An adaptive fault tolerance strategy for cloud storage

Yan Xiai[1], Zhang Dafang [1] and Yang Jinmin[2]
[1] Hunan University, College of Computer Science and Electronic Engineering,
Changsha, 410082, China
[e-mail: dfzhang@hnu.edu.cn]
[2] Hunan Police Academy, Department of Information Technology,
Changsha, 410138, China
[e-mail:yanxiai222@163.com]
*Corresponding author: Zhang Dafang

Received May 26, 2016; revised September 11, 2016; accepted September 27, 2016; published November 30, 2016

Abstract

With the growth of the massive amount of data, the failure probability of the cloud storage node is becoming more and more big. A single fault tolerance strategy , such as replication and erasure codes, has some unavoidable disadvantages, which can not meet the needs of the today's fault tolerance. Therefore, according to the file access frequency and size, an adaptive hybrid redundant fault tolerance strategy is proposed, which can dynamically change between the replication scheme and erasure codes scheme throughout the lifecycle. The experimental results show that the proposed scheme can not only save the storage space(reduced by 32% compared with replication), but also ensure the fast recovery of the node failures(increased by 42% compared with erasure codes).

Keywords: Replication; erasure codes; adaptive changes; fault tolerance; cloud storage

This research was supported by the National Natural Science Foundation of China（61472130, 61471169）, and the the Hunan Provincial Key Laboratory of Network Investigational Technology research projects of China（No. 2016WLZC006)

http://dx.doi.org/10.3837/tiis.2016.11.005 **ISSN : 1976-7277**

1. Introduction

Cloud storage makes a large amount of computation and storage resources connected together to form huge shared virtual storage pool to provide service for the user, it brought low operation and maintenance costs, the scalable performance profile and more efficient storage capacity, more and more users accepted it [1]. However, because of the complexity and openness of cloud storage environment, data failure problem has attracted the attention of the majority of users. For example: in 2011, Ali-cloud server disk failure, in the resumption operation in maintenance process resulted in a loss of data in the period; in 2012, Google email outbreak of large-scale data loss, about 15 million Gmail user data failure [2]. The primary task of cloud storage system is to ensure the high availability and high reliability of data [3,4], thus we must consider the fault-tolerant mechanism to construct a set of high performance and low cost.

Replication and erasure codes are often used in data redundancy fault-tolerant technique. Replication technology (such as GFS, HDFS) is mainly used in early fault tolerant for cloud storage, with the growth of data quantity, the storage space is growing exponentially, cloud storage tolerance gradually transform from the replication to the erasure codes [5]. Erasure codes can effectively reduce the redundancy data space, but the repair takes up more bandwidth and access to more data nodes, decoding complex, there is more time delay. Thus, single data redundancy method has been unable to meet the specific needs of the fault tolerance of different types of users, hybrid data redundancy adaptive method has become a research hotspot in the future cloud storage tolerance.

For the performance of cloud storage fault-tolerant system, users expected it close to the erasure codes means in storage overhead, and close to replication means in the repair efficiency. In fact, all files in the cloud storage fault-tolerant system achieves low redundancy, low bandwidth and high access efficiency are contradictory; the key is how to find balance between them.

Firstly, this paper analyzed replication and erasure codes in detail, then it proposed adaptive switching between replication and erasure codes (ASBRE) based on replication and erasure codes, ASBRE algorithm adaptively switch fault-tolerant strategy according to the amount of file access (including access frequency and file size), using replication fault tolerant for access number is high, using erasure codes fault tolerant for access number is low; it constructed cloud storage fault tolerance framework based on ASBRE algorithm, the experiments were verified and evaluated.

2. Related Work

The primary literature through hybrid redundancy to achieve cloud storage tolerance are: Fan et al [6] found that most of the data access operations on the data after the creation of a short period of time and in HDFS, life cycles over a certain period of files via a one process writes data of erasure codes block, replication data block transformation for erasure coded data block to save storage space, and testing the relationship between the time delay and the performance loss. Zhang Z et al. in Microsoft Research Institute [7] improved only support the replication of HDFS to fully support the replication and erasure codes, the user can choice fault tolerant way according to their own fault tolerance requirements, but for a specific file of single data redundancy method, and the time and the space will not occur. Yadi Ma et al. [8] presented a cloud computing environment replication and erasure codes combining fault tolerance schemes,

using the LRU (least recently used) method to replace, effectively realized the balance of space and time, LRU method existed in the locality, without considering the unit between the access frequency and one-time permanent conversion, data may exist secondary pollution. Aye Chan Ko et al. [9] HDFS presented replication method based on erasure codes , in order to enhance the reliability, hash of previously stored erasure codes, and then copy, between the access delay control in a certain range of threshold than full replication technique to save storage space of the 33%. Roy Friedman et al. [10] presented a flexible replication erasure codes method, for file access to low heat, with a single erasure codes are stored, for file access of high heat, in order to ensure access to high quality, according to the dynamic heat generating replica access, namely for accessing files of high heat is block replication and erasure codes coexistence. Yang Dongri et al. [11] presented replication and erasure codes fusion cloud storage file system fault tolerance mechanism, according to the access frequency to select fault tolerance, and focuses on the analysis of the reliability of the system, but did not on how to determine the file access frequency made multiple description. Song BaoYan et al. [12] presented using RS codes of HDFS storage fault tolerant strategy optimization, using RS code +N/2 replication blocks (n denotes the original HDFS copy block) of the fault tolerance strategy, and through changes to the generated matrix optimization of RS codes in the finite field multiplication computation difficult problem.

The current research shows that the existing research can be divided into three categories: the first category constructed to make copies of the cloud environment and erasure codes switching fault tolerance mechanism, is mainly based on FIFO (First in first out), LRU (Least recently used) switching method, switching is basically static, without considering with the file access heat changes in the adaptive adjustment, and the LFU (Least frequently used) method switch has no detailed description; second is to increase the cloud storage reliability and access efficiency, based on erasure codes, a copy is generated dynamically according to the heat of the file. the erasure codes and copies exist simultaneously for a long time ,which increases the storage pressure; the third is to build two kinds of fault tolerance methods for users to choose, access files are unable to automatically adjust according to the heat changes.

More and more studies show that access frequency of data file presents significant deviation in cloud storage system, and with the increasing storage capacity, the deviation is more obvious, the access frequency of data file is Zipf distribution [17]. If the N file ranking according to the access frequency from high to low, the access frequency of the file i is:

$$F_{(i)} = \frac{1/\sum_{i=1}^{N}\frac{1}{i}}{i}, i = 1,2,3...N$$

Fig. 1 is the access frequency distribution of 200 files, from the figure, less files access frequency is high, and most of the files access frequency is low.

For a few files which frequently accessed is high, user attention is access bandwidth and repair properties, should adopt the replication technology to fault tolerance, and for most of files which the access frequency is low, users are more concerned about the storage space, should adopt the erasure codes technology for fault-tolerant. File access frequency in the life cycle may change at any time, so it needs adaptive switching algorithm to change fault tolerant strategy.

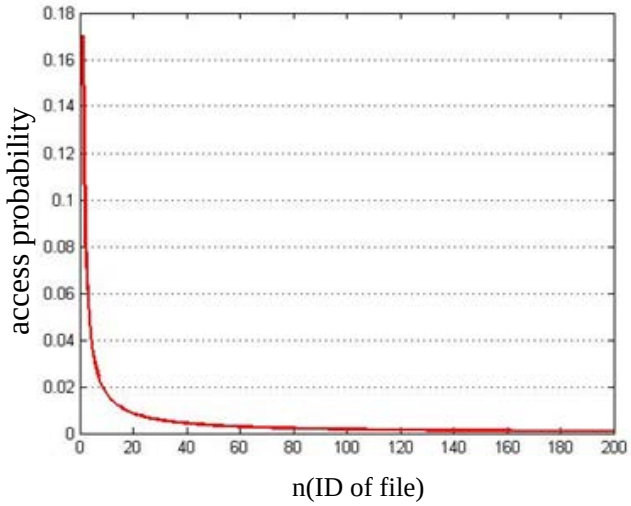


Fig. 1. File data access probability map in cloud storage

3. Replication and erasure codes comparison analysis.

3.1 Replication fault-tolerant technology

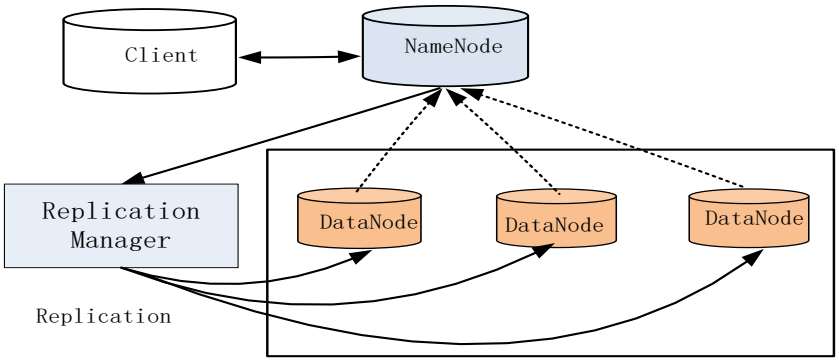


Fig. 2. HDFS fault-tolerant framework

Bhagwat D et al. [13] proposed to save several copies fault tolerance method according to the original data block level. Replication fault tolerance strategy is simple, efficient, easy to implement and deploy, because each replicate data block is a complete backup, data recovery efficiency is high, and can be very good support for concurrent access, so in practice has been widely used. In GFS and HDFS fault tolerant strategy currently is used by the replication technology. HDFS fault tolerance framework is shown in **Fig. 2**.

3.2 Erasure codes fault-tolerant technology

In storage fault-tolerant system based on erasure codes, N storage nodes are divided into two parts of the data nodes and check nodes, and check nodes data computed by the data node, defined erasure codes for a four tuple (n, k, W, k'), n denotes the number of data nodes, K shows the raw file slice number, n-k denotes check node number, w represents each data block contains the number of bits, k' denotes any access to a bar with k' data block can repair the

damaged file [14,15]. **Fig. 3** is the encoding and decoding process of fault tolerance based on erasure codes. In order to enable the system to have the optimal storage efficiency, erasure codes must have the MDS (Maximum Distance Separable) attribute, $k = k'$. There are many types of erasure codes, such as Reed-Solomon encoding (RS encoding), LDPC codes, array codes etc.

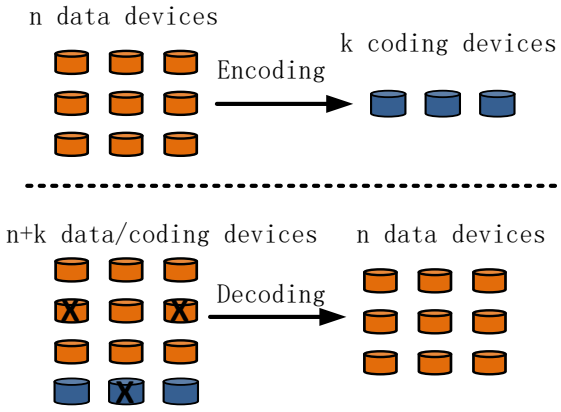


Fig. 3. Encoding and decoding process of fault tolerance based on erasure codes

3.3 Performance comparison of replication and erasure codes

3.3.1 Storage space and repair bandwidth

If the file size is T MB, it is first divided into k blocks (each size of T/k MB), then again for coding and replication, based on erasure codes R(n, k), $(n-k)\frac{T}{k}$ is the size of the redundant data, access to k data block can fix a failure data block, can tolerate $n-k$ data block failure. In the same capability of fault tolerance, if we use the method of replication, at least create $n-k+1$ copies for each data block.

From **Table 1**, block repair bandwidth based on erasure codes is high, but if the failure data is low, using erasure codes can save storage space effectively.

Table 1. Comparison of the storage space and data blocks repair bandwidth in the same fault tolerance capability

Comparison parameters	erasure codes	replication
Fault tolerant ability	$n-k$	$n-k$
storage space	$\frac{nT}{k}MB$	$k(n-k+1)MB$
Block repair bandwidth	$k\times\frac{T}{k}=TMB$	$\frac{T}{k}MB$

3.3.2 Reliability comparison

In the cloud storage fault tolerant system based on replication, the reliability of each data storage node is P, and the redundancy is t, R_{rep} is used to represent the reliability of the whole system, and R_{rep} can be computed by :

$$R_{rep} = 1-(1-P)^t$$

In the cloud storage fault tolerant system based on erasure codes, it is assumed that the reliability of each data storage node is P, the redundancy is $t = n/k$, the reliability of the whole system is expressed by R_{code} [16], and R_{code} can be computed by :

$$R_{code} = \sum_{i=0}^{kt-k} C_{kt}^{i} P^{kt-i} (1-P)^i$$

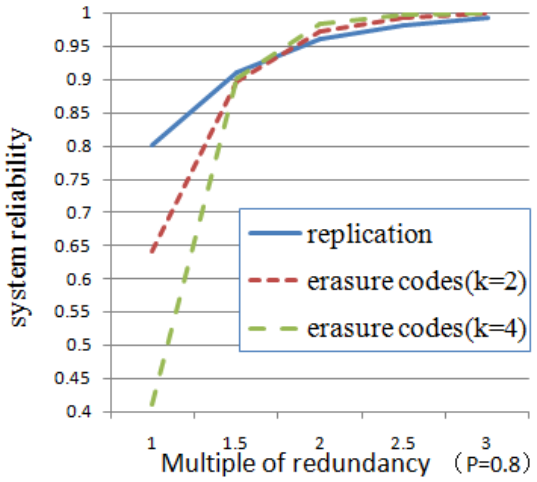


Fig. 4. Reliability comparison of replication and erasure codes

Fig. 4 is the comparison of the fault tolerance system reliability based on the replication and erasure codes, set $P = 0.8$. From the figure, when redundancy is low, the reliability of erasure codes is very low, and the file blocks more, the lower the reliability, but increase with redundancy, erasure codes can provide the reliability growth rate quickly than replication, when $t > 2$, erasure codes reliability is higher, in the same reliability, erasure codes redundancy is low.

4. Adaptive switching algorithms of replication and erasure codes

4.1 Switching methods

In computer systems, there are three common approaches to switching: FIFO (First in first out), LRU (Least recently used), LFU (Least frequently used) [18]. FIFO implement is the most simple, only need swapped out the object which is the first into the system, LRU is considered temporal locality, it swapped out recently not accessed object, LFU according to the access frequency, it swapped out the object which is the minimum access number.

Failure rate of files in cloud storage fault-tolerant system is closely related to access frequency, the client pays close attention to the overall access quality, only consider time to the FIFO, LRU is not suitable for such application scenarios. Therefore, it uses the LFU method with access frequency.

The file in system may have the following conditions:

(1) Some files access frequency is high before a period of time, but in recent period of time the access frequency is low, other files within a short time cannot reach the number of access, and files are not being swapped out frequency table.

(2) Some files access frequency is low before a period of time, but in recent period of time the access frequency is high, in a short period of time it cannot beyond other files access times in the frequency table space, and files are not being swapped in frequency table.

In order to avoid the influence of local access, it uses slicing LFU method. the time period T is divided into some slices of equal intervals, then decide whether to replace erasure codes with replication according to the weight of access frequency. In addition, file failure is also associated with the file size. the files is bigger, the more storage nodes are occupied and therefore the probability of failure will increase exponentially, so the file size must be taken into consideration. In our algorithm, the amount of access is the key factor, which includes access frequency and file size.

When the file is first written into the system, the amount of access is zero, so the file is stored using erasure codes.

4.2 Adaptive switching algorithm

In algorithm, the time axis is divided into a number of cycles(T_1,T_2…T_n…) according to a certain length, we only compare the amount of access between the previous cycle and this cycle to decide whether or not to switch. In this way, the earlier access is blocked, so that the cache pollution problem can be avoided in the LFU algorithm. The main idea of the algorithm : According to the change of the amount of access at a certain cycle, the file that the amount of access is fast-growing can be replace with replication codes as early as possible, the file that the amount of access is fast-falling can be replace with erasure codes as early as possible .

4.2.1 Definition and description

（1） System cycle： $T = \bigcup_{i=1,2,...N} \{T_i\}$.

（2）Global data file set： $D = \bigcup_{k=1,2,...N} \{D_k\}$, where, $D_k =< S_k \mid F_{ki} >$, S_k denote the size of file D_k , F_{ki} denote total access times of file D_k ,after each access， $F_{ki}++$.

（3）the number of times a file is accessed within a time interval: $C_{ki} = F_{ki} - F_{ki-1}$

（4）the replacement threshold of file access: w

（5）the increment of the number of access in the adjacent time interval: $R_{ki} = C_{ki} - C_{ki-1}$

（6）file set of high access table space： $Z = \bigcup_{j=1,2..N} \{Z_j\}$,

（7）the documents in the high access table space are divided into two sets: $Z_{up} = \{D_k \mid R_{ki} > 0\}$, $Z_{down} = \{D_k \mid R_{ki} < 0\}$, Z_{UP} is the rising set， Z_{down} is the set of down.

The recent accessed file is likely to be accessed again, so computing the access frequency is not simple sum, but the recent frequency of access is set greater weight. N_T is used to express

the number of time intervals, and the file access frequency $AF(D_k)$ [19] is computed based on formula (1):

$$AF(D_k) = \sum_{i=1}^{N_T} (C_{ki} \times 2^{-(N_T - i + 1)}) \qquad (1)$$

The larger the file, the more distributed nodes, the longer access delay, it also set greater weight. File access volume $A(D_k)$ is computed by formula (2):

$$A(D_k) = S_k \times AF(D_k) \qquad (2)$$

4.2.2 Switching process

When the *K* file is accessed, if the file is not located in a high access table space, determine if the condition $R_{ki} > 0.and.A(D_k) > w$ is satisfied. Where, $R_{ki} > 0.$, it is the first decision condition, if it is less than the previous access, the maintenance of the status. $A(D_k) > w$ is determine condition whether the amount of access to replace the threshold.

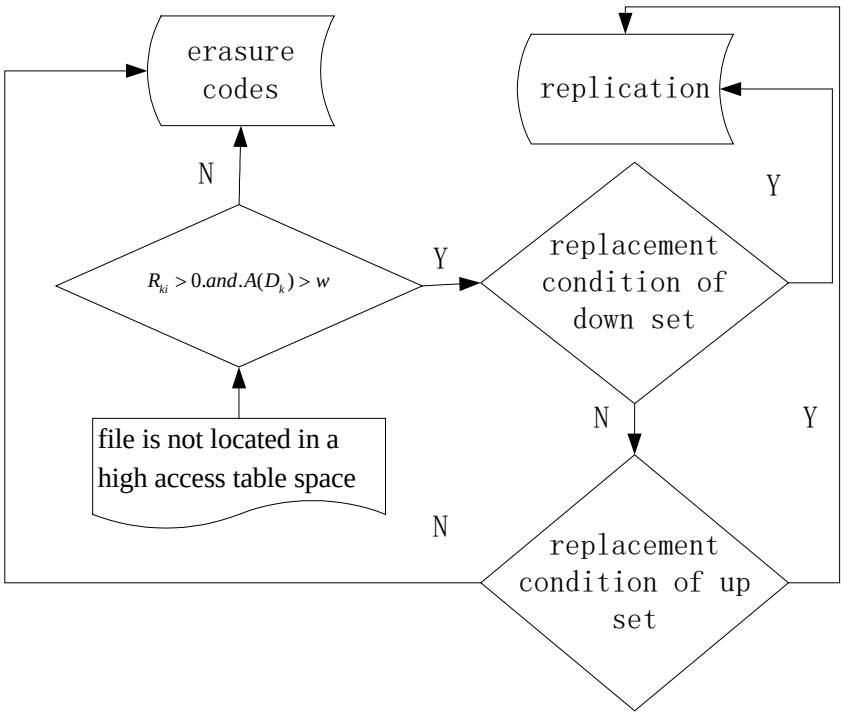


Fig. 5. Switching process

The number of file access is to achieve the threshold of replacement, the file does not necessarily switch, because it is possible that the access of all the files in the high access table are higher than the *K* file. Data show that in the same time interval files are accessed in similar number of times, in order to accelerate the file access volume comparison of *K* file and file in access scale space, it divides the files in high access scale table space into up set Z_{UP} and down

set Z_{down}, files in Z_{down} with tendency to exit the table space, and if you want to replace files in Z_{UP} need higher criterion. Replacement steps are as follows:

（1）find the document $D_{k'}$ which is the least amount of access in Z_{down};

（2）if it is satisfied $A(D_k) > A(D_{k'})$, D_k will be moved into Z_{down}, and $D_{k'}$ be moved out;

（3）find the document $D_{k''}$ which is the least amount of access in Z_{UP};

（4）if it is satisfied $A(D_k) - w > A(D_{k''})$, D_k will be moved into Z_{UP}, and $D_{k'}$ be moved out;

A brief switching process is shown in **Fig. 5**.

The specific switching algorithm is shown in **Table 2**.

Table 2. Adaptive switching algorithm

	0--- erasure codes, 1--- replication
1:	**function** switch ( )
2:	D_k is first write **then** select 0
3:	if D_k is not in the visit capacity table
4:	if $R_{ki} > 0.and.A(D_k) > w$ query the lowest $A(D_{k'})$ in set Z_{down}
5:	if $A(D_k) > A(D_{k'})$
6:	D_k replace with 1, $D_{k'}$ replace with 0
7:	$D_{k'}$ out Z_{down} D_k in Z_{down}
8:	if Z_{down} is not full, then set D_k in Z_{down} D_k replace with 1
9:	endif
10:	else query the lowest $A_{k''i}$ in set Z_{up}
11:	if $A(D_k) - w > A(D_{k''})$
12:	D_k replace with 1 , $D_{k''}$ replace with 0
13:	$D_{k''}$ out Z_{up} D_k in Z_{up}
14:	if Z_{up} is not full, then set D_k in Z_{up} D_k replace with 1
15:	endif
16:	endif
17:	endif
18:	endif
19:	endif
20:	F_{ki} ++

4.3 Cloud storage fault-tolerant architecture based on ASBRE

Cloud storage fault-tolerant architecture based on replication and erasure codes dynamic switching as shown in **Fig. 6**, the system is composed of three parts: the client, metadata server, data storage node.

When writing data of the client, it sends written request to metadata server, when it receives response, the file block write original data block nodes in the data frame, and through the fault-tolerant management module generation erasure codes stored in the erasure codes frame

of data nodes. The client in reading the file, first read the original data block to check, if the data block check failure, look up whether the file is located in the access list in the space, and if so, use replication technology to data repair, if not, use erasure codes decoding technology to data repair.

The metadata server is the core of system architecture; in order to enhance its reliability, it uses the hot backup mode. Two isomorphic servers work at the same time, and a task is sent to the master server and slave server at the same time. In the normal condition, the slave server does not command the work, but it will command the work when physical faults occurred in the master server. Master and slave server generates check-point periodically and to compare the check results (state) are the same or not, if it is the same, indicating that the system is normal, if it is not the same, indicating that at least one of the sever has a soft fault, then the two servers will roll back to last checkpoint state, redo it.

The metadata server consists of metadata management and fault management module. The metadata management module is responsible for processing the data block name space, index records, encoding block to the storage node information mapping etc. Fault management module store a file access list for replication and erasure codes adaptive switching. When the document writing in the first time will be stored as erasure codes, when the access times reached a certain threshold, the three replication stored into the corresponding data nodes and delete erasure codes, when access dropped to threshold, went to generate erasure codes and remove replication blocks of data, if in some time the amount of file access increased and reached the threshold, fault management module can be switch adaptively.

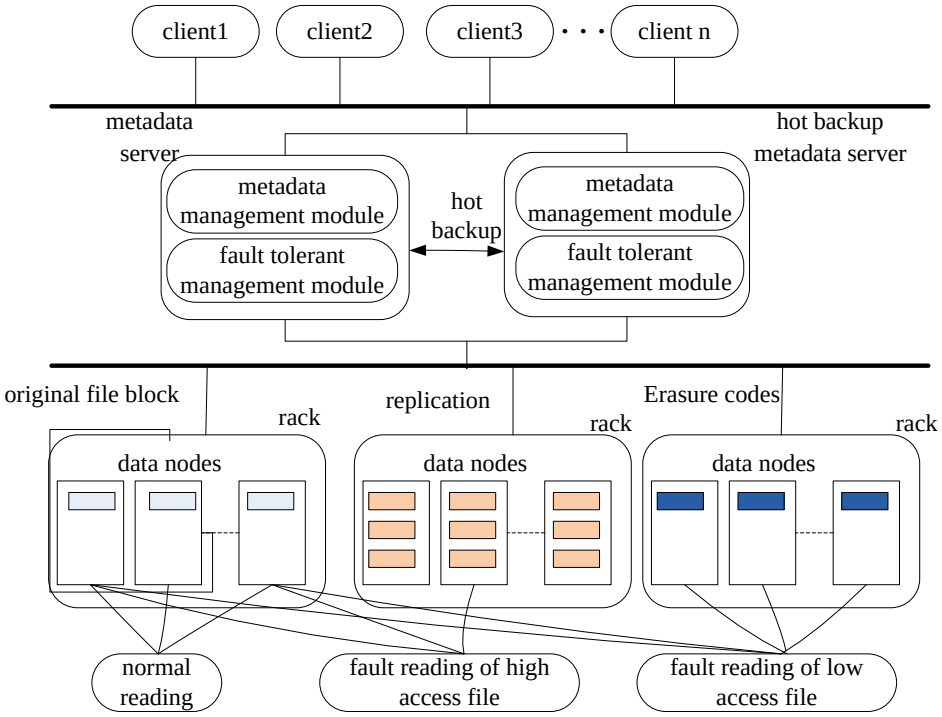


Fig. 6. Cloud storage fault-tolerant architecture based on ASBRE

5 Experimental performance evaluations

5.1 Experimental environment

In order to validate the feasibility of the project, we built a Hadoop cluster storage platform, platform server composed by 2 nameNodes and 12 dataNodes. The server CPU main frequency is 2.6GHz, memory size is 4GB, hard disk size is 1TB, the operating system is ubuntu-11.10, JDK version is 6u30-linux-i586, Hadoop version is 0.20.205, the network bandwidth is 100Mbps.

In experiment, it calls Hadoop benchmark test SWIM (Statistical Workload Injector for MapReduce) [20] to test, according to Facebook, 600 cluster nodes across the record of the access of six months, the access time adjusted according to the ratio of 20:1 to generate simulated access procedures, access 300 GB of storage access files. The fault tolerance module, the replication redundancy is 3, erasure codes using RS encoding (Reed-solomon), R (n, K) = R (9, 6), high access scale to maintain 1000 records. From two aspects of storage cost and repair time, to compare the ASBRE scheme, pure copy replication scheme (Allreplcatoin), pure erasure codes scheme (Allcode), literature [8] proposed the CAROM scheme and literature [10] proposed REC scheme. CAROM scheme uses the LRU methods to switch statically between replication and erasure codes. REC scheme first generates erasure codes for all files, then increase copy replication to the access frequency file in system operation. The comparison with CAROM, REC is maintained under the same reliability.

5.2 Comparison analysis of storage space

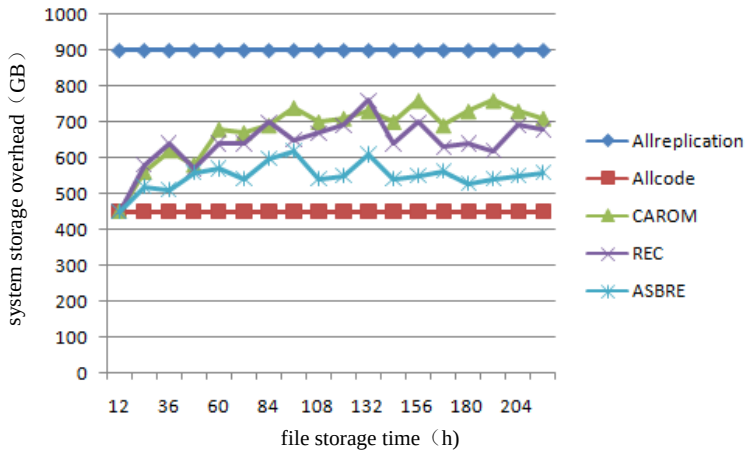


Fig. 7. comparison of system storage overhead

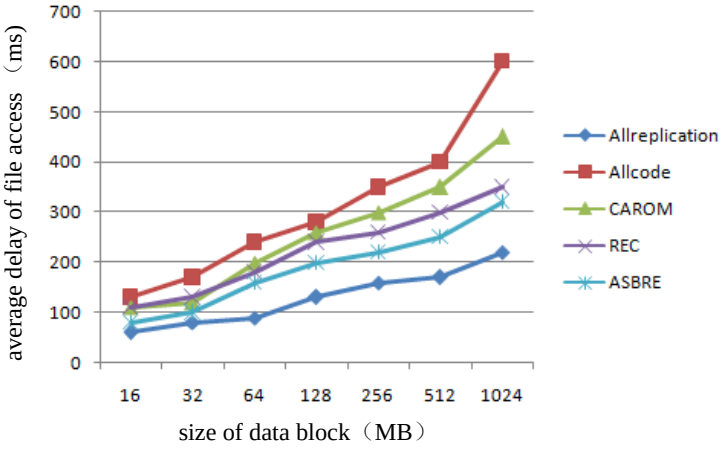
The block size is 64MB, the storage overhead of different fault tolerant schemes in different time as shown in **Fig. 7**.

The redundancy of pure copy replication scheme is 3, the storage overhead constant is 3 times of the original file system, the size is 900GB, using pure erasure codes scheme R (9, 6) storage overhead constant is 1.5 times of the original file system, the size is 450GB. CAROM scheme uses the LRU methods copy and delete code static switch, compared with full replication scheme save an average of 25% of the storage space around, and rec scheme is first erasure codes, system operation on the access frequency of a file copy, adaptive switching with respect to full replication scheme with average storage space saving 30%, ASBRE according to

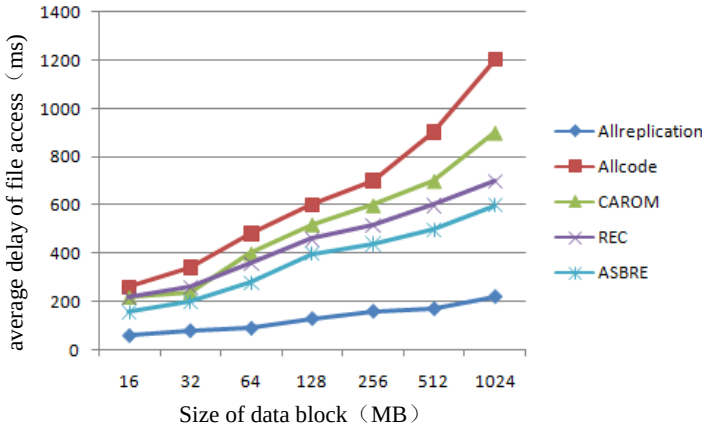
the size of access to replication and erasure codes, relative to full replication scheme with average storage space saving 32%.

5.3 Comparison analysis of fault tolerant effectiveness

In order to validate the system fault tolerance effectiveness, it calculated the file access delay in single node failure and double nodes failure. It randomly selected 1000 different access records to access file, it calculated the average delay of file access in different data block size case, node fault simulation using disk format. The average delay of the client access files as shown in **Fig. 8**.



(a)single node failure



(b)double nodes failure

Fig. 8. average delay of file access in fault

From **Fig. 8** when the storage overhead is not considered, full replication scheme is the best fault-tolerant effectiveness, only a survival node can quickly repair, it is not affect much by the failure node number; Allcode need complex algebra computing and reading more nodes, so either single node failure or double nodes failure, its validity is the worst; ASBRE adopts hybrid redundant fault-tolerant strategy, time is reduced by about 42% compared with Allcode; CAROM, REC, ASBRE solutions belong to hybrid redundancy category, where, ASBRE

considers the size of the file, the use of adaptive switching reduces data pollution, so performance is slightly better than the first two solutions; Fault-tolerant effectiveness of Allcode, CAROM, REC, ASBRE are closely related to data block size and the number of nodes.

5.4 CPU and memory resource consumption comparison

In order to verify the CPU and memory resources consumption, we randomly selected 1000 different access records to access file in different MTBF(Mean Time Between Failures). CPU and memory consumption of Allcode、ASBRE、Allreplcatoin are shown in **Table 3** and **Table 4**(average of each recovery process).

We analyze From **Table 3** and **Table 4**: CPU and memory consumption of Allreplcatoin is the lowest level, but it pay three times the storage overhead cost. For long MTBF, because the ASBRE algorithm needs to maintain the two tables and table lookup, CPU and memory resources consumption is slightly higher than the Allcode. But as the MTBF becomes shorter, the complexity of Allcode is very high because of encoding and decoding, while the ASBRE has a part of the encoding of the replication scheme, so the ASBRE has lower CPU and memory resources consumption than the Allcode.

Table 3. CPU resource consumption comparison(%)

MTBF(Unit:Min)	60	50	40	30	20	10
Allcode	20	23	27	31	38	43
Allreplcatoin	15	17	20	23	25	26
ASBRE	25	27	30	32	36	39

Table 4. Memory resource consumption comparison(%)

MTBF(Unit:Min)	60	50	40	30	20	10
Allcode	45	49	52	55	59	68
Allreplcatoin	37	40	45	47	55	57
ASBRE	50	52	55	57	58	62

6. Conclusions

With the development of cloud computing, there will be more and more users to migrate data to the cloud, cloud storage system fault tolerance is an essential part. The ideal fault tolerance mechanism of users expect: storage costs like erasure codes, and recovery performance like replication technology. The survey shows that, large-scale data access in Zipf distribution in the cloud storage system. This paper carries out in-depth comparative analysis of replication and erasure codes from the storage cost, repair bandwidth and reliability. And then combined with the file size and access frequency, it proposed adaptive cloud storage fault-tolerant mechanism ASBRE based on the full replication and erasure codes, it builds the corresponding cloud storage fault tolerance framework. Experimental results show that the ASBRE scheme is saved storage space 32% by completely copy technology, the repair effectiveness increased 42% by erasure codes, and slightly better than the CAROM and REC also belong to the hybrid data redundancy scheme. According to the analysis, the ASBRE scheme is especially suitable for fault tolerance system in which the frequency of failure is high and the file size is large. But ASBRE needs each server CPU utilization and memory usage overall to be slightly higher.

According to the data access distribution, this paper presents hybrid redundancy mechanism of adaptive switching, improves the access quality, effectively ease the contradiction between storage and efficient access. But it don't optimized for a single redundancy, such as the finite field multiplication problem, dynamic replication number problem, and further work also include solving the load balancing problem.

References

[1] YAO Wenbin, HAN Si, and LI Xiaoyong, "Security sharing scheme for encrypted data in cloud storage," *Journal on Communications*, Vol.36, NO.10, PP.1-8, October 2015. Article (CrossRef Link)

[2] WANG Hongyuan, ZHU Liehuang, and LILONG Yijia, "Group provable data possession with deduplication in cloud storage," *Journal of Software*, Vol,27, No, 6, PP.1-16. June 2016. Article (CrossRef Link)

[3] MICHELE A and STEFANO C, "Replication vs erasure coding in data centric storage for wireless sensor networks," *Computer Networks*, Vol. 77, No.11, PP.42-55. November 2015. Article (CrossRef Link)

[4] LUO J Q, MOCHAN S, and XU L H, et al, "Efficient encoding schedules for XOR-based erasure codes[J]," *IEEE Transactions on Computers*, Vol.63, No.9, PP.2259-2272. September 2014. Article (CrossRef Link).

[5] KHAN O，BURNS R，and PLANK J S, "Rethinking erasure codes for cloud file systems: minimizing I/O for recovery and degraded reads," in *Proc. of USENIX． FAST 2012: 10th USENIX Conference on File and Storage Technologies*． San Jose, CA，PP.1-14, 2012. Article (CrossRef Link).

[6] FAN B, TANTISIRIROJ W, and XIAO L, et al, "DiskReduce: RAID for data-intensive scalable computing," in *Proc. of the Petascale Data Storage Workshop* (PDSW 2009). Portland: ACM Press, PP. 6-10, 2009. Article (CrossRef Link).

[7] ZHANG Z, DESHPANDE A, and Ma X, et al, "Does erasure coding have a role to play in my data center," *Microsoft research* MSR-TR-2010, 52, 2010. Article (CrossRef Link).

[8] YADI M, THYAGA N, and KRISHNA P N, et al, "An ensemble of replication and erasure codes for cloud file systems," in *Proc. of 2013 Proceedings IEEE INFOCOM*, Italy, PP.1276-1284, 2013. Article (CrossRef Link).

[9] AYE C K and WINT T Z, "Fault tolerant erasure coded replication for HDFS based cloud storage," in *Proc. of 2014 IEEE Fourth International Conference on Big Data and Cloud Computing*, Sydney, PP.104-109, 2014. Article (CrossRef Link).

[10] ROY F,YOAV K, and AMIR K, " Replicated erasure codes for storage and repair traffic efficiency," in *Proc. of 14-th IEEE International Conference on Peer-to-Peer Computing*, London, PP.1-10, 2014 Article (CrossRef Link).

[11] YANG Dongri , WANG Ying, and LIU Peng, "Fault-tolerrant mechanism combined with replication and error correcting code for clould file systems," J*ournal of Tsinghua University(SCI &Technol)*,Vol.54, No1, PP.137-144, January 2014. Article (CrossRef Link)

[12] SONG Baoyan, WANG Junlu, and WANG Yan, "Optimized storage strategy research of HDFS based on vandermonde code," *Chinese Journal of Computers*, Vol.38, No.9, PP.:1825-1836, Septmeber 2015 Article (CrossRef Link)

[13] BHAGWAT D, POLLACK K , and LONG D D, et al, "Providing high reliability in a minimum redundancy archival strorage system," in *Proc. of the 14th IEEE International Symposium on MASCOTS*. PP.413-421,2016. Article (CrossRef Link)

[14] LACAN J and FIMES J., "Systematic MDS erasure codes baesd on vandermonde matrices," *IEEE Communications Letters*, Vol. 8, No. 9, PP. 570-582, Septmeber 2004. Article (CrossRef Link).

[15] PLANK J S, "A tutorial on reed-solomon coding for fault-tolerance in RAID-like systems," *Software: Practice and Experience*, Vol.27, No.9, PP.995-1012, Septmeber 1997. Article (CrossRef Link).
[16] ZHANG Zhiyun, MENG Huiping, and LI Dun, et al, "Research on dynamic replication redundancy storage strategy based on erasure code," *Computer Engineering and Design*, Vol.35, NO. 9, PP.3085-3090, Septmeber 2014. Article (CrossRef Link).
[17] DANIEL F and HAIPING X., "A raid-based secure and fault-tolerant model for cloud information storage," *International Journal of Software Engineering & Knowledge Engineering*, Vol. 23, No. 5, PP.627-654, May 2013 Article (CrossRef Link).
[18] LI Yong, WU Fei, and CHEN Fujie., "Design and management of large scale hierarchical VOD storage system," *Journal of Software*, Vol.10, No.4, PP.355-358, April 1994. Article (CrossRef Link).
[19] DU Z, HU J and CHEN Y, "Optimized Qos-aware replica placement heuristics and applications in astronomy data grid," *Journal of Systems and Software*, Vol.84, No.7, PP.1224-1232, July 2011. Article (CrossRef Link).
[20] CHEN Y, GANAPATHI A, and GRIFFITH R, et al, "The case for evaluating mapReduce performance using workload suites," in *Proc. of 19th Annual IEEE International Symposium on Modelling*, Singapore, PP.390-399, 2011. Article (CrossRef Link).

Xiai Yan received his M.S.degree in Software Engineering from the Hunan University,China, in2007,.Currently, he is working in Hunan Police Academy as professor. He has been a Ph.D.Candidate in Computer Science and Technology for four years in Hunan University. His research interests include software engineering, fault-tolerance computing.

Dafang Zhang received his Ph.D. degree in applied mathematics from Hunan University, in 1997. He is a professor in the College of Computer Science and Electronics Engineering of the University. His current research interests include wireless network, distribute computation,and DPI..

Jinmin Yang received his Ph.D. degree in computer from Hunan University, in 2004. He is a professor in the College of Computer Science and Electronics Engineering of the University. His current research interests include software engineering, fault-tolerance computing.