



- (51) **International Patent Classification:**  
G06F 17/00 (2006.01)
- (21) **International Application Number:**  
PCT/US2011/058679
- (22) **International Filing Date:**  
31 October 2011 (31.10.2011)
- (25) **Filing Language:** English
- (26) **Publication Language:** English
- (30) **Priority Data:**  
12/951,760 22 November 2010 (22.11.2010) US
- (71) **Applicant (for all designated States except US):** MICROSOFT CORPORATION [US/US]; One Microsoft Way, Redmond, Washington 98052-6399 (US).
- (72) **Inventors:** KALACH, Ran; Microsoft Corporation, LCA - International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). DICKSON, Mathew James; Microsoft Corporation, LCA - International Pat-

ents, One Microsoft Way, Redmond, Washington 98052-6399 (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, 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, IS, JP, KE, KG, KM, 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, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, 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, MD, 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,

[Continued on next page]

(54) Title: HETEROGENEOUS FILE OPTIMIZATION

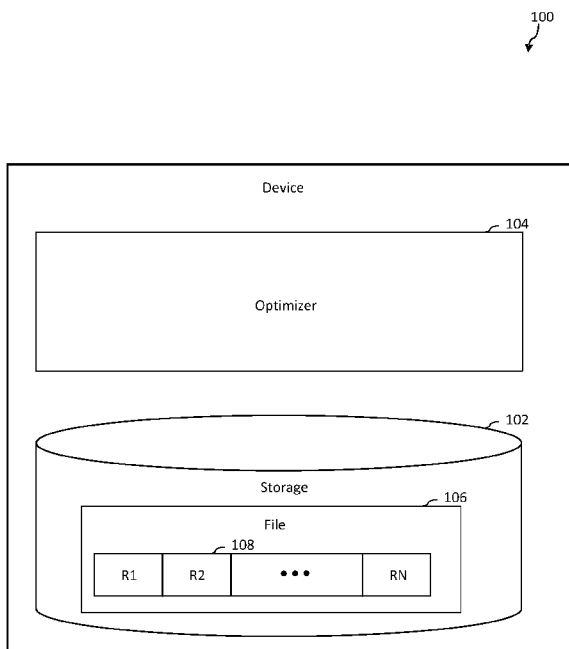


FIG. 1

(57) **Abstract:** Techniques are described herein that are capable of heterogeneously optimizing a file. Heterogeneous optimization involves optimizing regions of a file non-uniformly. For example, the regions of the file may be optimized to different extents. In accordance with this example, a different optimization technique may be used to optimize each region or subset of the regions. In one aspect, optimization designations are assigned to respective regions of a file based on access patterns that are associated with the respective regions. The file may be a database file, a virtualized storage file, or other suitable type of file. Each optimization designation indicates an extent to which the respective region is to be optimized. Each region may be optimized to the extent that is indicated by the respective optimization designation that is assigned to that region.

WO 2012/071140 A2



SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ,  
GW, ML, MR, NE, SN, TD, TG).

— *as to the applicant's entitlement to claim the priority of  
the earlier application (Rule 4.17(iii))*

**Declarations under Rule 4.17:**

— *as to applicant's entitlement to apply for and be granted  
a patent (Rule 4.17(ii))*

**Published:**

— *without international search report and to be republished  
upon receipt of that report (Rule 48.2(g))*

## HETEROGENEOUS FILE OPTIMIZATION

### BACKGROUND

[0001] Data optimization is the act of reducing an amount of data that is stored on a storage device (e.g., a disk) or transmitted across a network without compromising the fidelity and integrity of the original data. Data optimization often involves a combination of techniques for eliminating redundancy in and between persistently stored files. Data de-duplication (dedup) is one such technique in which identical regions (a.k.a. chunks) of data in one or more files are stored as a single region. Compression is another such technique in which data is encoded to include fewer bits (or other information-bearing units) than the original data.

[0002] Once data is optimized, the data may be accessed by reversing the effects of the optimization (i.e., de-optimizing the optimized data), for example by performing an inverse dedup operation and/or a decompression operation with respect to the optimized data. However, de-optimization causes a delay with respect to accessing the data. A greater amount of data results in a longer latency. Moreover, such latency may occur each time the data is accessed unless a de-optimized version of the data is stored for access on a storage device. Furthermore, de-optimization often consumes substantial resources (e.g., memory, central processing unit (CPU), disk I/O, etc.) of a device, which may negatively affect a main workload that is running on the device. Accordingly, frequent de-optimization may result in relatively inefficient utilization of the device's resources.

[0003] For example, if data in a file is fully optimized, the latency that is associated with accessing the data may unduly degrade the performance of a device that accesses the data and/or a workload that is running on the device, especially if the data is frequently accessed. In another example, it may not be desirable to optimize some regions of a file and/or some types of data. However, the various regions of the file may not be visible to a device that attempts to optimize the file. Accordingly, the device may have no way to know whether the regions of the file are optimizable.

### SUMMARY

[0004] Various approaches are described herein for, among other things, heterogeneously optimizing a file. Heterogeneous optimization involves optimizing regions of a file non-uniformly. For example, the regions of the file may be optimized to different extents. In accordance with this example, a different optimization technique may be used to optimize each region or subset of the regions.

[0005] An example method is described in which optimization designations are assigned to respective regions of a file based on access patterns that are associated with the respective regions. Each optimization designation indicates an extent to which the respective region is to be optimized. Each region is optimized to the extent that is indicated by the respective optimization designation that is assigned to that region.

[0006] Another example method is described in which access indicators are assigned to respective regions of a file. The access indicators correspond to respective access patterns that are associated with the respective regions. For example, the access patterns may be monitored using a file system filter driver. Optimization designations are assigned to the respective regions based on the respective access indicators that are assigned to the respective regions. Each optimization designation indicates an extent to which the respective region is to be optimized. Each region is optimized to the extent that is indicated by the respective optimization designation that is assigned to that region.

[0007] Yet another example method is described in which a virtualized storage file that includes multiple regions is mounted to provide a mounted virtualized storage file. The mounted virtualized storage file includes data sequences that correspond to the respective regions. Each data sequence is included in a collection of one or more respective files. A disk filter is executed with respect to the mounted virtualized storage file to monitor an access pattern of each collection. Optimization designations are assigned to the respective regions based on the access patterns of the respective collections that include the corresponding data sequences. Each region is optimized to the extent that is indicated by the respective optimization designation that is assigned to that region.

[0008] An example system is described that includes an assignment module and an optimization module. The assignment module is configured to assign optimization designations to respective regions of a file based on access patterns that are associated with the respective regions. Each optimization designation indicates an extent to which the respective region is to be optimized. The optimization module is configured to optimize each region to the extent that is indicated by the respective optimization designation that is assigned to that region.

[0009] Another example system is described that includes an assignment module and an optimization module. The assignment module is configured to assign access indicators to respective regions of a file. The access indicators correspond to respective access patterns that are associated with the respective regions. The assignment module is further configured to assign optimization designations to the respective regions based on the

respective access indicators that are assigned to the respective regions. Each optimization designation indicates an extent to which the respective region is to be optimized. The optimization module is configured to optimize each region to the extent that is indicated by the respective optimization designation that is assigned to that region.

5 **[0010]** Yet another example system is described that includes a mounting module, an access monitor, an assignment module, and an optimization module. The mounting module is configured to mount a virtualized storage file that includes multiple regions to provide a mounted virtualized storage file. The mounted virtualized storage file includes data sequences that correspond to the respective regions. Each data sequence is included  
10 in a collection of one or more respective files. The access monitor is configured to execute a disk filter with respect to the mounted virtualized storage file to monitor an access pattern of each collection. The assignment module is configured to assign optimization designations to the respective regions based on the access patterns of the respective collections that include the corresponding data sequences. The optimization  
15 module is configured to optimize each region to the extent that is indicated by the respective optimization designation that is assigned to that region.

**[0011]** An example computer program product is described that includes a computer-readable medium having computer program logic recorded thereon for heterogeneously optimizing a file. The computer program product includes first and second program logic  
20 modules. The first program logic module is for enabling the processor-based system to assign optimization designations to respective regions of a file based on access patterns that are associated with the respective regions. Each optimization designation indicates an extent to which the respective region is to be optimized. The second program logic module is for enabling the processor-based system to optimize each region to the extent  
25 that is indicated by the respective optimization designation that is assigned to that region.

**[0012]** Another example computer program product is described that includes a computer-readable medium having computer program logic recorded thereon for heterogeneously optimizing a file. The computer program product includes first, second, and third program logic modules. The first program logic module is for enabling the  
30 processor-based system to assign access indicators to respective regions of a file. The access indicators correspond to respective access patterns that are associated with the respective regions. The second program logic module is for enabling the processor-based system to assign optimization designations to the respective regions based on the respective access indicators that are assigned to the respective regions. Each optimization

designation indicates an extent to which the respective region is to be optimized. The third program logic module is for enabling the processor-based system to optimize each region to the extent that is indicated by the respective optimization designation that is assigned to that region.

5 [0013] Yet another example computer program product is described that includes a computer-readable medium having computer program logic recorded thereon for heterogeneously optimizing a file. The computer program product includes first, second, third, and fourth program logic modules. The first program logic module is for enabling the processor-based system to mount a virtualized storage file that includes multiple  
10 regions to provide a mounted virtualized storage file. The mounted virtualized storage file includes data sequences that correspond to the respective regions. Each data sequence is included in a collection of one or more respective files. The second program logic module is for enabling the processor-based system to execute a disk filter with respect to the mounted virtualized storage file to monitor an access pattern of each collection. The third  
15 program logic module is for enabling the processor-based system to assign optimization designations to the respective regions based on the access patterns of the respective collections that include the corresponding data sequences. The fourth program logic module is for enabling the processor-based system to optimize each region to the extent that is indicated by the respective optimization designation that is assigned to that region.

20 [0014] This Summary is provided to introduce a selection of concepts in a simplified form that are further described below in the Detailed Description. This Summary is not intended to identify key features or essential features of the claimed subject matter, nor is it intended to be used to limit the scope of the claimed subject matter. Moreover, it is noted that the invention is not limited to the specific embodiments described in the  
25 Detailed Description and/or other sections of this document. Such embodiments are presented herein for illustrative purposes only. Additional embodiments will be apparent to persons skilled in the relevant art(s) based on the teachings contained herein.

#### BRIEF DESCRIPTION OF THE DRAWINGS/FIGURES

[0015] The accompanying drawings, which are incorporated herein and form part of the  
30 specification, illustrate embodiments of the present invention and, together with the description, further serve to explain the principles involved and to enable a person skilled in the relevant art(s) to make and use the disclosed technologies.

[0016] FIG. 1 is a block diagram of an example device in accordance with an embodiment.

[0017] FIG. 2 is a block diagram of an example host device in accordance with an embodiment.

[0018] FIGS. 3-6 depict flowcharts of example methods for heterogeneously optimizing a file in accordance with embodiments.

5 [0019] FIG. 7 is a block diagram of an example implementation of an optimizer shown in FIG. 1 in accordance with an embodiment.

[0020] FIG. 8 depicts an example computer in which embodiments may be implemented.

[0021] The features and advantages of the disclosed technologies will become more  
10 apparent from the detailed description set forth below when taken in conjunction with the drawings, in which like reference characters identify corresponding elements throughout. In the drawings, like reference numbers generally indicate identical, functionally similar, and/or structurally similar elements. The drawing in which an element first appears is indicated by the leftmost digit(s) in the corresponding reference number.

15 **DETAILED DESCRIPTION**

I. Introduction

[0022] The following detailed description refers to the accompanying drawings that illustrate exemplary embodiments of the present invention. However, the scope of the present invention is not limited to these embodiments, but is instead defined by the  
20 appended claims. Thus, embodiments beyond those shown in the accompanying drawings, such as modified versions of the illustrated embodiments, may nevertheless be encompassed by the present invention.

[0023] References in the specification to "one embodiment," "an embodiment," "an example embodiment," or the like, indicate that the embodiment described may include a  
25 particular feature, structure, or characteristic, but every embodiment may not necessarily include the particular feature, structure, or characteristic. Moreover, such phrases are not necessarily referring to the same embodiment. Furthermore, when a particular feature, structure, or characteristic is described in connection with an embodiment, it is submitted that it is within the knowledge of one skilled in the relevant art(s) to implement such  
30 feature, structure, or characteristic in connection with other embodiments whether or not explicitly described.

II. Example Embodiments

[0024] Example embodiments described herein are capable of heterogeneously optimizing a file. Heterogeneous optimization involves optimizing regions of a file non-

uniformly. For example, the regions of the file may be optimized to different extents. In accordance with this example, a different optimization technique may be used to optimize each region or subset of the regions.

**[0025]** In example embodiments, optimization designations are assigned to respective regions of a file based on access patterns that are associated with the respective regions. An access pattern indicates and/or describes access(es) and/or modification(s) with respect to a region with which the access pattern is associated. The file may be a database file, a virtualized storage file, or other suitable type of file. A database file is a file that includes multiple record files and/or multiple log files. A virtualized storage file is a file that is configured to be mounted as a disk or a volume to provide a file system interface for accessing hosted files. In accordance with these example embodiments, each optimization designation indicates an extent to which the respective region is to be optimized. Each region may be optimized to the extent that is indicated by the respective optimization designation that is assigned to that region.

**[0026]** Optimization designations may be defined in any suitable manner. For instance, a first optimization designation may indicate that a first region of a file is to be compressed but not de-duplicated. A second optimization designation may indicate that a second region is to be de-duplicated but not compressed. A third optimization designation may indicate that a third region is to be compressed and de-duplicated. A fourth optimization designation may indicate that a fourth region is to be neither compressed nor de-duplicated. Fifth and sixth optimization designations may indicate that fifth and sixth regions are to be compressed using respective first and second compression techniques. Seventh and eighth optimization designations may indicate that seventh and eighth regions are to be de-duplicated using respective first and second de-duplication techniques, and so on.

**[0027]** In an example embodiment, each optimization designation indicates a respective optimization policy or a respective optimization level within a global policy. An optimization policy is a set of rules that defines a manner in which regions of a file are to be optimized based on designated criteria. The optimization policy may define multiple optimization levels. Each optimization level indicates one or more data optimization techniques that are to be performed with respect to the regions that satisfy a respective subset of the designated criteria. For example, a first optimization level may indicate that no optimization is to be performed with respect to regions that are associated with the first optimization level. In accordance with this example, the extent to which a region is

optimized may increase as the optimization level that is associated with the region increases. In an aspect, increasing an optimization level that is associated with a region may increase storage savings but may consume more computational resources and/or add latency to data access operations that are performed with respect to the region.

5 Accordingly, selecting an optimization level to be associated with a region may involve balancing storage savings with increased resource consumption and/or latency. Such balancing may be based on or influenced by the data optimization technique that provides the greatest optimization for the type of data that is to be optimized. It will be recognized that an optimization designation that indicates an optimization policy may further indicate  
10 an optimization level that is defined by that optimization policy.

**[0028]** Example techniques described herein have a variety of benefits as compared to conventional techniques for optimizing a file. For instance, some example techniques may optimize the various regions of a file to different degrees. Some example techniques may partially optimize a file, meaning that one or more regions of the file are not optimized.

15 The time and/or amount of resources that is consumed by a device to access a file that is optimized in accordance with one or more of the example techniques described herein may be less than the time and/or amount of resources that is consumed by the device to access the file if it were optimized using conventional techniques.

**[0029]** FIG. 1 is a block diagram of an example device 100 in accordance with an  
20 embodiment. Device 100 is a processing system that is capable of optimizing a file. An example of a processing system is a system that includes at least one processor that is capable of manipulating data in accordance with a set of instructions. For instance, a processing system may be a computer, a personal digital assistant, etc.

**[0030]** Device 100 includes storage 102 and an optimizer 104. Storage 102 stores a file  
25 106. File 106 includes multiple regions 108. File 106 is shown to include a vector of N regions (labeled as R1, R2, ... , RN) for illustrative purposes and is not intended to be limiting. It will be recognized that file 106 may include any suitable number and/or configuration of regions. For instance, regions 108 need not necessarily be stored contiguously on storage 102. Moreover, a region need not necessarily be stored using  
30 contiguous bits of storage 102. Each region may correspond to any suitable offset in file 106. The number of bits in each region may be based on any of a variety of factors, such as an amount of memory that is available for tracking the regions. Each region may include any number of bits (i.e., may be any size), and the number of bits in each region

may be the same or different. Furthermore, the number of bits in each region may be fixed or variable.

[0031] Regions 108 may be defined based on access patterns that are associated with the regions 108, access indicators that are assigned to the respective regions 108, and/or any other suitable factor(s). An access pattern indicates and/or describes access(es) and/or modification(s) with respect to a region with which the access pattern is associated. For instance, an access pattern may include a time at which a region was most recently accessed, a number of times that the region is accessed, a frequency with which the region is accessed, a time at which the region was most recently modified, a number of times that the region is modified, a frequency with which the region is modified, an indication as to whether the region is accessed during a system boot operation (e.g., with respect to device 100), whether the region is accessed by a specified application (e.g., a database application), etc.

[0032] Access patterns may correspond to access indicators. Each access indicator is associated with one or more respective criteria. An access indicator is assigned to a region if an access pattern that is associated with the region satisfies the one or more criteria that are associated with that access indicator. Accordingly, multiple instances of an access indicator may be assigned among the region 108 of file 106, though the scope of the example embodiments is not limited in this respect. Access indicators are discussed in greater detail below with reference to FIGS. 4 and 5.

[0033] Optimizer 104 is configured to optimize regions 108 of file 106 based on optimization designations that are assigned to the regions 108. The optimization designations may be based on access patterns that are associated with the regions 108 and/or any other suitable factor(s). Techniques for optimizing regions (e.g., regions 108) are described in detail below with reference to FIGS. 3-6.

[0034] FIG. 2 is a block diagram of an example host device 200 in accordance with an embodiment. Host device 200 is a processing system that is capable of mounting a virtualized storage file to provide a virtual disk. Host device 200 includes storage 202 and an optimizer 204. Storage 202 stores a virtualized storage file 206, which includes multiple regions 208. Regions 208 may be defined based on access patterns that are associated with the regions 208, offsets in virtualized storage file 206 that correspond to hosted files that are stored on a virtual disk (e.g., virtual disk 210), and/or any other suitable factor(s). The number of bits in each region may be based on any of a variety of factors, including but not limited to an average number of bits included in hosted files that

correspond to the regions, an amount of memory that is available for tracking the regions, etc. Regions 208 are shown to be arranged as a vector of N regions (labeled as R1, R2, ..., RN) for illustrative purposes and are not intended to be limiting. It will be recognized that virtualized storage file 206 may include any suitable number and/or configuration of regions.

[0035] Optimizer 204 is configured to optimize regions 208 of virtualized storage file 206 based on optimization designations that are assigned to the regions 208. The optimization designations may be based on access patterns that are associated with the regions 208, properties of the hosted files, and/or any other suitable factor(s). Example properties of a hosted file include but are not limited to an access pattern of the hosted file, heuristics regarding the hosted file, a classification of the hosted file, a format of the hosted file, a type of the hosted file, an intended use of the hosted file (e.g., whether the hosted file is to be used to execute virtual machine 212 or during a system boot operation with regard to host device 200), whether the hosted file is accessed by a specified application (e.g., a database application), etc. Example formats of a hosted file include but are not limited to an Adobe® PDF format, a Microsoft® Office (e.g., Word®, Excel®, Visio®, etc.) format, a WordPerfect® format, an extensible markup language (XML) format, etc.

[0036] In some example embodiments, optimizer 204 is capable of mounting virtualized storage file 206 to provide virtual disk 210, as indicated by arrow 214. Virtual disk 210 is shown in FIG. 2 to be mounted on a virtual machine 212, which is configured to execute on host device 200. It will be recognized, however, that virtual disk 210 may be mounted on host device 200, rather than on virtual machine 212. For example, host device 200 need not necessarily include virtual machine 212. Mounting virtualized storage file 206 on host device 200 or virtual disk 210 may enable optimizer 204 to recognize virtual disk 210 as storage, rather than as a file. For instance, mounting virtualized storage file 206 may enable optimizer 204 to determine logical volume(s) and/or file system(s) that are associated with virtual disk 210 for purposes of optimizing regions 208 or hosted files that are included in virtual disk 210. Techniques for optimizing regions (e.g., regions 208) are described in detail below with reference to FIGS. 3-6.

[0037] FIGS. 3-6 depict flowcharts 300, 400, 500, and 600 of example methods for optimizing a file in accordance with embodiments. Flowcharts 300, 400, 500, and 600 may be performed by optimizer 104 of device 100 shown in FIG. 1 and/or by optimizer 204 of host device 200 shown in FIG. 2, for example. For illustrative purposes, flowcharts

300, 400, 500, and 600 are described with respect to an optimizer 700 shown in FIG. 7, which is an example of an optimizer 104 or 204, according to an embodiment. As shown in FIG. 7, optimizer 700 includes an assignment module 702, an optimization module 704, an access monitor 706, and a mounting module 708. Further structural and operational  
5 embodiments will be apparent to persons skilled in the relevant art(s) based on the discussion regarding flowcharts 300, 400, 500, and 600. It will be recognized that any one or more of assignment module 702, optimization module 704, access monitor 706, and/or mounting module 708 may be implemented in virtual machine 212.

**[0038]** As shown in FIG. 3, the method of flowchart 300 begins at step 302. In step 302,  
10 optimization designations are assigned to respective regions of a file based on access patterns that are associated with the respective regions. Each optimization designation may be assigned to the respective region based on any of a variety of factors, including but not limited to a latency that is associated with accessing that region, an amount of system resources (e.g., bandwidth, runtime, etc. of a processor, storage, or network in a system)  
15 that is utilized to access that region, whether that region is accessed in response to mounting an operating system that uses that region (e.g., to provide a virtual disk), whether that region is accessed by a specified application (e.g., a database application), etc. The file may be a virtualized storage file, a database file, or other suitable type of file. Each optimization designation indicates an extent to which the respective region is to be  
20 optimized. For instance, each optimization designation may indicate a respective optimization policy and/or optimization level in an optimization policy to be applied to the region to which that optimization designation is assigned. In an example implementation, assignment module 702 assigns the optimization designations to the respective regions.

**[0039]** In an example embodiment, an access pattern that is associated with a region  
25 indicates an extent to which the region is accessed. For instance, the extent may be based on a number of times that the region is accessed, a frequency with which the region is accessed, a time at which the region is most recently accessed, and/or a combination thereof. It will be recognized that the extent to which a region is access may be based on factor(s) in addition to or in lieu of the example factors recited above.

**[0040]** In another example embodiment, an access pattern that is associated with a  
30 region indicates an extent to which the region is modified. For example, the extent may be based on a number of times that the region is modified, a frequency with which the region is modified, a time at which the region is most recently modified, and/or a combination

thereof. It will be recognized that the extent to which a region is modified may be based on factor(s) in addition to or in lieu of the example factors recited above.

[0041] In yet another example embodiment, each access pattern indicates a specific extent to which the respective region is accessed or modified. The specific extent is a single value that represents the actual extent (or an estimate thereof) to which the  
5 respective region is accessed or modified. For example, if a region is modified seven times, the corresponding access pattern may indicate that the region is modified seven times, approximately zero times, approximately ten times, etc. In another example, if a region is accessed 253 times per hour, the corresponding access pattern may indicate that  
10 the region is accessed 253 time per hour, approximately 200 times per hour, approximately 250 times per hour, etc.

[0042] In still another example embodiment, each access pattern indicates a general extent that includes the specific extent to which the respective region is accessed or modified. For example, if a region is modified seven times, the corresponding access  
15 pattern may indicate that the region is modified between zero and ten times, between five and twenty times, or any other suitable range of times. In another example, if a region is accessed 253 times per hour, the corresponding access pattern may indicate that the region is accessed between zero and three-hundred times, between two-hundred and four-hundred times, or any other suitable range of times

[0043] For instance, first access pattern(s) may indicate a first general extent of zero to  
20 100 instances of access or modification. Second access pattern(s) may indicate a second general extent of 101 to 250 instances of access or modification. Third access pattern(s) may indicate a third general extent of 251 to 500 instances of access or modification, and so on. The example ranges of the general extents described herein are provided for  
25 illustrative purposes and are not intended to be limiting. It will be recognized that a general extent may indicate any suitable range of access instances, access frequencies, access times, modification instances, modification frequencies, modification times, etc.

[0044] At step 304, each region is optimized to the extent that is indicated by the respective optimization designation that is assigned to that region. In an example  
30 implementation, optimization module 704 optimizes each region.

[0045] In an example embodiment, each region is iteratively optimized to the extent that is indicated by the respective optimization designation that is assigned to that region. For example, optimization module 704 may monitor the regions to determine changes with respect to the regions since the most recent optimization of the regions. In accordance

with this example, optimization module 704 may create a differential file that includes the changes. For instance, optimization module 704 may optimize the regions on a periodic basis.

5 [0046] As shown in FIG. 4, the method of flowchart 400 begins at step 402. In step 402, access indicators are assigned to the respective regions of a file. The access indicators correspond to respective access patterns that are associated with the respective regions. In an example implementation, assignment module 702 assigns the access indicators to the respective regions.

10 [0047] In an example embodiment, access indicators having a first common value (e.g., one) are assigned to a first subset of the regions. Regions that are included in the first subset are associated with respective access patterns that satisfy at least one first criterion (e.g., most recent access time more than one day ago). Access indicators having a second common value (e.g., two) are assigned to a second subset of the regions. Regions that are included in the second subset are associated with respective access patterns that satisfy at  
15 least one second criterion (e.g., most recent access time more than one hour ago but less than one day ago). Access indicators having a third common value (e.g., three) are assigned to a third subset of the regions. Regions that are included in the third subset are associated with respective access patterns that satisfy at least one third criterion (e.g., most recent access time less than one hour ago). Three subsets of regions are described with  
20 respect to this example embodiment for illustrative purposes and are not intended to be limiting. It will be recognized that the example embodiment may include any number of subsets of regions corresponding to any number of respective criteria.

[0048] At step 404, optimization designations are assigned to the respective regions based on the respective access indicators that are assigned to the respective regions. Each  
25 optimization designation indicates an extent to which the respective region is to be optimized. In an example implementation, assignment module 702 assigns the optimization designations to the respective regions.

[0049] In an example embodiment, each optimization designation may be associated with a respective latency. For instance, if a first optimization designation indicates that a  
30 first type of optimization operation is to be performed with respect to a region to which the first optimization designation is assigned, a first latency that is expected with regard to the first type of optimization operation may be associated with the first optimization designation. If a second optimization designation indicates that a second type of optimization operation is to be performed with respect to a region to which the second

optimization designation is assigned, a second latency that is expected with regard to the second type of optimization operation may be associated with the second optimization designation, and so on.

5 [0050] In accordance with this example embodiment, a latency threshold may be associated with each access indicator. For instance, it may be determined that a system is capable of tolerating a first latency with respect to accessing regions to which respective access indicators that have a first value are assigned. It may be further determined that a system is capable of tolerating a second latency with respect to accessing regions to which respective access indicators that have a second value are assigned, and so on. The first  
10 latency may correspond to a first latency threshold; the second latency may correspond to a second latency threshold, and so on. Accordingly, each optimization designation may be assigned to the respective region based on a latency that is associated with the respective optimization designation being less than a latency threshold that is associated with the access indicator that is assigned to that region.

15 [0051] In another example embodiment, each optimization designation may be associated with a respective amount of system resource consumption. For instance, if a first optimization designation indicates that a first type of optimization operation is to be performed with respect to a region to which the first optimization designation is assigned, a first amount of system resource consumption that is expected with regard to the first type  
20 of optimization operation may be associated with the first optimization designation. If a second optimization designation indicates that a second type of optimization operation is to be performed with respect to a region to which the second optimization designation is assigned, a second amount of system resource consumption that is expected with regard to the second type of optimization operation may be associated with the second optimization  
25 designation, and so on.

[0052] In accordance with this example embodiment, a consumption threshold may be associated with each access indicator. For instance, it may be determined that a system is capable of tolerating a first amount of system resource consumption with respect to accessing regions to which respective access indicators that have a first value are assigned.  
30 It may be further determined that a system is capable of tolerating a second amount of system resource consumption with respect to accessing regions to which respective access indicators that have a second value are assigned, and so on. The first amount of system resource consumption may correspond to a first consumption threshold; the second amount of system resource consumption may correspond to a second consumption

threshold, and so on. Accordingly, each optimization designation may be assigned to the respective region based on an amount of system resource consumption that is associated with the respective optimization designation being less than a consumption threshold that is associated with the access indicator that is assigned to that region.

5 [0053] In yet another example embodiment, first optimization designations are assigned to a first subset of the regions based on first access indicators that correspond to a first range of values being assigned to the respective regions that are included in the first subset. Second optimization designations are assigned to a second subset of the regions based on second access indicators that correspond to a second range of values being  
10 assigned to the respective regions that are included in the second subset, and so on. In accordance with this example embodiment, the first optimization designations may indicate a first common extent to which the respective regions that are included in the first subset are to be optimized. The second optimization designations may indicate a second common extent to which the respective regions that are included in the second subset are  
15 to be optimized, and so on.

[0054] At step 406, each region is optimized to the extent that is indicated by the respective optimization designation that is assigned to that region. In an example implementation, optimization module 704 optimizes each region.

[0055] In some example embodiments, one or more steps 402, 404, and/or 406 of  
20 flowchart 400 may not be performed. Moreover, steps in addition to or in lieu of steps 402, 404, and/or 406 may be performed.

[0056] In an example embodiment, instead of performing step 402 of flowchart 400, the steps of flowchart 500 in FIG. 5 are performed. As shown in FIG. 5, the method of flowchart 500 begins at step 502. In step 502, access patterns that are associated with  
25 respective regions of a file are monitored using a file system filter driver. A file system filter driver intercepts requests that are targeted at a file system or another file system filter driver. By intercepting a request before it reaches its intended target, a file system filter driver can extend and/or replace functionality provided by the original target of the request. Examples of a file system filter driver include but are not limited to an anti-virus  
30 filter, a backup agent, an encryption module, etc. In an example implementation, access monitor 706 monitors the extent to which each of the plurality of regions is accessed.

[0057] At step 504, access indicators are assigned to the respective regions. Each access indicator corresponds to the access pattern that is associated with the respective region. In

an example implementation, assignment module 702 assigns the access indicators to the respective regions.

[0058] In another example embodiment, instead of performing steps 402 and 404 of flowchart 400, the steps of flowchart 600 in FIG. 6 are performed. As shown in FIG. 6, the method of flowchart 600 begins at step 602. In step 602, a virtualized storage file that includes multiple regions is mounted to provide a mounted virtualized storage file that includes data sequences that correspond to the respective regions. Each data sequence is included in a collection of one or more respective files. In an example implementation, mounting module 708 mounts the virtualized storage file to provide the mounted virtualized storage file.

[0059] At step 604, a disk filter is executed with respect to the mounted virtualized storage file to monitor an access pattern of each collection. A disk filter intercepts requests that are targeted at a virtual or physical disk. By intercepting requests before they reach their intended target, the disk filter can determine information (e.g., access patterns) regarding hosted files that are included in the intended target. In an example implementation, access monitor 706 executes the disk filter with respect to the mounted virtualized storage file.

[0060] At step 606, optimization designations are assigned to the respective regions based on the access patterns of the respective collections that include the corresponding data sequences. In an example implementation, assignment module 702 assigns the optimization designations to the respective regions.

[0061] It will be recognized that optimizer 700 may not include one or more of assignment module 702, optimization module 704, access monitor 706, and/or mounting module 708. Furthermore, optimizer 700 may include modules in addition to or in lieu of assignment module 702, optimization module 704, access monitor 706, and/or mounting module 708.

[0062] Assignment module 702, optimization module 704, access monitor 706, and mounting module 708 may be implemented in hardware, software, firmware, or any combination thereof. For example, assignment module 702, optimization module 704, access monitor 706, and/or mounting module 708 may be implemented as computer program code configured to be executed in one or more processors. In another example, assignment module 702, optimization module 704, access monitor 706, and/or mounting module 708 may be implemented as hardware logic/electrical circuitry.

[0063] FIG. 8 depicts an example computer 800 in which embodiments may be implemented. Any one or more of devices 100 and 200 shown in respective FIGS. 1 and 2 (or any one or more subcomponents thereof shown in FIG. 7) may be implemented using computer 800, including one or more features of computer 800 and/or alternative features.

5 Computer 800 may be a general-purpose computing device in the form of a conventional personal computer, a mobile computer, or a workstation, for example, or computer 800 may be a special purpose computing device. The description of computer 800 provided herein is provided for purposes of illustration, and is not intended to be limiting.

Embodiments may be implemented in further types of computer systems, as would be  
10 known to persons skilled in the relevant art(s).

[0064] As shown in FIG. 8, computer 800 includes a processing unit 802, a system memory 804, and a bus 806 that couples various system components including system memory 804 to processing unit 802. Bus 806 represents one or more of any of several types of bus structures, including a memory bus or memory controller, a peripheral bus, an  
15 accelerated graphics port, and a processor or local bus using any of a variety of bus architectures. System memory 804 includes read only memory (ROM) 808 and random access memory (RAM) 810. A basic input/output system 812 (BIOS) is stored in ROM 808.

[0065] Computer 800 also has one or more of the following drives: a hard disk drive  
20 814 for reading from and writing to a hard disk, a magnetic disk drive 816 for reading from or writing to a removable magnetic disk 818, and an optical disk drive 820 for reading from or writing to a removable optical disk 822 such as a CD ROM, DVD ROM, or other optical media. Hard disk drive 814, magnetic disk drive 816, and optical disk drive 820 are connected to bus 806 by a hard disk drive interface 824, a magnetic disk  
25 drive interface 826, and an optical drive interface 828, respectively. The drives and their associated computer-readable storage media provide nonvolatile storage of computer-readable instructions, data structures, program modules and other data for the computer. Although a hard disk, a removable magnetic disk and a removable optical disk are described, other types of computer-readable storage media can be used to store data, such  
30 as flash memory cards, digital video disks, random access memories (RAMs), read only memories (ROM), and the like.

[0066] A number of program modules may be stored on the hard disk, magnetic disk, optical disk, ROM, or RAM. These programs include an operating system 830, one or more application programs 832, other program modules 834, and program data 836.

Application programs 832 or program modules 834 may include, for example, computer program logic for implementing assignment module 702, optimization module 704, access monitor 706, mounting module 708, flowchart 300 (including any step of flowchart 300), flowchart 400 (including any step of flowchart 400), flowchart 500 (including any step of flowchart 500), and/or flowchart 600 (including any step of flowchart 600), as described herein.

[0067] A user may enter commands and information into the computer 800 through input devices such as keyboard 838 and pointing device 840. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 802 through a serial port interface 842 that is coupled to bus 806, but may be connected by other interfaces, such as a parallel port, game port, or a universal serial bus (USB).

[0068] A display device 844 (e.g., a monitor) is also connected to bus 806 via an interface, such as a video adapter 846. In addition to display device 844, computer 800 may include other peripheral output devices (not shown) such as speakers and printers.

[0069] Computer 800 is connected to a network 848 (e.g., the Internet) through a network interface or adapter 850, a modem 852, or other means for establishing communications over the network. Modem 852, which may be internal or external, is connected to bus 806 via serial port interface 842.

[0070] As used herein, the terms “computer program medium” and “computer-readable medium” are used to generally refer to media such as the hard disk associated with hard disk drive 814, removable magnetic disk 818, removable optical disk 822, as well as other media such as flash memory cards, digital video disks, random access memories (RAMs), read only memories (ROM), and the like.

[0071] As noted above, computer programs and modules (including application programs 832 and other program modules 834) may be stored on the hard disk, magnetic disk, optical disk, ROM, or RAM. Such computer programs may also be received via network interface 850 or serial port interface 842. Such computer programs, when executed or loaded by an application, enable computer 800 to implement features of embodiments discussed herein. Accordingly, such computer programs represent controllers of the computer 800.

[0072] Example embodiments are also directed to computer program products comprising software (e.g., computer-readable instructions) stored on any computer useable medium. Such software, when executed in one or more data processing devices, causes a

data processing device(s) to operate as described herein. Embodiments may employ any computer-useable or computer-readable medium, known now or in the future. Examples of computer-readable mediums include, but are not limited to storage devices such as RAM, hard drives, floppy disks, CD ROMs, DVD ROMs, zip disks, tapes, magnetic  
5 storage devices, optical storage devices, MEMS-based storage devices, nanotechnology-based storage devices, and the like.

### III. Conclusion

[0073] While various embodiments have been described above, it should be understood that they have been presented by way of example only, and not limitation. It will be  
10 apparent to persons skilled in the relevant art(s) that various changes in form and details can be made therein without departing from the spirit and scope of the invention. Thus, the breadth and scope of the present invention should not be limited by any of the above-described example embodiments, but should be defined only in accordance with the following claims and their equivalents.

15

## CLAIMS

1. A method comprising:  
assigning a plurality of optimization designations to a plurality of respective regions of a file based on a plurality of access patterns that is associated with the plurality of respective regions, each optimization designation indicating an extent to which the  
5 respective region is to be optimized; and  
optimizing each region to the extent that is indicated by the respective optimization designation that is assigned to that region.
2. The method of claim 1, wherein assigning the plurality of optimization  
10 designations comprises:  
assigning the plurality of optimization designations to the plurality of respective regions of a virtual storage file.
3. The method of claim 1, wherein assigning the plurality of optimization  
designations comprises:  
15 assigning the plurality of optimization designations to the plurality of respective regions of a database file.
4. The method of claim 1, wherein assigning the plurality of optimization  
designations comprises:  
assigning each optimization designation to the respective region further based on a  
20 latency that is associated with accessing that region.
5. The method of claim 1, wherein assigning the plurality of optimization  
designations comprises:  
assigning each optimization designation to the respective region further based on  
an amount of system resources that is utilized to access that region.
- 25 6. The method of claim 1, wherein assigning the plurality of optimization  
designations comprises:  
assigning each optimization designation to the respective region based on at least  
one of a number of times that the respective region is accessed or a number of times that  
the respective region is modified.
- 30 7. The method of claim 1, wherein assigning the plurality of optimization  
designations comprises:  
assigning each optimization designation to the respective region based on at least  
one of a frequency with which the respective region is accessed or a frequency with which  
the respective region is modified.

8. The method of claim 1, wherein assigning the plurality of optimization designations comprises:

assigning each optimization designation to the respective region based on at least one of a time at which the respective region is most recently accessed or a time at which the respective region is most recently modified.

9. The method of claim 1, wherein assigning the plurality of optimization designations comprises:

assigning each optimization designation to the respective region based on whether the respective region is accessed in response to mounting an operating system that uses the respective region.

10. The method of claim 1, wherein assigning the plurality of optimization designations comprises:

assigning each optimization designation to the respective region based on whether the respective region is accessed by a specified application.

11. The method of claim 1, further comprising:

monitoring the plurality of access patterns that is associated with the plurality of respective regions using a file system filter driver;

wherein assigning the plurality of optimization designations comprises:

assigning the plurality of optimization designations to the plurality of respective regions in response to monitoring the plurality of access patterns that is associated with the plurality of respective regions.

12. The method of claim 1, wherein the file is a virtualized storage file; wherein the method further comprises:

mounting the virtualized storage file to provide a mounted virtualized storage file that includes a plurality of data sequences that corresponds to the plurality of respective regions, each data sequence being included in a collection of one or more respective files, and

executing a disk filter with respect to the mounted virtualized storage file to monitor an access pattern of each collection; and

wherein assigning the plurality of optimization designations comprises:

assigning each optimization designation to the respective region of the virtualized storage file based on the access pattern of the respective collection that includes the corresponding data sequence.

13. The method of claim 1, wherein optimizing each region comprises:  
iteratively optimizing each region to the extent that is indicated by the respective optimization designation that is assigned to that region.

14. The method of claim 1, further comprising:

5 assigning a plurality of access indicators to the plurality of respective regions, each access indicator corresponding to the access pattern that is associated with the respective region;

wherein assigning the plurality of optimization designations comprises:

10 assigning each optimization designation to the respective region based on a latency that is associated with the respective optimization designation being less than a latency threshold that is associated with the access indicator that is assigned to that region.

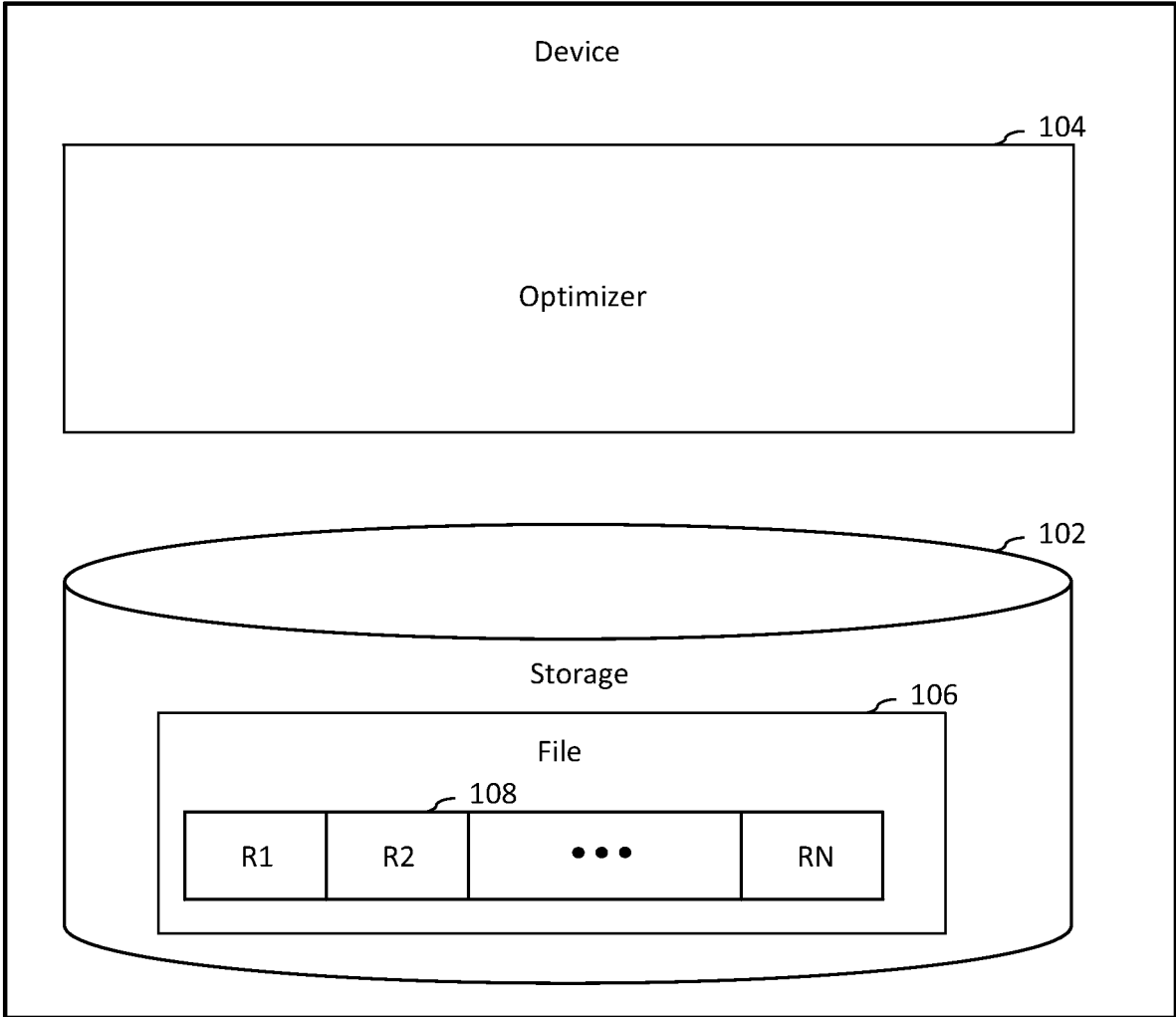
15. A system comprising:

15 an assignment module configured to assign a plurality of optimization designations to a plurality of respective regions of a file based on a plurality of access patterns that is associated with the plurality of respective regions, each optimization designation indicating an extent to which the respective region is to be optimized; and

an optimization module configured to optimize each region to the extent that is indicated by the respective optimization designation that is assigned to that region.

20

100  
↙



**FIG. 1**

200  
↙

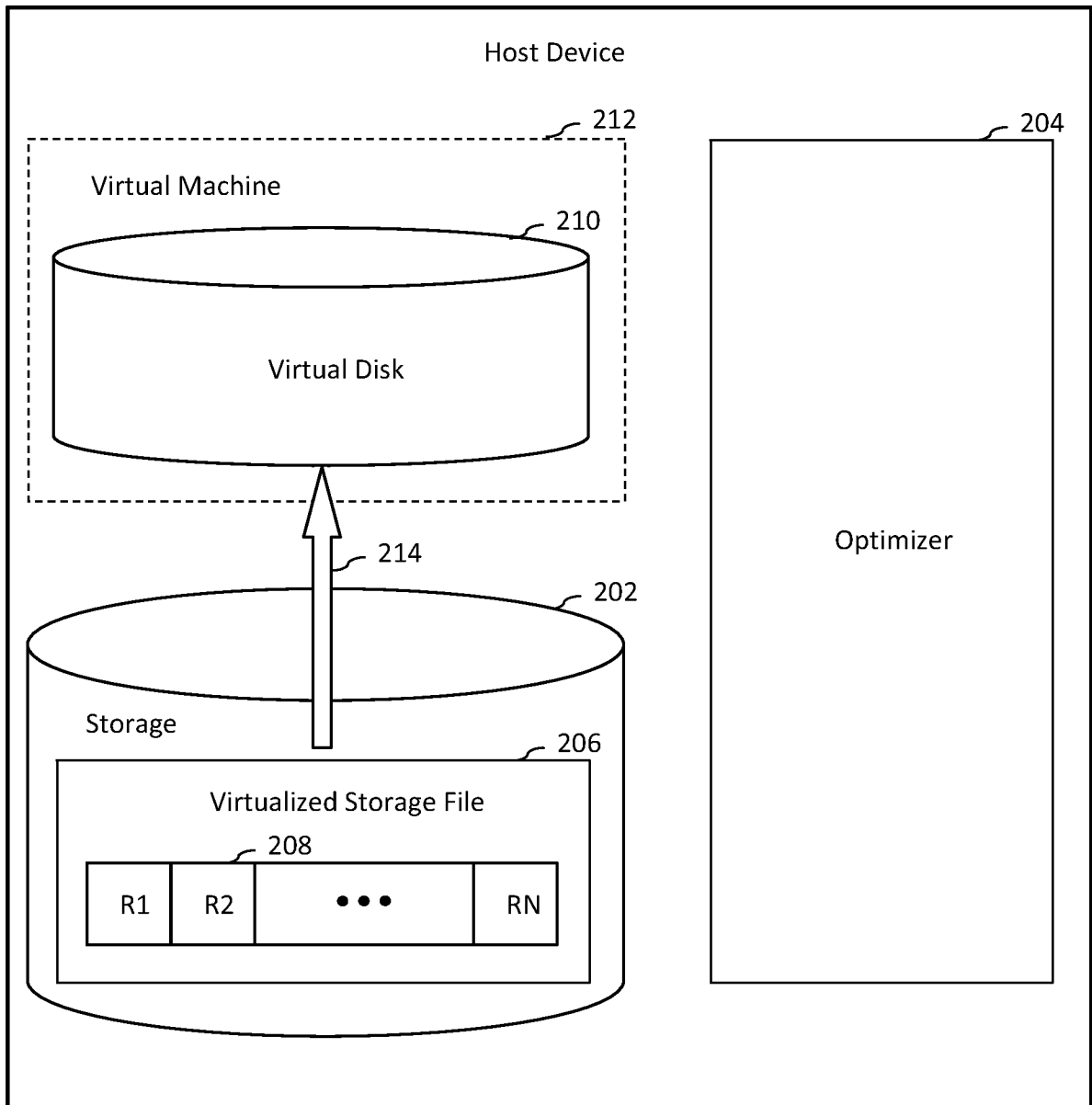
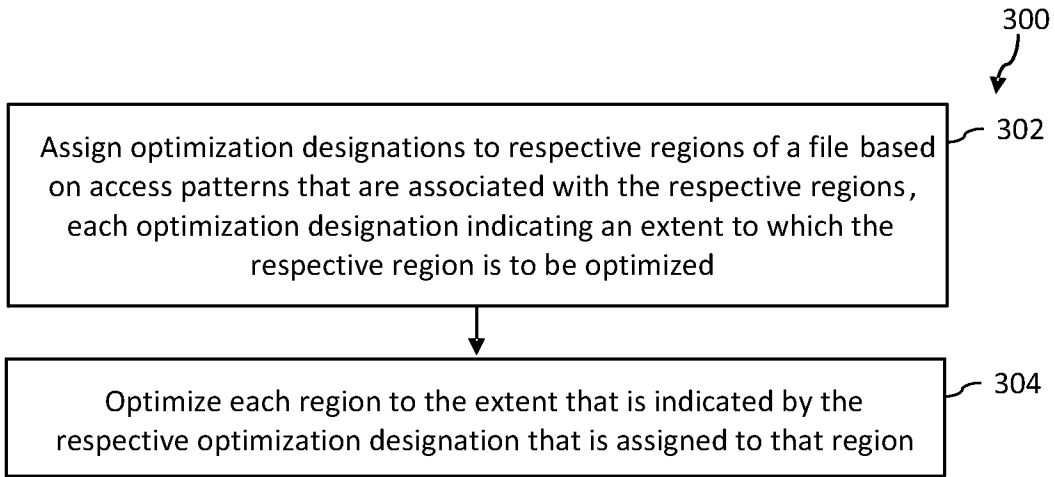
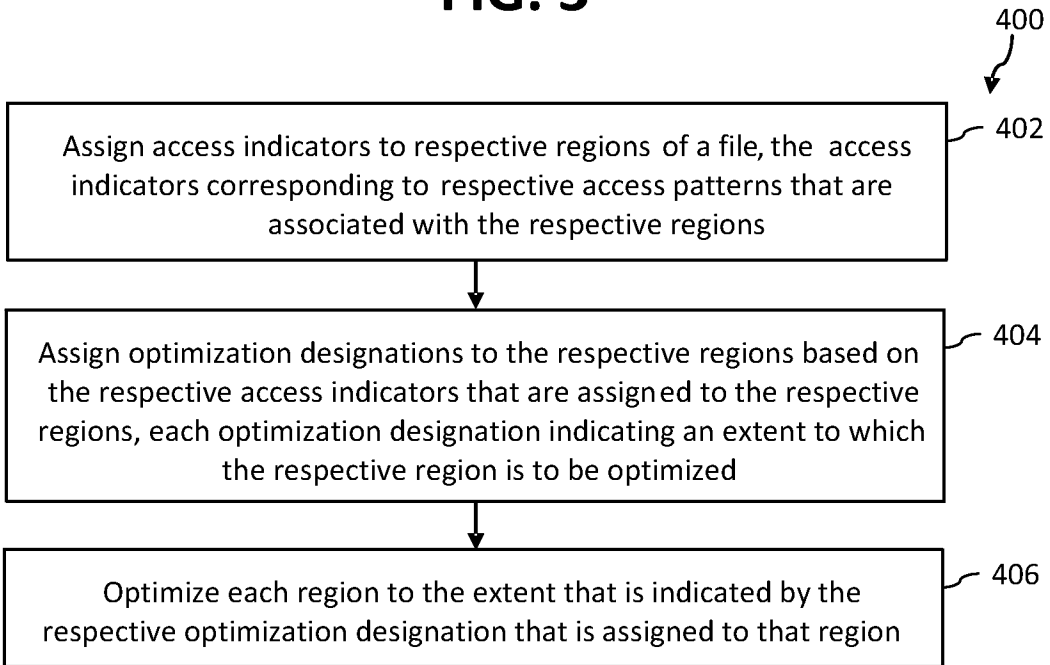


FIG. 2

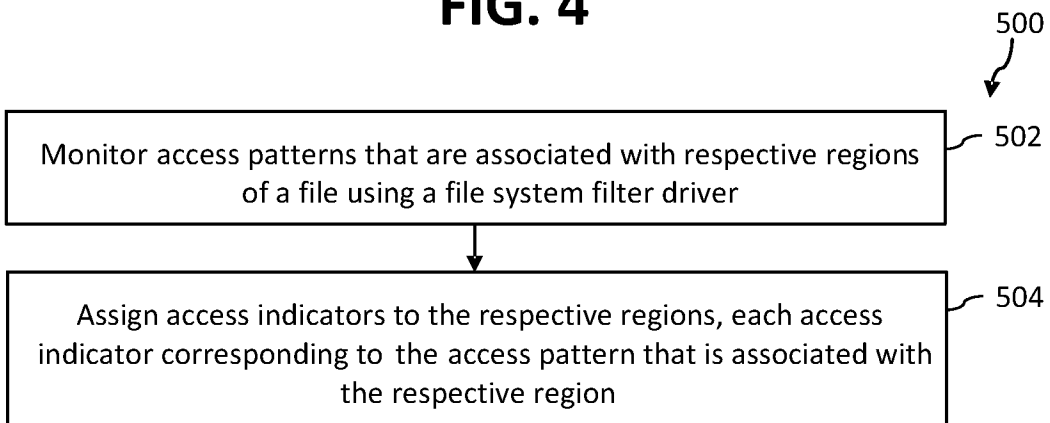
3/5



**FIG. 3**



**FIG. 4**



**FIG. 5**

4/5

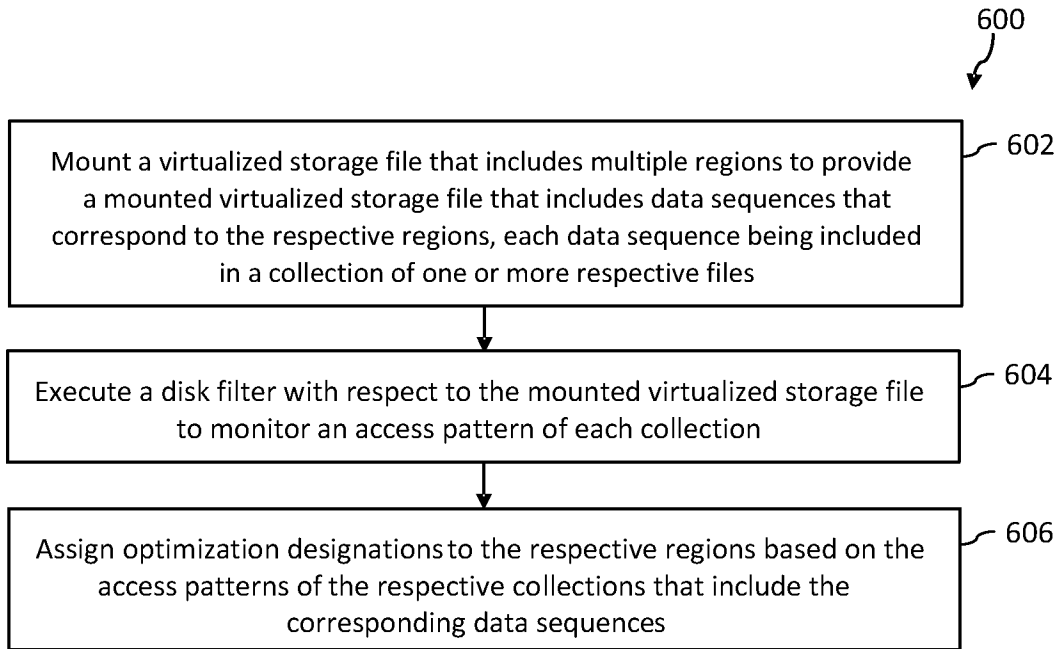


FIG. 6

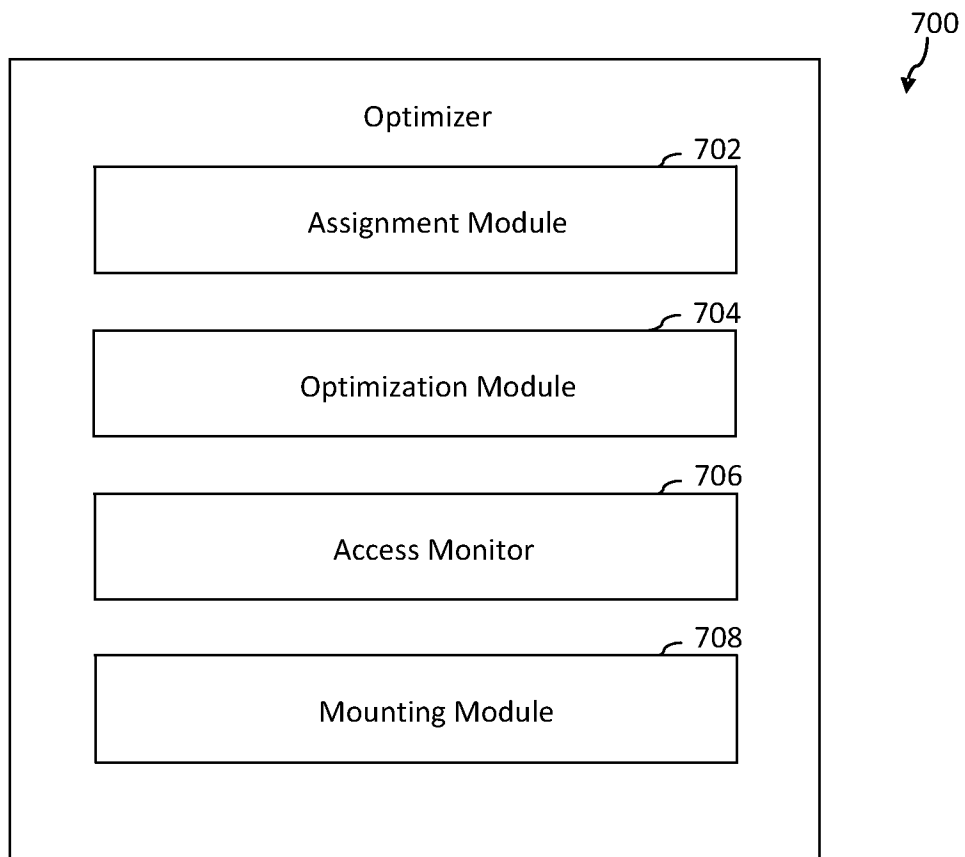


FIG. 7

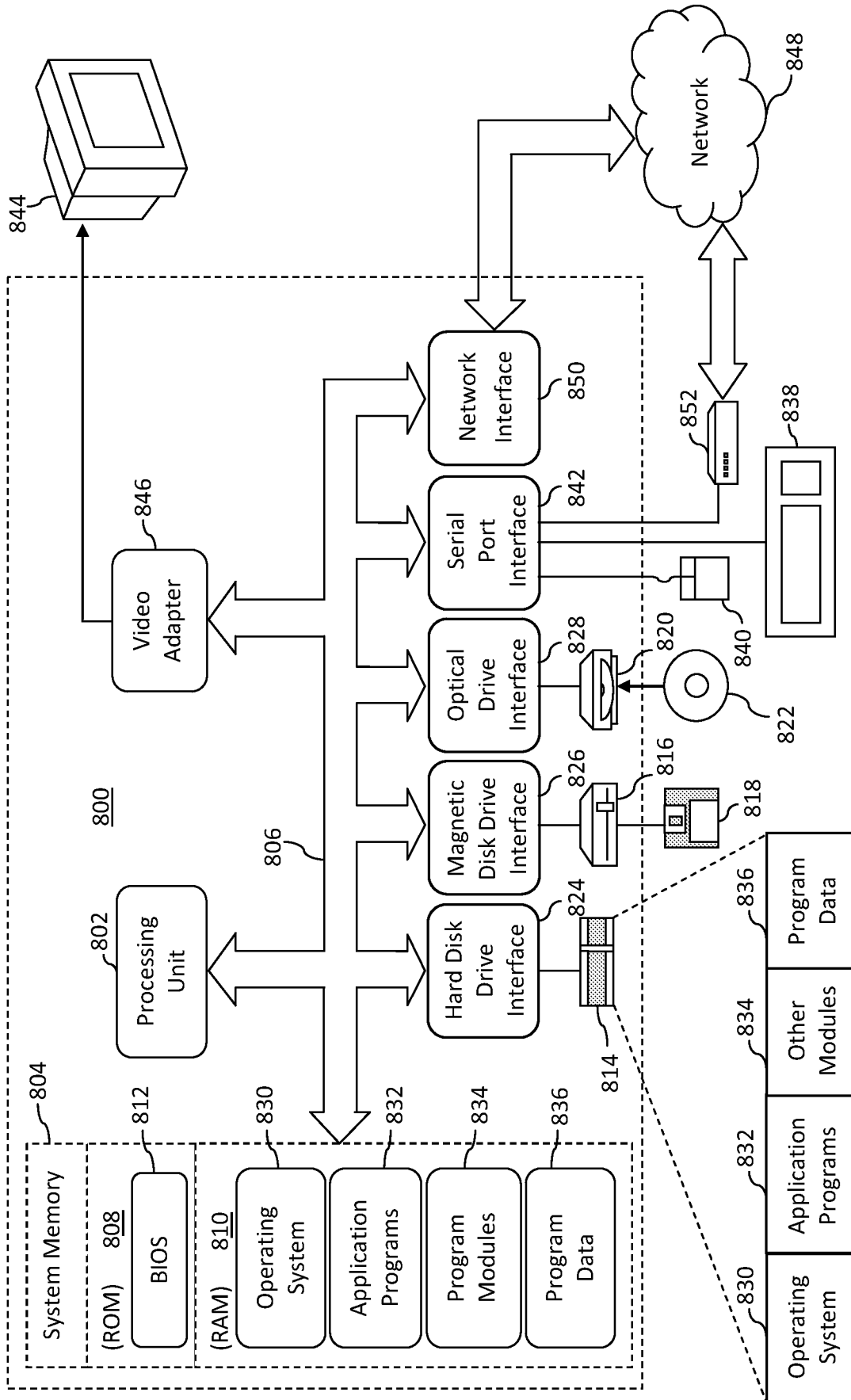


FIG. 8