A Fast Recovery Method for Single Disk Failure Based on EVENODD

Feng Xiao, Di Fan, and Dan Tang
(Corresponding author: Dan Tang)

School of Software Engineering, Chengdu University of Information Technology
No. 24, Xue-Fu Road, Chengdu 610225, China
(Email: 398879332@qq.com)

(Received Nov. 23, 2017; Revised and Accepted Apr. 28, 2018; First Online Jan. 12, 2019)

Abstract

In an array storage system based on EVENODD, the traditional single disk failure recovery requires reading all the remaining data. Even if the hybrid recovery method is used, the theoretical limit of the data reading cost can be reduced by only 25%.To further reduce the read overhead for single disk failure recovery, improve the reliability of storage system, this paper proposes an improved method for EVENODD single disk failure recovery. This method is to reduce the amount of data that needed to read during recovery by adding a local redundant.The experimental results show that the improved method has significantly improved the recovery time comparing with the EVENODD.

Keywords: EVENODD; Reading Overhead; Single Disk Failure

1 Introduction

In recent years, with the rapid development of society and technology, the amount of data in the world is increasing at an explosive rate. As a result, the scale of storage systems is also growing accordingly. As the scale increases, the stability of the storage system is also facing many great challenges. The main reason is that while the number of disks is increasing, the probability of data loss caused by disk failures is also increasing. In order to protect the data, it is necessary to recover the data even after the disk failed. Therefore, the time of data recovery and the amount of the data that needed to be read during the recovery process become the key to the stability of the system.

Array codes is a kind of erasure code which is widely applied. Its advantage is the calculation is XOR so the encoding and decoding speed are pretty fast. The typical array codes include EVENODD [5], RDP [3], STAR [7], RTP [4] and so on. STAR and RTP are expanded by EVENODD and RDP respectively. However, there are still many problems to be solved. Aiming at the problem of update efficiency, Chen put forward the Inverse Code [2] by constructing low density matrix. Aiming at the problem of data storage reliability and expansibility, Teng proposed the Random Array Code [10]. In order to accelerate the speed of data recovery, Sun proposed a new coding theory named ESRC [9], which is based on local check and global check. In order to solve the problem that the MDS array codes have a length limitation, Huang proposed a new code named Symmetric code [6].

EVENODD code is one of the most commonly used array codes in storage systems which can tolerate two errors. Its principle is to ensure the integrity of the data by adding two redundant disks. EVENODD can recover the original data correctly after any two of the disks failed.

For EVENODD coding, the advantages are that the structure is simple, and the encoding and decoding process are based on XOR operation, so the speed is very fast. However, one of the major defect is that the single disk failure recovery requires too much data to read. All the remaining raw data is need to be read. Considering that the recovery is based on a single check disk, if a local check disk is computed only by some of the data disks, then when the data disk fails, only part of the data disks that generate the local check disk can be recovered. Therefore, a fast recovery method for single disk failure based on EVENODD is proposed.

2 Introduction of EVENODD

2.1 Encoding Process

The encoding process of EVENODD is based on an original data array, and the size of the original data array is $(p-1)*p$, p is a prime number. The data block in the array is assumed to be $a_{i,j}$, where i and j represent row and column coordinates of elements. The coding formulas are shown in Equations (1),(2)and (3):

$$a_{i,p} = \bigoplus_{j=0}^{p-1} a_{i,j} \tag{1}$$

$$S = \bigoplus_{j=1}^{p-1} a_{p-1-j,j} \quad (2)$$

$$a_{i,p+1} = S \oplus \left(\bigoplus_{j=0}^{p-1} a_{<i-j>_p,j} \right). \quad (3)$$

The final coding process is shown in Figure 1.

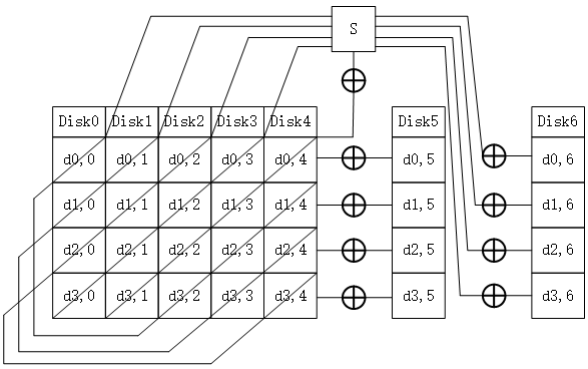

Figure 1: Horizontal redundancy and diagonal redundancy calculation process

Finally, an array of $(p-1)*(p+2)$ with two redundant columns is obtained.

2.2 Single Disk Failure Recovery

The traditional single disk failure recovery method can be divided into two categories according to the position of the failed disk: data disk failure and redundant disk failure.

The method of recovering from data disk failure is to determine the location of the failed disk first, and then read out all the other data blocks in data disks and horizontal redundant disk, and finally run XOR operation through these data, get the lost data disk. The data required to be read during recovery is shown in Figure 2, the "X" indicates that the data block is lost and "O" means the data block need to be read.

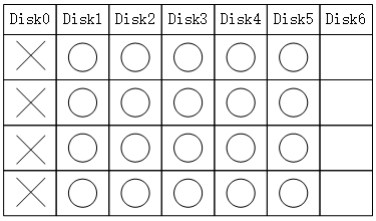

Figure 2: Data to be read from single disk failure recovery

The recovery of the redundant disk failure is exactly the same as the coding process, that is, when the horizontal or diagonal redundant disk fails, it can be rebuilt by using encoding algorithm.

Therefore, for the method of traditional single disk failure recovery, whether the data disk failed or redundant disk failed, the number of data disks required is p.

2.3 Single Disk Failure Hybrid Recovery Algorithm

In order to reduce the recovery read overhead of single disk failure, literature [1] proposes that, for EVENODD when a data disk fails, the horizontal redundancy and diagonal redundancy can be used simultaneously to recover. Because there are some repeated data blocks needed in using different methods to recover, so the recovery read overhead can be reduced by caching these blocks. Theoretical deduction is also carried out in this paper, and it is found that the best recovery method is to recover half of the data using horizontal redundancy, and recover the other half data using diagonal redundancy. And the lower bound of the theoretical recovery read overhead is 3/4 of the total data, which means it can save 25% of the recovery read overhead. Similarly, a hybrid recovery algorithm for RDP single disk failure is proposed in the literature [11]. The lower bound of the read overhead is also 3/4 of the total data.

As it's shown in Figure 3, when the Disk0 fails, the first two data blocks are recovered by horizontal redundant columns, and the latter two data blocks are recovered by diagonal redundancy. Among them the circle indicates that the data blocks used by horizontal redundant columns while square means the data blocks used by diagonal redundant columns, and the data blocks which have both symbols mean they are repeated.

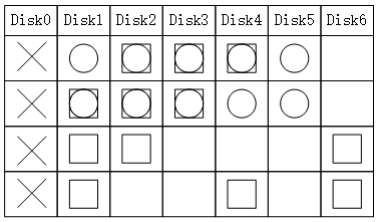

Figure 3: Using hybrid algorithm to recover single disk failure

3 Local Repair Method of EVENODD

In order to reduce the recovery read overhead of EVENODD code in the single disk failure further, this paper has modified this code. The transformation method is called EVENODD local repair method. And the expanded EVENODD is called LREVENODD (Local Repairable EVENODD).

3.1 Encoding Process

Local repairable EVENODD code is an extension of EVENODD. A redundant column x is added on the basis of the original EVENODD with two redundant columns. Redundant column x is obtained by horizontal XOR computation of the previous $\frac{p-1}{2}$data columns. The computation formula is Equation (4):

$$a_{i,p+2} = \bigoplus_{j=0}^{\frac{p-1}{2}} a_{i,j} \tag{4}$$

Encoding process of Local repairable EVENODD code is shown in Figure 4. The same shape of white blocks generate the corresponding black check blocks:

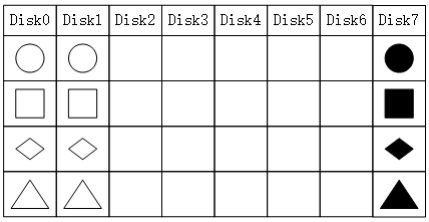

Figure 4: Encoding process of LREVENODD

3.2 Encoding Overhead

For the EVENODD local repair method, encoding process only adds a step in the traditional process of EVENODD encoding. In the calculation of horizontal redundant columns, after XORing the first half of the data, the XOR result as local redundant column and save to disk X. Although this step increases the storage overhead and the encoding time, it does not increase the computational complexity. Moreover, with the increase of prime number, the proportion of increased overhead to the overall coding overhead will continue to decline. Because the value p in the real environment represents the number of disks in the storage cluster, it is generally larger, so the overhead caused by the increased local redundant column x is relatively smaller.

3.3 Correctness Proof

Based on the definition in the upper section, the former $p+2$ column of the Local Repairable EVENODD code is the EVENODD code, so the nature of the former $p+2$ column is no longer discussed at next, and only the impact of the new local redundant column x loss is discussed

3.3.1 Only the X Column is Missing

If only the X column is missing, it means that all the data in the front $p+2$ columns is in good condition, and the X column can be generated by encoding Equation (4). At this point, the data integrity can be guaranteed.

3.3.2 X and Another Column Y Missing

First, suppose that another missing column is a data column, then the horizontal redundant column is intact. So the data column can be recovered by XOR of all the remaining data columns and the horizontal redundant column. The computation formula is Equation (5):

$$a_{i,y} = \bigoplus_{j=0}^{p} a_{i,j} \tag{5}$$

When all the data columns have been rebuilt, the situation is consistent with the loss of the X column, and then the X column can be generated again by encoding.

If another missing column is a redundant column, which means all data columns are in good condition. So at first, the missing horizontal redundant column or diagonal redundant column can be regenerated by EVENODD encoding. If the horizontal redundancy column is lost, the coding formula of recovery is Equation (1). If the missing redundant column is diagonal redundant column, the coding formula of recovery is shown in Equation (2) and Equation (3).

After restoring the horizontal redundant column or diagonal redundant column, only the X column still needs to be recovered. So the X column can be restored by Equation (4).

Through the analysis above, it can be concluded that after adding a local redundant column x, the new array code can still recover any two errors.

3.4 Relative Properties

3.4.1 MDS Property

It can be found that although the extended EVENODD code adds a redundancy, but it can only tolerate any two errors (in some cases it can tolerate three errors) and lose the MDS property. The reason is that the local fault tolerance is in conflict with MDS property. For example, RS codes remove the lost blocks and decode them with the rest of the global blocks, so the locality is poor, but there is good code distance, and has the best fault tolerance. EVENODD and all the MDS codes are like this. The literature [8] pointed out that if adds some local redundant columns, it means that there exists errors(missing columns is not related to these columns) that the local redundant columns can not be solved. Thus it can be seen that the local fault tolerance and minimum code distance are conflicted, which means it is inevitable to improve the local tolerance while sacrificing MDS property.

Furthermore, it is proved that the Local Repairable Codes with additional local redundant columns can be infinitely approximated to MDS codes. Then, for a MDS code with only one redundant column, it can be proved that, when the value p is increased, the LREVENODD code can also be close to the MDS codes.

3.4.2 The Relationship Between New Redundancy and Overhead Reduction

By discussing MDS property, it can be concluded that increasing local redundant columns and ensuring MDS property are conflicted. The paper [12] pointed out that appropriate data redundancy can improve system performance. Therefore, it is necessary to explore the relationship between redundant columns number and reduced overhead to determine how much local redundancy is added to achieve optimal.

In order not to increase the complexity of the encoding process, the redundant columns are selected by sharing the original data columns. The number of local redundant columns is n, and the percentage of saved overhead is cp, then the ratio of new redundancy and overhead is $l = \frac{cp}{n}$. Assuming that the additional local redundancy columns are 1, 2, 3, 4 and 5, the obtained results are shown in Table 1:

Table 1: Relationship between new redundancy and overhead reduction

symbol	value				
cp	50%	67%	75%	80%	83%
n	1	2	3	4	5
l	50%	33%	25%	20%	17%

It can be seen from the table that with the increase of the number of local redundant columns, the saved read overhead is also increasing, but the reduction of the overhead caused by the new redundancy is decreasing, that is, the efficiency is decreasing. And adding redundant columns will bring more storage overhead, and will also influence the efficiency of encoding and decoding. So the choice of adding only a local redundant column can significantly reduce the single disk failure recovery read overhead and does not increase the excessive storage overhead and lead to system performance degradation.

4 Decoding Process

4.1 Single Disk Failure

For traditional EVENODD coding, the recovery of single disk failure (non redundant columns) is done by reading the remaining data columns and the horizontal redundant columns to xor. For the EVENODD local repair method, there are two situations to repair a single disk failure.

First, suppose the data column fails, then it can be divided into two cases, one is the missing data column is in the first half, and the other is in the second half. If the missing column is in the first $\frac{p-1}{2}$ columns, it is necessary to read the first half part of the remaining data column and the new local redundant column x for XOR restoration. The data block needed to recover is shown in Figure 5, the "X" indicates that the data block is lost, "O" means the data blocks which are required in recover.

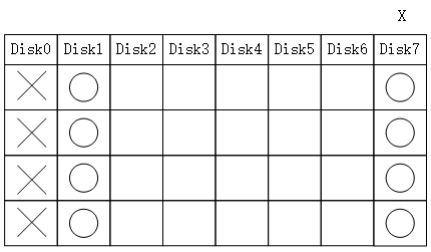

Figure 5: Data need to read when missing column is in first part

If the missing column is in the posterior $\frac{p+1}{2}$ columns, the local redundancy column X and the horizontal redundant column p can be read first, then XOR them, and the result is the XOR of the data column in the second half. Then read the data columns in the second half, and XOR with the previous result, to get the missing data column. The data blocks needed for recovery are shown in Figure 6.

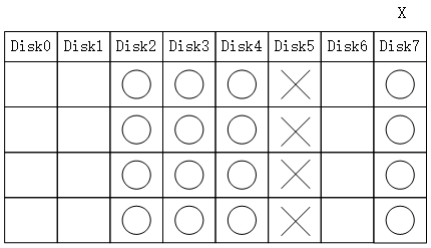

Figure 6: Data need to read when missing column is in second part

It can be concluded that the amount of data needed to recover from single disk failure is $\frac{p-1}{2}$ and $\frac{p+1}{2}$ respectively in two cases. When p is increasing, the reduced read overhead tends to approach 50%.

If the local redundant column fails, it is necessary to read the first $\frac{p-1}{2}$ data columns for restoration, which is completely equivalent to the encoding process. This situation does not exist in the EVENODD codes, and the recovery read overhead remains $\frac{p-1}{2}$. When p is increasing, the proportion of local redundant column fails to all single disk failures will continue to decrease.

In case of horizontal redundant column fails, only the posteriors $\frac{p+1}{2}$ data columns and the local redundant column are needed to recover. Read overhead is only $\frac{p+1}{2}+1$. When p is increasing, it can reduce the data reading overhead approaching to 50%. The data blocks required for the recovery of horizontal redundant column failure are shown in Figure 7.

For the case of diagonal redundant column fails, the EVENODD local repair method is consistent with the traditional EVENODD recovery algorithm, that is, read all

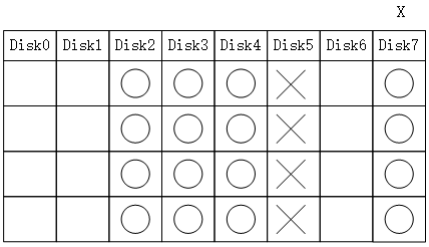

Figure 7: Data need to read when horizontal redundant column is missing

data columns to encode and get the redundant columns, and the read overhead is p. Similarly, the ratio of this failure to all single disk failures will continue to decrease as p continues to increase. Therefore, for single disk failure recovery, assuming that the probability of each disk failure is consistent, the average overhead of the LREVENODD single disk failure recovery is shown in Equation (6):

$$c = \frac{\frac{p-1}{2} * \frac{p-1}{2} + \frac{p+1}{2} * \frac{p+3}{2} + \frac{p+3}{2} + p + \frac{p-1}{2}}{p+3} \tag{6}$$

According to the analysis above, it can be concluded that the single disk failure recovery overhead of LREVENODD approach nearly $\frac{p}{2}$, and Figure 8 is a theoretical comparison between the EVENODD and the LREVENODD single disk failure recovery overhead when p takes different values.

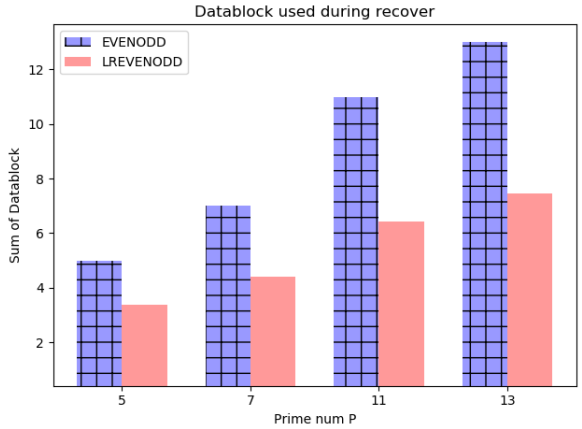

Figure 8: Comparison of single disk failure recovery reading overhead

Compared with the traditional EVENODD recovery algorithm, the proposed method saves 50% of the data read overhead in the single disk failure recovery, while the recovery of redundant columns is almost consistent with the traditional EVENODD read overhead. Because the number of data columns in practical applications is much larger than the number of redundant columns, it can be concluded that the method reduces the data read overhead by nearly 50% for single disk failure recovery.

4.2 Double Disk Failures Recovery

4.2.1 Two Data Column Missing

Like traditional recovery methods, recovery is done by reading the remaining data and using horizontal redundancy and diagonal redundancy when two data columns failed in LREVENODD. Data read overhead is shown as Equation (7):

$$cp = p - 2 + 2 = p. \tag{7}$$

4.2.2 One Data Column and Local Redundant Column Missing

First, by reading the remaining data columns and horizontal redundant column to recover the lost data column, and cache the xor result of the previous $\frac{p-1}{2}$ data columns(don't xor the missing column, but xor it with the cache result when restored it), the final xor result is the missing local redundant column.

As shown in Figure 9, when the second data column and the local redundant column are lost, the remaining intact data columns and horizontal redundant columns are first read to recover the lost second data column. At the same time, the XOR results are saved in the process. After calculating the second column data, xor the result with the second data column, and the local redundant column is obtained.

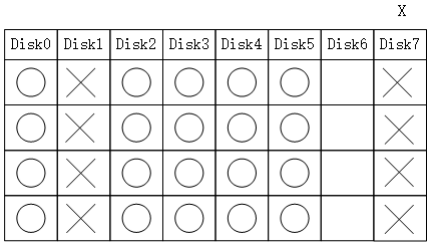

Figure 9: Data need to read when local redundant column and one data column are missing

As shown in Figure 9, the read overhead of recovery in this case is still p columns.

4.2.3 One Data Column and Horizontal or Diagonal Redundant Column Missing

The recovery method in this situation is consistent with the traditional EVENODD recovery method. The reading overhead is p.

4.2.4 Two Redundant Columns Missing

The recovery method in this situation is consistent with the encoding process. So obviously, the reading overhead is p.

In summary, the EVENODD local repair method can greatly reduce the read overhead and improve the performance when the single data disk fails. For other cases,

the EVENODD local repair method does not produce too much overhead, and is basically consistent with the traditional EVENODD.

4.3 Triple Disk Failures Recovery

Although the extended EVENODD loses its MDS property, it can still be successfully restored for most of the three disk failures. Then analysis in which case the three disk failure can be restored when $p = 7$.

The three redundant columns in the extended EVENODD can be represented by the following equations:

$$\begin{aligned} a_{i,p} &= \bigoplus_{j=0}^{p-1} a_{i,j} \quad (i = 0, 1, ..., p-2) \\ a_{j,p+1} &= S \oplus \left(\bigoplus_{j=0}^{p-1} a_{<i-j>_p,j} \right) \quad (i = 0, 1, ..., p-2) \\ a_{i,p+2} &= \bigoplus_{j=0}^{\frac{p-1}{2}} a_{i,j} \quad (i = 0, 1, ..., p-2). \end{aligned}$$

Meanwhile, assume the probability of each disk failure is γ.

4.3.1 Three Data Disks Missing

When three disks are lost and the three disks are all data disks, the probability is shown as Equation (8).

$$\lambda = C_p^3 * \gamma^3 \tag{8}$$

When the lost disks are all data disks, for the horizontal redundant disk, each of the equations contains 3 unknown quantities. For diagonal redundant disk, each equation contains at least 2 unknown quantities. The problem then turns into whether the situation can be translated into two disks lost, i.e. whether a disk exists in the three lost disks can be recovered by the local redundant disk. We discuss the situation according to the position distribution of the three data disks:

First, the three column data disks are in the front $\frac{p-1}{2}$ disks or back $\frac{p+1}{2}$ disks, the probability of such a situation is shown as Equation (9).

$$\lambda = C_{\frac{p-1}{2}}^3 * \gamma^3 + C_{\frac{p+1}{2}}^3 * \gamma^3 \tag{9}$$

First, suppose that three disks are in the previous $\frac{p-1}{2}$ disks, as shown in Figure 10. Assuming each column represents a data disk, for a local redundant disk, each of its equations contains 3 unknown quantities and is not solvable. And if three disks are in posteriors $\frac{p+1}{2}$ disks is also similar, because the disks in the back are not a linear relationship with the local redundant disk, it is impossible to recover the local redundant disk. And only through the other two redundant disks to restore has too many unknown disks, so this situation is not solvable.

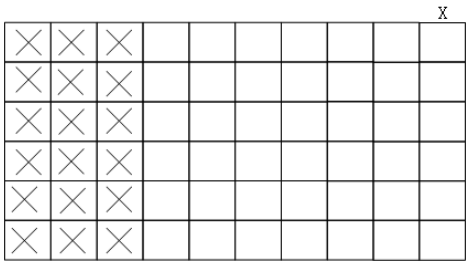

Figure 10: Can not recover from missing three data disks in same part

Assuming there is a disk in the three data disks not in the same half with the other two disks, the probability of this kind of case is shown as Equation (10).

$$\lambda = C_{\frac{p-1}{2}}^1 * C_{\frac{p+1}{2}}^2 * \gamma^3 + C_{\frac{p-1}{2}}^2 * C_{\frac{p+1}{2}}^1 * \gamma^3 \tag{10}$$

First, suppose that there is a lost disk in the previous $\frac{p-1}{2}$ disks and the other two lost disks are in the second half, then there is only one unknown quantity in each equation that constitutes a local redundant disk. So it can be solved. When the first lost disk is restored, the problem is transformed into an ordinary double disk failure, so the situation is solvable.

If there are two disks lost in the previous $\frac{p-1}{2}$ disks and one in the second half. Then the equations of the horizontal redundant disk and the equations that constitute the local redundant disk can be subtracted, and a set of equations about the latter $\frac{p+1}{2}$ disks can be obtained, shown as Equation (11).

$$a_{i,p} - a_{i,p+2} = \bigoplus_{j=\frac{p-1}{2}}^{p-1} a_{i,j} \tag{11}$$

In these equations, each has only one unknown quantity and can be solved. When the missing data is restored, the problem is transformed into an ordinary double disk failure again. So the situation is also solvable.

4.3.2 Two Data Disks and One Redundant Disk Missing

When there are three disks lost, two of them are data disks, and the other one is redundant disk, and the probability of this case is shown as Equation (12).

$$\lambda = C_p^2 * C_3^1 * \gamma^3 \tag{12}$$

This situation can be divided into several types according to the redundant disk type, in which the probability of each case is shown as Equation (13).

$$\lambda = C_p^2 * \gamma^3 \tag{13}$$

First, if the lost disk is a local redundant disk, it is equivalent to the normal EVENODD double disk failure

recovery. After the completion of the restoration, the local redundant disk can be restored by encoding.

If the lost disk is horizontal redundant disk, then it can be classified according to the distribution of the data disks.

When the two disks are in the first half, the property of EVENODD shows that there is always only one unknown $a_{i,j}$ in the equations of the diagonal redundant disk. Through diagonal redundancy can calculate this unknown quantity $a_{i,j}$. And then there is only the unknown quantity $a_{i,k}$ in the same row, and it can be calculated through the local redundancy. And after $a_{i,k}$ is calculated, there is only one unknown quantity on the line and it located which slope is 1. And by repeating these steps, it can solve all the unknowns.

If the two disks are in the second half, because the local redundancy is independent of the latter $\frac{p+1}{2}$ disks, they can only be solved by diagonal redundancy. The previous discussion shows that although the diagonal redundancy can solve an unknown quantity $a_{i,j}$, but the other unknown quantity in the same line depends on other redundancy to solute. While there is no other redundant existence at the same time, so it can not be solved. The probability of this case is shown as Equation (14).

$$\lambda = C^2_{\frac{p+1}{2}} * \gamma^3 \tag{14}$$

Assuming there is a data disks in the previous $\frac{p-1}{2}$ disks while the other is in the second part, the probability of such a situation is shown as Equation (15).

$$\lambda = C^1_{\frac{p-1}{2}} * C^1_{\frac{p+1}{2}} * \gamma^3 \tag{15}$$

There is only one unknown quantity existed in each equation that constitutes a local redundant disk. So it can be solved. When the first lost disk is restored, there is still one data disk and horizontal redundant disk are missing, which can be restored by the traditional RDP double disk recovery method. So this situation is solvable.

If the lost disk is redundant disk, then it can be classified according to the distribution of the data disk.

Assuming the two disks are both in the previous $\frac{p-1}{2}$ disks or in the second half, the probability of this happening is shown as Equation (16).

$$\lambda = \left(C^2_{\frac{p-1}{2}} + C^2_{\frac{p+1}{2}}\right) * \gamma^3 \tag{16}$$

When the two lost data disks are in the first half, even though there are two redundant disks, the two redundant disks are linearly related, so the two missing disks can not be restored. And when the two disks are in the second half, there is only horizontal disk related to the lost disk, so it can not be recovered either.

If one of the two disks is in the first half and the other in the second half, they can be restored. Because the lost disk in the first half can be restored by the local redundant disk, the remaining lost data disk can be restored by the horizontal redundant disk. Finally, the diagonal redundant column is restored by the encoding algorithm.

4.3.3 One Data Disk and Two Redundant Disks Missing

Assuming there are three disks lost, one is a data disk, and the other two are redundant disks, the probability of this case is shown as Equation (17).

$$\lambda = C^1_p * C^2_3 * \gamma^3 \tag{17}$$

In consideration of that only one data disk is lost, if the intact redundant disk is a horizontal redundant disk or a diagonal redundant disk, the data disk can be restored, and then the lost redundant disks can be restored by encoding.

Assuming the remaining redundant disk is local redundant disk, the probability of such a situation is shown as Equation (18).

$$\lambda = C^1_p * \gamma^3 \tag{18}$$

This situation needs to be classified according to the location of the data disk.

If the lost data disk is in the front half, the data disk can be recovered through a local redundant disk, and then other redundant disks are restored by encoding.

And assuming the missing data disk is in the second half, the probability of such a situation is shown as Equation (19).

$$\lambda = C^1_{\frac{p+1}{2}} * \gamma^3 \tag{19}$$

Because the local redundant disk is linear independent of the latter part, and it is impossible to recover the lost data disk only by diagonal redundancy, so the situation is not solvable.

4.3.4 Three Redundant Disks Missing

If the lost three disks are redundant disks, they can be restored directly by encoding again.

From the above discussion, we can get the sum of the probability of the non-solvable cases, which are shown as Equation (20).

$$\begin{aligned}\lambda \quad = \quad & \Big(C^3_{\frac{p-1}{2}} + C^3_{\frac{p+1}{2}} + C^2_{\frac{p+1}{2}} + \\ & (C^2_{\frac{p-1}{2}} + C^2_{\frac{p+1}{2}}) + C^1_{\frac{p+1}{2}}\Big) * \gamma^3\end{aligned} \tag{20}$$

And the probability of three disk failures is shown as Equation (21).

$$\lambda = C^3_{p+3} * \gamma^3 \tag{21}$$

Therefore, the proportion of non-recoverable cases of all three disk failures is shown as the following equation:

$$\Omega \quad = \quad \frac{\left(C^3_{\frac{p-1}{2}} + C^3_{\frac{p+1}{2}} + C^2_{\frac{p+1}{2}} + (C^2_{\frac{p-1}{2}} + C^2_{\frac{p+1}{2}}) + C^1_{\frac{p+1}{2}}\right)}{C^3_{p+3}}$$

From the above equation, it can be concluded that when the p tends to infinity, the probability of unable to

recover in the three disk failure is $\Omega = \frac{1}{4}$, that is to say, for the LREVENODD, 75% of the three disk failures can be recovered. And because in practice, the probability of three disk failures is much lower than the single disk failure, so the EVENODD extended by using the local repair algorithm is closer to MDS codes.

5 Performance Test

This section tests and compares the performance of the single disk failure recovery between EVENODD and LREVENODD. The machine parameters in the experimental environment cluster are: CPU Intel Core i7-3632, memory 8GB, disk 500GB, test file size 10M.

5.1 Encoding Test

Figure 11 gives the coding time of EVENODD and LREVENODD when the prime takes different values.

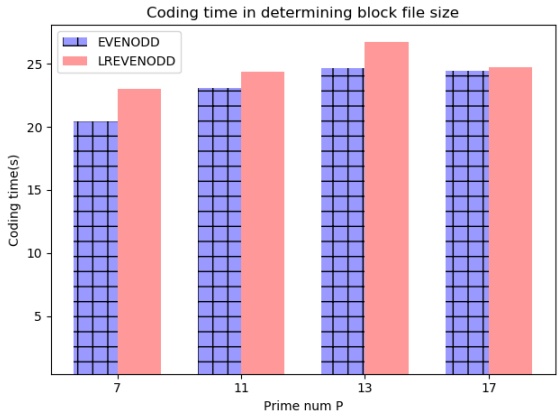

Figure 11: Coding time comparison

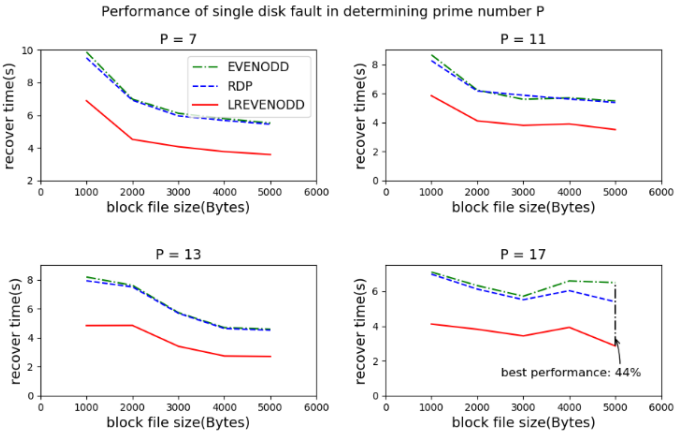

Figure 12: Single disk failure recover time in determing p

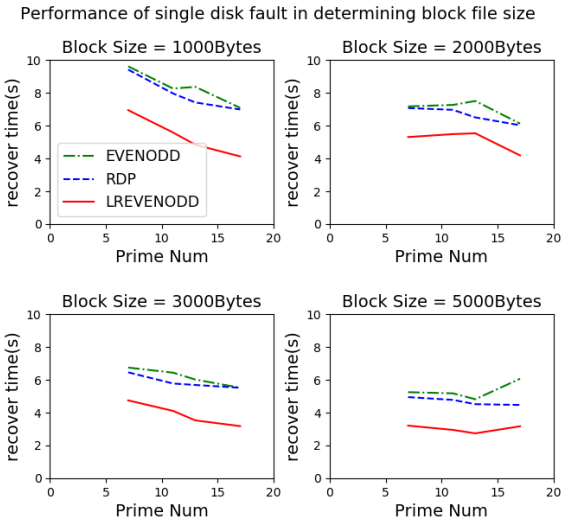

Figure 13: Single disk failure recover time in determing block file size

5.2 Single Disk Failure Recovery Test

Figure 12 gives a single disk failure recovery time when p is determined and block size is different. And Figure 13 gives a single disk failure recovery time when p takes different values.

Through the test results, it can be found that when the p increases and the block file size increases, the recovery overhead of single disk failure decreases. When p=17 and block file size is 5000Bytes, the overhead is reduced by 44%. And Figure 14 gives a comparison of the file block used in single disk failure recovery process between EVENODD and LREVENODD.

As shown in Figure 14, it can be found that when the number of file blocks increases, the number of data blocks required for single disk failure recovery in LREVENODD is closer to 50% of the number in EVENODD.

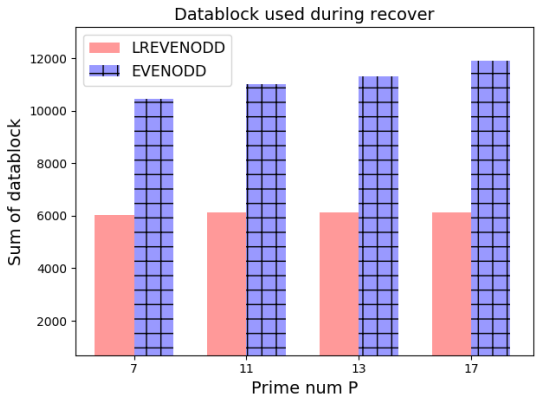

Figure 14: Single disk failure recovery reading overhead

5.3 Double Disk Failure Recovery Test

Figure 15 shows the recover time in double disk failure when the prime number p is different while block size is determined.

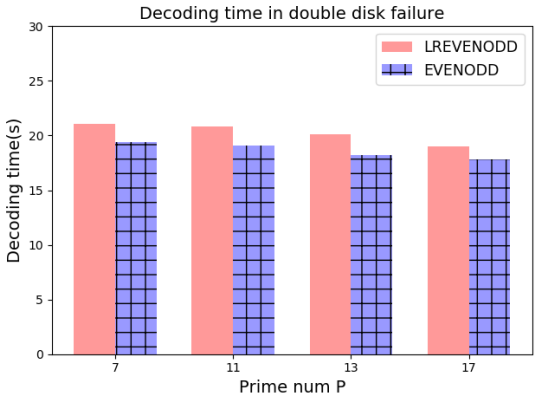

Figure 15: Double disk failures recover time

5.4 Triple Disk Failure Recovery Test

Figure 16 shows the unrecoverable times when 10000 triple disk failures occured with different prime number.

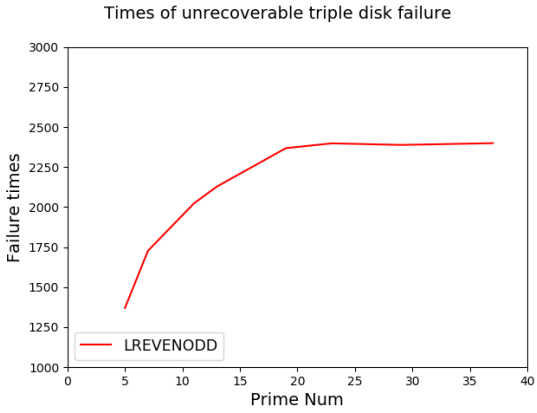

Figure 16: Times of unrecoverable triple disk failure

By Figure 16 knowable, when p increases, the number of non recoverable times is gradually approaching 2500 times, which is 25% of the total number of triple disk failure. Figure 17 shows the average recovery time of the triple disk failure when the file size is large and the prime number p takes different values.

6 Conclusions

Aiming at the problem that the read overhead of single disk failure recovery in EVENODD is too large, this paper proposes a method to reform it. Combining LRC with

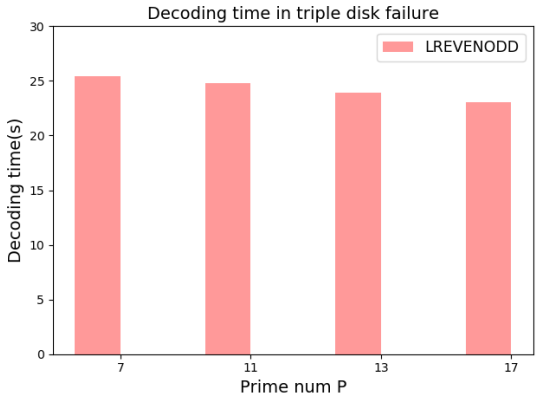

Figure 17: Decoding time in triple disk failure

RS codes, local repair method is used to improve EVENODD codes. The local repair method reduces the read overhead of single disk failure recovery by adding a new redundant column. The experimental results showed that the improved EVENODD code with local repair method can effectively reduce the read overhead of single disk failure recovery and improve the system performance. In fact, this method is not only applicable to EVENODD, but also can be used for RDP, STAR and RTP codes and many other slope codes, and the relevant proof and test need to be further completed.

Acknowledgments

This study was supported by the National Natural Science Fund of China 61501064 and Science and technology program of Sichuan 2018GZ0099. The authors gratefully acknowledge the anonymous reviewers for their valuable comments and teacher's hard cultivation.

References

[1] Q. Chang, Y. Xu, L. Xiang, and Y. Pan, "A hybrid recovery algorithm for single disk failure in evenodd (in chinese)," *Computer Applications and Software*, vol. 28, no. 6, pp. 15–18, 2011.

[2] L. Chen, D. Yuan, P. Teng, and X. Wang, "Inverse code: A low density mds horizontal array code that can accommodate 3 errors (in chinese)," *Journal of Sichuan University (Engineering Science Edition)*, no. 5, pp. 135–142, 2017.

[3] P. Corbett, B. English, A. Goel, T. Grcanac, S. Kleiman, J. Leong, and S. Sankar, "Row-diagonal parity for double disk failure correction," in *Usenix Conference on File and Storage Technologies*, pp. 1–1, 2004.

[4] A. Goel and P. Corbett, "Raid triple parity," *Acm Sigops Operating Systems Review*, vol. 46, no. 3, pp. 41–49, 2012.

[5] R. Hu, G. Liu, and J. Jiang, "An efficient coding scheme for tolerating double disk failures," in *IEEE International Conference on High Performance Computing and Communications*, pp. 707–712, 2010.

[6] Z. Huang, *Research on MDS Array Code in Fault-Tolerant Storage System*, PhD thesis, Huazhong University of Science and Technolog, 2016.

[7] C. Huang and L. Xu, "Star: An efficient coding scheme for correcting triple storage node failures," in *Conference on Usenix Conference on File and Storage Technologies*, pp. 15–15, 2005.

[8] M. Sathiamoorthy, M. Asteris, D. Papailiopoulos, A. G. Dimakis, R. Vadali, S. Chen, and D. Borthakur, "Xoring elephants: Novel erasure codes for big data," *Proceedings of the Vldb Endowment*, vol. 6, no. 5, pp. 325–336, 2013.

[9] D. Sun, *Research on Verification Update and Repair Optimization Technology for Fault-Tolerant Storage System*, PhD thesis, University of Science and Technology of China, 2017.

[10] P. Teng, J. Zhang, L. Chen, and X. Wang, "Random array code: A raid storage disaster recovery method with high disaster tolerance and scalability," *Journal of Sichuan University (Engineering Science Edition)*, vol. 49, no. 3, pp. 110–116, 2017.

[11] L. Xiang, Y. Xu, J. C. S. Lui, and Q. Chang, "Optimal recovery of single disk failure in rdp code storage systems," *Acm Sigmetrics Performance Evaluation Review*, vol. 38, no. 1, pp. 119–130, 2010.

[12] L. Xu and J. Bruck, "Highly available distributed storage systems," *A Caltech Library Service*, vol. 249, pp. 307–330, 1999.

Biography

Feng Xiao, master candidate. He is currently an Master student in Chengdu University of Information Technology, Chengdu, China. His research interests include coding theory and database theory.

Di Fan, master candidate. She is currently an Master student in Chengdu University of Information Technology, Chengdu, China. Her research interests include coding theory and information security.

Dan Tang received his Ph.D. degree from Graduate University of Chinese Academy of Sciences (CAS), Beijing, China in 2010. He is currently an associate professor with Chengdu University of Information Technology, Chengdu, China. His research interests include coding theory and secret sharing scheme.