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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
3 August 2017 (03.08.2017)

WIPO | PCT

(10) International Publication Number
WO 2017/131648 A1

(51) International Patent Classification:
G06F 12/08 (2006.01) *G06F 13/16* (2006.01)

(21) International Application Number:
PCT/US2016/015030

(22) International Filing Date:
27 January 2016 (27.01.2016)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **HEWLETT PACKARD ENTERPRISE DEVELOPMENT LP** [US/US]; 11445 Compaq Center Drive West, Houston, Texas 77070 (US).

(72) Inventors: **BROEDER, Sean L.**; 3000 Hanover Street, Palo Alto, California 94304 (US). **MARKT, Dennis E.**; 10810 Farnam Drive, Omaha, Nebraska 68154 (US). **TUNG, Shang-Sheng**; 3000 Hanover Street, Palo Alto, California 94304 (US). **CASSOU, Ronald M.**; 3000 Hanover Street, Palo Alto, California 94304 (US).

(74) Agents: **POSSETT, Ramya** et al.; Hewlett Packard Enterprise, 3404 E. Harmony Road, M/S 79, Fort Collins, Colorado 80528 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, 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).

Declarations under Rule 4.17:

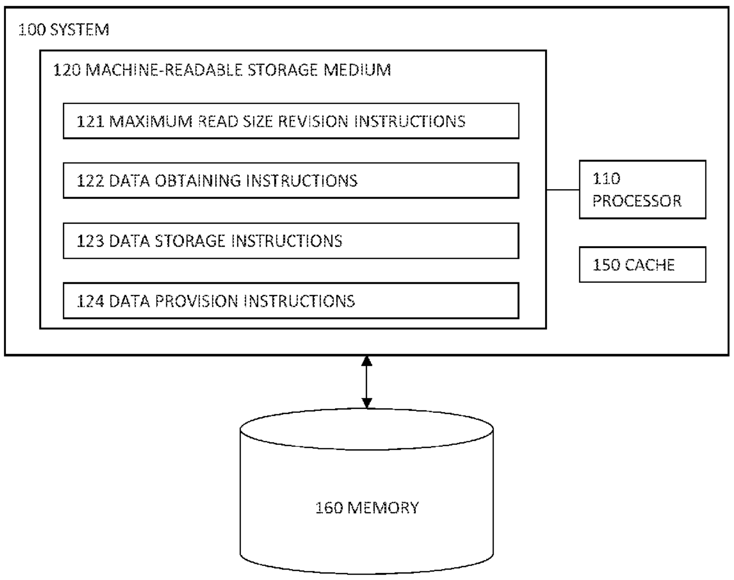
— *as to the identity of the inventor (Rule 4.17(i))*

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

[Continued on next page]

(54) Title: ADAPTIVE REQUEST MANAGEMENT

FIG. 1


100 SYSTEM
120 MACHINE-READABLE STORAGE MEDIUM
121 MAXIMUM READ SIZE REVISION INSTRUCTIONS
122 DATA OBTAINING INSTRUCTIONS
123 DATA STORAGE INSTRUCTIONS
124 DATA PROVISION INSTRUCTIONS
110 PROCESSOR
150 CACHE
160 MEMORY

(57) **Abstract**: Examples relate to adaptive request management for memory. One example relates to a system for adaptive request management. They system may comprise a physical processor implementing machine readable instructions stored on a non-transitory machine-readable storage medium that cause the system to responsive to receiving a request from a client, revise a maximum read amount of data to be read from a memory communicably coupled to the system, obtain, from the memory, a first amount of data corresponding to the received request from the client, wherein a size of the obtained first amount of data is the revised maximum read amount of data; and store the obtained data from the memory in a cache of the system.

WO 2017/131648 A1

Published:

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

ADAPTIVE REQUEST MANAGEMENT

BACKGROUND

[0001] Data may be stored in memory and accessed by a system via a cache. In situations where numerous requests exist to read data are received from clients, the system may experience performance issues when accessing data in the memory, storing it in the cache, and providing it to clients.

BRIEF DESCRIPTION OF THE DRAWINGS

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

[0003] FIG. 1 is a block diagram of an example system for adaptive request management;

[0004] FIG. 2 is a block diagram of an example system for adaptive request management;

[0005] FIG. 3 is an example table of thresholds and corresponding scale factors for adaptive request management; and

[0006] FIG. 4 is a flowchart of an example method for adaptive request management.

DETAILED DESCRIPTION

[0007] The following detailed description refers to the accompanying drawings. Wherever possible, the same reference numbers are used in the drawings and the following description to refer to the same or similar parts. While several examples are described in this document, modifications, adaptations, and other implementations are possible. Accordingly, the following detailed description does not limit the disclosed examples. Instead, the proper scope of the disclosed examples may be defined by the appended claims.

[0008] As mentioned above, data may be stored in memory and accessed by a system via a cache. For example, when the system receives a request for data, the system may request that data from the memory, and the data obtained from the memory may be stored in the cache and accessed by the system via the cache. In some examples, it may take numerous attempts to obtain data from the memory and store it in the cache to fulfill a single read request for data. In situations where numerous requests exist to read data are received from clients, the system may experience performance issues when accessing data in the memory, storing it in the cache, and providing it to clients.

[0009] For example, a static algorithm to fetch data from a memory may cause performance issues when the size of a cache is limited. In this example, system performance could be significantly degraded if a large amount of data (e.g., more than would fit in the cache) is being read. Performance could suffer even more if a large number of read requests are pending.

[0010] Performance issues may arise because the total size of the cache used by the server is bounded, so as data is stored in the cache, it overwrites existing data in the cache. Also, the reply size is a fraction of the maximum read size of data that may be stored in the cache, so more data may be stored in the cache than can be returned to the requesting client. As such, when the server goes to the memory to retrieve data from the memory, the server obtains more data from the memory to be stored in the cache than can be returned in a single reply. The cost of a large read is small compared to initiating a read in the first place. Given that, servers often obtain larger amounts of data to store in the cache than can be returned to the client in a reply message.

[0011] An example of poor system performance with a read request may occur with a large table scan that is requested with other concurrent read requests. If the size of data in the read request exceeds the size of the cache or a size of data that can be returned in a single reply, the read request may be satisfied by sending multiple requests to the memory for the requested data.

[0012] In this example, the server tries to anticipate the data that is needed so that the data is in the cache when the next request for the data is received. Accordingly, the server may try to read more data than can be returned in a single reply to the requester or may schedule data to be pre-fetched before the next request is read. When the request for data is received, the server may obtain the relevant data from the memory, store it in the cache, and return a portion of the stored data to the requesting client in a reply message. As such, multiple repeated requests may be needed to return all of the data requested in the large table scan.

[0013] The server, however, is handling more requests than just this single table scan. At any given time, there may be tens or hundreds of requests in the server's queue for different data. Because other requests are being satisfied, when the server tries to obtain data from the cache for the large table scan request, that data may no longer be in the cache. The server will have to go obtain the relevant data from the memory again and store it in the cache, displacing data that has been read for another request.

[0014] In this example, the same effort has to be made repeatedly to satisfy the request for the large table scan. The data retrieved for the large table scan will be replaced by data retrieved for other read requests, so the same large amount of data may be obtained numerous times to satisfy the request for the table scan. These efforts may be duplicated for numerous requests where the size of data exceeds the cache size or reply size. The concurrency of requests triggers this duplication of efforts, with a larger request queue size of concurrent requests increasing the likelihood that previously retrieved data for a first request will be replaced by data retrieved for a second request before the first request can be fully serviced. These duplicated efforts cause a major technical challenge in that the server is continuously busy even though not a lot of data is returned to clients.

[0015] A new technical solution to this technical challenge involves an adaptive algorithm that limits the maximum read size of data stored in the cache. By limiting the amount of data read to the cache, the server may achieve higher performance at scale by ensuring that some of the previously retrieved data is still in the cache when another request for that data is received. Using an adaptive solution allows the amount of data stored in the cache to become smaller when needed and return to a predetermined maximum read size as appropriate.

[0016] This solution addresses the issue of poor performance in numerous ways. The system may perform adaptive request management, where the maximum read size of data read from the memory and stored in the cache responsive to receiving a request from the client may be revised. For example, the system may have a predetermined maximum read size that may be revised responsive to the request queue size of requests reaching a predetermined threshold. In this example, the maximum read size may be reduced by a predetermined scale factor. In some examples, a set of predetermined thresholds and corresponding set of scale factors may be used.

[0017] In some examples, the personality of worker threads in the system may be altered. For example, when the request queue is small (and the maximum read size is the predetermined maximum read size), the system may have separate threads for prefetching data and for storing data in the cache. Responsive to determining that adaptive request management should be used, the thread for prefetching data may no longer be used, and only one thread may be used to obtain data from the memory and store it in the cache.

[0018] The system may determine that adaptive request management should be used responsive to the request size exceeding a predetermined threshold, responsive to the amount of data in the cache corresponding to the request being less than a predetermined threshold size, responsive to data stored in the cache corresponding to the request being less than a predetermined multiple of the reply size, and/or responsive to other conditions that may cause poor system performance.

[0019] A computer system implementing adaptive request management may revise, responsive to receiving a request from a client, a maximum read size of data to be read from a memory communicably coupled to the system. The system may also obtain, from the memory, a first amount of data corresponding to the received request from the client, wherein a size of the obtained first amount of data is the revised maximum read amount of data. The system may store the obtained data from the memory in a cache of the system.

[0020] Referring now to the drawings, FIG. 1 is a block diagram of an example system 100 for adaptive request management. In the example depicted in FIG. 1, system 100 includes a non-transitory machine-readable storage medium 120 and a processor 110.

[0021] Referring now to the drawings, FIG. 1 is a block diagram of an example system 100 for adaptive request management. System 100 may comprise a cloud server, a mainframe, notebook, desktop, tablet, workstation, mobile device, and/or any other device suitable for executing the functionality described below. In the embodiment of FIG. 1, system 100 includes a non-transitory machine-readable storage medium 120 and a processor 110.

[0022] Processor 110 may be one or more central processing units (CPUs), microprocessors, and/or other hardware devices suitable for retrieval and execution of instructions stored in machine-readable storage medium 120. Processor 110 may fetch, decode, and execute program instructions 121, 122, 123, 124, and/or other instructions to enable adaptive request management, as described below. As an alternative or in addition to retrieving and executing instructions, processor 110 may include one or more electronic circuits comprising a number of electronic components for performing the functionality of one or more of instructions 121, 122, 123, 124, and/or other instructions.

[0023] In one example, the program instructions 121, 122, 123, 124, and/or other instructions can be part of an installation package that can be executed by processor 110 to implement the functionality described herein. In this case, memory 120 may be a portable medium such as a CD, DVD, or flash drive or a memory maintained by a computing device from which the installation package can be downloaded and installed. In another example, the program instructions may be part of an application or applications already installed on system 100.

[0024] Non-transitory machine-readable storage medium 120 may be any hardware storage device for maintaining data accessible to system 100. For example, machine-readable storage medium 120 may include one or more hard disk drives, solid state drives, tape drives, and/or any other storage devices. The storage devices may be located in system 100 and/or in another device in communication with system 100. For example, machine-readable storage medium 120 may be any electronic, magnetic, optical, or other physical storage device that stores executable instructions. Thus, machine-readable storage medium 120 may be, for example, Random Access Memory (RAM), an Electrically-Erasable Programmable Read-Only Memory (EEPROM), a storage drive, an optical disc, and the like. As described in detail below, machine-readable storage medium 120 may be encoded with executable instructions for adaptive request management. As detailed below, storage medium 120 may maintain and/or store the data and information described herein.

[0025] Cache 150 may comprise a non-transitory high-speed storage device for maintaining data accessible to system 100 and received from a non-transitory machine-readable storage medium 160. In some examples, cache 150 may be part of storage medium 120, may be a separate hardware component of system 100, and/or may be a separate hardware component communicably coupled to system 100. Cache 150 may be a L1 cache, L2 cache, and/or any other type of non-transitory high-speed storage device suitable for storing data obtained from storage medium 160 and accessible to system 100.

[0026] Non-transitory machine-readable storage medium 160 may be any hardware storage device for maintaining data accessible to system 100. For example, machine-readable storage medium 160 may include one or more hard disk drives, solid state drives, tape drives, and/or any other storage devices. The storage devices may be located in system 100 and/or in another device in communication with system 100. For example, machine-readable storage medium 160 may be any electronic, magnetic, optical, or other physical storage device that stores data. Thus, machine-readable storage medium 160 may be, for example, Random Access Memory (RAM), an Electrically-Erasable Programmable Read-Only Memory (EEPROM), a storage drive, an optical disc, and the like. As described in detail below, machine-readable storage medium 160 may maintain and/or store data and/or information for a set of clients that obtain the data via the system 100.

[0027] System 100 may receive a request from a client for data stored in the storage medium 160. System 100 may store the request in a queue of requests. The set of request may comprise requests from one or multiple clients. The system 100 may process requests in the queue in a first come, first serve order or any other order suitable for processing requests. Each request may comprise an identification of the data to be read, a size of the data to be read, and/or other information related to the data to be read.

[0028] The system 100 may determine a next request to be processed from the queue of requests. To process the request, the system 100 may determine a size of data to obtain from the storage medium 160, may obtain relevant data of that size from the storage medium 160, may store the obtained data in the cache 150, and may provide some or all of the stored data in the cache 150 to the requesting client.

[0029] Maximum read size revision instructions 121, when executed by processor 110, may begin to process the request by determining the size of data to obtain from the storage medium 160. The non-transitory storage medium 120 (and/or cache 150) may store a maximum read amount that indicates the size of data that system 100 (and/or data obtaining instructions 122) should obtain from the storage medium 160 each time a request is processed. The maximum read amount may be a predetermined, standard read amount or may be revised by maximum read size revision instructions 121, when executed by processor 110.

[0030] The maximum read size revision instructions 121, when executed by processor 110, may determine whether to revise the maximum read amount of data to be read from the storage medium 160. The maximum read size revision instructions 121, when executed by processor 110, may determine whether to revise the maximum read amount of data based on one or more factors.

[0031] In some examples, the maximum read size revision instructions 121, when executed by processor 110, may determine whether to revise the maximum read amount of data by determining whether a request queue size exceeds a predetermined threshold. The non-transitory storage medium 120 (and/or cache 150) may store a threshold request queue size as the predetermined threshold. The maximum read size revision instructions 121, when executed by processor 110, may determine a number of requests in the queue as the request queue size and compare the request queue size to the predetermined threshold. Responsive to the request queue size exceeding the predetermined threshold, the maximum read size revision instructions 121, when executed by processor 110, may determine that the maximum read amount should be revised and may revise the maximum read amount (as described in further detail below). Responsive to the request queue size not exceeding the predetermined threshold, the maximum read size revision instructions 121, when executed by processor 110, may use the predetermined, standard read amount stored in the non-transitory machine readable storage medium 120 (and/or the cache 150) as the maximum read amount.

[0032] In some examples, the maximum read size revision instructions 121, when executed by processor 110, may determine whether to revise the maximum read amount of data by determining whether a request queue size exceeds a predetermined threshold. The non-transitory storage medium 120 (and/or cache 150) may store a threshold request queue size as the predetermined threshold. The maximum read size revision instructions 121, when executed by processor 110, may determine a number of requests in the queue as the request queue size and compare the request queue size to the predetermined threshold. Responsive to the request queue size exceeding the predetermined threshold, the maximum read size revision instructions 121, when executed by processor 110, may determine that the maximum read amount should be revised and may revise the maximum read amount (as described in further detail below). Responsive to the request queue size not exceeding the predetermined threshold, the maximum read size revision instructions 121, when executed by processor 110, may use the predetermined amount of maximum read amount stored in the non-transitory machine readable storage medium 120 (and/or the cache 150) as the maximum read amount.

[0033] The maximum read size revision instructions 121, when executed by processor 110, may revise the maximum read amount of data by scaling down the maximum read amount of data by a scale factor corresponding to the predetermined threshold (of request queue size). In some examples, the storage medium 120 (and/or cache 150) may store a set of predetermined thresholds of request queue size and a corresponding set of scale factors. The maximum read size revision instructions 121, when executed by processor 110, may revise the maximum read amount of data by a scale factor corresponding to a threshold exceeded in the set of predetermined thresholds. In these examples, the maximum read size revision instructions 121, when executed by processor 110, may determine the largest threshold exceeded by the request queue size and may determine the scale factor corresponding to that threshold.

[0034] FIG. 3 is an example table of thresholds and corresponding scale factors for adaptive request management. For example, the maximum read size revision instructions 121, when executed by processor 110, may determine, responsive to a first threshold (e.g., 100 requests) being exceeded by the request queue size, whether the request queue size exceeds a second threshold (e.g., 200 requests) larger than the first threshold. Responsive to the request size exceeding the second threshold (but not a third threshold (e.g. 300 requests) larger than the second threshold), the maximum read size revision instructions 121, when executed by processor 110, may revise the maximum read amount of data by scaling down the maximum read amount of data by a second scale factor (e.g., 4) corresponding to the exceeded second predetermined threshold.

[0035] Returning to FIG. 1, in another example, the maximum read size revision instructions 121, when executed by processor 110, may determine whether to revise the maximum read amount of data by determining whether a cached amount of data corresponding to the received request is less than a reply size amount of data to be returned to the client. The cached amount of data may comprise an amount of data stored in the cache 150 corresponding to the received request. In some examples, the cached amount of data may be only the amount of data being obtained by the read request to the storage medium 160, but in other examples, the cached amount of data may include data already stored in the cache 150 that was stored as part of previous processing of the request received from the client.

[0036] In some examples, responsive to determining that the maximum read amount should be revised (and/or revising the maximum read amount), the maximum read size revision instructions 121, when executed by processor 110, may stop pre-fetching, from storage medium 160, data corresponding to the request from the client.

[0037] In some examples, responsive to determining that the maximum read amount should not be revised, the maximum read size revision instructions 121, when executed by processor 110, may revise the stored maximum read amount in the storage medium 120 (and/or cache 150) to the predetermined, standard read amount.

[0038] Responsive to the maximum read size revision instructions 121, when executed by processor 110, determining the maximum read amount of data, the data obtaining instructions 122, when executed by processor 110, may obtain, from the storage medium 160, the determined maximum read amount of data corresponding to the received request from the client.

[0039] In some examples, the data storage instructions 123, when executed by processor 110, may store the obtained data from the storage medium 160 in the cache 150. As mentioned above, the data storage instructions 123, when executed by processor 110, may store the obtained data in the cache 150 by overwriting existing data in the cache 150.

[0040] In some examples, the data provision instructions 124, when executed by processor 110, may provide, as a reply response to the received request, a subset of the stored data in the cache 150 to the client. The reply response may have a size that is smaller than the amount of data obtained and stored in the cache 150 responsive to receiving the request. As explained above, the cost of initiating a read request is much higher than reading a large amount of data. As such, more data may be stored in the cache than may be returned to the client in a single reply response.

[0041] FIG. 2 is a block diagram of an example system 200 for adaptive request management. As with system 100, system 200 may comprise a cloud server, a mainframe, notebook, desktop, tablet, workstation, mobile device, and/or any other device suitable for executing the functionality described below. As with processor 110 of FIG. 1, processor 210 may be one or more CPUs, microprocessors, and/or other hardware devices suitable for retrieval and execution of instructions. Cache 250 may be the same as or similar to the cache 150 of FIG. 1. Non-transitory storage medium 260 of FIG. 2 may be the same as or similar to the non-transitory storage medium 160 of FIG. 1.

[0042] As detailed below, system 200 may include a series of engines 220-240 for adaptive request management. Each of the engines may generally represent any combination of hardware and programming. For example, the programming for the engines may be processor executable instructions stored on a non-transitory machine-readable storage medium and the hardware for the engines may include at least one processor of the system 200 to execute those instructions. In addition or as an alternative, each engine may include one or more hardware devices including electronic circuitry for implementing the functionality described below.

[0043] System 200 may receive and process requests in a manner the same as or similar to system 100.

[0044] Maximum read size revision engine 220 may revise a maximum read amount of data to be read from a memory communicably coupled to the system, responsive to receiving a request from a client. The maximum read size revision engine 220 may determine whether to revise the maximum read amount of data before revising the maximum read amount of data. In some examples, the maximum read size revision engine 220 may revise the maximum read amount of data in a manner the same as or similar to that of the maximum read size revision instructions 122 of system 100. Further details regarding an example implementation of maximum read size revision engine 220 are provided above in connection with maximum read size revision instructions 121 of FIG. 1.

[0045] Data obtaining engine 230 may obtain, from the memory, a first amount of data corresponding to the received request from the client, wherein a size of the obtained first amount of data is the revised maximum read amount of data. In some examples, the data obtaining engine 230 may obtain the first amount of data corresponding to the received request from the client in a manner the same as or similar to that of the data obtaining instructions 122 of system 100. Further details regarding an example implementation of data obtaining engine 230 are provided above in connection with data obtaining instructions 122 of FIG. 1.

[0046] Data storage engine 240 may store the obtained data from the memory in a cache of the system. In some examples, the data storage engine 240 may store the obtained data from the memory in a cache (e.g., cache 250) in a manner the same as or similar to that of the system 100. Further details regarding an example implementation of data storage engine 240 are provided above in connection with data storage instructions 123 of FIG. 1.

[0047] FIG. 4 is a flowchart of an example method for execution by a computing device for adaptive request management.

[0048] Although execution of the methods described below are with reference to system 100 of FIG. 1, and/or system 200 of FIG. 2, other suitable devices for execution of this method will be apparent to those of skill in the art. The method described in FIG. 4 and other figures may be implemented in the form of executable instructions stored on a machine-readable storage medium, such as storage medium 120, by one or more engines described herein, and/or in the form of electronic circuitry.

[0049] In an operation 400, responsive to receiving a request from a client, the method determines whether to revise a maximum read amount of data to be read from a memory communicably coupled to the system. For example, the system 100 (and/or the maximum read size revision instructions 121, the maximum read size revision engine 220, or other resource of the system 100) may determine whether to revise the maximum read amount. The system 100 may determine whether to revise the maximum read amount in a manner similar or the same as that described above in relation to the execution of the maximum read size revision instructions 121, the maximum read size revision engine 220, and/or other resource of the system 100.

[0050] In an operation 410, responsive to determining that the maximum read amount of data should be revised, the maximum read amount of data to be read from the memory may be revised. For example, the system 100 (and/or the maximum read size revision instructions 121, the maximum read size revision engine 220, or other resource of the system 100) may revise the maximum read amount. The system 100 may revise the maximum read amount in a manner similar or the same as that described above in relation to the execution of the maximum read size revision instructions 121, the maximum read size revision engine 220, and/or other resource of the system 100.

[0051] In an operation 420, a first amount of data corresponding to the received request from the client may be obtained from the memory, wherein a size of the obtained first amount of data is the revised maximum read amount of data. For example, the system 100 (and/or the data obtaining instructions 122, the data obtaining engine 230, or other resource of the system 100) may obtain the first amount of data. The system 100 may obtain the first amount of data in a manner similar or the same as that described above in relation to the execution of the data obtaining instructions 122, the data obtaining engine 230, and/or other resource of the system 100.

[0052] In an operation 430, the data obtained from the memory may be stored in a cache of the system. For example, the system 100 (and/or the data storage instructions 123, the data storage engine 240, or other resource of the system 100) may store the obtained data in the cache. The system 100 may store the obtained data in the cache in a manner similar or the same as that described above in relation to the execution of the data storage instructions 123, the data storage engine 240, or other resource of the system 100.

[0053] The foregoing disclosure describes a number of example embodiments for adaptive request management. The disclosed examples may include systems, devices, computer-readable storage media, and methods for adaptive request management. For purposes of explanation, certain examples are described with reference to the components illustrated in FIGS. 1-4. The functionality of the illustrated components may overlap, however, and may be present in a fewer or greater number of elements and components. Further, all or part of the functionality of illustrated elements may co-exist or be distributed among several geographically dispersed locations. Moreover, the disclosed examples may be implemented in various environments and are not limited to the illustrated examples.

[0054] Further, the sequence of operations described in connection with FIGS. 1-4 are examples and are not intended to be limiting. Additional or fewer operations or combinations of operations may be used or may vary without departing from the scope of the disclosed examples. Furthermore, implementations consistent with the disclosed examples need not perform the sequence of operations in any particular order. Thus, the present disclosure merely sets forth possible examples of implementations, and many variations and modifications may be made to the described examples. All such modifications and variations are intended to be included within the scope of this disclosure and protected by the following claims.

Claims

We claim:

1. A system for adaptive request management, the system comprising:

a physical processor implementing machine readable instructions stored on a non-transitory machine-readable storage medium that cause the system to:

responsive to receiving a request from a client, revise a maximum read amount of data to be read from a memory communicably coupled to the system;

obtain, from the memory, a first amount of data corresponding to the received request from the client, wherein a size of the obtained first amount of data is the revised maximum read amount of data; and

store the obtained data from the memory in a cache of the system.

2. The system of claim 1, wherein the physical processor implements machine readable instructions that cause the system to:

provide, as a reply response to the received request, a subset of the stored data in the cache to the client.

3. The system of claim 1, wherein the physical processor implements machine readable instructions that cause the system to:

responsive to receiving the request from the client, determine whether to revise the maximum read amount of data to be read from the memory.

4. The system of claim 3, wherein the physical processor implements machine readable instructions that cause the system to:

determine whether to revise the maximum read amount of data by determining whether a request queue size exceeds a predetermined threshold, wherein the request queue size comprises a number of requests received for data in the memory; and

revise the maximum read amount of data by scaling down the maximum read amount of data by a scale factor corresponding to the predetermined threshold.

5. The system of claim 3, wherein the non-transitory machine readable storage medium comprises a set of predetermined thresholds and a corresponding set of scale factors, and

wherein the physical processor implements machine readable instructions that cause the system to:

determine whether to revise the maximum read amount of data by determining whether a request queue size exceeds a first predetermined threshold of the set of predetermined thresholds, wherein the request queue size comprises a number of requests received for data in the memory; and

revise the maximum read amount of data by scaling down the maximum read amount of data by a first scale factor corresponding to the exceeded first predetermined threshold.

6. The system of claim 5, wherein the physical processor implements machine readable instructions that cause the system to:

determine whether to revise the maximum read amount of data by determining whether the request queue size exceeds a second predetermined threshold of the set of predetermined thresholds; and

revise the maximum read amount of data by scaling down the maximum read amount of data by a second scale factor corresponding to the exceeded second predetermined threshold.

7. The system of claim 1, wherein the physical processor implements machine readable instructions that cause the system to:

determine whether to revise the maximum read amount of data by determining whether a cached amount of data corresponding to the received request is less than a reply size amount of data to be returned to the client, wherein the cached amount of data comprises data stored in the cache corresponding to the received request.

8. The system of claim 1, wherein the physical processor implements machine readable instructions that cause the system to:

responsive to revising the maximum read amount of data, stop pre-fetching, from the memory, data corresponding to the received request.

9. The system of claim 1, wherein the physical processor implements machine readable instructions that cause the system to:

revise the maximum read amount of data to be read from the memory to a predetermined standard read amount.

10. A method for adaptive request management, the method being implemented by a computing system comprising a physical processor implementing computer readable instructions, the method comprising:

responsive to receiving a request from a client, determining whether to revise a maximum read amount of data to be read from a memory communicably coupled to the system;

responsive to determining that the maximum read amount of data should be revised, revising the maximum read amount of data to be read from the memory;

obtaining, from the memory, a first amount of data corresponding to the received request from the client, wherein a size of the obtained first amount of data is the revised maximum read amount of data;

storing the obtained data from the memory in a cache of the system.

11. The method of claim 10, wherein determining whether to revise the maximum read amount of data comprises:

determining whether a request queue size exceeds a predetermined threshold, wherein the request queue size comprises a number of requests received for data in the memory; and

wherein the method further comprises:

revising the maximum read amount of data by scaling down the maximum read amount of data by a scale factor corresponding to the predetermined threshold.

12. The method of claim 10, wherein determining whether to revise the maximum read amount of data comprises:

determining whether a cached amount of data corresponding to the received request is less than a reply size amount of data to be returned to the client, wherein the cached amount of data comprises data stored in the cache corresponding to the received request.

13. The method of claim 10, wherein the system comprises a non-transitory machine readable storage medium that comprises a set of predetermined thresholds and a corresponding set of scale factors,

wherein determining whether to revise the maximum read amount of data comprises determining whether a request queue size exceeds one of a set of predetermined thresholds, wherein the request queue size comprises a number of requests received for data in the memory; and

wherein the method further comprises:

responsive to determining that the request queue size exceeds the predetermined threshold, revising the maximum read amount of data by scaling down the maximum read amount of data by a scale factor corresponding to the exceeded predetermined threshold.

.

14. The method of claim 10, further comprising:

responsive to determining that the maximum read amount of data should not be revised, revising the maximum read amount of data to be read from the memory to a predetermined standard read amount.

15. A non-transitory machine-readable storage medium comprising instructions for adaptive request management, the instructions executable by a physical processor of a system to:

responsive to receiving a request from a client, revise a maximum read amount of data to be read from a memory communicably coupled to the system;

obtain, from the memory, a first amount of data corresponding to the received request from the client, wherein a size of the obtained first amount of data is the revised maximum read amount of data;

store the obtained data from the memory in a cache of the system; and

provide, as a reply response to the received request, a subset of the stored data in the cache to the client.

FIG. 1

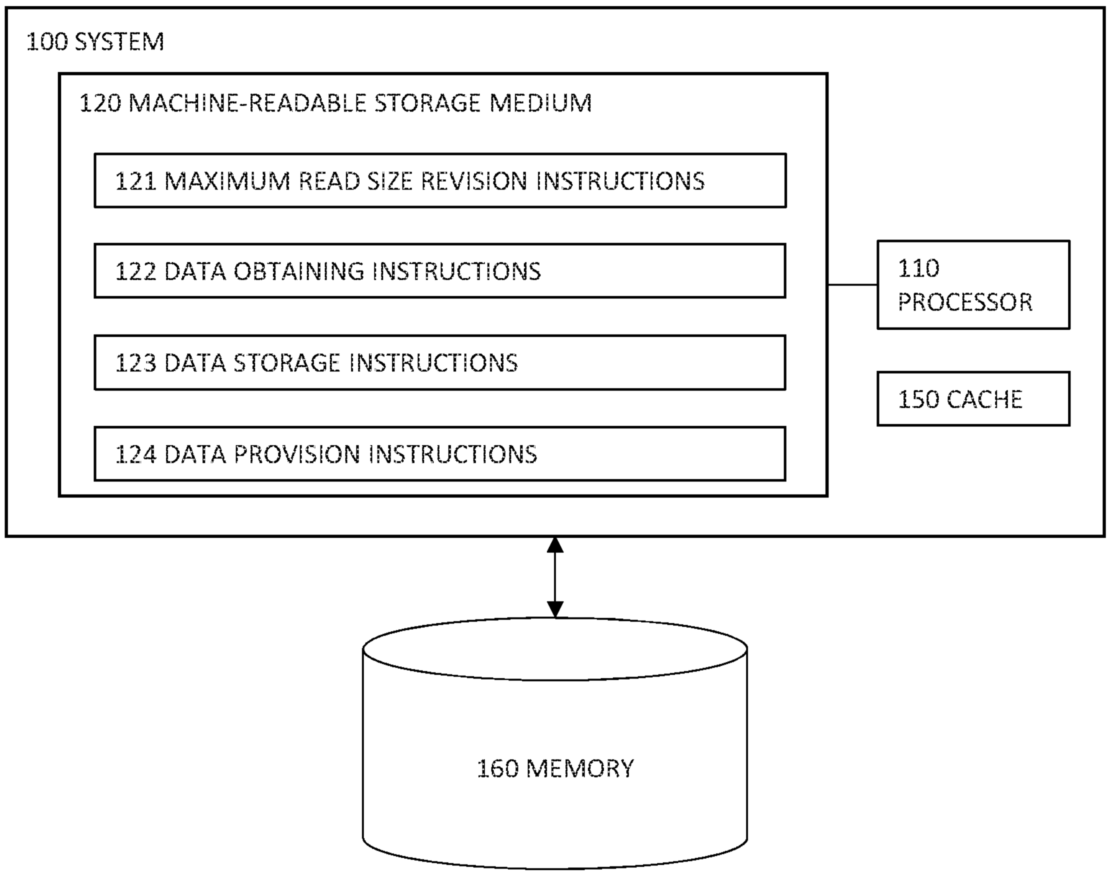

100 SYSTEM
120 MACHINE-READABLE STORAGE MEDIUM
121 MAXIMUM READ SIZE REVISION INSTRUCTIONS
122 DATA OBTAINING INSTRUCTIONS
123 DATA STORAGE INSTRUCTIONS
124 DATA PROVISION INSTRUCTIONS
110 PROCESSOR
150 CACHE
160 MEMORY

FIG. 2

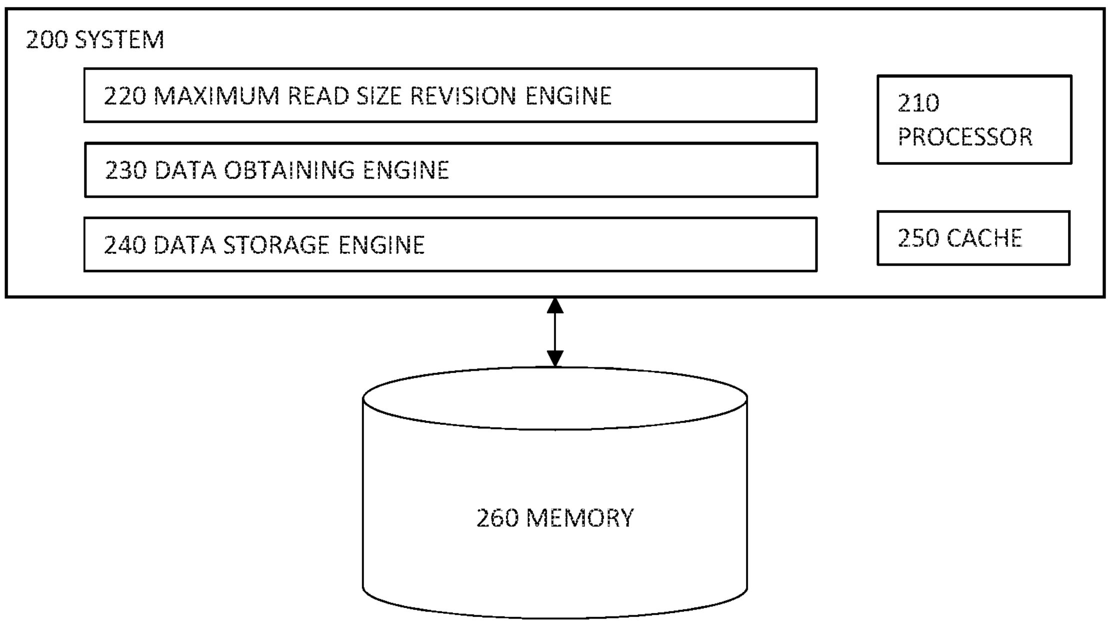
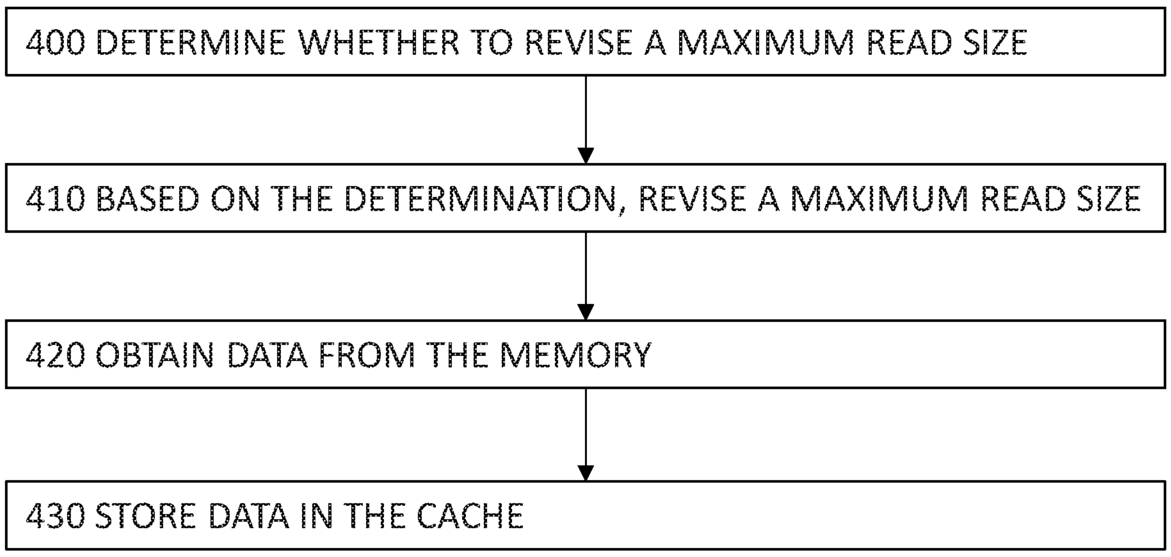

200 SYSTEM
220 MAXIMUM READ SIZE REVISION ENGINE
230 DATA OBTAINING ENGINE
240 DATA STORAGE ENGINE
210 PROCESSOR
250 CACHE
260 MEMORY

FIG. 3

Request Size Threshold 300	Scale Factor 310
100	2
200	4
300	8
400	16
500	32

FIG. 4


400 DETERMINE WHETHER TO REVISE A MAXIMUM READ SIZE
410 BASED ON THE DETERMINATION, REVISE A MAXIMUM READ SIZE
420 OBTAIN DATA FROM THE MEMORY
430 STORE DATA IN THE CACHE

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2016/015030

A. CLASSIFICATION OF SUBJECT MATTER

G06F 12/08(2006.01)i, G06F 13/16(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F 12/08; G06F 12/16; G06F 3/00; G06F 12/00; G06F 13/28; G06F 13/16

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: request, memory, revise, maximum read amount of data, size, cache

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2009-0063731 A1 (KEVIN C. GOWER et al.) 05 March 2009 See paragraphs [0018], [0031]-[0036], [0055]-[0057]; claims 1, 12; and figures 1, 4.	1-3,9-10,14-15
Y		8
A		4-7,11-13
Y	US 2014-0089602 A1 (APPLE INC.) 27 March 2014 See paragraphs [0077]-[0083]; claim 1; and figures 6, 8.	8
A	US 2005-0193023 A1 (LABEEB K. ISMAIL) 01 September 2005 See paragraphs [0031]-[0044], [0052]-[0067]; and figures 3A-3B, 4B-5.	1-15
A	US 2009-0063729 A1 (KEVIN C. GOWER et al.) 05 March 2009 See paragraphs [0056]-[0064], [0090]-[0097]; claims 1, 13; and figures 5-7.	1-15
A	JP 5797342 B2 (MITSUBISHI ELECTRIC CORP.) 21 October 2015 See paragraphs [0015]-[0020], [0087]-[0098]; and figures 3, 12.	1-15

☐ Further documents are listed in the continuation of Box C.   ☒ See patent family annex.

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

Date of the actual completion of the international search 19 October 2016 (19.10.2016)	Date of mailing of the international search report **20 October 2016 (20.10.2016)**
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 CHIN, Sang Bum Telephone No. +82-42-481-8398

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/US2016/015030

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2009-0063731 A1	05/03/2009	US 7558887 B2	07/07/2009
US 2014-0089602 A1	27/03/2014	US 9218286 B2	22/12/2015
US 2005-0193023 A1	01/09/2005	CA 02557770 A1	06/10/2005
		CA 2557770 C	25/03/2014
		US 2015-0319105 A1	05/11/2015
		US 8965936 B2	24/02/2015
		WO 2005-091773 A2	06/10/2005
		WO 2005-091773 A3	30/11/2006
US 2009-0063729 A1	05/03/2009	US 7861014 B2	28/12/2010
JP 5797342 B2	21/10/2015	CN 104769558 A	08/07/2015
		DE 112012007102 T5	16/07/2015
		TW 201418980 A	16/05/2014
		TW I475383 B	01/03/2015
		US 2015-0186208 A1	02/07/2015
		WO 2014-068789 A1	08/05/2014