip comprises valid data in data strips D_{ix} that are in the R stripes and that are distributed on a same data storage node;
the generation unit is configured to generate parity data of a parity strip P_{Ky} of a new stripe S_K for data of the data strip comprising valid data in the R stripes, wherein the new stripe S_K comprises the data strip comprising valid data in the R stripes and the parity strip P_{Ky}, and K is an integer different from 1 to R; and
the storage unit is configured to store the parity data of the parity strip P_{Ky} on the y^{th} storage node in the M parity storage nodes.

10. The stripe server according to claim 9, wherein the stripe server further comprises an indication unit, configured to indicate a data storage node on which a data strip that stores garbage data of the R stripes is distributed to perform garbage collection.

11. The stripe server according to claim 10, wherein the stripe server further comprises a releasing unit, configured to release the R stripes.

12. The stripe server according to claim 9, wherein the stripe server further comprises a determining unit, configured to determine the stripe S_i, wherein a quantity of data strips comprising garbage data in the stripe Si meets a reassembly threshold.

13. A computer program product, wherein the computer program product comprises a computer instruction, the computer program product is used in a storage system, the storage system comprises N data storage nodes that store data strips and M parity storage nodes that store parity strips; R stripes are distributed on the N+M storage nodes, each stripe S_i comprises N data strips and M parity strips, a data strip D_{ix} is distributed on an x^{th} storage node in the N data storage nodes, and a parity strip P_{iy} is distributed on a y^{th} storage node in the M parity storage nodes; N, M, and R are positive integers, R is not less than 2, a value of i is an integer ranging from 1 to R, a value of x is an integer ranging from 1 to N, and a value of y is an integer ranging from 1 to M; and a stripe server that is used in the storage system executes the computer instruction, to perform the following operations:

selecting the R stripes, wherein a maximum of one data strip comprises valid data in data strips D_{ix} that are in the R stripes and that are distributed on a same data storage node;
generating parity data of a parity strip P_{Ky} of a new stripe S_K for data of the data strip comprising valid data in the R stripes, wherein the new stripe S_K comprises the data strip comprising valid data in the R stripes and the parity strip P_{Ky}, and K is an integer different from 1 to R; and
storing the parity data of the parity strip P_{Ky} on the y^{th} storage node in the M parity storage nodes.

14. The computer program product according to claim 13, wherein the stripe server executes the computer instruction, to further indicate a data storage node on which a data strip that stores garbage data of the R stripes is distributed to perform garbage collection.

15. The computer program product according to claim 14, wherein the stripe server executes the computer instruction, to further release the R stripes.

16. The computer program product according to claim 13, wherein the stripe server executes the computer instruction, to further determine the stripe S_i, wherein a quantity of data strips comprising garbage data in the stripe S_i meets a reassembly threshold.

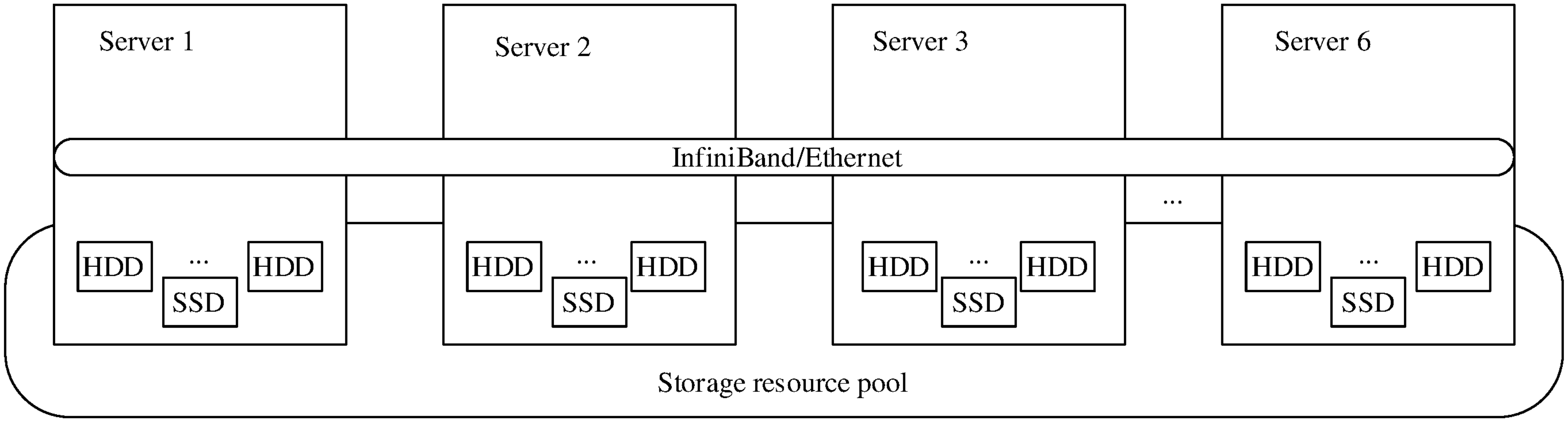

Server 1
Server 2
Server 3
Server 6
InfiniBand/Ethernet
...
HDD
...
HDD
SSD
HDD
...
HDD
SSD
HDD
...
HDD
SSD
HDD
...
HDD
SSD
Storage resource pool

FIG. 1

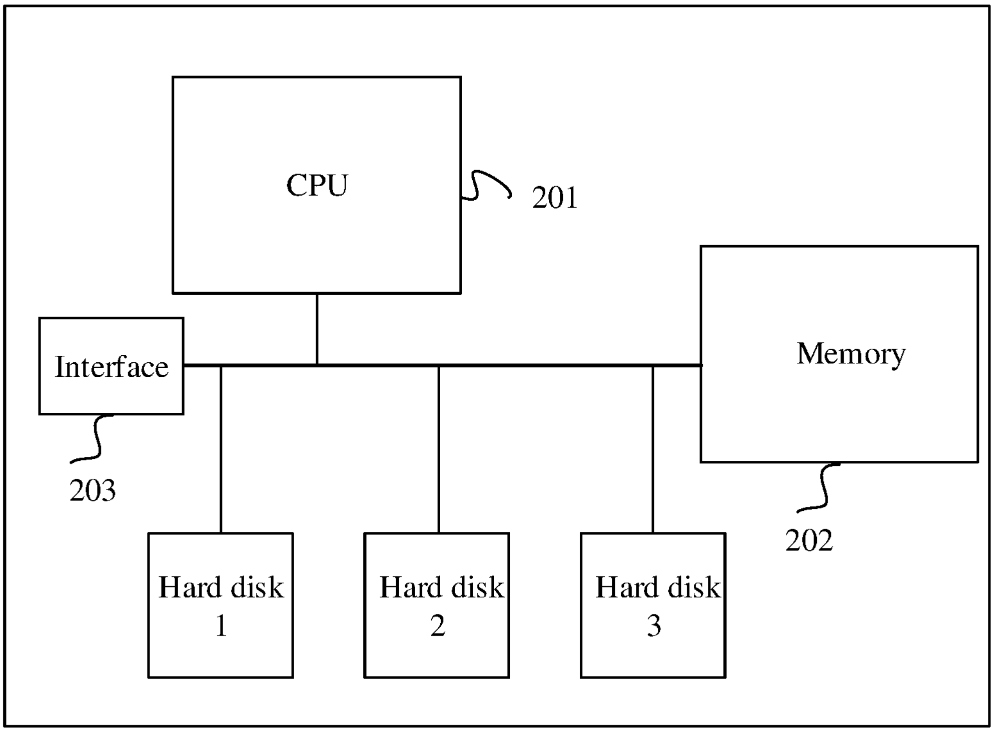

CPU
201
Interface
203
Memory
202
Hard disk 1
Hard disk 2
Hard disk 3

FIG. 2

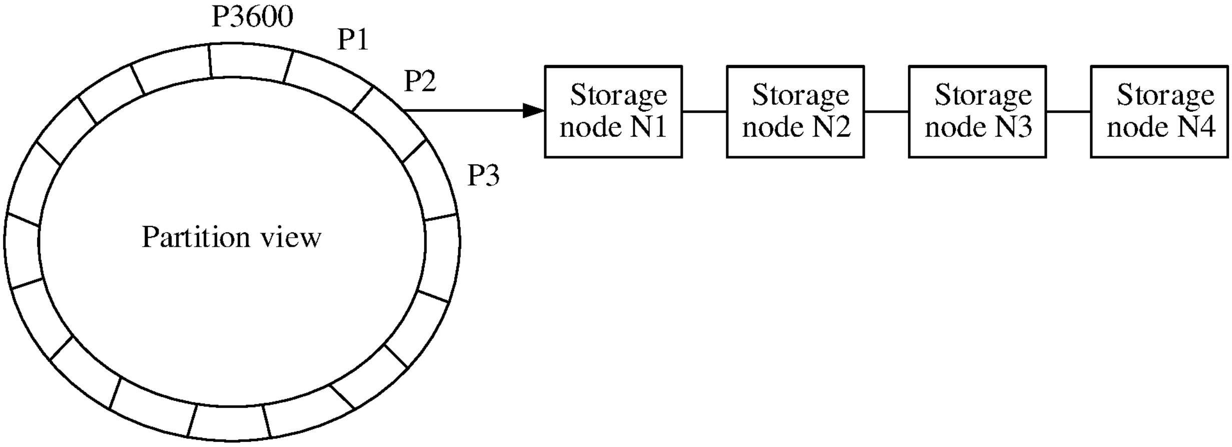

P3600
P1
P2
P3
Partition view
Storage node N1
Storage node N2
Storage node N3
Storage node N4

FIG. 3

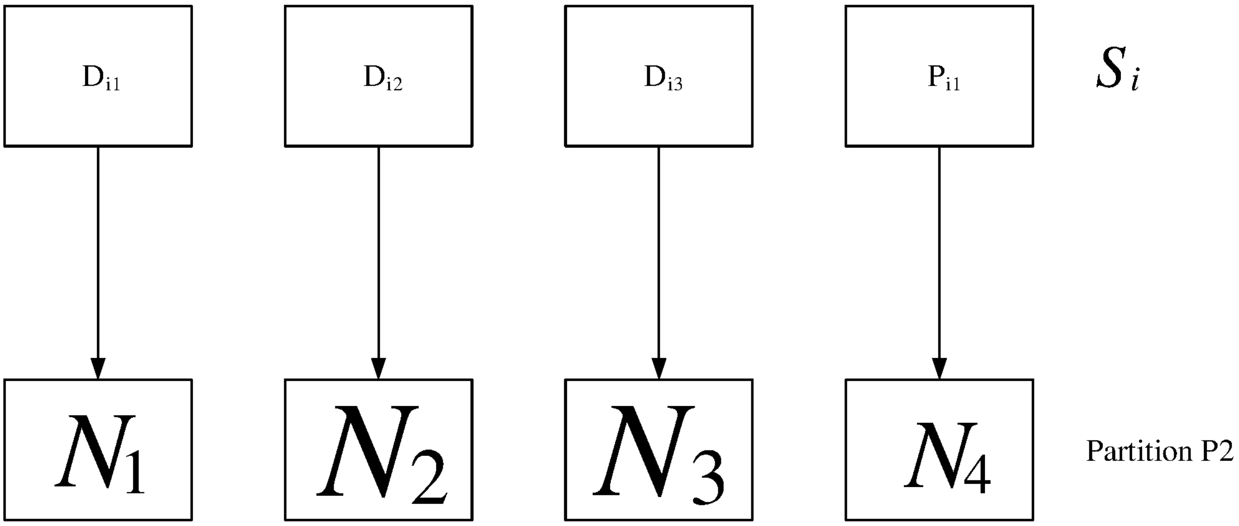

D_{i1}
D_{i2}
D_{i3}
P_{i1}
S_i
N_1
N_2
N_3
N_4
Partition P2

FIG. 4

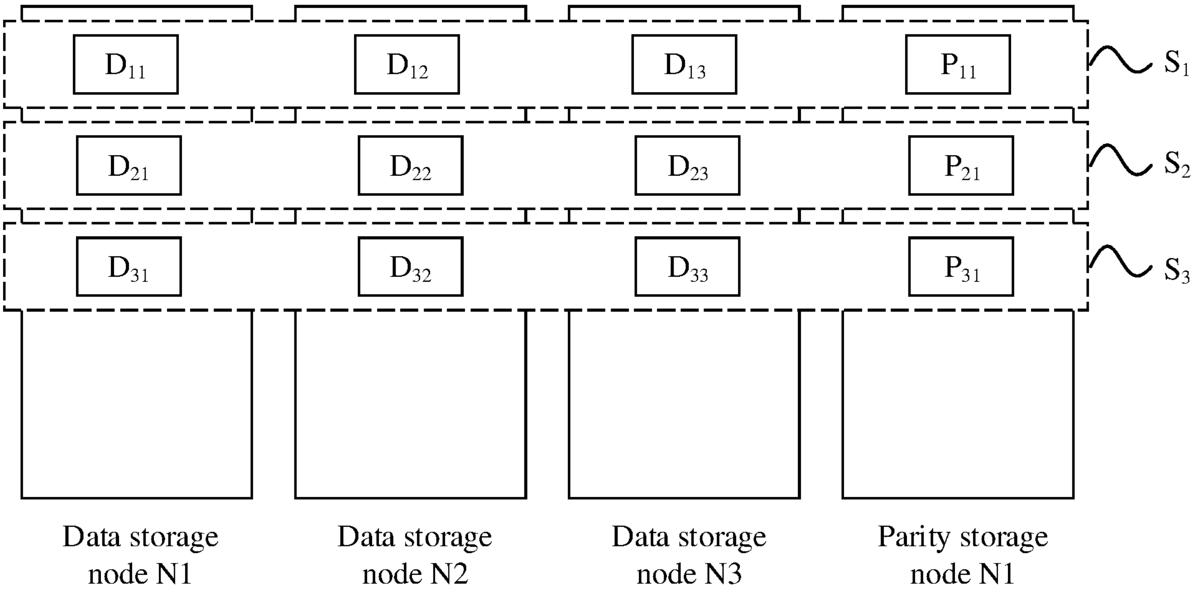

D_{11}
D_{12}
D_{13}
P_{11}
S_1
D_{21}
D_{22}
D_{23}
P_{21}
S_2
D_{31}
D_{32}
D_{33}
P_{31}
S_3
Data storage node N1
Data storage node N2
Data storage node N3
Parity storage node N1

FIG. 5

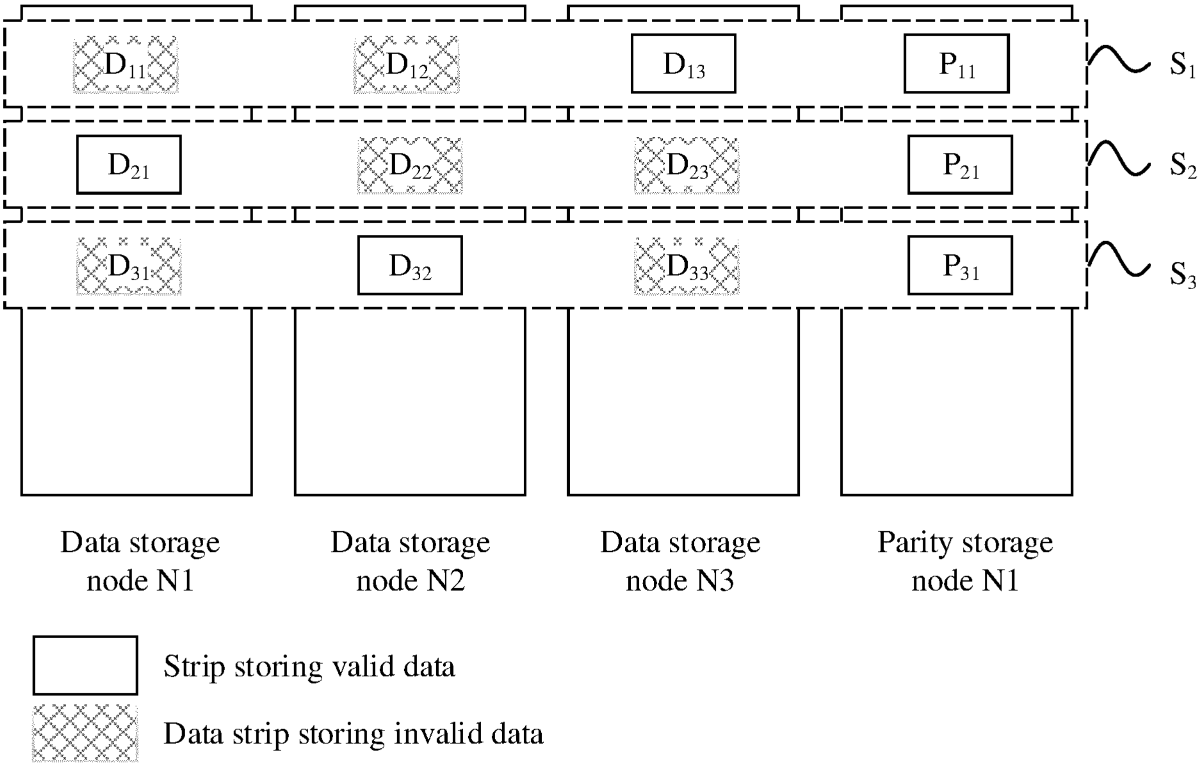
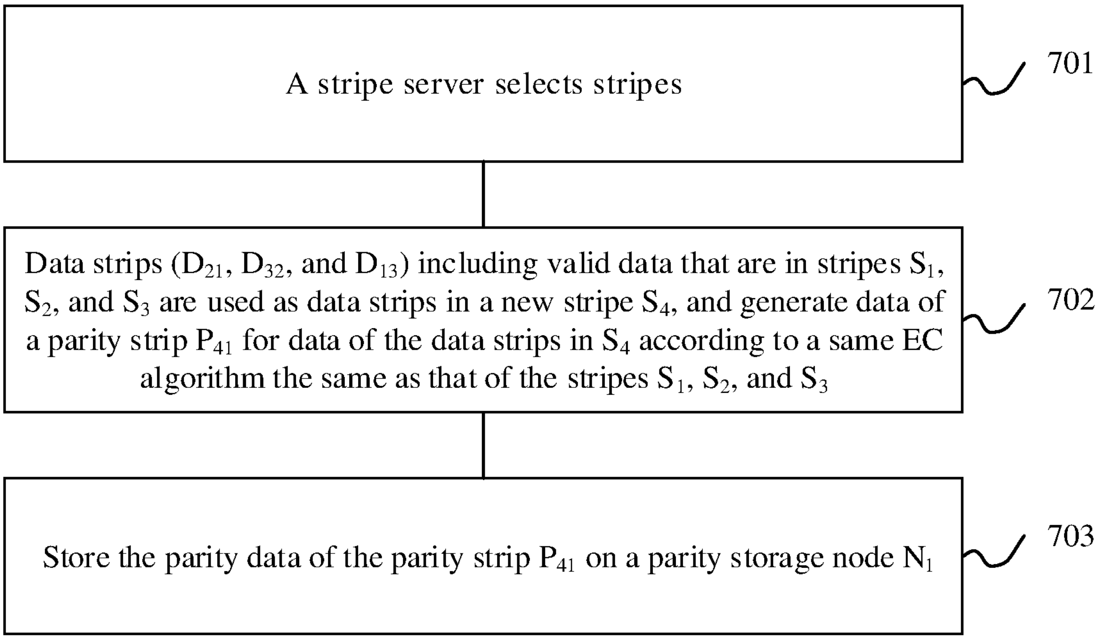


FIG. 6



701 A stripe server selects stripes

702 Data strips (D_{21}, D_{32}, and D_{13}) including valid data that are in stripes S_1, S_2, and S_3 are used as data strips in a new stripe S_4, and generate data of a parity strip P_{41} for data of the data strips in S_4 according to a same EC algorithm the same as that of the stripes S_1, S_2, and S_3

703 Store the parity data of the parity strip P_{41} on a parity storage node N_1

FIG. 7

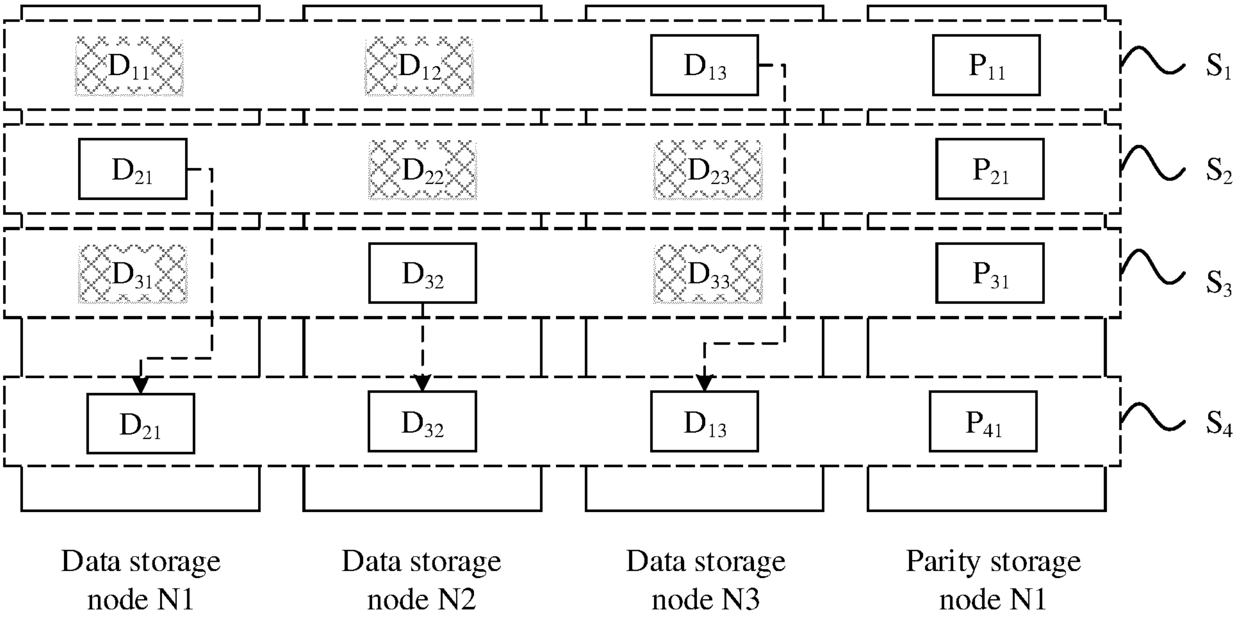

D11
D12
D13
P11
S1
D21
D22
D23
P21
S2
D31
D32
D33
P31
S3
D21
D32
D13
P41
S4
Data storage node N1
Data storage node N2
Data storage node N3
Parity storage node N1

FIG. 8

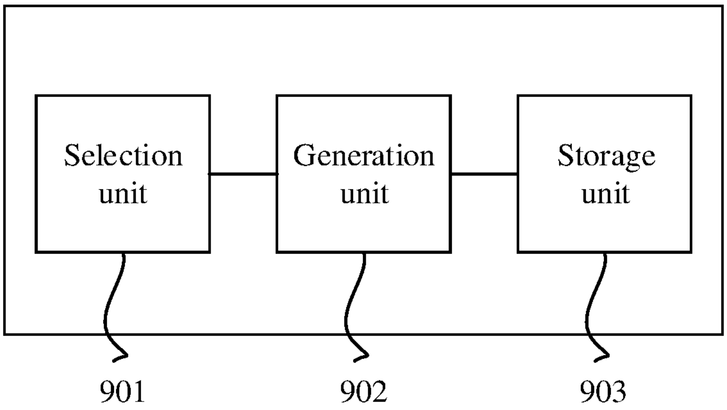

Selection unit
Generation unit
Storage unit
901
902
903

FIG. 9

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CN2019/103281

A. CLASSIFICATION OF SUBJECT MATTER

G06F 12/00(2006.01)i

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

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)

WPI; EPODOC; CNPAT; CNKI; IEEE: 分布式, 存储, 分条, 条带, 重组, 合并, 校验, 垃圾, 回收, 数据搬移, distribution, storage, stripe, block, unit, strip, parity, garbage, collection, data migration

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
PX	CN 109814805 A (HUAWEI TECHNOLOGIES CO., LTD.) 28 May 2019 (2019-05-28) claims 1-16	1-16
A	WO 2017173623 A1 (HUAWEI TECHNOLOGIES CO., LTD.) 12 October 2017 (2017-10-12) description, pp. 7, 8, and 13, and figures 1 and 2	1-16
A	CN 103902465 A (HUAWEI TECHNOLOGIES CO., LTD.) 02 July 2014 (2014-07-02) entire document	1-16
A	US 2015205670 A1 (DSSD, INC.) 23 July 2015 (2015-07-23) entire document	1-16

□ 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 **01 November 2019**	Date of mailing of the international search report **28 November 2019**
Name and mailing address of the ISA/CN **China National Intellectual Property Administration (ISA/ CN)** **No. 6, Xitucheng Road, Jimenqiao Haidian District, Beijing 100088** **China** Facsimile No. **(86-10)62019451**	Authorized officer Telephone No.

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2019/103281

Patent document cited in search report	Publication date (day/month/year)	Patent family member(s)	Publication date (day/month/year)
CN 109814805 A	28 May 2019	None	
WO 2017173623 A1	12 October 2017	US 2018239671 A1	23 August 2018
		EP 3336706 A1	20 June 2018
		CN 107484427 A	15 December 2017
CN 103902465 A	02 July 2014	CN 106681934 A	17 May 2017
US 2015205670 A1	23 July 2015	EP 2899627 A1	29 July 2015
		US 2016217038 A1	28 July 2016
		US 8949692 B1	03 February 2015
		CN 104809032 A	29 July 2015