



- (51) **International Patent Classification:**
G06F 12/02 (2006.01) *G06F 12/08* (2006.01)
- (21) **International Application Number:**
PCT/US2014/039578
- (22) **International Filing Date:**
27 May 2014 (27.05.2014)
- (25) **Filing Language:** English
- (26) **Publication Language:** English
- (71) **Applicant:** HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P. [US/US]; 11445 Compaq Center Drive W., Houston, Texas 77070 (US).
- (72) **Inventor:** NAZARI, Siamak; 4245 Technology Drive, Fremont, California 94538 (US).
- (74) **Agents:** HA, Miranda J. et al.; Hewlett-Packard Company, Intellectual Property Administration, 3404 E. Harmony Road, Mail Stop 35, Fort Collins, Colorado 80528 (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, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT,

HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, 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, 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).

Declarations under Rule 4.17:

- *as to the identity of the inventor (Rule 4.17(i))*
- *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

Published:

- *with international search report (Art. 21(3))*

(54) **Title:** STORING COMPRESSED DATA

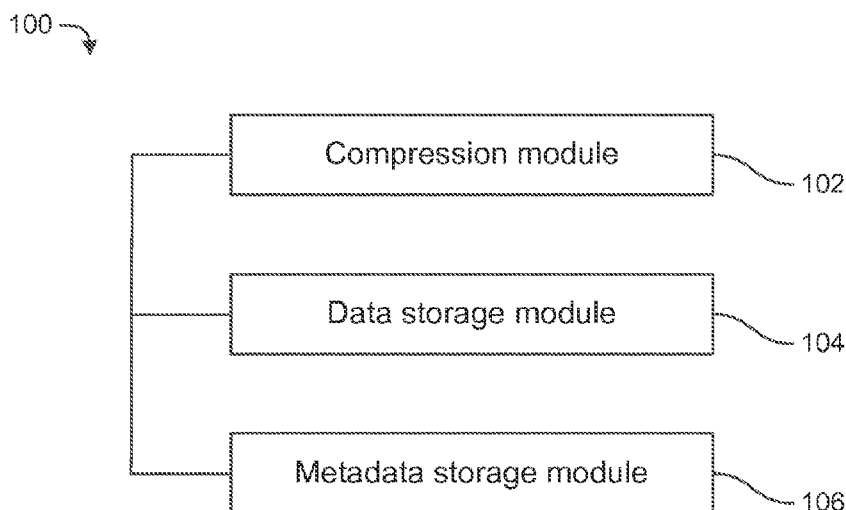


FIG. 1

(57) **Abstract:** Example implementations relate to storing compressed data. In example implementations, a plurality of data blocks may be compressed into a data unit. The size of the data unit may be less than or equal to a fixed size. The data unit may be stored in a physical location.

- 1 -

STORING COMPRESSED DATA

BACKGROUND

[0001] Storage servers may be used to store data. Data may be compressed before being stored. When stored data is retrieved, the retrieved data may be decompressed.

BRIEF DESCRIPTION OF THE DRAWINGS

[0002] The following detailed description references the drawings, wherein:

[0003] FIG. 1 is an example system for storing compressed data;

[0004] FIG. 2 is an example system for tracking and locating stored data;

[0005] FIG. 3 is a block diagram of an example system that includes a machine-readable storage medium encoded with instructions to enable storing of compressed data;

[0006] FIG. 4 is a block diagram of an example system that includes a machine-readable storage medium encoded with instructions to store pointers;

[0007] FIG. 5 is a block diagram of an example system that includes a machine-readable storage medium encoded with instructions to enable locating a desired data block;

[0008] FIG. 6 is a flowchart of an example method for storing compressed data;

[0009] FIG. 7 is a flowchart of an example method for maintaining pointers;

[0010] FIG. 8 is a flowchart of an example method for locating a desired data block; and

[0011] FIG. 9 is a flowchart of an example method for making a physical location available for data storage.

DETAILED DESCRIPTION

[0012] To save storage space, data may be compressed before being stored in a storage server. In some storage servers, each piece of data that is received may be individually compressed, resulting in compressed data of various sizes. Tracking various sizes of compressed data may consume more resources than tracking compressed data of the same size, and may result in more fragmentation

- 2 -

in a storage server. The present disclosure provides for compressing multiple pieces of data together, and for a uniform maximum size of compressed data that is stored, reducing fragmentation and simplifying management of stored compressed data.

[0013] Referring now to the drawings, FIG. 1 is an example system 100 for storing compressed data. In some implementations, system 100 may be implemented in a storage server or other storage system. As used herein, the term "storage system" should be understood to refer to a system capable of receiving data, storing data, and retrieving previously stored data. An entity that sends data to be stored in a storage system, and/or requests previously stored data from a storage system, may be referred to herein as a "host".

[0014] In FIG. 1, system 100 includes compression module 102, data storage module 104, and metadata storage module 106. As used herein, the terms "include", "have", and "comprise" are interchangeable and should be understood to have the same meaning. A module may include a set of instructions encoded on a machine-readable storage medium and executable by a processor. In addition or as an alternative, a module may include a hardware device comprising electronic circuitry for implementing the functionality described below.

[0015] Compression module 102 may compress data blocks into data units. As used herein, the term "compress" should be understood to refer to transforming data such that the data may be stored using fewer bits than were in the data's original form. Compression methods may exploit statistical redundancy to represent data more concisely without losing information, allowing the compression process to be reversed. The act of reversing the compression process to obtain data in its original form may be referred to herein as "decompressing". As used herein, the term "data block" should be understood to refer to a quantity of data that is handled together in communications between a host and a storage system. For example, a host may send a request to a storage server to store a data block at a particular address (e.g., write to an address), or to retrieve a data block from a specified address (e.g., read from an address). In some implementations, a host may send a request to a storage server to store or retrieve multiple data blocks in a specified range of addresses (each data block being stored in/retrieved from a respective addresses).

- 3 -

[0016] A data unit may include a plurality of compressed data blocks. A respective size of each data unit may be less than or equal to a fixed size. For example, a group of data blocks that are each 512 bytes in size may be compressed together into a data unit that is equal to or less than 16 kB in size. If, after being compressed, a group of data blocks takes up more storage space than the fixed size, the compressed data may be split between multiple data units. Continuing with the example of the fixed size being 16 kB, if a group of data blocks, after being compressed together, is 20 kB in size, 16 kB of the compressed data may be in a first data unit, and the remaining 4 kB of compressed data may be in a second data unit. It should be understood that the first and second data units may each include less than 16 kB of compressed data; for example, 15 kB of the compressed data may be in a first data unit, and the remaining 5 kB may be in a second data unit. Each data unit may include compressed data from more than one data block.

[0017] Data storage module 104 may store each data unit in a respective physical location. A physical location may be a storage space in a storage device, such as a hard-disk drive (HDD), solid-state drive (SSD), tape drive, or any other suitable data storage device. Capping the size of each data unit at the fixed size may lead to more than one physical location being used to store a group of data blocks that are compressed together. In the above example, the first data unit may be stored in a first physical location, and the second data unit may be stored in a second physical location. The first and second physical locations may or may not be physically adjacent to one another.

[0018] Metadata storage module 106 may track data blocks in the physical locations. For example, metadata storage module 106 may track which physical locations data blocks are stored in once compressed. In some implementations, metadata storage module 106 may store position indicators. Position indicators may indicate positions of data blocks in decompressed data segments obtained by decompressing data units. Position indicators are discussed further below with respect to FIG. 2.

[0019] An addressing scheme used by a host in instructing a storage system to read/write data may be different from an addressing scheme used by the storage system for physical locations. For example, a host may send a plurality of data blocks

- 4 -

to a storage system and specify a block address (e.g., a logical block address, or LBA) at which each of the plurality of data blocks are to be stored, or otherwise associate data blocks with respective block addresses. The storage system may compress the plurality of data blocks together into a data unit that is stored in a single physical location. The host may later send a read request to the storage system to retrieve certain data blocks of the plurality of data blocks, and the read request may include the respective block addresses associated with the desired data blocks. Although the read request from the host may include multiple block addresses, the storage system may read a single physical location since the desired data blocks have been stored in the same physical location. The compression of data blocks and storage of multiple (compressed) data blocks in a single physical location may be transparent to the host.

[0020] In light of the above, in some implementations, metadata storage module 106 may store pointers. Each pointer may correspond to a respective block address. Each block address may be associated with a respective data block. For example, a write request from a host may associate a block address with a data block. Each pointer may include a mapping of the respective block address to a physical location in which a data unit, into which the respective data block is compressed, is stored. When a storage system receives a read request that includes a block address, the pointer corresponding to the block address may be used to determine which physical location in the storage system to access to retrieve the desired data block. Pointers and position indicators may be referred to as "metadata".

[0021] FIG. 2 is an example system 200 for tracking and locating stored data. In some implementations, system 200 may be implemented in a storage server or other storage system. In FIG. 2, system 200 includes compression module 202, data storage module 204, metadata storage module 206, counter module 208, and locator module 210. A module may include a set of instructions encoded on a machine-readable storage medium and executable by a processor. In addition or as an alternative, a module may include a hardware device comprising electronic circuitry for implementing the functionality described below.

[0022] Modules 202, 204, and 206 of FIG. 2 may be analogous to (e.g., have functions and/or components similar to) modules 102, 104, and 106, respectively, of

- 5 -

FIG. 1. Counter module 208 may maintain a plurality of reference counters. Each of the plurality of reference counters may correspond to a respective physical location. Each of the plurality of reference counters may indicate how many pointers comprise a mapping to the respective physical location.

[0023] Maintaining reference counters may include updating reference counters as write requests are received. For example, a host may send a write request to a storage system (e.g., system 200). The write request may include a first plurality of data blocks to be stored, and may associate each of the first plurality of data blocks with a respective one of a first plurality of block addresses. Compression module 202 may compress the first plurality of data blocks into a first data unit, and data storage module 204 may store the first data unit in a first physical location. Metadata storage module 206 may store a first plurality of pointers, corresponding to respective ones of the first plurality of block addresses, that each include a mapping of a respective block address to the first physical location. Counter module 208 may set a reference counter, that corresponds to the first physical location, to a value equal to the number of pointers in the first plurality of pointers (which may be equal to the number of block addresses in the first plurality of block addresses). Later, the host may send another write request that includes a second plurality of data blocks, and that associates each of the second plurality of data blocks with a respective one of a second plurality of block addresses. The second plurality of block addresses may include one of the first plurality of block addresses (e.g., a subsequent write request may modify a data block associated with a certain block address). Compression module 202 may compress the second plurality of data blocks into a second data unit, and data storage module 204 may store the second data unit in a second physical location different from the first physical location. Metadata storage module 206 may store a second plurality of pointers, corresponding to respective ones of the second plurality of block addresses, that each include a mapping of a respective block address to the second physical location. A pointer corresponding to the block address included in the first plurality of block addresses and the second plurality of block addresses may be updated to include a mapping to the second physical location rather than to the first physical location. Counter module 208 may decrement the reference counter corresponding to the first physical location since

- 6 -

there is now one less pointer that includes a mapping to the first physical location. If the second plurality of block addresses includes multiple ones of the first plurality of block addresses, the counter module 208 may decrement the reference counter corresponding to the first physical location by the number of overlapping block addresses, as that may be the number of pointers in the first plurality of pointers that are updated to include a mapping to the second physical location rather than to the first physical location. Counter module 208 may also set a reference counter, that corresponds to the second physical location, to a value equal to the number of pointers in the second plurality of pointers (which may be equal to the number of block addresses in the second plurality of block addresses). Pointers and reference counters may be stored/modified in a similar manner in response to subsequent write requests.

[0024] Locator module 210 may locate, based on a position indicator, a data block, that is associated with a block address specified in a read request, in a decompressed data segment obtained by decompressing a data unit. For example, in response to a write request that includes a plurality of data blocks each associated with a respective one of a plurality of block addresses, the plurality of data blocks may be compressed (e.g., by compression module 202) into a data unit and the data unit stored (e.g., by data storage module 204) in a physical location, as discussed above. Metadata storage module may store a plurality of position indicators, each position indicator corresponding to one of the plurality of data blocks. When system 200 receives a read request that includes one of the plurality of block addresses, which may be associated with a data block desired by a host, the pointer corresponding to the one of the plurality of block addresses may be used to identify the appropriate physical location. The data unit stored in the physical location may be decompressed to obtain a decompressed data segment. The position indicator corresponding to the desired data block may be used to locate the desired data block in the decompressed data segment. For example, the position indicator may include an offset from a known point in the decompressed data segment (e.g., beginning or end of the decompressed data segment). Locator module 210 may use the offset to determine where in the decompressed data segment the desired data block is.

- 7 -

[0025] FIG. 3 is a block diagram of an example system 300 that includes a machine-readable storage medium encoded with instructions to enable storing of compressed data. In some implementations, system 300 may be implemented in a storage system (e.g., storage server). In FIG. 3, system 300 includes processor 302 and machine-readable storage medium 304.

[0026] Processor 302 may include a central processing unit (CPU), microprocessor (e.g., semiconductor-based microprocessor), and/or other hardware device suitable for retrieval and/or execution of instructions stored in machine-readable storage medium 304. Processor 302 may fetch, decode, and/or execute instructions 306, 308, and 310 to enable storing of compressed data, as described below. As an alternative or in addition to retrieving and/or executing instructions, processor 302 may include an electronic circuit comprising a number of electronic components for performing the functionality of instructions 306, 308, and/or 310.

[0027] Machine-readable storage medium 304 may be any suitable electronic, magnetic, optical, or other physical storage device that contains or stores executable instructions. Thus, machine-readable storage medium 304 may include, for example, a RAM, an Electrically Erasable Programmable Read-Only Memory (EEPROM), a storage device, an optical disc, and the like. In some implementations, machine-readable storage medium 304 may include a non-transitory storage medium, where the term "non-transitory" does not encompass transitory propagating signals. As described in detail below, machine-readable storage medium 304 may be encoded with a set of executable instructions 306, 308, and 310.

[0028] Instructions 306 may compress a plurality of data blocks into a data unit. The plurality of data blocks may accompany or be part of a write request from a host. In some implementations, the plurality of data blocks may be aggregated from multiple write requests. The plurality of data blocks may be compressed together until the size of the resulting data unit is less than a fixed size. In some cases, the plurality of data blocks may be compressed into multiple data units, as discussed above with respect to FIG. 1.

[0029] Instructions 308 may store a data unit in one of a plurality of physical locations. Data units stored in the plurality of physical locations may have respective

- 8 -

sizes that are each less than or equal to a fixed size. An addressing scheme used by system 300 for physical locations may be different from an addressing scheme used by a host in instructing system 300 to read/write data, as discussed above with respect to FIG. 1.

[0030] Instructions 310 may store a plurality of position indicators. Each of the position indicators may indicate a respective position, of a respective one of a plurality of data blocks, in a decompressed data segment obtained by decompressing a data unit. In some implementations, the plurality of position indicators may be stored with other metadata in volatile memory to facilitate quick access to the metadata. A copy of the metadata may be stored in non-volatile memory so that the metadata may be recovered if power to system 300 is lost.

[0031] FIG. 4 is a block diagram of an example system 400 that includes a machine-readable storage medium encoded with instructions to store pointers. In some implementations, system 400 may be implemented in a storage system (e.g., storage server). In FIG. 4, system 400 includes processor 402 and machine-readable storage medium 404.

[0032] As with processor 302 of FIG. 3, processor 402 may include a CPU, microprocessor (e.g., semiconductor-based microprocessor), and/or other hardware device suitable for retrieval and/or execution of instructions stored in machine-readable storage medium 404. Processor 402 may fetch, decode, and/or execute instructions 406, 408, 410, 412, 414, and 416. As an alternative or in addition to retrieving and/or executing instructions, processor 402 may include an electronic circuit comprising a number of electronic components for performing the functionality of instructions 406, 408, 410, 412, 414, and/or 416.

[0033] As with machine-readable storage medium 304 of FIG. 3, machine-readable storage medium 404 may be any suitable physical storage device that stores executable instructions. Instructions 406, 408, and 410 on machine-readable storage medium 404 may be analogous to instructions 306, 308, and 310, respectively, on machine-readable storage medium 304. Instructions 412 may store a plurality of pointers. Each of the plurality of pointers may correspond to a respective one of a plurality of block addresses. Each of the plurality of block addresses may be associated with a respective one of a plurality of data blocks. For example, a write

request from a host may associate a block address with a data block. Each of the plurality of pointers may comprise a mapping of the respective one of the plurality of block addresses to one of a plurality of physical locations. The physical location to which a block address is mapped may be the physical location in which a data unit, into which the respective data block is compressed, is stored.

[0034] Instructions 414 may update one of the plurality of pointers in response to a write request that comprises one of the plurality of block addresses. Updating a pointer may include changing a mapping of the corresponding block address from a first physical location to a second physical location, as discussed above with respect to FIG. 2. In some implementations, if a pointer is updated, a reference counter may be modified. For example, if a pointer is updated to include a mapping to a second physical location rather than to a first physical location, a reference counter corresponding to the first physical location may be decremented.

[0035] Instructions 416 may make, in response to a determination that no pointer comprises a mapping to one of the plurality of physical locations, the one of the plurality of physical locations available to store another data unit. Such a determination may mean that all data blocks compressed into the data unit stored in the one of the plurality of physical locations are no longer current (e.g., after a write command in response to which the data unit was stored, a host sent additional write commands that associated all the block addresses, that were associated with data blocks compressed into the data unit, with other data blocks). In some implementations, a determination that no pointer comprises a mapping to one of the plurality of physical locations may be made by identifying a reference counter whose value is zero. In some implementations, reference counters may not be used, and a garbage collection mechanism may periodically analyze stored pointers to identify which physical locations are mapped to.

[0036] FIG. 5 is a block diagram of an example system 500 that includes a machine-readable storage medium encoded with instructions to enable locating a desired data block. In some implementations, system 500 may be implemented in a storage system (e.g., storage server). In FIG. 5, system 500 includes processor 502 and machine-readable storage medium 504.

- 10 -

[0037] As with processor 302 of FIG. 3, processor 502 may include a CPU, microprocessor (e.g., semiconductor-based microprocessor), and/or other hardware device suitable for retrieval and/or execution of instructions stored in machine-readable storage medium 504. Processor 502 may fetch, decode, and/or execute instructions 506, 508, 510, 512, 514, 516, 518, and 520 to enable locating a desired data block, as described below. As an alternative or in addition to retrieving and/or executing instructions, processor 502 may include an electronic circuit comprising a number of electronic components for performing the functionality of instructions 506, 508, 510, 512, 514, 516, 518, and/or 520.

[0038] As with machine-readable storage medium 304 of FIG. 3, machine-readable storage medium 504 may be any suitable physical storage device that stores executable instructions. Instructions 506, 508, and 510 on machine-readable storage medium 504 may be analogous to instructions 306, 308, and 310, respectively, on machine-readable storage medium 304. Instructions 512 on machine-readable storage medium 504 may be analogous to instructions 412 on machine-readable storage medium 404. Instructions 514 may receive a read request. The read request may include one of a plurality of block addresses.

[0039] Instructions 516 may identify, based on one of a plurality of pointers, one of a plurality of physical locations that maps to the one of the plurality of block addresses. For example, the read request may be parsed to identify the one of the plurality of block addresses, and the pointer corresponding to the one of the plurality of block addresses may be retrieved. The retrieved pointer may be analyzed to identify the physical location to which the one of the plurality of block addresses is mapped.

[0040] Instructions 518 may decompress a data unit, that is stored in one of a plurality of physical locations, to obtain a decompressed data segment. The data unit may be one into which a desired data block (e.g., a data block associated with a block address specified in a read request) is compressed. The one of the plurality of physical locations may be identified using a pointer, as discussed above.

[0041] Instructions 520 may locate, based on one of a plurality of position indicators, a data block in a decompressed data segment. The data block may be associated with a block address specified in a read request. In some

implementations, the one of the plurality of position indicators may include an offset from a known point in the decompressed data segment (e.g., beginning or end of the decompressed data segment).

[0042] Methods related to compressing, storing, and tracking data blocks are discussed with respect to FIGS. 6-9. FIG. 6 is a flowchart of an example method 600 for storing compressed data. Although execution of method 600 is described below with reference to processor 302 of FIG. 3, it should be understood that execution of method 600 may be performed by other suitable devices, such as processors 402 and 502 of FIGS. 4 and 5, respectively. Method 600 may be implemented in the form of executable instructions stored on a machine-readable storage medium and/or in the form of electronic circuitry.

[0043] Method 600 may start in block 602, where processor 302 may receive a plurality of data blocks. The plurality of data blocks may accompany or be part of a write request from a host. In some implementations, the plurality of data blocks may be aggregated from multiple write requests.

[0044] In block 604, processor 302 may compress the plurality of data blocks into a data unit. The plurality of data blocks may be compressed together until the size of the resulting data unit is less than or equal to a fixed size. In some cases, the plurality of data blocks may be compressed into multiple data units, as discussed above with respect to FIG. 1.

[0045] Next, in block 606, processor 302 may store the data unit in a physical location of a plurality of physical locations. Data units stored in the plurality of physical locations may have respective sizes that are each less than or equal to a fixed size. An addressing scheme used by a storage system (e.g., system 300) for physical locations may be different from an addressing scheme used by a host in instructing the storage system to read/write data, as discussed above with respect to FIG. 1.

[0046] Finally, in block 608, processor 302 may store a set of position indicators. Each of the set of position indicators may indicate a respective position, of a respective one of the plurality of data blocks, in a decompressed data segment obtained by decompressing the data unit. In some implementations, the set of position indicators may be stored with other metadata in volatile memory, and a copy

- 12 -

of the metadata may be stored in non-volatile memory, as discussed above with respect to FIG. 3.

[0047] FIG. 7 is a flowchart of an example method 700 for maintaining pointers. Although execution of method 700 is described below with reference to processor 402 of FIG. 4, it should be understood that execution of method 700 may be performed by other suitable devices, such as processors 302 and 502 of FIGS. 3 and 5, respectively. Some blocks of method 700 may be performed in parallel with and/or after method 600. Method 700 may be implemented in the form of executable instructions stored on a machine-readable storage medium and/or in the form of electronic circuitry.

[0048] Method 700 may start in block 702, where processor 402 may receive a first write request (e.g., from a host). The first write request may include a first plurality of data blocks to be stored, and may associate each of the first plurality of data blocks with a respective one of a first plurality of block addresses. The first plurality of data blocks may be compressed into a first data unit. The size of the first data unit may be less than or equal to a fixed size.

[0049] In block 704, processor 402 may store the first data unit in a first physical location of a plurality of physical locations. In block 706, processor 402 may store a first set of position indicators. Each of the first set of position indicators may indicate a respective position, of a respective one of the first plurality of data blocks, in a first decompressed data segment obtained by decompressing the first data unit.

[0050] In block 708, processor 402 may store a plurality of pointers. Each of the plurality of pointers may correspond to a respective one of a first plurality of block addresses. Each of the first plurality of block addresses may be associated with a respective one of the first plurality of data blocks. Each of the plurality of pointers may comprise a mapping of the respective one of the first plurality of block addresses to the first physical location. Although block 708 is shown below block 706 in FIG. 7, it should be understood that the plurality of pointers may be stored before or in parallel with the first set of position indicators.

[0051] In block 710, processor 402 may receive a subsequent write request that includes a second plurality of data blocks and associates one of the second plurality of data blocks with one of the first plurality of block addresses. It should be

- 13 -

understood that the subsequent write request may associate multiple ones of the second plurality of data blocks with respective multiple ones of the first plurality of block addresses. The second plurality of data blocks may be compressed into a second data unit. The size of the second data unit may be less than or equal to a fixed size.

[0052] In block 712, processor 402 may store the second data unit, into which the second plurality of data blocks is compressed, in a second physical location of the plurality of physical locations. In block 714, processor 402 may update a pointer, of the plurality of pointers, that corresponds to the one of the first plurality of block addresses (i.e., one of the first plurality of block addresses that the subsequent write request associates with one of the second plurality of data blocks). The updated pointer may comprise a mapping of the one of the first plurality of block addresses to the second physical location.

[0053] In block 716, processor 402 may store a second set of position indicators. Each of the second set of position indicators may indicate a respective position of a respective one of the second plurality of data blocks in a second decompressed data segment obtained by decompressing the second data unit. Although block 716 is shown below block 714 in FIG. 7, it should be understood that elements of block 716 may be executed before or in parallel with elements of block 714.

[0054] FIG. 8 is a flowchart of an example method 800 for locating a desired data block. Although execution of method 800 is described below with reference to processor 502 of FIG. 5, it should be understood that execution of method 800 may be performed by other suitable devices, such as processors 302 and 402 of FIGS. 3 and 4, respectively. Some blocks of method 800 may be performed in parallel with and/or after method 600 or 700. Method 800 may be implemented in the form of executable instructions stored on a machine-readable storage medium and/or in the form of electronic circuitry.

[0055] Method 800 may start in block 802, where processor 502 may receive a write request that includes a second plurality of data blocks and associates one of the second plurality of data blocks with one of a plurality of block addresses that was associated, by a previous write request, with one of a first plurality of data blocks. The

second plurality of data blocks may be compressed into a data unit. The size of the data unit may be less than or equal to a fixed size.

[0056] In block 804, processor 502 may store the data unit, into which the second plurality of data blocks is compressed, in a physical location of a plurality of physical locations. In block 806, processor 502 may store a set of position indicators. Each of the set of position indicators may indicate a respective position of a respective one of the second plurality of data blocks in a decompressed data segment obtained by decompressing the data unit.

[0057] In block 808, processor 502 may receive a read request that includes the one of the first plurality of block addresses (e.g., a block address that was associated, by a previous write request, with one of the first plurality of data blocks). Next, in block 810, processor 502 may decompress, in response to the read request, the data unit to obtain the decompressed data segment. The physical location in which the data unit is stored may be identified based on a pointer, as discussed above. Finally, in block 812, processor 502 may locate, in the decompressed data segment, and based on the set of position indicators, the one of the second plurality of data blocks associated with the one of the first plurality of block addresses. Locating a desired data block in a decompressed data segment is discussed above with respect to FIG. 5.

[0058] FIG. 9 is a flowchart of an example method 900 for making a physical location available for data storage. Although execution of method 900 is described below with reference to processor 402 of FIG. 4, it should be understood that execution of method 900 may be performed by other suitable devices, such as processors 302 and 502 of FIGS. 3 and 5, respectively. Some blocks of method 900 may be performed in parallel with and/or after method 600, 700, or 800. Method 900 may be implemented in the form of executable instructions stored on a machine-readable storage medium and/or in the form of electronic circuitry.

[0059] Method 900 may start in block 902, where processor 402 may receive write requests (e.g., from hosts). Write requests may associate data blocks with respective block addresses. In block 904, processor 402 may update pointers in response to write requests, as discussed above with respect to FIG. 7. In block 906, processor 402 may determine whether any pointer comprises a mapping to one of a

- 15 -

plurality of physical locations. The determination of block 906 may be made based on a reference counter or garbage collection mechanism, as discussed above with respect to FIG. 4.

[0060] If, in block 906, processor 402 determines that a pointer comprises a mapping to the one of the plurality of physical locations, method 900 may loop back to 902. If, in block 906, processor 402 determines that no pointer comprises a mapping to the one of the plurality of physical locations, method 900 may proceed to block 908, in which processor 402 may make the one of the plurality of physical locations available to store a data unit. If a data unit is already stored in the one of the plurality of physical locations, processor 402 may allow the data unit to be overwritten with another data unit. Method 900 may then loop back to block 902.

[0061] The foregoing disclosure describes storing compressed data. Example implementations described herein enable compression of multiple data blocks together into a data unit, and tracking data blocks in physical locations that store data units whose sizes are uniformly capped. Thus, fragmentation in storage systems may be reduced, and management of stored compressed data may be simplified.

- 16 -

I claim:

1. A system for storing compressed data, the system comprising:
 - a compression module to compress data blocks into data units, wherein a data unit comprises a plurality of compressed data blocks, and wherein a respective size of each data unit is less than or equal to a fixed size;
 - a data storage module to store each data unit in a respective physical location; and
 - a metadata storage module to track data blocks in the physical locations.
2. The system of claim 1, wherein:
 - the metadata storage module is to store position indicators; and
 - position indicators indicate positions of data blocks in decompressed data segments obtained by decompressing data units.
3. The system of claim 2, further comprising a locator module to locate, based on a position indicator, a data block, that is associated with a block address specified in a read request, in a decompressed data segment obtained by decompressing a data unit.
4. The system of claim 1, wherein:
 - the metadata storage module is to store pointers;
 - each pointer corresponds to a respective block address;
 - each block address is associated with a respective data block; and
 - each pointer comprises a mapping of the respective block address to a physical location in which a data unit, into which the respective data block is compressed, is stored.
5. The system of claim 4, further comprising a counter module to maintain a plurality of reference counters, each of the plurality of reference counters corresponding to a respective physical location, wherein each of the plurality of reference counters indicates how many pointers comprise a mapping to the respective physical location.

- 17 -

6. A machine-readable storage medium encoded with instructions executable by a processor of a system for storing compressed data, the machine-readable storage medium comprising:

instructions to compress a plurality of data blocks into a data unit;

instructions to store the data unit in one of a plurality of physical locations, wherein data units stored in the plurality of physical locations have respective sizes that are each less than or equal to a fixed size; and

instructions to store a plurality of position indicators, wherein each of the position indicators indicates a respective position, of a respective one of the plurality of data blocks, in a decompressed data segment obtained by decompressing the data unit.

7. The machine-readable storage medium of claim 6, further comprising instructions to store a plurality of pointers, wherein:

each of the plurality of pointers corresponds to a respective one of a plurality of block addresses;

each of the plurality of block addresses is associated with a respective one of the plurality of data blocks; and

each of the plurality of pointers comprises a mapping of the respective one of the plurality of block addresses to the one of the plurality of physical locations.

8. The machine-readable storage medium of claim 7, further comprising:

instructions to update one of the plurality of pointers in response to a write request that comprises one of the plurality of block addresses; and

instructions to make, in response to a determination that no pointer comprises a mapping to the one of the plurality of physical locations, the one of the plurality of physical locations available to store another data unit.

9. The machine-readable storage medium of claim 7, further comprising:

instructions to receive a read request, wherein the read request comprises one of the plurality of block addresses;

- 18 -

instructions to identify, based on one of the plurality of pointers, the one of the plurality of physical locations that maps to the one of the plurality of block addresses;

instructions to decompress the data unit, that is stored in the one of the plurality of physical locations, to obtain the decompressed data segment; and

instructions to locate, based on one of the plurality of position indicators, the data block in the decompressed data segment.

10. A method for storing compressed data, the method comprising:

receiving a first plurality of data blocks;

compressing the first plurality of data blocks into a first data unit;

storing the first data unit in a first physical location of a plurality of physical locations, wherein data units stored in the plurality of physical locations have respective sizes that are each less than or equal to a fixed size; and

storing a first set of position indicators, wherein each of the first set of position indicators indicates a respective position, of a respective one of the first plurality of data blocks, in a first decompressed data segment obtained by decompressing the first data unit.

11. The method of claim 10, further comprising storing a plurality of pointers, wherein:

each of the plurality of pointers corresponds to a respective one of a first plurality of block addresses;

each of the first plurality of block addresses is associated with a respective one of the first plurality of data blocks; and

each of the plurality of pointers comprises a mapping of the respective one of the first plurality of block addresses to the first physical location.

12. The method of claim 11, wherein a first write request comprises the first plurality of data blocks and associates the first plurality of data blocks with the first plurality of block addresses, the method further comprising:

- 19 -

receiving a subsequent write request that comprises a second plurality of data blocks and associates one of the second plurality of data blocks with one of the first plurality of block addresses;

storing a second data unit, into which the second plurality of data blocks is compressed, in a second physical location of the plurality of physical locations; and

updating a pointer, of the plurality of pointers, that corresponds to the one of the first plurality of block addresses, wherein the updated pointer comprises a mapping of the one of the first plurality of block addresses to the second physical location.

13. The method of claim 12, further comprising storing a second set of position indicators, wherein each of the second set of position indicators indicates a respective position of a respective one of the second plurality of data blocks in a second decompressed data segment obtained by decompressing the second data unit.

14. The method of claim 13, further comprising:

receiving, after the subsequent write request is received, a read request comprising the one of the first plurality of block addresses;

decompressing, in response to the read request, the second data unit to obtain the second decompressed data segment; and

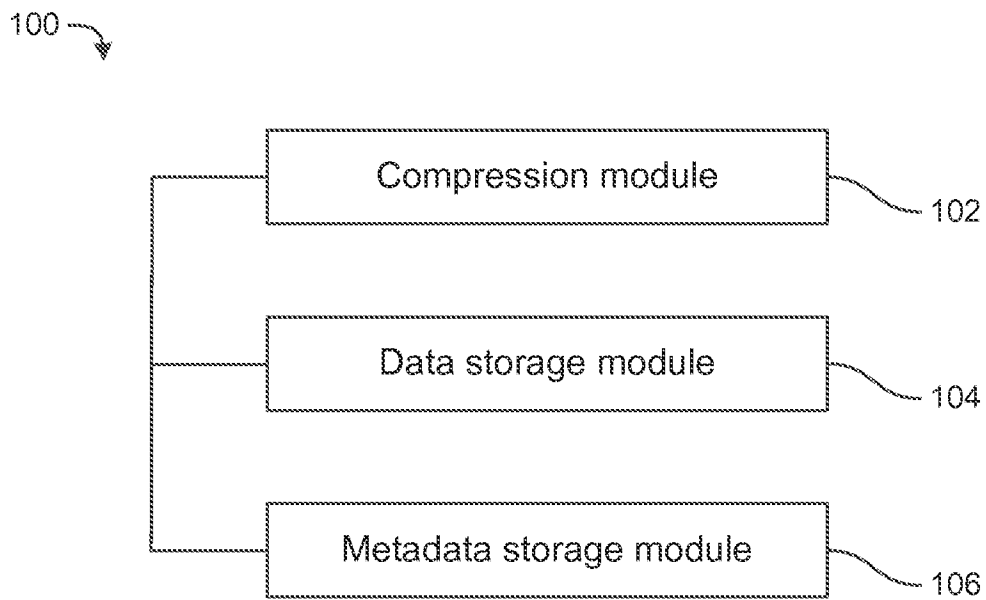
locating, in the second decompressed data segment, and based on the second set of position indicators, the one of the second plurality of data blocks associated with the one of the first plurality of block addresses.

15. The method of claim 12, further comprising:

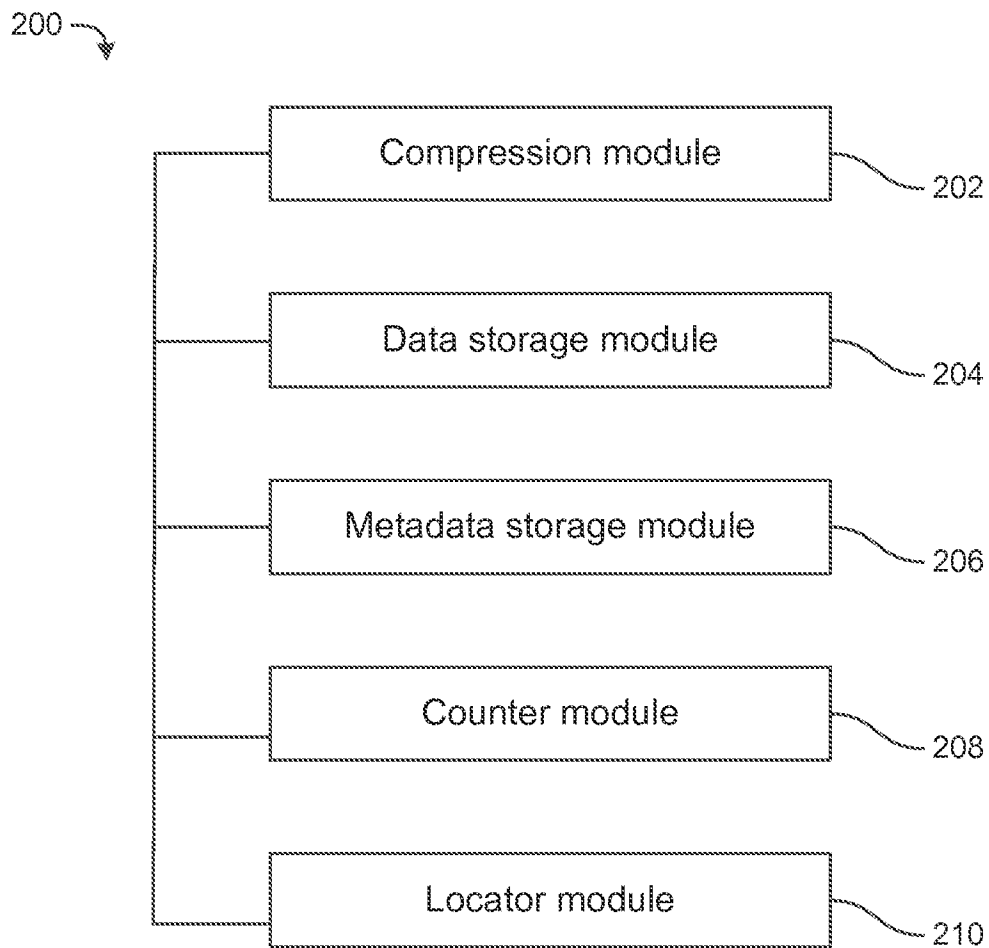
determining whether any pointer comprises a mapping to one of the plurality of physical locations; and

in response to a determination that no pointer comprises a mapping to the one of the plurality of physical locations, making the one of the plurality of physical locations available to store a third data unit.

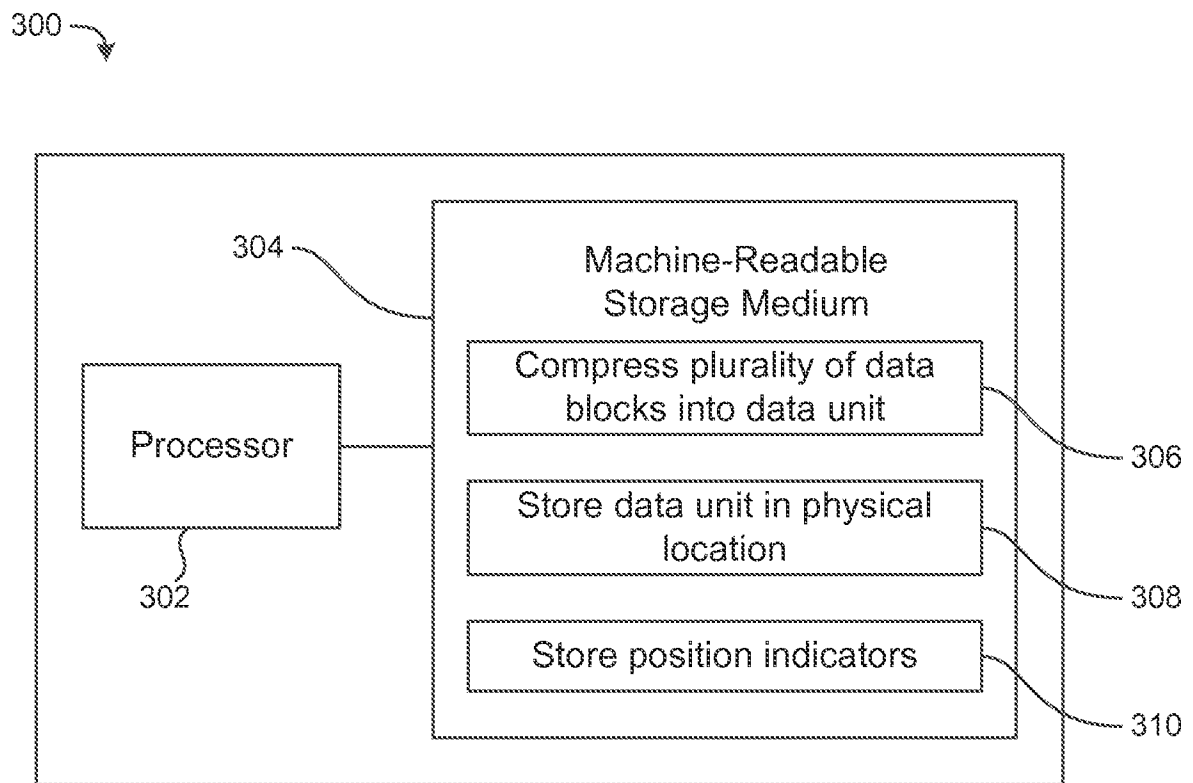
1/9

**FIG. 1**

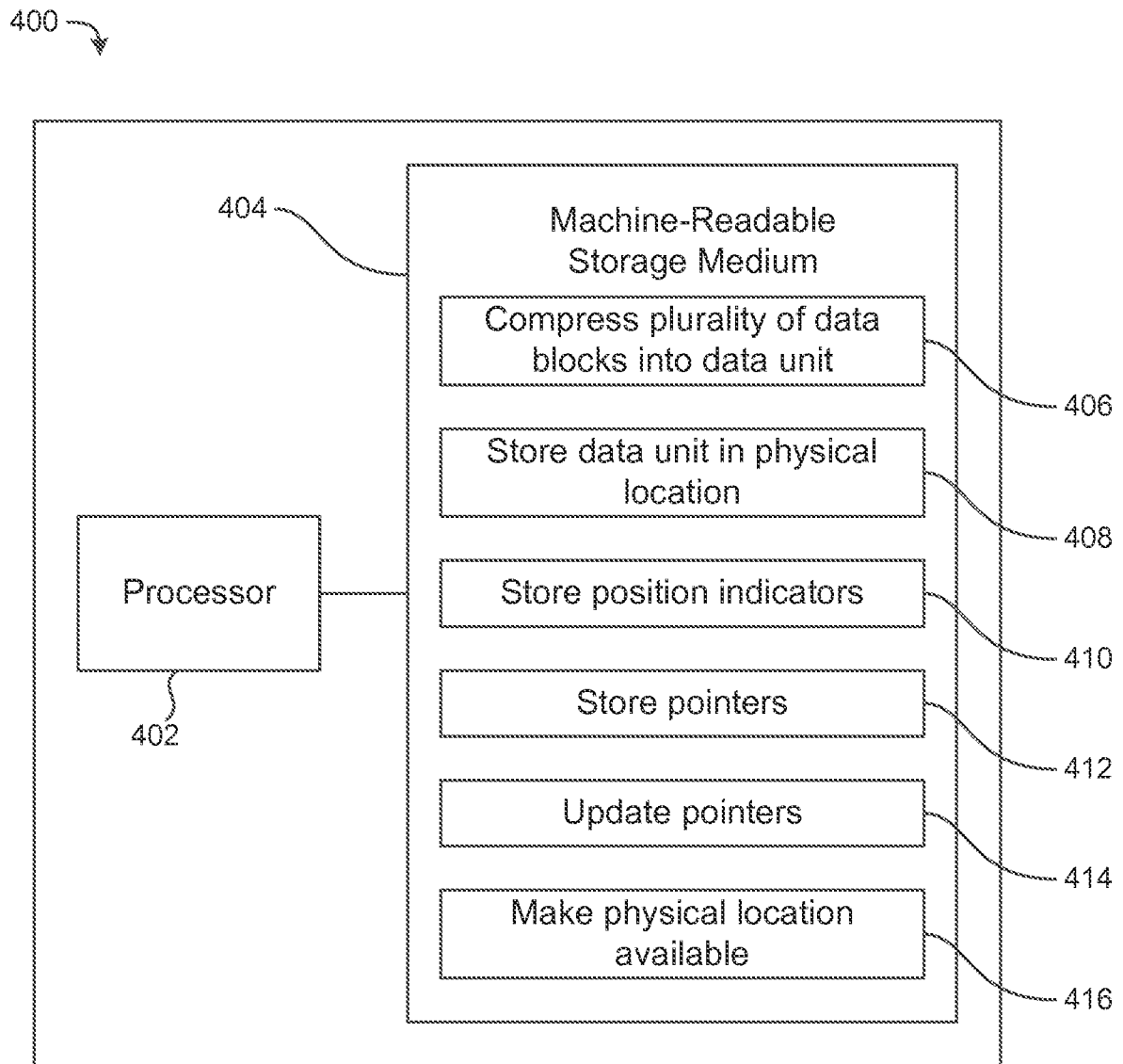
2/9

**FIG. 2**

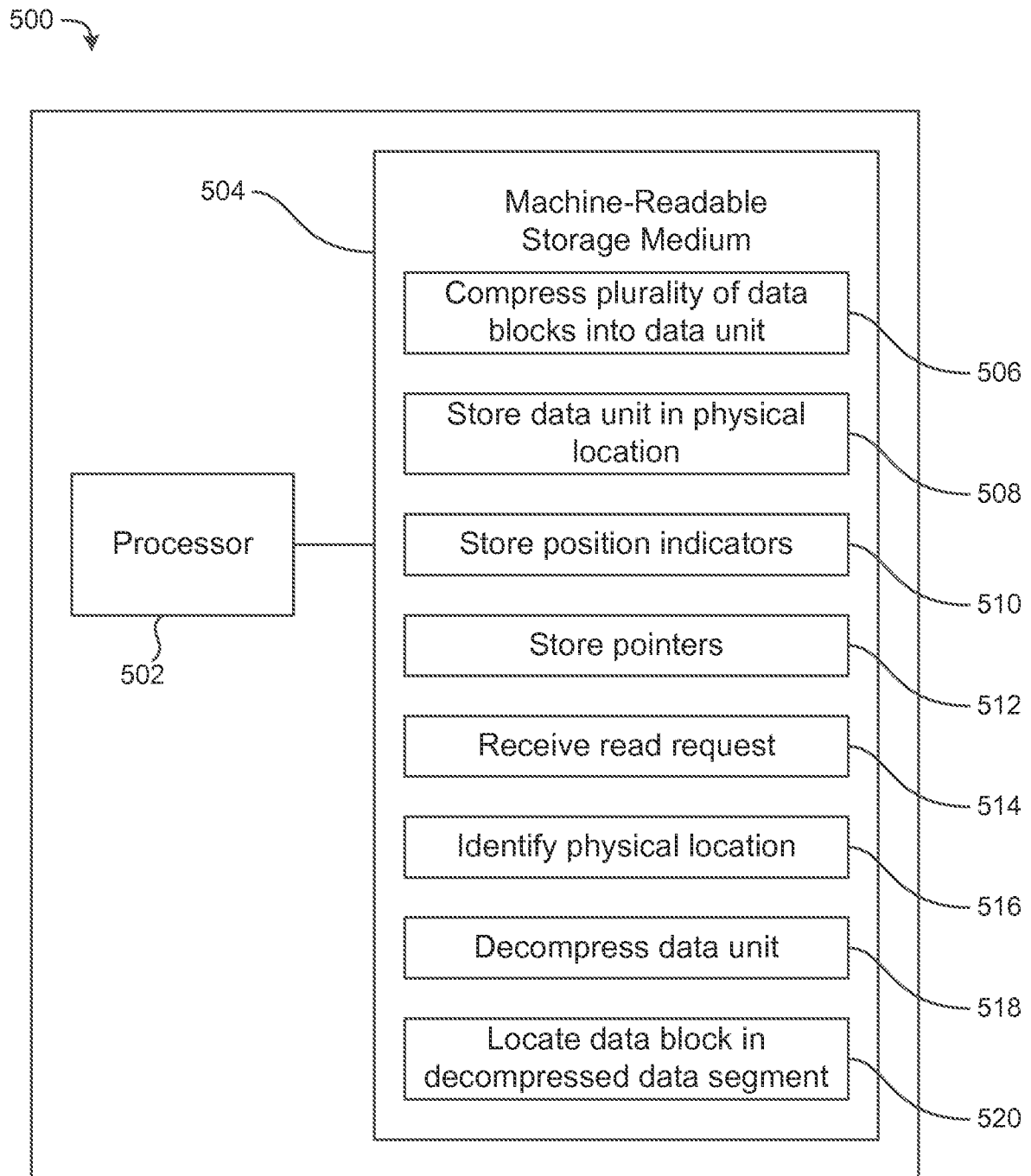
3/9

**FIG. 3**

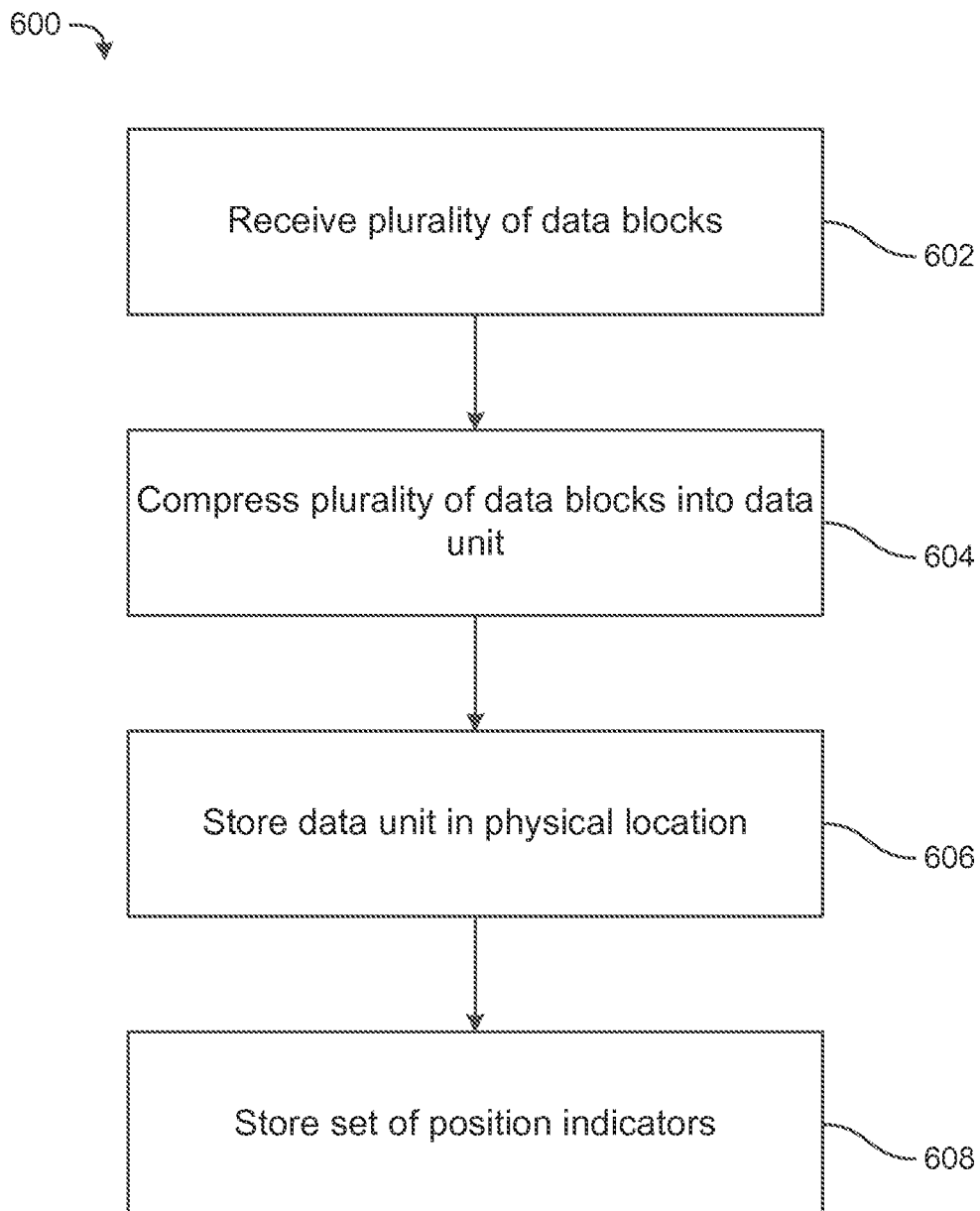
4/9

**FIG. 4**

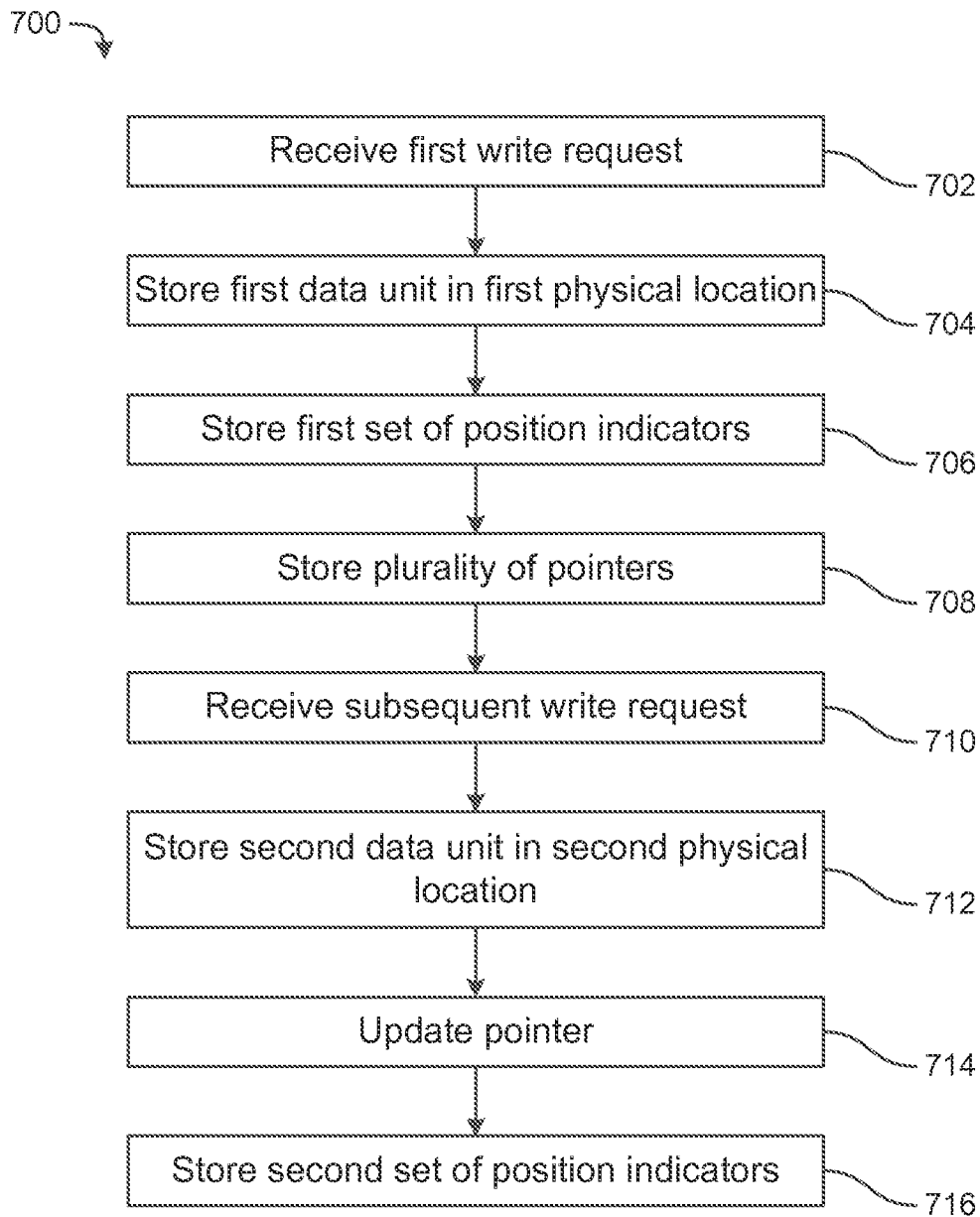
5/9

**FIG. 5**

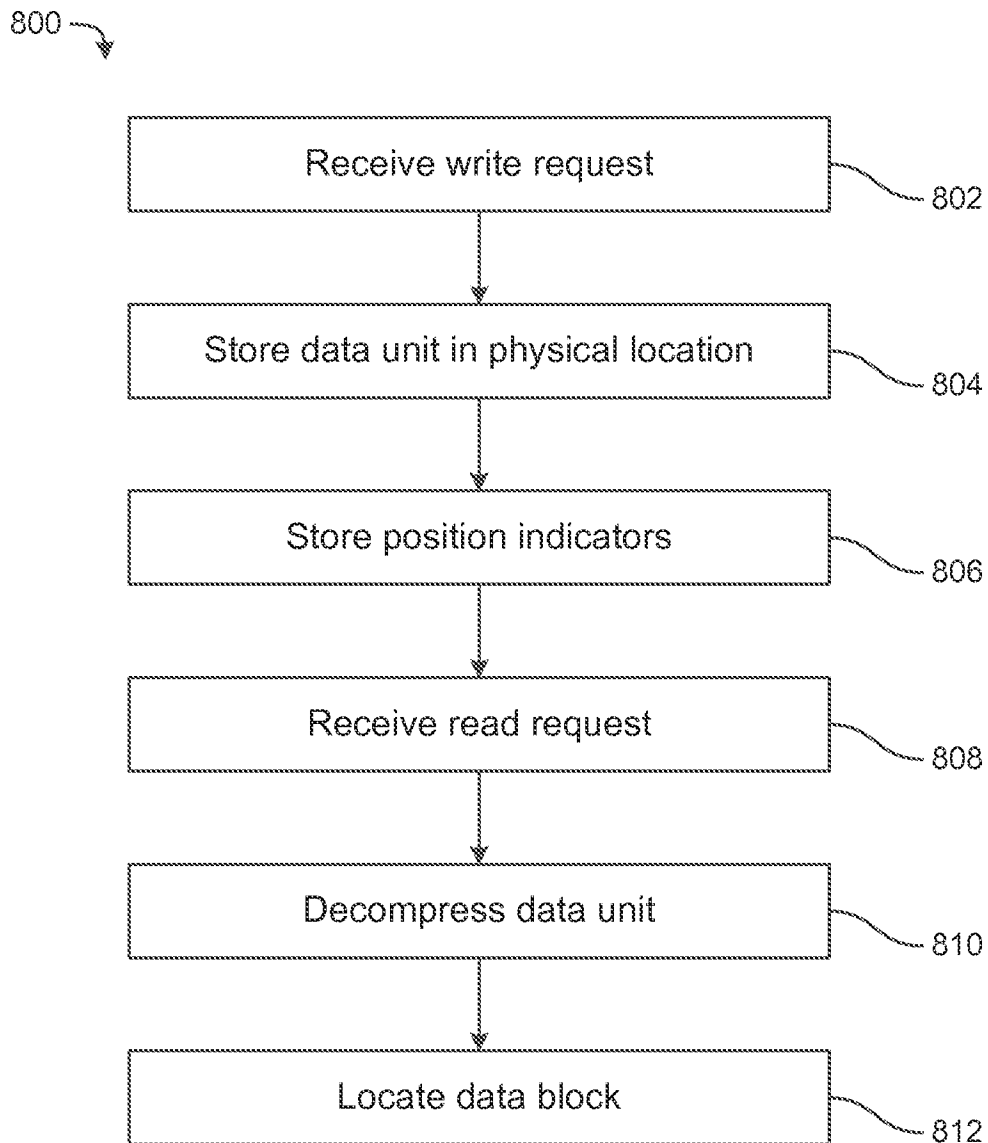
6/9

**FIG. 6**

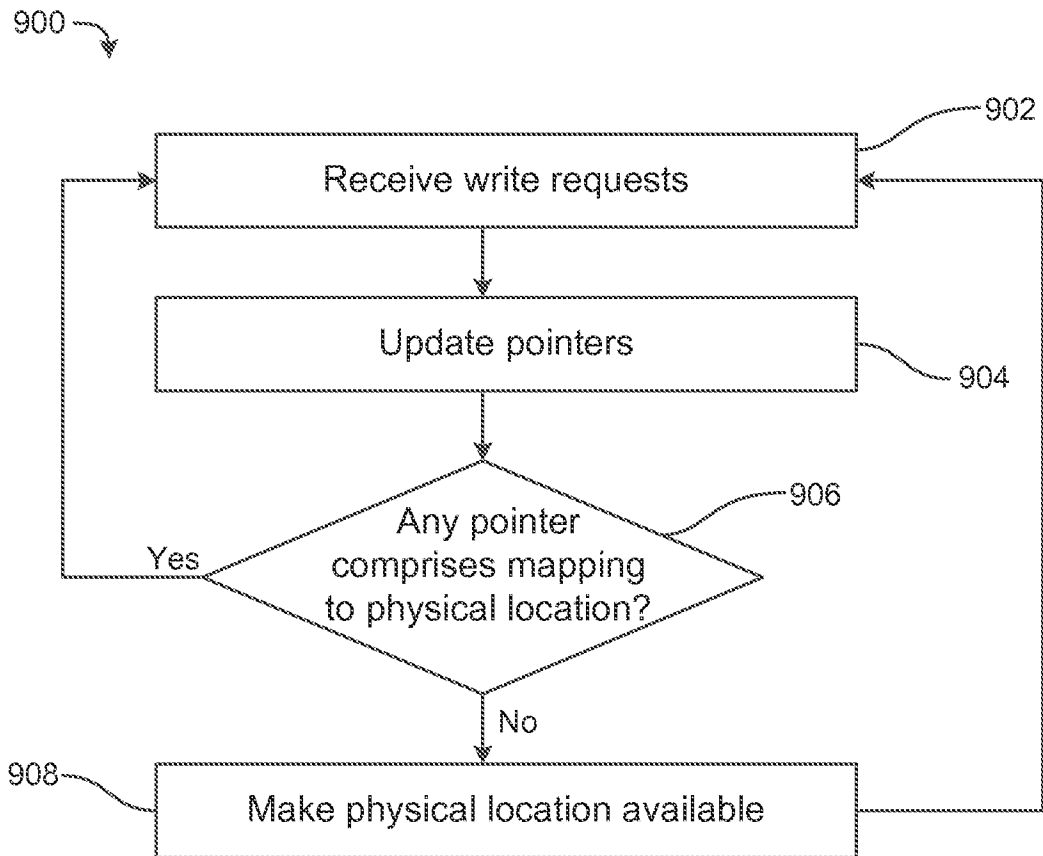
7/9

**FIG. 7**

8/9

**FIG. 8**

9/9

**FIG. 9**

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2014/039578**A. CLASSIFICATION OF SUBJECT MATTER****G06F 12/02(2006.01)i, G06F 12/08(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 12/02; G06F 12/12; G06F 12/06; H04L 29/08; G06F 17/30; G06F 12/08

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

Korean utility models and applications for utility models

Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

eKOMPASS(KIPO internal) & Keywords: compress, storage, fixed, size

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2011-0218970 A1 (JONATHAN AMIT et al.) 08 September 2011 See paragraphs [0002], [0022], [0060]-[0065], [0095]-[0107]; claims 1-2, 8; and figures 1, 7-9.	1-4, 6-7, 9-11
Y		5, 12-14
A		8, 15
Y	US 2012-0324170 A1 (PAUL E. MCKENNEY) 20 December 2012 See paragraphs [0049], [0060], [0071]; claim 1; and figure 1.	5, 12-14
A	US 2014-0108493 A1 (JASON K. RESCH et al.) 17 April 2014 See paragraphs [0011], [0064]; claim 1; and figure 1.	1-15
A	US 2008-0307191 A1 (GEOFF LANE et al.) 11 December 2008 See paragraphs [0011], [0038]; claim 1; and figure 4.	1-15
A	EP 2725764 A1 (HUAWEI TECHNOLOGIES CO., LTD.) 30 April 2014 See paragraphs [0001], [0009]; claim 1; and figure 1.	1-15



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

10 February 2015 (10.02.2015)

Date of mailing of the international search report

11 February 2015 (11.02.2015)

Name and mailing address of the ISA/KR

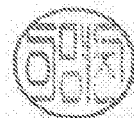
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan City, 302-701,
Republic of Korea

Facsimile No. ++82 42 472 3473

Authorized officer

LEE, Myung Jin

Telephone No. +82-42-481-8474



INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2014/039578

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2011-0218970 A1	08/09/2011	CN 101346883 A IL 191082 A IL 191082 D0 IL 191083 D0 US 2011-218975 A1 US 2011-219144 A1 US 2012-265740 A1 US 8285898 B2 US 8473652 B2 US 8656075 B2 US 8677039 B2	14/01/2009 31/07/2013 29/12/2008 29/12/2008 08/09/2011 08/09/2011 18/10/2012 09/10/2012 25/06/2013 18/02/2014 18/03/2014
US 2012-0324170 A1	20/12/2012	US 2014-089596 A1	27/03/2014
US 2014-0108493 A1	17/04/2014	US 2011-106904 A1 US 2011-107165 A1 US 2013-290809 A1 US 2014-280681 A1 US 2014-317224 A1 US 8479078 B2 US 8631303 B2 US 8769035 B2	05/05/2011 05/05/2011 31/10/2013 18/09/2014 23/10/2014 02/07/2013 14/01/2014 01/07/2014
US 2008-0307191 A1	11/12/2008	US 8250335 B2	21/08/2012
EP 2725764 A1	30/04/2014	CN 102541475 A EP 2725764 A4 US 2014-0013051 A1 WO 2013-135105 A1	04/07/2012 03/09/2014 09/01/2014 19/09/2013