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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
15 September 2016 (15.09.2016)

WIPO | PCT

(10) International Publication Number
WO 2016/144432 A1

(51) International Patent Classification:
G06F 17/30 (2006.01) *G06F 3/06* (2006.01)

(21) International Application Number:
PCT/US2016/014945

(22) International Filing Date:
26 January 2016 (26.01.2016)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
14/656,453 12 March 2015 (12.03.2015) US

(71) Applicant: **INTEL CORPORATION** [US/US]; 2200 Mission College Boulevard, Santa Clara, California 95054 (US).

(72) Inventors: **WANG, Ren**; 9137 NW Esson Ct., Portland, Oregon 97229 (US). **ZHAO, Weishuang**; 5817 Douglas Street, Apt 2, Pittsburgh, Pennsylvania 15217 (US). **SHEN, Wei**; 821 NE Eaglenest Ct., Hillsboro, Oregon 97124 (US). **MESNIER, Michael P.**; 51711 SW Old Portland Rd ., Apt. B, Scappoose, Oregon 97056 (US). **TAI, Tsung-Yuan C.**; 12709 NW Majestic Sequoia Way, Portland, Oregon 97229 (US). **ERGIN, Mesut A.**; 5651 NW 178th Ave., Portland, Oregon 97229 (US).

(74) Agents: **AUYEUNG, Al** et al.; 1211 SW 5th Avenue, Suite 1500-1900, Portland, Oregon 97204 (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, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

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

Published:

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

(54) **Title**: COMPUTING METHOD AND APPARATUS ASSOCIATED WITH CONTEXT-AWARE MANAGEMENT OF A FILE CACHE

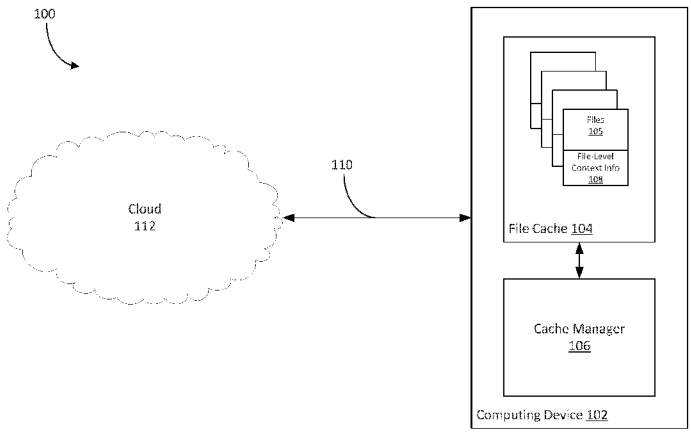

Figure 1

(57) **Abstract**: Computer-readable storage media, computing devices and methods associated with file cache management are discussed herein. In embodiments, a computing device may include a file cache and a file cache manager coupled with the file cache. The file cache manager may be configured to implement a context-aware eviction policy to identify a candidate file for deletion from the file cache, from a plurality of individual files contained within the file cache, based at least in part on file-level context information associated with the individual files. In embodiments, the file-level context information may include an indication of access recency and access frequency associated with the individual files. In such embodiments, identifying the candidate file for deletion from the file cache may be based, at least in part, on both the access recency and the access frequency of the individual files. Other embodiments may be described and/or claimed.

WO 2016/144432 A1

COMPUTING METHOD AND APPARATUS ASSOCIATED WITH CONTEXT-AWARE MANAGEMENT OF A FILE CACHE

RELATED APPLICATION

This application claims priority to U.S. Application No. 14/656,453, entitled "COMPUTING METHOD AND APPARATUS ASSOCIATED WITH CONTEXT-AWARE MANAGEMENT OF A FILE CACHE," filed March 12, 2015.

TECHNICAL FIELD

Embodiments of the present disclosure are related to the field of cache management, and in particular, to context-aware management of a file cache.

BACKGROUND

The background description provided herein is for the purpose of generally presenting the context of the disclosure. Unless otherwise indicated herein, the materials described in this section are not prior art to the claims in this application and are not admitted to be prior art by inclusion in this section.

Cloud storage is rapidly gaining popularity in recent years due to its manageability, convenience, and scalability. Accessing files from the remote cloud, especially when over slow wireless links, however, may incur long latency which may negatively impact users' experiences.

Caching the cloud storage content in a storage disk (for example a local hard drive) of a local client is one method of combatting network latency. Under the current state of the art, cloud storage services may employ relatively naïve methods for caching cloud content. An example of one such method is the "mirror" approach, used by a majority of cloud storage services (e.g., GoogleDrive, etc.). With this approach all cloud content may be cached, or "mirrored," on the local disk. Because of this, the mirror approach may only be practical when cloud storage capacity is smaller than local storage. A second example of such a naïve method involves manual selection of important files that should be cached on the local disk (e.g., "offline files" method of GoogleDrive). This approach is also designed to deal with offline case when network connection is not available or is experiencing delays from network latency. Cloud storage, however, is becoming larger and larger making these naïve approaches impractical.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 depicts an example computing environment in which a file cache manager that implements a context-aware eviction policy may be implemented, in accordance with

various embodiments of the present disclosure.

Figure 2 illustrates an example process flow for a computer-implemented context-aware eviction policy based file cache management, in accordance with various embodiments of the present disclosure.

Figure 3 is a schematic illustration of an example computing device, in accordance with various embodiments of the present disclosure.

Figure 4 illustrates an example non-transitory computer-readable storage medium having instructions configured to practice all or selected ones of the operations associated with the processes described above.

DETAILED DESCRIPTION OF ILLUSTRATIVE EMBODIMENTS

Methods, computer-readable media, and computing devices associated with context-aware management of a file cache are described herein. In the following detailed description, reference is made to the accompanying drawings which form a part hereof wherein like numerals designate like parts throughout, and in which is shown, by way of illustration, embodiments that may be practiced. It is to be understood that other embodiments may be utilized and structural or logical changes may be made without departing from the scope of the present disclosure. Therefore, the following detailed description is not to be taken in a limiting sense, and the scope of embodiments is defined by the appended claims and their equivalents.

Various operations may be described as multiple discrete actions or operations in turn, in a manner that is most helpful in understanding the claimed subject matter. However, the order of description should not be construed as to imply that these operations are necessarily order dependent. In particular, these operations may not be performed in the order of presentation. Operations described may be performed in a different order than the described embodiment. Various additional operations may be performed and/or described operations may be omitted in additional embodiments.

For the purposes of the present disclosure, the phrase "A and/or B" means (A), (B), or (A and B). For the purposes of the present disclosure, the phrase "A, B, and/or C" means (A), (B), (C), (A and B), (A and C), (B and C), or (A, B and C). The description may use the phrases "in an embodiment," or "in embodiments," which may each refer to one or more of the same or different embodiments. Furthermore, the terms "comprising," "including," "having," and the like, as used with respect to embodiments of the present disclosure, are synonymous.

Figure 1 depicts an example computing environment 100 in which a file cache manager 106 that implements a context-aware eviction policy may be practiced in accordance with various embodiments of the present disclosure. In embodiments, computing environment 100 may include a computing device 102 and a computing cloud 112, hereinafter referred to simply as cloud 112. Computing device 102 may include a file cache 104, and file cache manager 106 coupled with file cache 104 to manage file cache 104. File cache 104 may be configured to store local copies of files that have been recently accessed from one or more file repositories (e.g., cloud 112) by one or more applications on computing device 102 or client devices (not shown) serviced by computing device 102. In embodiments, file cache 104 may provide faster or more efficient access to the files 105 stored within the file cache than would be achievable by accessing the file from the repository. For example, as depicted, cloud 112 and computing device 102 may be coupled with one another by way of a network connection 110. Network connection 110 may represent any wired or wireless wide area network or any combination thereof. Because computing device 102 may need to retrieve files not stored locally in file cache 104 from cloud 112, the retrieval of any files from cloud 112 may be impacted by latency of network 110, or any components thereof, and/or latency of cloud 112 in processing requests for the files. Examples of client devices (not shown) may be client devices coupled with computing device 102 via a local or personal network. These client devices may include, but are not limited to, tablets, smartphones, or various wearable devices.

Because the storage capacity of cloud 112 may be larger than the storage capacity of file cache 104, only a selection of files (e.g., files 105) may be able to be stored, or cached, in file cache 104. As a result, occasionally, one or more of files 105 may need to be evicted or deleted, to make room for a new file that may need to be cached with files 105. Because of this, file cache manager 106 may be configured to evict, or delete, one or more of files 105, in response to a determination that file cache 104 has reached a threshold of available capacity. Such a threshold may be reached by the addition of one or more files to file cache 104 or through a reduction of capacity allocated to file cache 104. To accomplish this, file cache manager 106 may be configured to automatically evict, or delete, one or more of files 105 to make room in file cache 104 for the new file or account for the reduction in capacity. In evicting a file from file cache 104, it is important to keep, if possible, those files that have a higher likelihood of being accessed in an effort to ensure

that any latency in file accesses is reduced or minimized. As such, file cache manager 106 may be configured to implement a context-aware eviction policy, such as that depicted by process flow 200 of Figure 2, to identify a candidate file for deletion from file cache. Such a process flow may enable automatic identification of a candidate file based on file-level context information 108 associated with the individual files, as opposed to block-level or object-level context information which may lose important and useful file-level information. As used herein, automatic may refer to a process or procedure carried out autonomously, or without human intervention.

In embodiments, file-level context information 108 may include an indication of access recency and/or access frequency associated with the individual files of files 105. In such embodiments, file cache manager 106 may automatically identify a candidate file for deletion, or eviction, from file cache 104 based, at least in part, on the access recency and/or the access frequency of the individual files. Taking into account recency of file access may be beneficial as the most recently accessed files might be accessed again in a relatively short amount of time. Taking into account access frequency may be beneficial by reducing the weight given to a recently accessed file that has only ever been accessed a relatively small number of times, thereby giving the files that are accessed more frequently more weight in determining the candidate for eviction.

In some embodiments, the access recency and access frequency may be indicated by a frequency-recency value (FRV) associated with the individual files. Such a value may be based on any formula taking into account both recency and frequency. For example, in some embodiments, the FRV may be calculated as a weighted exponentially moving average (WEMA). Such a WEMA may be represented, for example, by the equation

$$FRV_k = \alpha * FRV_{k-1} + (1- \alpha)*sample,$$

where FRV_k may represent a current, or new, FRV;

FRV_{k-1} may represent an immediately preceding FRV;

α may represent a degree of weighting decrease to be applied to discount older observations; and

sample may be utilized to take into account access recency.

For example, sample may represent a value of 1 for an individual file that is currently accessed or -1 for an individual file that is not currently accessed. The weighting decrease for α may be, for example, a value chosen between 0 and 1, where a higher value

for α may discount older observations faster. By doing this, the current sample (represented as recency) is smoothed overtime, while the WEMA gives more weight to recent samples. Thus both frequency and recency information may be captured by this equation. It will be appreciated there are a multitude of ways in which the recency and frequency may be captured and that the above example is meant to be illustrative only.

In some embodiments, file-level context information 108 may include an indication of application-level relationships between files 105. In such embodiments, file cache manager 106 may be configured to identify the candidate file for deletion from the file cache based on these application-level relationships. For example, in some embodiments, these application level relationships may be generated from file access traces that indicate a sequence, or pattern, in which individual files are accessed based on execution of an application. In such embodiments, file cache manager 106 may be configured to eliminate for consideration those individual files that are indicated, by such a file access pattern, as being accessed after an individual file that was recently accessed by the application.

In some embodiments, for example, file cache manager 106 may be configured to observe file access patterns during execution of the application, and may be further configured to generate the application-level relationships between the individual files based at least in part on the observed file access patterns. These application level relationships may be maintained in file level context info 108 or may be stored in a central repository, such as one or more of the mass storage devices discussed below in reference to figure 3. In other embodiments, the application-level relationships may be maintained in a database in cloud 112, in such embodiments, file access patterns may be generated by one or more other computing devices of cloud 112 that are configured to observe file access patterns during execution of the application, and generate the application-level relationships between the individual files based at least in part on the observed file access patterns.

In some embodiments, the file-level context information 108 may include an indication of file size of the individual files. In such embodiments file cache manager 106 may be configured to identify the candidate file for deletion from file cache 104 based on the respective file sizes of files 105. For example, file cache manager 106 may be configured to assign a smaller weight to a file with a relatively large size in an effort to increase or maximize the number of files that may be able to be stored in file cache 104.

In such an example, a file with a relatively smaller size may be assigned a correspondingly greater weight.

In some embodiments, the file-level context information may include an indication of user preference of the individual files. For example, in some embodiments, the user may select one or more files that the user regularly accesses in an effort to ensure that these one or more files are cached and thereby quickly accessible. In, other embodiments, the user may assign weights, or priorities, to the files for the file cache manager to take into account when identifying the candidate file for eviction. In such embodiments, the file cache manager may be configured to identify the candidate file for deletion from the file cache based on the user preference of the individual files.

In some embodiments, file-level context information 108 may include, any combination of the above discussed file-level context information. Such other file-level context information. While depicted here as a cloud storage cache, it will be appreciated that file cache 104 may be any suitable type of file cache including, but not limited to a disk cache, web cache, etc.

Figure 2 illustrates an example process flow 200 for a computer-implemented context-aware eviction policy based file cache management, in accordance with various embodiments of the present disclosure. Process flow 200 may begin at block 202, after which either a file access operation 204 or a caching capacity adjustment 206 may be detected by a cache manager (e.g., file cache manager 106) of a file cache. As depicted, in the event that a file access operation 204 is detected, a frequency-recency value (FRV) for each file of an associated file cache, such as that discussed in greater detail above in reference to figure 1, may be updated, by the file cache manager, at block 208. In addition, in embodiments where file access patterns are observed and generated by a file cache manager, any application-level file relationships may be updated by the file cache manager at block 208.

At block 210 a determination may be made by the file cache manager as to whether one or more files need to be evicted. Such a determination may be made based upon whether a file cache has reached a threshold of capacity by the addition of one or more files to the file cache or through a reduction of capacity allocated to the file cache. If the file cache has not reached such a threshold of capacity, process flow 200 may continue to block 220, where the process may end. If, on the other hand, the file cache has reached such a threshold of capacity, the process may proceed to block 212. At block 212, an

initial or next candidate file for eviction may be selected based upon a combination of the respective FRVs and file sizes of the individual files. As discussed above, in reference to Figure 1, the file size may be taken into account by applying a weight, or priority, to the individual files based on the respective file size and an initial candidate file may be selected based upon this weight. For example, the file cache manager may assign a smaller weight to a file with a relatively large size in an effort to increase or maximize the number of files that may be able to be stored in the file cache. In such an example, a file with a relatively smaller size may be assigned a correspondingly greater weight. As such, in this example, a file with a relatively larger size, but a relatively high FRV may be selected as an initial or next candidate for eviction, based on the combination of the file size and FRV, while a relatively smaller file with a lower FRV may not.

Once an initial or next candidate file is selected at block 212, the process may proceed to block 214, where a determination may be made as to whether the candidate file is included in a user selected file list. If the file is in a user selected file list, the process may proceed to block 212 where a next candidate file may be selected. If the file is not in the user selected file list, the process may proceed to block 216. It will be appreciated that the user selected file list is merely an example mechanism by which a user preference may be implemented, other methods may be utilized, such as those discussed above in reference to figure 1. For example, in embodiments where the user has assigned a priority value to the files, the priority value may be considered in conjunction with the FRV and file size at block 212, or may be separately considered at block 214.

At block 216, a determination may be made as to whether the candidate file is closely related to a recently accessed file. As used here, a closely related file may be a file that is generally accessed, at least according to file access patterns of an application, in synchronization with an access to the candidate file. If the candidate file is closely related to a recently accessed file, the process may proceed to block 212 where a next candidate file may be selected. If the file is not closely related to a recently accessed file, the process may proceed to block 218, where the candidate file may be evicted from the file cache by the file cache manager. Finally, at block 220, the process may end.

Referring now to **Figure 3**, wherein an example computing device suitable to implement any of the computing devices discussed herein, in accordance with various embodiments, is illustrated. As shown, computing device 300 may include one or more processors or processor cores 302, and system memory 304. In embodiments, multiple

processor cores 302 may be disposed on one die. For the purpose of this application, including the claims, the terms "processor" and "processor cores" may be considered synonymous, unless the context clearly requires otherwise. Additionally, computing device 300 may include mass storage device(s) 306 (such as flash drive, diskette, hard drive, compact disc read-only memory (CD-ROM), and so forth), input/output (I/O) device(s) 308 (such as camera, display device, keyboard, cursor control, gyroscope, accelerometer, and so forth), and communication interfaces 310 (such as network interface cards, modems, and so forth). In embodiments, a display device may be touch screen sensitive and may include a display screen, one or more processors, storage medium, and communication elements. Further, it may be removably docked or undocked from a base platform having the keyboard. The elements may be coupled to each other via system bus 312, which may represent one or more buses. In the case of multiple buses, the buses may be bridged by one or more bus bridges (not shown).

Each of these elements may perform its conventional functions known in the art. In particular, system memory 304 and mass storage device(s) 306 may be employed to store a working copy and a permanent copy of programming instructions implementing the operations described earlier (e.g., but not limited to, operations associated with file cache manager 106 of Figure 1 or the context-aware eviction policy depicted by process flow 200 of Figure 2) generally referred to as computational logic 322, or to store a file cache, such as file cache 104 of Figure 1. The various operations may be implemented by assembler instructions supported by processor(s) 302 or high-level languages, such as, for example, C, that may be compiled into such instructions.

The permanent copy of the programming instructions may be placed into permanent mass storage device(s) 306 in the factory, or in the field, through, for example, a distribution medium (not shown), such as a compact disc (CD), or through communication interface 310 (from a distribution server (not shown)). That is, one or more distribution media having an implementation of any of the operations described earlier may be employed to distribute these components to various computing devices.

The number, capability, and/or capacity of these elements 302-312 may vary, depending on the intended use of example computing device 300, e.g., whether example computer 300 is a smartphone, tablet, ultrabook, laptop, desktop, server, etc. The constitutions of these elements 310-312 are otherwise known, and accordingly will not be further described.

Figure 4 illustrates an example non-transitory computer-readable storage medium having instructions configured to practice all or selected ones of the operations associated with the processes described above. As illustrated, non-transitory computer-readable storage medium 402 may include a number of programming instructions 404. Programming instructions 404 may be configured to enable a device, e.g., computing device 400, in response to execution of the programming instructions, to perform one or more operations of the processes described in reference to Figures 1-2, in particular, the operations associated with file cache manager 106 for managing file cache 104. In alternate embodiments, programming instructions 404 may be disposed on multiple non-transitory computer-readable storage media 402 instead. In still other embodiments, programming instructions 404 may be encoded in transitory computer-readable signals.

Referring back to Figure 3, for one embodiment, at least one of processors 302 may be packaged together with memory having computational logic 322 (in lieu of storing in mass storage device(s) 306) configured to perform one or more operations of the processes described with reference to Figures 1-2. For one embodiment, at least one of processors 302 may be packaged together with memory having computational logic 322 configured to practice aspects of the methods described in reference to Figures 1-2 to form a System in Package (SiP). For one embodiment, at least one of processors 302 may be integrated on the same die with memory having computational logic 322 configured to perform one or more operations of the processes described in reference to Figures 1-2. For one embodiment, at least one of processors 302 may be packaged together with memory having computational logic 322 configured to perform one or more operations of the process described in reference to Figures 1-2 to form a System on Chip (SoC). Such a SoC may be utilized in any suitable computing device.

Although certain embodiments have been illustrated and described herein for purposes of description, a wide variety of alternate and/or equivalent embodiments or implementations calculated to achieve the same purposes may be substituted for the embodiments shown and described without departing from the scope of the present disclosure. This application is intended to cover any adaptations or variations of the embodiments discussed herein. Therefore, it is manifestly intended that embodiments described herein be limited only by the claims.

Where the disclosure recites "a" or "a first" element or the equivalent thereof, such

disclosure includes one or more such elements, neither requiring nor excluding two or more such elements. Further, ordinal indicators (e.g., first, second, or third) for identified elements are used to distinguish between the elements, and do not indicate or imply a required or limited number of such elements, nor do they indicate a particular position or order of such elements unless otherwise specifically stated.

Embodiments of the disclosure can take the form of an entirely hardware embodiment, an entirely software embodiment or an embodiment containing both hardware and software elements. In various embodiments, software, may include, but is not limited to, firmware, resident software, microcode, and the like. Furthermore, the disclosure can take the form of a computer program product accessible from a computer-usable or computer-readable medium providing program code for use by or in connection with a computer or any instruction execution system. As used herein, module may refer to a software module, a hardware module, or any number or combination thereof.

As used herein, the term module includes logic that may be implemented in a hardware component or device, software or firmware that may be run or running on a processor, or a combination of processors. The modules may be distinct and independent components integrated by sharing or passing data, or the modules may be subcomponents of a single module, or be split among several modules. The components may be processes running on, or implemented on, a single compute node or distributed among a plurality of compute nodes running in parallel, concurrently, sequentially or a combination, as described more fully in conjunction with the flow diagrams in the figures.

For the purposes of this description, a computer-usable or computer-readable medium can be any apparatus that can contain, store, communicate, propagate, or transport the program for use by or in connection with the instruction execution system, apparatus, or device. The medium can be an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system (or apparatus or device) or a propagation medium. Examples of a computer-readable medium include a semiconductor or solid state memory, magnetic tape, a removable computer diskette, a random access memory (RAM), a read-only memory (ROM), a rigid magnetic disk and an optical disk. Current examples of optical disks include compact disk - read only memory (CD-ROM), compact disk - read/write (CD-R/W) and DVD.

Some non-limiting examples are:

Example 1 may be a computing device comprising: a file cache; and a file cache

manager coupled with the file cache. The file cache manager may be configured to implement a context-aware eviction policy to identify a candidate file for deletion from the file cache, from a plurality of individual files contained within the file cache, based at least in part on file-level context information associated with the individual files. The file-level context information may include an indication of access recency and access frequency associated with the individual files, and identify the candidate file for deletion from the file cache may be based, at least in part, on both the access recency and the access frequency of the individual files.

Example 2 may be example 1, wherein the file cache manager may be configured to use a frequency-recency value (FRV) associated with each individual file to determine access recency and access frequency, wherein the file cache manager may be configured to caclulate the FRV as a weighted exponentially moving average (WEMA) represented by the equation

$$FRV_{NEW} = \alpha * FRV_{OLD} + (1- \alpha)*sample,$$

where FRV_{NEW} represents a new FRV,

FRV_{OLD} represents an immediately preceding FRV,

α represents a weighting decrease between 0 and 1, and

sample represents a value of 1 for an individual file that is currently accessed or -1 for an individual file that is not currently accessed.

Example 3 may be example 1, wherein the file-level context information may further include an indication of application-level relationships between the individual files, and the file cache manager may be further configured to identify the candidate file for deletion from the file cache based at least in part on the application-level relationships.

Example 4 may be example 3, wherein the file cache manager may be further configured to observe file access patterns during execution of one or more applications; and generate the application-level relationships between the individual files based at least in part on the observed file access patterns.

Example 5 may be example 3, wherein the application-level context information may include a file access pattern of an application under execution on the computing device, and to identify a candidate file for deletion from the file cache, based at least in part on the application-level relationships, the file cache manager may be further configured to eliminate for consideration those individual files that are indicated, by the file access pattern, as being accessed after an individual file that was recently accessed by

the application.

Example 6 may be example 1, wherein the file-level context information may be further configured to include an indication of file size of the individual files, and wherein the file cache manager may be further configured to identify the candidate file for deletion from the file cache based at least in part on the respective file sizes of the individual files.

Example 7 may be example 1, wherein the file-level context information may further include an indication of user preference of the individual files, and the file cache manager may be further configured to identify the candidate file for deletion from the file cache based at least in part on the user preference of the individual files.

Example 8 may be any one of examples 1-7, wherein the file cache manager may be further configured to implement the context-aware eviction policy in response to a determination that the file cache has reached a threshold of available capacity.

Example 9 may be example 8, wherein the file cache is one of: a disk cache; a web cache; or a cloud storage cache.

Example 10 may be a computer-implemented method comprising: determining, by a file cache manager, that a file cache of a computing device has reached a threshold of available capacity; and implementing, by the file cache manager, in response to the determining, a context-aware eviction policy to identify a candidate file for deletion from the file cache based at least in part on file-level context information associated with individual files contained within the cache, wherein the file-level context information includes an indication of application-level relationships between the individual files, and wherein to identify a candidate file for deletion from the file cache is based at least in part further on the application-level relationships of the individual files.

Example 11 may be example 10, wherein the file-level context information may include an indication of access recency and access frequency associated with the individual files, and identifying the candidate file for deletion from the file cache may be based, at least in part, on both the access recency and the access frequency of the individual files.

Example 12 may be example 11, wherein implementing may include computing a frequency-recency value (FRV) associated with each individual file to indicate access recency and access frequency, wherein calculating FRV may include calculating a weighted exponentially moving average (WEMA) represented by the equation:

$$FRV_{NEW} = \alpha * FRV_{OLD} + (1- \alpha)*sample,$$

where FRV_{NEW} represents a new FRV,

FRV_{OLD} represents an immediately preceding FRV,

α represents a weighting decrease between 0 and 1, and

sample represents a value of 1 for an individual file that is currently accessed or -1 for an individual file that is not currently accessed.

Example 13 may be example 11, further comprising: observing, by the file cache manager, file access patterns during execution of one or more applications; and generating the application-level relationships between the individual files based at least in part on the observed file access patterns.

Example 14 may be example 12, wherein the application-level context information may include a file access pattern of an application under execution on the computing device, and wherein identifying a candidate file for deletion from the file cache, based at least in part on the application-level relationships, may further include eliminating for consideration those individual files that are indicated, by the file access pattern, as being accessed after an individual file that was recently accessed by the application.

Example 15 may be example 10, wherein the file-level context information may further include an indication of file size of the individual files, and wherein identifying the candidate file for deletion from the file cache may be based at least in part on the respective file sizes of the individual files.

Example 16 may be any one of examples 10-15, wherein the file-level context information may further include an indication of user preference of the individual files, and wherein identifying the candidate file for deletion from the file cache may be based at least in part on the user preference of the individual files.

Example 17 may be example 16, wherein the file cache is one of: a disk cache; a web cache; or a cloud storage cache.

Example 18 may be one or more computer-readable media having instructions stored thereon which, in response to execution by a computing device, provide the computing device with a file cache manager to: determine that a file cache of the computing device has reached a threshold of available capacity; and implement, in response to the determine, a context-aware eviction policy to identify a candidate file for deletion from the file cache based at least in part on file-level context information associated with individual files contained within the cache, wherein the file-level context information includes an indication of access recency and access frequency associated with

the individual files, and wherein to identify the candidate file for deletion from the file cache is based, at least in part, on both the access recency and the access frequency of the individual files.

Example 19 may be example 18, wherein to implement may include to calculate a frequency-recency value (FRV) associated with each individual file to indicate access recency and access frequency, wherein to calculate may include to calculate the FRV as a weighted exponentially moving average (WEMA) represented by the equation:

$FRV_{NEW} = \alpha * FRV_{OLD} + (1- \alpha)*sample$,

where FRV_{NEW} represents a new FRV,

FRV_{OLD} represents an immediately preceding FRV,

α represents a weighting decrease between 0 and 1, and

sample represents a value of 1 for an individual file that is currently accessed or -1 for an individual file that is not currently accessed.

Example 20 may be example 18, wherein the file-level context information may further include an indication of application-level relationships between the individual files, and wherein the file cache manager may be further configured to identify the candidate file for deletion from the file cache based at least in part on the application-level relationships.

Example 21 may be example 20, wherein the file cache manager may be further configured to observe file access patterns during execution of one or more applications; and generate the application-level relationships between the individual files based at least in part on the observed file access patterns.

Example 22 may be example 20, wherein the application-level context information may further include a file access pattern of an application under execution on the computing device, and wherein to identify a candidate file for deletion from the file cache, based at least in part on the application-level relationships, the file cache manager may be further configured to eliminate for consideration those individual files that are indicated, by the file access pattern, as being accessed after an individual file that was recently accessed by the application. \

Example 23 may be example 18, wherein the file-level context information may further include an indication of file size of the individual files, and wherein the file cache manager may be further configured to identify the candidate file for deletion from the file cache based at least in part on the respective file sizes of the individual files.

Example 24 may be any one of examples 18-23, wherein the file-level context information may further include an indication of user preference of the individual files, and wherein the file cache manager may be further configured to identify the candidate file for deletion from the file cache based at least in part on the user preference of the individual files.

Example 25 may be example 24, wherein the file cache is one of: a disk cache; a web cache; or a cloud storage cache.

Example 26 may be a computing device comprising: means for determining that a file cache of a computing device has reached a threshold of available capacity; and means for implementing, in response to the determining, a context-aware eviction policy to identify a candidate file for deletion from the file cache based at least in part on file-level context information associated with individual files contained within the cache, wherein the file-level context information includes an indication of application-level relationships between the individual files, and wherein means for implementing includes means for identifying a candidate file for deletion from the file cache is based at least in part further on the application-level relationships of the individual files.

Example 27 may be example 26, wherein the file-level context information may include an indication of access recency and access frequency associated with the individual files, and wherein means for identifying may comprise means for identifying the candidate file for deletion from the file cache is based, at least in part, on both the access recency and the access frequency of the individual files.

Example 28 may be example 27, wherein means for identifying may comprise means for calculating a frequency-recency value (FRV) associated with each individual file to indicate access recency and access frequency, wherein the FRV may be calculated as a weighted exponentially moving average (WEMA) represented by the equation:

$FRV_{NEW} = \alpha * FRV_{OLD} + (1- \alpha)*sample$,

where FRV_{NEW} represents a new FRV,

FRV_{OLD} represents an immediately preceding FRV,

α represents a weighting decrease between 0 and 1, and

sample represents a value of 1 for an individual file that is currently accessed or -1 for an individual file that is not currently accessed.

Example 29 may be example 26, further comprising: means for observing file access patterns during execution of one or more applications; and means for generating the

application-level relationships between the individual files based at least in part on the observed file access patterns.

Example 30 may be example 29, wherein the application-level context information may include a file access pattern of an application under execution on the computing device, and wherein means for identifying a candidate file for deletion from the file cache, based at least in part on the application-level relationships, may further include means for eliminating for consideration those individual files that are indicated, by the file access pattern, as being accessed after an individual file that was recently accessed by the application.

Example 31 may be example 26, wherein the file-level context information may further include an indication of file size of the individual files, and wherein means for identifying may comprise means for identifying the candidate file for deletion from the file cache, based at least in part on the respective file sizes of the individual files.

Example 32 may be any one of examples 26-31, wherein the file-level context information may further include an indication of user preference of the individual files, and wherein means for identifying may comprise means for identifying the candidate file for deletion from the file cache, based at least in part on the user preference of the individual files.

Example 33 may be example 32, wherein the file cache is one of: a disk cache; a web cache; or a cloud storage cache.

Although specific embodiments have been illustrated and described herein, it will be appreciated by those of ordinary skill in the art that a wide variety of alternate and/or equivalent implementations may be substituted for the specific embodiments shown and described, without departing from the scope of the embodiments of the disclosure. This application is intended to cover any adaptations or variations of the embodiments discussed herein. Therefore, it is manifestly intended that the embodiments of the disclosure be limited only by the claims and the equivalents thereof.

Claims

What is claimed is:

1. A computing device comprising:

a file cache; and

a file cache manager coupled with the file cache, to implement a context-aware eviction policy to identify a candidate file for deletion from the file cache, from a plurality of individual files contained within the file cache, based on file-level context information associated with the individual files, wherein the file-level context information includes an indication of access recency and access frequency associated with the individual files, and wherein to identify the candidate file for deletion from the file cache is based, at least in part, on both the access recency and the access frequency of the individual files.

2. The computing device of claim 1, wherein the file cache manager is to use a frequency-recency value (FRV) associated with each individual file to determine access recency and access frequency, wherein the cache manage is to caclulate the FRV as a weighted exponentially moving average (WEMA) represented by the equation

$FRV_{NEW} = \alpha * FRV_{OLD} + (1- \alpha)*sample$,

where FRV_{NEW} represents a new FRV,

FRV_{OLD} represents an immediately preceding FRV,

α represents a weighting decrease between 0 and 1, and

sample represents a value of 1 for an individual file that is currently accessed or -1 for an individual file that is not currently accessed.

3. The computing device of claim 1, wherein the file-level context information further includes an indication of application-level relationships between the individual files, and wherein the file cache manager is further to identify the candidate file for deletion from the file cache based on the application-level relationships.

4. The computing device of claim 3, wherein the file cache manager is further to observe file access patterns during execution of one or more applications; and generate the application-level relationships between the individual files based at least in part on the observed file access patterns.

5. The computing device of claim 3, wherein the application-level context

information includes a file access pattern of an application under execution on the computing device, and wherein to identify a candidate file for deletion from the file cache, based on the application-level relationships, the file cache manager is further to eliminate for consideration those individual files that are indicated, by the file access pattern, as being accessed after an individual file that was recently accessed by the application.

6. The computing device of claim 1, wherein the file-level context information further includes an indication of file size of the individual files, and wherein the file cache manager is further to identify the candidate file for deletion from the file cache based on the respective file sizes of the individual files.

7. The computing device of claim 1, wherein the file-level context information further includes an indication of user preference of the individual files, and wherein the file cache manager is further to identify the candidate file for deletion from the file cache based on the user preference of the individual files.

8. The computing device of any one of claims 1-7, wherein the file cache manager is to implement the context-aware eviction policy in response to a determination that the file cache has reached a threshold of available capacity.

9. The computing device of claim 8, wherein the file cache is one of:

a disk cache;

a web cache; or

a cloud storage cache.

10. A computer-implemented method comprising:

determining, by a file cache manager, that a file cache of a computing device has reached a threshold of available capacity; and

implementing, by the file cache manager, in response to the determining, a context-aware eviction policy to identify a candidate file for deletion from the file cache based on file-level context information associated with individual files contained within the cache, wherein the file-level context information includes an indication of application-level relationships between the individual files, and wherein to identify a candidate file for deletion from the file cache is based further on the application-level relationships of the individual files.

11. The computer-implemented method of claim 10, wherein the file-level context information includes an indication of access recency and access frequency

associated with the individual files, and wherein identifying the candidate file for deletion from the file cache is based, at least in part, on both the access recency and the access frequency of the individual files.

12. The computer-implemented method of claim 11, wherein implementing includes computing a frequency-recency value (FRV) associated with each individual file to indicate access recency and access frequency, wherein calculating FRV includes calculating a weighted exponentially moving average (WEMA) represented by the equation:

$FRV_{NEW} = \alpha * FRV_{OLD} + (1- \alpha)*sample$,

where FRV_{NEW} represents a new FRV,

FRV_{OLD} represents an immediately preceding FRV,

α represents a weighting decrease between 0 and 1, and s

ample represents a value of 1 for an individual file that is currently accessed or -1 for an individual file that is not currently accessed.

13. The computer-implemented method of claim 11, further comprising: observing, by the file cache manager, file access patterns during execution of one or more applications; and generating the application-level relationships between the individual files based at least in part on the observed file access patterns.

14. The computer-implemented method of claim 12, wherein the application-level context information includes a file access pattern of an application under execution on the computing device, and wherein identifying a candidate file for deletion from the file cache, based on the application-level relationships, further includes eliminating for consideration those individual files that are indicated, by the file access pattern, as being accessed after an individual file that was recently accessed by the application.

15. The computer-implemented method of claim 10, wherein the file-level context information further includes an indication of file size of the individual files, and wherein identifying the candidate file for deletion from the file cache is based on the respective file sizes of the individual files.

16. The computer-implemented method of claim 10, wherein the file-level context information further includes an indication of user preference of the individual files, and wherein identifying the candidate file for deletion from the file cache is based on the user preference of the individual files.

17. The computer-implemented method of claim 10, wherein the file cache is one of:

a disk cache;

a web cache; or

a cloud storage cache.

18. One or more computer-readable media having instructions stored thereon which, in response to execution by a computing device, provide the computing device with a file cache manager to practice one of the methods of claims 10-17.

19. A computing device comprising:

means for determining that a file cache of a computing device has reached a threshold of available capacity; and

means for implementing, in response to the determining, a context-aware eviction policy to identify a candidate file for deletion from the file cache based on file-level context information associated with individual files contained within the cache, wherein the file-level context information includes an indication of application-level relationships between the individual files, and wherein means for implementing includes means for identifying a candidate file for deletion from the file cache is based further on the application-level relationships of the individual files.

20. The computing device of claim 19, wherein the file-level context information includes an indication of access recency and access frequency associated with the individual files, and wherein means for identifying comprises means for identifying the candidate file for deletion from the file cache is based, at least in part, on both the access recency and the access frequency of the individual files.

21. The computing device of claim 20, wherein means for identifying comprises means for calculating a frequency-recency value (FRV) associated with each individual file to indicate access recency and access frequency, wherein the FRV is calculated as a weighted exponentially moving average (WEMA) represented by the equation:

$FRV_{NEW} = \alpha * FRV_{OLD} + (1- \alpha)*sample$,

where FRV_{NEW} represents a new FRV,

FRV_{OLD} represents an immediately preceding FRV,

α represents a weighting decrease between 0 and 1, and

sample represents a value of 1 for an individual file that is currently accessed or -1 for an individual file that is not currently accessed.

22. The computing device of claim 19, further comprising: means for observing file access patterns during execution of one or more applications; and means for generating the application-level relationships between the individual files based at least in part on the observed file access patterns.

23. The computing device of claim 22, wherein the application-level context information includes a file access pattern of an application under execution on the computing device, and wherein means for identifying a candidate file for deletion from the file cache, based on the application-level relationships, further includes means for eliminating for consideration those individual files that are indicated, by the file access pattern, as being accessed after an individual file that was recently accessed by the application.

24. The computing device of claim 19, wherein the file-level context information further includes an indication of file size of the individual files, and wherein means for identifying comprises means for identifying the candidate file for deletion from the file cache, based on the respective file sizes of the individual files.

25. The computing device of any one of claims 19-24, wherein the file-level context information further includes an indication of user preference of the individual files, and wherein means for identifying comprises means for identifying the candidate file for deletion from the file cache, based on the user preference of the individual files.

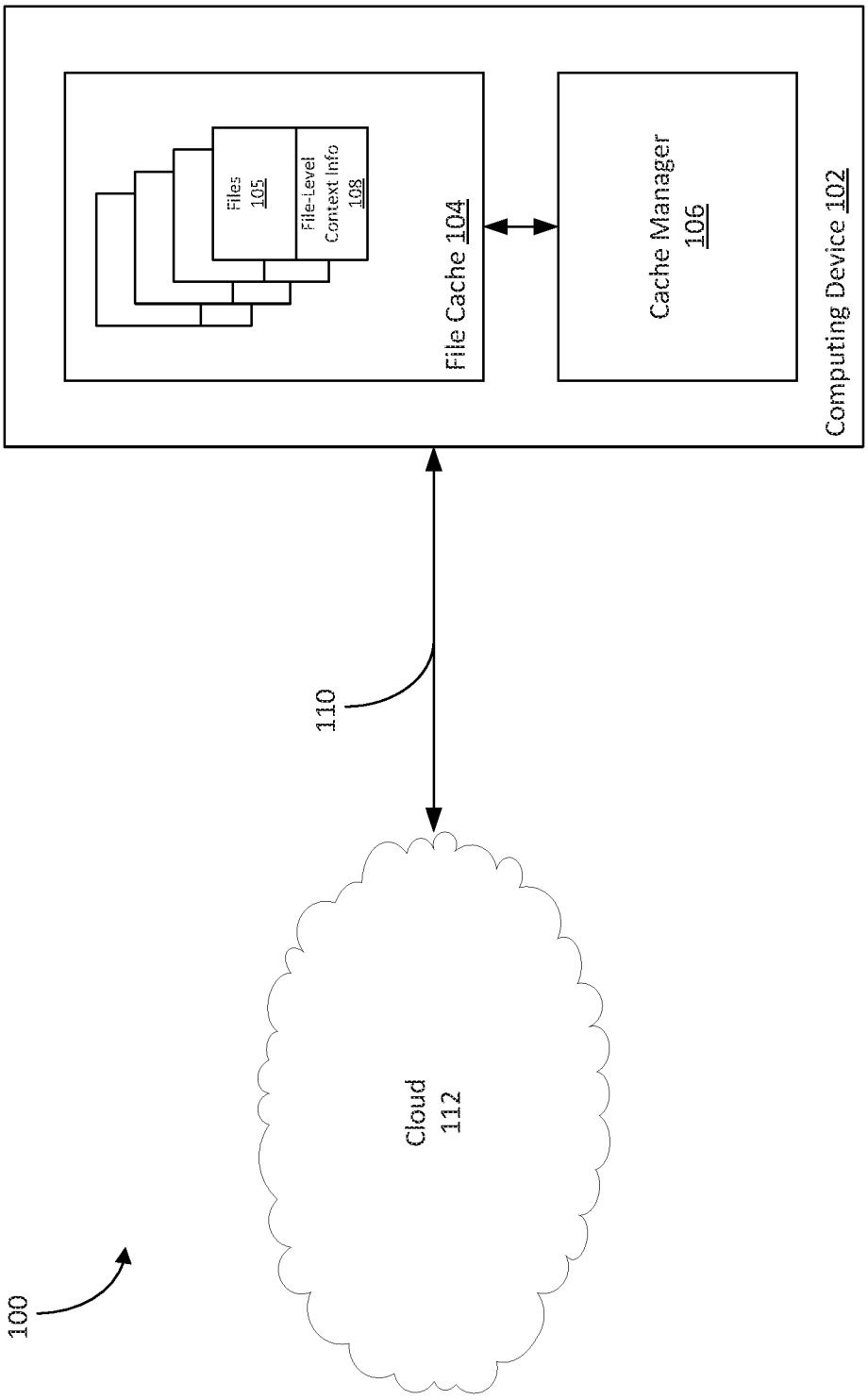
Computing Device 102
Cache Manager 106
File Cache 104
Files 105
File-Level Context Info 108
110
Cloud 112
100

Figure 1

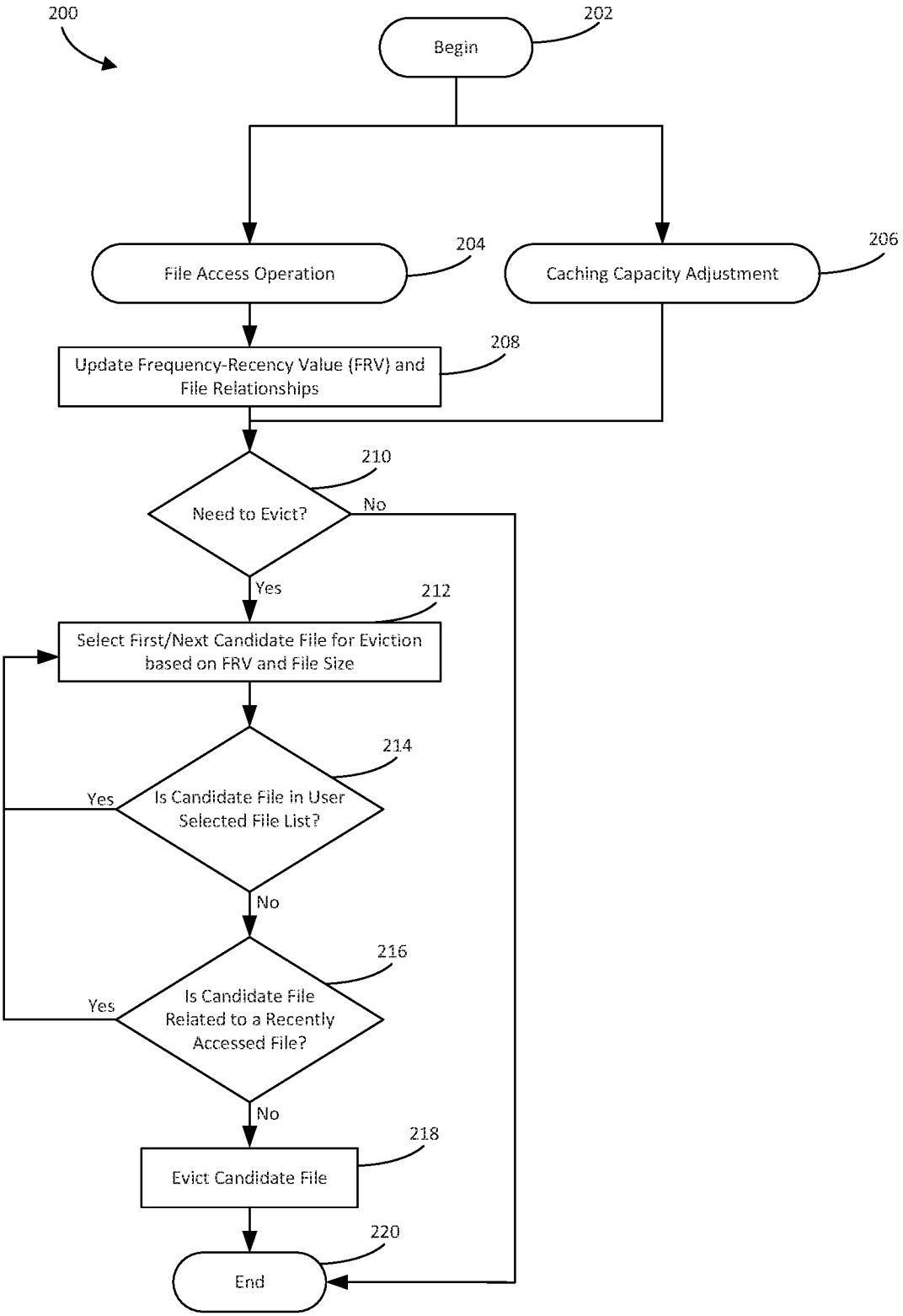
200
202
Begin
204
File Access Operation
206
Caching Capacity Adjustment
208
Update Frequency-Recency Value (FRV) and File Relationships
210
Need to Evict?
No
Yes
212
Select First/Next Candidate File for Eviction based on FRV and File Size
214
Is Candidate File in User Selected File List?
Yes
No
216
Is Candidate File Related to a Recently Accessed File?
Yes
No
218
Evict Candidate File
220
End

Figure 2

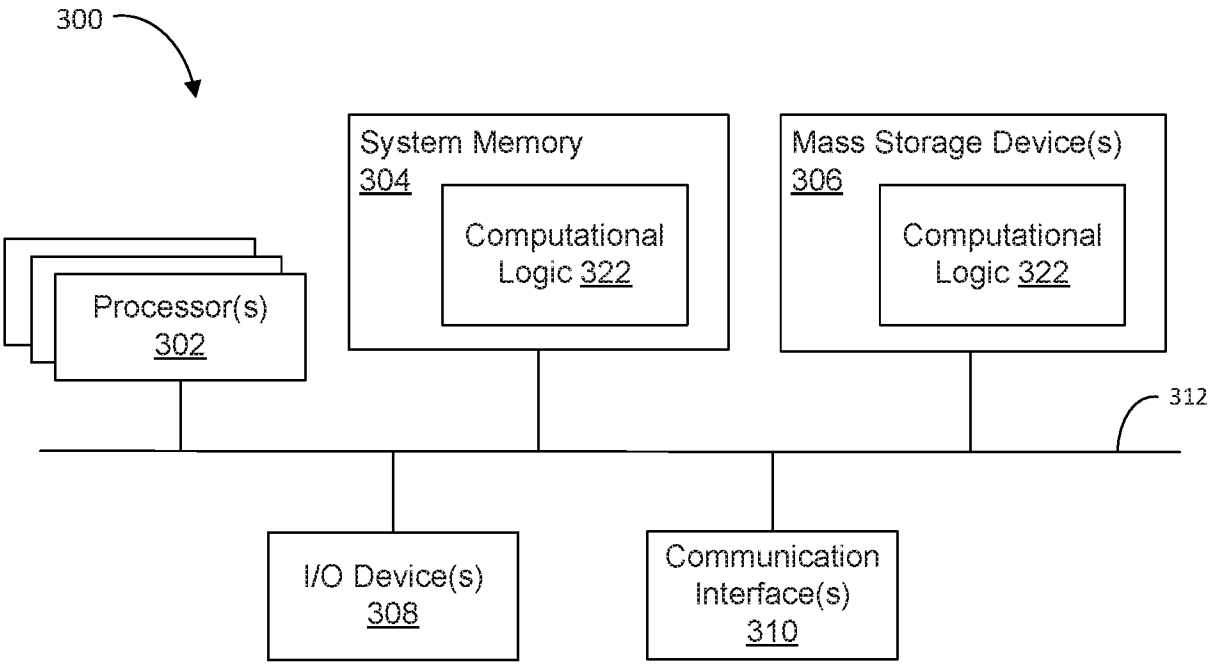
300
System Memory
304
Computational
Logic 322
Mass Storage Device(s)
306
Computational
Logic 322
Processor(s)
302
312
I/O Device(s)
308
Communication
Interface(s)
310

Figure 3

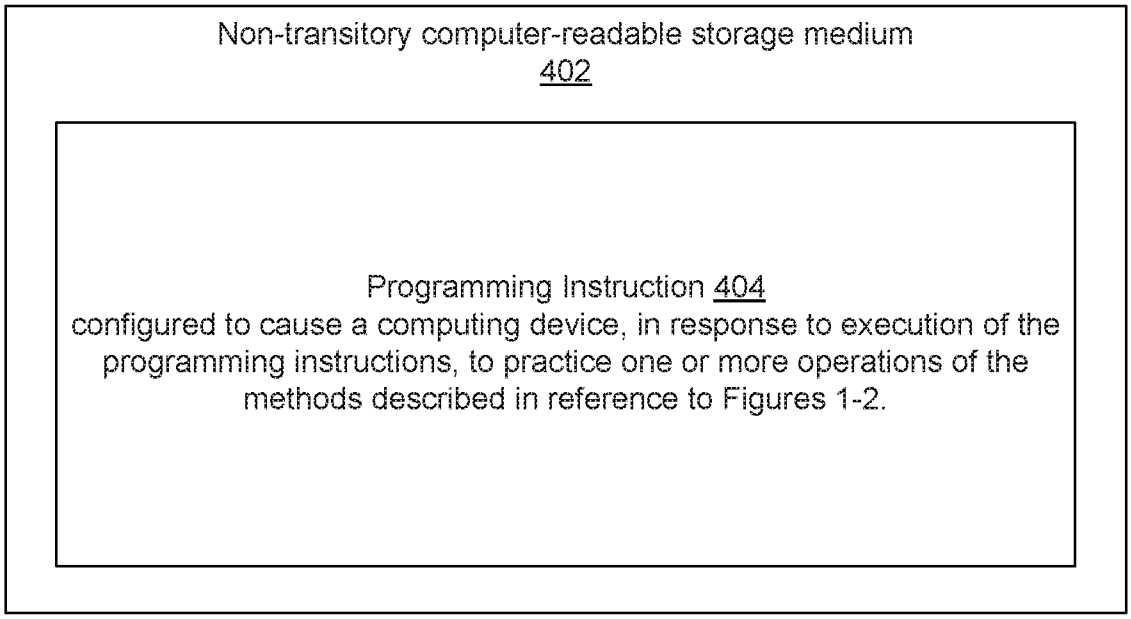
Non-transitory computer-readable storage medium
402
Programming Instruction 404
configured to cause a computing device, in response to execution of the programming instructions, to practice one or more operations of the methods described in reference to Figures 1-2.

Figure 4

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2016/014945

A. CLASSIFICATION OF SUBJECT MATTER

G06F 17/30(2006.01)i, G06F 3/06(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 17/30; G06F 12/12; H04L 29/08; G06F 12/00; G06F 13/00; G06F 12/08; G06F 9/45; G06F 3/06

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: evict, cache, management, access, recency, frequency, file, size, level, and similar terms.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2009-0100224 A1 (WENGUANG WANG) 16 April 2009 See paragraphs [0014]-[0015], [0022]-[0025], [0029], [0035] and [0038]; claim 10; and figures 1-2.	1-2,6,8-9
Y		3-5,7,10-25
Y	US 2012-0210068 A1 (VIKRAM JOSHI et al.) 16 August 2012 See paragraphs [0243] and [0256]; and figures 31A and 32.	3-5,10-25
Y	US 2014-0337458 A1 (DROPBOX, INC.) 13 November 2014 See paragraph [0054]; claim 12; and figure 2.	7,16,25
A	US 2007-0113014 A1 (SVETOSLAV MANOLOV et al.) 17 May 2007 See paragraphs [0021]-[0024] and figure 3A.	1-25
A	WO 2006-071792 A2 (INTEL CORPORATION) 06 July 2006 See paragraphs [0037]-[0044] and figure 3a.	1-25

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

☒ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents,such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
13 June 2016 (13.06.2016)	**13 June 2016 (13.06.2016)**

Name and mailing address of the ISA/KR
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon, 35208, Republic of Korea
Facsimile No. +82-42-481-8578

Authorized officer
NHO, Ji Myong
Telephone No. +82-42-481-8528

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2016/014945

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2009-0100224 A1	16/04/2009	US 8161240 B2	17/04/2012
US 2012-0210068 A1	16/08/2012	US 2012-0210066 A1	16/08/2012
		US 8996807 B2	31/03/2015
		US 9003104 B2	07/04/2015
		WO 2013-023090 A2	14/02/2013
		WO 2013-023090 A3	25/04/2013
US 2014-0337458 A1	13/11/2014	None	
US 2007-0113014 A1	17/05/2007	None	
WO 2006-071792 A2	06/07/2006	CN 100437523 C	26/11/2008
		CN 1804816 A	19/07/2006
		EP 1831791 A2	12/09/2007
		JP 2008-525919 A	17/07/2008
		US 2006-0143396 A1	29/06/2006
		WO 2006-071792 A3	04/01/2007