

DATA CACHING AND READING METHOD, AND DATA ACCESS SYSTEM

[1] The present application claims priority to Chinese patent application No. 202210037805.8, filed with the Chinese Patent Office on January 13, 2022, and entitled “Data Caching and Reading Method, and Data Access System”, which is incorporated herein by reference in its entirety.

TECHNICAL FIELD

[2] The present application relates to a technical field of data caching, in particular, to data caching and reading.

BACKGROUND

[3] In recent years, in the scenarios of big data analyses, cloud-based object storage services are used continually and extensively. When large-scale distributed analysis services read data from the cloud-based object storage services, a substantial amount of hot topic data occupies most of the bandwidths provided by the object storage services, resulting in a poor overall utilization rate of clusters.

[4] As such, a distributed caching system is established by using the spare storage space in a cluster to improve the efficiency of data reading so as to accelerate big data analyses on the object storage services. This is an ideal solution in terms of cost and efficiency.

[5] However, the requirements for strong consistency, high performance, and high availability in big data analyses bring great challenges to the global design, performance indicators, and other aspects of a data caching system.

[6] FIG. 1 schematically shows the architecture of an existing caching system Alluxio.

[7] Alluxio is an abstract file caching system between a computing service and an underlying file (object) storage, mainly providing the caching capability for the underlying file storage.

[8] When use of Alluxio is needed in the computing service to read a file in the object storage services, it is necessary to acquire metadata of a file through the Alluxio service first.

[9] In Step S1, the client in the computing service sends a data request to the host of the data caching service on the basis of a file name maintained by the data caching service, e.g., “alluxio:///mount_path/directory/file1”.

[10] The indexing relationship between a file in the object storage service system and that in a data caching service system is established mainly through <file name + update time> of the underlying file within Alluxio, i.e., corresponding to the file name maintained by the stored data caching system and the file name and file update time of a file in the corresponding object storage service system as shown in FIG. 1.

[11] In Step S2, the host of the data caching service system requests for the update time of a corresponding underlying file (“oss://bucket/directory/file1”) from the object storage service system, and receives, in Step S3, information regarding the update time of the underlying file returned by the object storage service system.

[12] Afterwards, by comparing the update time received from the object storage service system with that of a file recorded by the data caching service system, the host determines whether the update times are consistent so as to determine whether the file cached by the data caching service system is consistent with data stored in the object storage service system.

[13] No matter whether there is a cached file which is consistent with the underlying file requested, a node of a specified working machine (Alluxio Worker) needs to be connected again. The working machine caches data in a size of 64M by default for storage, and then responds to a read request from the computing service.

[14] The host notifies, in Step S4, the client of the computing service of the specified working machine.

[15] In Step S5, the client requests data from the working machine.

[16] In the event that there is a cached file which is consistent with the requested underlying file, the working machine directly sends, in Step S8, cached data (or a cached file) to the client.

[17] In the event that there is no cached file which is consistent with the requested underlying file, the working machine requests, in Step S6, data of the underlying file from the object storage service system, and receives, in Step S7, data from the object storage service system, and caches the same. Then in Step S8, the cached data (or a cached file) is sent to the client.

[18] However, in the existing technical solution, the validity of caching depends on “an update time” of a requested file recorded in the underlying storage. The update time of a file system is generally on the order of seconds. For millisecond-level update operations such as put and append, it is highly likely to occur within 1 second. When such a millisecond-level update operation occurs and the file system has not been updated, there would be consistency problems if data is read, causing the read data to be incorrect.

[19] Moreover, the solution strongly depends on the node of the Alluxio service, thereby lowering the high availability index of the system. For example, when the Alluxio service is abnormal, the entire storage system will also be inaccessible.

[20] In addition, the underlying object storage service system needs an Alluxio data caching service system for mapping intermediate metadata. The native access path cannot be used in the service. An access through a special path modified by Alluxio is required, and this cannot support transparent caching.

[21] Accordingly, a data caching and reading solution is still needed to solve at least one aspect of the above problems.

SUMMARY

[22] A technical problem to be solved by the disclosure is to provide data caching and reading solution which is capable of verifying the consistency between data cached by a data caching system and corresponding data stored in a data storage system conveniently, quickly, timely, and accurately.

[23] According to a first aspect of the disclosure, it is provided a data caching method for a data caching system, including: receiving a request for target data to be acquired sent by a client, the request including metadata of the target data stored in a data storage system; and determining, based on the metadata of the target data, whether cached target data which is consistent with the target data stored in the data storage system is cached in the data caching system.

[24] Optionally, the method may further include: sending location information of the cached target data to the client in the case of determining that the cached target data which is consistent with the target data of the data storage system is cached; and/or sending a consistence failure message to the client in the case of determining that the cached target data which is consistent with the target data of the data storage system is not cached.

[25] Optionally, in the case of determining that the cached target data which is consistent with the target data of the data storage system is not cached, the method may further include: receiving, from the client, the target data acquired by the client from the data storage system; and caching the target data.

[26] Optionally, the step of determining whether the cached target data which is consistent with the target data stored in the data storage system is cached in the data caching system may include: determining, based on identification information of the target data, whether the cached target data corresponding to the target data is cached; and determining, based on metadata of the cached target data and the metadata of the target data, whether the cached target data is consistent with the target data in the case of determining that the cached target data corresponding to the target data is cached.

[27] Optionally, the metadata of the target data is acquired by the client from the data storage system.

[28] Optionally, the metadata of the target data includes at least one of an update time, a cyclic redundancy check value, and a data size of the target data.

[29] According to a second aspect of the disclosure, it is provided a data reading method, including: sending a first request for target data to a data storage system; receiving metadata of the target data from the data storage system; and sending a second request for the target data to a data caching system, the second request including the metadata.

[30] Optionally, the method may further include: acquiring the target data from the data caching system in a case where it is determined by the data caching system, based on the metadata, that data which is consistent with the target data of the data storage system is cached; and/or acquiring the target data from the data storage system in a case where it is determined by the data caching system, based on the metadata, that data which is consistent with the target data of the data storage system is not cached.

[31] Optionally, the method may further include: sending the acquired target data to the data caching system after acquiring the target data from the data storage system.

[32] According to a third aspect of the disclosure, it is provided a data caching apparatus, including a request receiving apparatus for receiving a request for target data to be acquired sent by a client, the request including metadata of target data stored in a data storage system; and a consistency determination apparatus for determining, based on the metadata of the target data, whether cached target data which is consistent with the target data stored in the data storage system is cached in the data caching system.

[33] According to a fourth aspect of the disclosure, it is provided a data reading apparatus, including a first request apparatus for sending a first request for target data to a data storage system; a metadata receiving apparatus for receiving metadata of target data from the data storage system; and a second request apparatus for sending a second request for target data to a data caching system, the second request including the metadata.

[34] According to a fifth aspect of the disclosure, it is provided a data access system, including a data storage system, a data caching system, and a client, wherein the client sends a first request for target data to the data storage system, the data storage system returns metadata of the target data to the client in response to the first request, the client sends a second request for the target data to the data caching system, the second request including the metadata, the data caching system determines, based on the metadata of the target data, whether cached target data which is consistent with the target data stored in the data storage system is cached in the data caching system, the data caching system sends location information of the cached target data to the client in the case of determining that the cached target data which is consistent with the target data of the data storage system is cached, the client acquires the target data from the data caching system, the data caching system sends a consistency failure message to the client in the

case of determining that the cached target data which is consistent with the target data of the data storage system is not cached, the client acquires the target data from the data storage system and sends the acquired target data to the data caching system, and the data caching system caches the target data.

[35] According to a sixth aspect of the disclosure, it is provided a computing device, including a processor; and a memory stored thereon with executable codes which, when executed by the processor, enable the processor to execute the method in the first aspect described above.

[36] According to a seventh aspect of the disclosure, it is provided a computer program product including executable codes which, when executed by a processor of an electronic device, enable the processor to execute the method in the first aspect described above.

[37] According to an eighth aspect of the disclosure, it is provided a non-transitory machine-readable storage medium stored thereon with executable codes which, when executed by a processor of an electronic device, enable the processor to execute the method in the first aspect described above.

[38] Hence, by acquiring metadata of target data via a client from a data storage system as the basis for determining the consistency of cached target data, it is possible to conveniently and quickly, but in real time and accurately, determine whether the data cached by the data caching system is consistent with the corresponding data stored in the data storage system, thereby ultimately ensuring the accuracy of the data acquired by the client.

BRIEF DESCRIPTION OF THE DRAWINGS

[39] The above and other objects, features, and advantages of the present disclosure will become more apparent by describing the exemplary embodiments of the present disclosure in greater detail in conjunction with the accompanying drawings, in which the same reference numerals generally represent the same components in the exemplary embodiments of the present disclosure.

[40] FIG. 1 schematically shows the architecture of an existing caching system, Alluxio.

[41] FIG. 2 schematically shows the architecture of a data access system according to the present disclosure.

[42] FIG. 3 is a schematic block diagram of a data reading apparatus which may be used in a client in a computing service according to the present disclosure.

[43] FIG. 4 is a schematic block diagram of a data caching apparatus which may be used in a data caching system according to the present disclosure.

[44] FIG. 5 is a schematic flow diagram of a data caching and reading method according to the present disclosure.

[45] FIG. 6 is a schematic structural diagram of a computing device which may be used to implement the above data reading and caching method according to an embodiment of the present invention.

DETAILED DESCRIPTION

[46] The preferred embodiments of the present disclosure will be described in more detailed with reference to the accompanying drawings. Although preferred embodiments of the disclosure are shown in the accompanying drawings, it should be understood that the disclosure may be implemented in various forms, and should not be limited to the embodiments illustrated here. In contrary, these embodiments are provided to enable the disclosure to be more thorough and complete, and to fully convey the scope of the disclosure to those skilled in the art.

[47] The disclosure provides a process of constructing data caching on demand on the basis of a client in a computing service with regard to the scenarios of big data analyses in, for example, a large-scale distributed system, which basically maintains the original access mode and high availability of data, thereby performing transparent acceleration. At the same time, it solves difficulties in determining consistency of distributed caching and the issue regarding cached data update.

[48] FIG. 2 schematically shows the architecture of a data access system according to the present disclosure.

[49] As shown in FIG. 2, the data access system according to the present disclosure includes a data storage system, a data caching system, and a client in a computing service.

[50] FIG. 3 is a schematic block diagram of a data reading apparatus which may be used in a client in a computing service according to the present disclosure.

[51] As shown in FIG. 3, the data reading apparatus 100 may include a first request apparatus 110, a metadata receiving apparatus 120, and a second request apparatus 130. In some embodiments, the data reading apparatus 100 may further include a first data acquisition apparatus 155, a second data acquisition apparatus 180, and a target data sending apparatus 190.

[52] FIG. 4 is a schematic block diagram of a data caching apparatus which may be used in a data caching system according to the present disclosure.

[53] As shown in FIG. 4, the data caching apparatus 200 may include a request receiving apparatus 210, a consistency determination apparatus 220, a first information sending apparatus 230, and a second information sending apparatus 240.

[54] The functions of these apparatus will be described in detail with reference to the flow diagrams of the data access method as shown in FIG. 2 and FIG. 5.

[55] FIG. 5 is a schematic flow diagram of a data caching and reading method according to the present disclosure.

[56] As shown in FIG. 5, in Step S110, the client in the computing service sends a first request for target data to a data storage system through a first request apparatus 110 for example.

[57] Here, the computing service requests for acquiring metadata of target data from the data storage system through a first request initiated by the client. Metadata of the target data may include, for example, an update time (update_time), a CRC check value of a file (checksum), a file size (length), and so forth of the target data (a data file).

[58] The first request may directly request to acquire metadata of the target data on the basis of a file name such as “oss://bucket/directory/file1” maintained by the data storage system.

[59] In Step S310, the data storage system receives a first request, and sends, in Step S320, the metadata of the requested target data (a data file such as “oss://bucket/directory/file1”) to the client.

[60] In Step S120, the client receives metadata of the target data from the data storage system through the metadata receiving apparatus 120 for example.

[61] In Step S130, the client sends a second request for target data to the data caching system through a second request apparatus 130 for example, the second request including metadata. Accordingly, through the second request, the client requests the data caching system for acquiring a caching location of the target data.

[62] In Step S210, the name space, for example, of the data caching system receives, through the request receiving apparatus 210 for example, a first request for target data to be acquired sent by the client. As stated above, the first request includes metadata of the target data stored in the data storage system.

[63] In Step S220, the name space, for example, of the data caching system determines, through the consistency determination apparatus 220 for example, whether cached target data which is consistent with target data stored in the data storage system is cached in the data caching system on the basis of metadata of the target data.

[64] Here, it is possible to first determine whether cached target data corresponding to the target data is cached on the basis of identification information of target data such as a file name.

[65] Afterwards, in the case where it is determined that cached target data corresponding to the target data is cached, it is determined whether the cached target data is consistent with the target data on the basis of metadata of the cached target data and metadata of the target data. In the case where the metadata of the cached target data is consistent with that of the target data, it is determined that the cached target data is consistent with the target data. Otherwise, it is determined that the cached target data is inconsistent with the target data.

[66] As shown in FIG. 2, the data caching system may maintain the correspondence between target data (files) and metadata acquired when the target data was previously cached. For example, the file name and metadata such as cyclic redundancy check value, file size, and update time of each cached file such as “oss://bucket/directory/file1” can be recorded accordingly for the above consistency determination.

[67] Here, the data caching system may determine whether there is cached target data corresponding to expected target data through a file name (file_name), and accurately determine whether the target data stored in the current data storage system

is consistent with cached target data within the data caching system according to metadata such as an update time (update_time), a CRC check value of a file (checksum), and a file size (length).

[68] In the event of determining that cached target data which is consistent with the target data of the data storage system is cached (consistency) in Step S220, the name space, for example, of the data caching system sends, in Step S230, location information of the cached target data to the client through the first information sending apparatus 230 for example. For example, the information of a cache node that caches this cache target data may be returned to the client as the location information of the cache target data.

[69] Under this circumstance, the client may acquire target data from the data caching system.

[70] Specifically, for example, the client may receive location information of the cached target data, such as the information about a cache node that caches the cached target data, from for example, the name space of the data caching system in Step S140.

[71] In Step S150, the client requests the cached target data from the location (cache node) where the cached target data is located.

[72] In Step S250, a cache node of the data caching system receives the request for the cached target data sent by the client, and in Step S260, the cache node sends the cached target data to the client of the computing service.

[73] In Step S155, the client receives the cached target data from the cache node of the data caching system through the first data acquisition apparatus 155 for example. Accordingly, the acquired cached target data may be directly used for the computing service.

[74] In a case where it is determined in Step S220 that cached target data which is consistent with the target data of the data storage system is not cached (inconsistency), in Step S240, the name space, for example, of the data caching system sends a consistence failure message to the client through the second information sending apparatus 240 for example.

[75] In Step S160, the client receives the consistence failure message from the name space, for example, of the data caching system.

[76] Under this circumstance, the client may acquire the target data directly from the data storage system.

[77] Specifically, for example, a target data request may be sent to the data storage system in Step S170 to request for acquiring the target data.

[78] The data storage system receives the target data request from the client in Step S330, and sends the requested target data to the client in Step S340.

[79] In Step S180, the client receives target data sent from the data storage system through the second data acquisition apparatus 180 for example.

[80] Afterwards, after acquiring target data from the data storage system, the client may further send, in Step S190, the acquired target data to the data caching system through a target data sending apparatus 190, for example. Accordingly, the acquired target data may be directly used in the computing service.

[81] In Step S290, the data caching system receives target data sent from the client, and caches the same to a cache node.

[82] In addition, the name space of the data caching system may further record metadata of target data sent by the client in Step S130 in association with the file name of the cached target data as the basis for later determining the consistency of the target data.

[83] In this way, whether the target data cached by the data caching system is consistent with the target data stored in the data storage system may be directly determined on the basis of metadata of the target data acquired from the data storage system by the client. In the event that there is consistent target data, the client acquires cached target data from a cache node of the data caching system. In the event that there is no consistent target data, the client acquires the target data from the data storage system directly, and sends the acquired target data to the data caching system for caching for later use in reading.

[84] In addition, in the event of determining that cached target data which is consistent with target data of the data storage system is not cached (inconsistent) in Step S220, upon requesting the data storage system for reading target data in Step S170, the client may logically block (LogicBlock) target data (e.g., a file) according to the file size (length) in metadata acquired in Step S120, that is, logically segmenting target data

(a file) into a plurality of logic blocks such as logic block 1, logic block 2, logic block 3, ... logic block n as shown in FIG. 2 according to preset logical data block sizes (Block size), where n is a positive integer.

[85] In this way, the client may directly read corresponding data at a specified location of the data storage system according to logic blocks.

[86] Specifically, when reading data in the data storage system, the client may first determine which logic fragment (a logic block) of the target data (a file) is to be read according to a reading offset (position), and may then directly read, according to the returned metadata in Step S120, data at a specified location (e.g., an initial location of a file + a location corresponding to the offset) without further retrieving metadata services.

[87] On the other hand, after receiving logic blocks of target data from a client, the data caching system may also perform caching according to the logic blocks, for example, caching a logic block on the same caching node, or caching different logic blocks of a same file (target data) on different caching nodes. The data caching system may maintain a table of the correspondence between the logic blocks of the file (target data) and cache nodes. For example, as shown in FIG. 2, for the file “oss://bucket/directory/file1”, the correspondence between a plurality of logic blocks and cache nodes, e.g., <logic block 1, node 1>, <logic block 2, node 1> ..., are recorded. Accordingly, the indexing relationships between respective logic blocks of a file and cache nodes may be established.

[88] In this way, when requesting for the target data again, the client may also determine which logic block of the target data (a file) is to be read according to the offset of data requested currently by the client in the target data, and perform reading directly at a corresponding cache node in the event of determining, based on metadata, that the cached target data of the data caching system is consistent with the target data stored in the data storage system.

[89] So far, the present disclosure provides a highly efficient data caching solution which is capable of automatically checking whether raw data in a data storage system is changed upon reading data, and caching and updating data efficiently, and may also perform blocked caching efficiently when necessary.

[90] In addition, with regard to the characteristics of cloud-based data storage, the life-cycle management of data caching may also be implemented. When it is found that the cached data is inconsistent with the raw data, timely updating can be implemented.

[91] Accordingly, the data reading and caching solution of the present disclosure may not only efficiently solve the cache consistency problem arising from changes in the underlying data stored in the data storage system, but also introduce a caching mechanism without reducing the availability of the system, thereby solving the bandwidth bottleneck caused by a large amount of hot spot data, and improving the overall resource utilization rate of a cluster.

[92] The data caching and reading solution of the present disclosure can implement strong consistency between cached data and raw data. Through metadata information requested by the client in advance, it can accurately determine whether the current target data (a file) is consistent with the cached target data in a timely and accurate manner.

[93] Moreover, the solution proposed in the present disclosure has good compatibility. A data caching mechanism is introduced on the client side, which is compatible with the original direct path access to the data storage system. When initiating a request for target data, the client may directly use a file name (and a path) maintained by the data storage system. As such, this may avoid changes in paths caused by introducing caching, and thus modifications to existing computing frameworks are not necessary.

[94] Additionally, the solution of the disclosure is based on the client's capability for having direct access to the data storage system, and thus the embodiments of the present disclosure may also have high availability. In case that the data caching system fails, the client at an upper layer may determine, at its discretion, whether to wait for restoration of the data caching system or directly connect the data storage system at the distal end and the like according to strategies so as to avoid reduction in the availability of the system due to introduction of a caching system, which more conforms to the large-scale production scenarios.

[95] It should be understood that the data caching and reading solution of the present disclosure is not only applicable to the data storage system of the Object Storage Services (OSS) mentioned above, but can also be applied to various other data storage

systems such as Simple Storage Service (S3), Hadoop distributed file system (HDFS), and Network Attached Storage (NAS).

[96] In fact, various data storage systems at present often record metadata of files and data. Hence, a client requesting data may request metadata corresponding to data from a data storage system, send the same to the caching system for comparison so as to determine the consistency of the cached data. As such, various data storage systems may be used for the data caching and data reading solution of the present disclosure.

[97] FIG. 6 shows a schematic structural diagram of a computing device which may be used to implement the data reading and caching method described above according to an embodiment of the present invention.

[98] With reference to FIG. 6, a computing device 600 includes a memory 610 and a processor 620.

[99] The processor 620 may be a multi-core processor, and may also include a plurality of processors. In some embodiments, the processor 620 may include a general-purpose primary processor and one or more special-purpose coprocessors such as graphics processing unit (GPU) and digital signal processor (DSP). In some embodiments, the processor 620 may be implemented by using customized circuits such as an Application Specific Integrated Circuit (hereinafter referred to as ASIC) or a Field Programmable Gate Array (hereinafter referred to as FPGA).

[100] The memory 610 may include various types of storage units such as a system memory, a read-only memory (ROM), and a permanent storage apparatus, wherein the ROM may store static data or instructions required by the processor 620 or other modules of the computer. The permanent storage apparatus may be a readable and writable storage apparatus, and may be a non-volatile storage device which will not lose instructions and data stored even though the computer is powered off. In some embodiments, a mass storage apparatus (such as a magnetic or compact disk and a flash memory) is used as a permanent storage apparatus. In yet further embodiments, the permanent storage apparatus may be a removable storage device (such as a floppy disk and a compact disk drive). The system memory may be a readable and writable storage device or a volatile readable and writable storage device such as dynamic random-access memory. Instructions and data required when some or all processors are operating may be stored in the system memory. In addition, the memory 610 may

include any combinations of computer readable storage media, including various types of semiconductor memory chips (DRAM, SRAM, SDRAM, flash memory, and programmable read only memory). Magnetic disks and/or compact disks may also be used. In some embodiments, the memory 610 may include a readable and/or writable removable storage device such as laser disk (CD), read-only digital versatile optical disk (e.g., DVD-ROM and bi-layered DVD-ROM), read-only blue-ray disk, ultra-density optical disk, flash memory card (e.g., SD card, min SD card, Micro-SD card, and the like), magnetic floppy disk and so forth. The computer readable storage media do not include carrier waves and transitory electronic signals through wireless or wired transmission.

[101] The memory 610 is stored thereon with executable codes which, when executed by the processor 620, may enable the processor 620 to implement the data reading and caching methods mentioned above.

[102] The data reading and caching solution according to the present invention are described in detailed above with reference to the accompanying drawings.

[103] In addition, the methods according to the present invention may further be implemented as a computer program or computer program product, the computer program or the computer program product including computer program code instructions for executing respective steps above defined in the above methods of the present invention.

[104] As an alternative, the present invention may further be implemented as a non-transitory machine-readable storage medium (or a computer-readable storage medium or a machine-readable storage medium) stored thereon with executable codes (or computer programs or computer instruction codes) which, when executed by a processor of an electronic device (or a computing device, a server and the like), enable the processor to execute respective steps of the above methods according to the present invention.

[105] Those skilled in the art would also understand that various exemplary logic blocks, modules, circuits, and algorithmic steps as described in connection with the present disclosure may be implemented as electronic hardware, computer software or a combination thereof .

[106] The flow diagrams and block diagrams in the accompanying drawings show the architecture, functionality, and operation of possible implementations of systems and methods according to various embodiments of the present invention. In this regard, each block in a flow diagram or block diagram may represent a module, a program segment or a portion of codes including one or more executable instructions for implementing specified logical functions. It should also be noted that in some alternative implementations, the marked functions in the blocks may also occur in a different order from that marked in the accompanying drawings. For example, two successive blocks may be executed basically in parallel as a matter of fact. They may also be executed in a contrary order sometimes, depending on the functions involved. It should also be noted that each block in the block diagrams and/or flow diagrams and a combination of blocks in the block diagrams and/or flow diagrams may be implemented by a special-purpose system based on hardware for executing specified functions or operations, or may be implemented by a combination of special-purpose hardware and computer instructions.

[107] Respective embodiments of the present inventions are described above. The above description is illustrative, not exhaustive, and is not limited to respective embodiments disclosed. Without deviating from the scope and spirit of respective embodiments illustrated, multiple modifications and changes are apparent to those skilled in the art. The terminology used herein is chosen to best explain the principles, practical applications, or improvements to the technology in the marketplace of the embodiments, or to enable others of ordinary skill in the art to understand the embodiments disclosed herein.

CLAIMS

What is claimed is:

1. A data caching method for a data caching system, comprising:

receiving a request for target data to be acquired sent by a client, the request comprising metadata of the target data stored in a data storage system; and

determining, based on the metadata of the target data, whether cached target data which is consistent with the target data stored in the data storage system is cached in the data caching system.

2. The data caching method according to claim 1, further comprising:

sending location information of the cached target data to the client in the case of determining that the cached target data which is consistent with the target data of the data storage system is cached; and/or

sending a consistence failure message to the client in the case of determining that the cached target data which is consistent with the target data of the data storage system is not cached.

3. The data caching method according to claim 2, in the case of determining that the cached target data which is consistent with the target data of the data storage system is not cached, the method further comprising:

receiving, from the client, the target data acquired by the client from the data storage system; and

caching the target data.

4. The data caching method according to claim 1, the step of determining whether the cached target data which is consistent with the target data stored in the data storage system is cached in the data caching system comprising:

determining, based on identification information of the target data, whether the cached target data corresponding to the target data is cached; and

determining, based on metadata of the cached target data and the metadata of the target data, whether the cached target data is consistent with the target data in the

case of determining that the cached target data corresponding to the target data is cached.

5. The data caching method according to claim 1, wherein

the metadata of the target data is acquired by the client from the data storage system; and/or

the metadata of the target data comprises at least one of an update time, a cyclic redundancy check value, and a data size of the target data.

6. A data reading method, comprising:

sending a first request for target data to a data storage system;

receiving metadata of the target data from the data storage system; and

sending a second request for the target data to a data caching system, the second request comprising the metadata.

7. The data reading method according to claim 6, further comprising:

acquiring the target data from the data caching system in a case where it is determined by the data caching system, based on the metadata, that data which is consistent with the target data of the data storage system is cached; and/or

acquiring the target data from the data storage system in a case where it is determined by the data caching system, based on the metadata, that data which is consistent with the target data of the data storage system is not cached.

8. The data reading method according to claim 7, further comprising:

sending the acquired target data to the data caching system after acquiring the target data from the data storage system.

9. A data access system, comprising: a data storage system, a data caching system, and a client, wherein

the client sends a first request for target data to the data storage system;

the data storage system returns metadata of the target data to the client in response to the first request;

the client sends a second request for the target data to the data caching system, the second request comprising the metadata;

the data caching system determines, based on the metadata of the target data, whether cached target data which is consistent with the target data stored in the data storage system is cached in the data caching system;

the data caching system sends location information of the cached target data to the client in the case of determining that the cached target data which is consistent with the target data of the data storage system is cached, the client acquires the target data from the data caching system;

the data caching system sends a consistence failure message to the client in the case of determining that the cached target data which is consistent with the target data of the data storage system is not cached, the client acquires the target data from the data storage system and sends the acquired target data to the data caching system, and the data caching system caches the target data.

10. A computing device, comprising:

a processor; and

a memory stored thereon with executable codes which, when executed by the processor, enable the processor to implement the method of any one of claims 1-8.

11. A non-transitory machine-readable storage medium stored thereon with executable codes, which, when executed by a processor of an electronic device, enable the processor to implement the method of any one of claims 1-8.