A Dynamic Erasure Code Based on Block Code

Yulong Meng, Lingling Zhang, Dong Xu,
Zhiyun Guan, Long Ren

Harbin Engineering University
Harbin 150001, China
xudong@hrbeu.edu.cn

Abstract

Aiming at the problem that the parameters of existing erasure codes are poorly dynamic in adjustment, this paper proposes a dynamic erasure code based on the idea of block codes, named DLRC (Dynamic Local Reconstruction Code). We encode the data blocks in group according to four parameters which proposed in DLRC. The dynamic balance of storage overhead, fault-tolerance ability, fault-tolerance rate, and reconstruction overhead is achieved by adjusting the values of parameters which can meet the different performance requirements of the distributed storage systems. The experiments show that DLRC achieves different proportions of four main performances mentioned above by adjusting the values of four parameters. In addition, we retain global parity blocks and put them in the calculation of local parity blocks which makes all coded blocks can be locally reconstructed within the group. In this case, DLRC ensures high fault tolerance ability and fault tolerance rate while reducing the overall reconstruction over-head.

1 Introduction

In recent years, with the transformation of traditional industries and the rapid development of information technology such as cloud computing, big data, Internet of things and artificial intelligence, the amount of data has grown geometrically. According to IDC [1], the amount of global data is expected to reach 44 ZB by 2020. In China, the data volume will reach 8060 EB which is 18 percentages of global data volume. With the advent of the big data era, the storage cluster sizes have increased, interconnected storage devices have increased, and storage node failures have become more frequent. Therefore, how to ensure the reliability of data in distributed storage systems has become an urgent problem to be solved. At present, there are two main types of fault-tolerant technologies commonly used in distributed storage systems. One is multi-copy technology, which is reconstructed through replication, but the cost is very expensive. The other is erasure code technology, which is reconstructed through coding [2–4]. Compared with multi-copy technology, erasure code can provide the same or even higher fault tolerance ability while significantly reducing storage overhead [5, 6]. So that erasure code technology has attracted extensive attention and become a research hotspot in the storage field.

Among all erasure codes, the block code is a new type of erasure code based on the idea of grouping to reduce the reconstruction overhead. LRC [8] used in Microsoft Cloud Storage System (Windows Azure Storage) [7] reduces the number of coded blocks need to be read during reconstruction by coding data blocks into local parity blocks in groups. This reduces the bandwidth and I/O required for reconstruction. However, LRC can only optimize the reconstruction efficiency of a single data block. For the global parity block's failure, it can only be reconstructed by all data blocks. In this case, the amount of data blocks that needs to be read is large. Literature [9] proposed a multi-level block code, named Pyramid. When multiple blocks in a group failed, Pyramid can effectively reduce the average reconstruction overhead, but its storage overhead is large. Miyamae et al. proposed SHEC (Shingled Erasure Code) [10], which makes each group overlap like tiles on the roof. SHEC effectively avoids the situation that the amount of downloaded data increases sharply while the number of failed coded blocks increases. However, since SHEC only has the local parity blocks but no global parity blocks, its fault tolerance ability is bound to be reduced.

Based on the idea of block code, this paper proposes a dynamic erasure code, named DLRC (Dynamic Local Reconstruction Code), which can adjust the values of parameters to achieve dynamic balance of storage overhead, fault tolerance ability, fault tolerance rate and reconstruction overhead which meets different performance requirements of distributed storage systems.

International Conference on Embedded Wireless
Systems and Networks (EWSN) 2019
25–27 February, Beijing, China

ISBN: 978-0-9949886-3-8

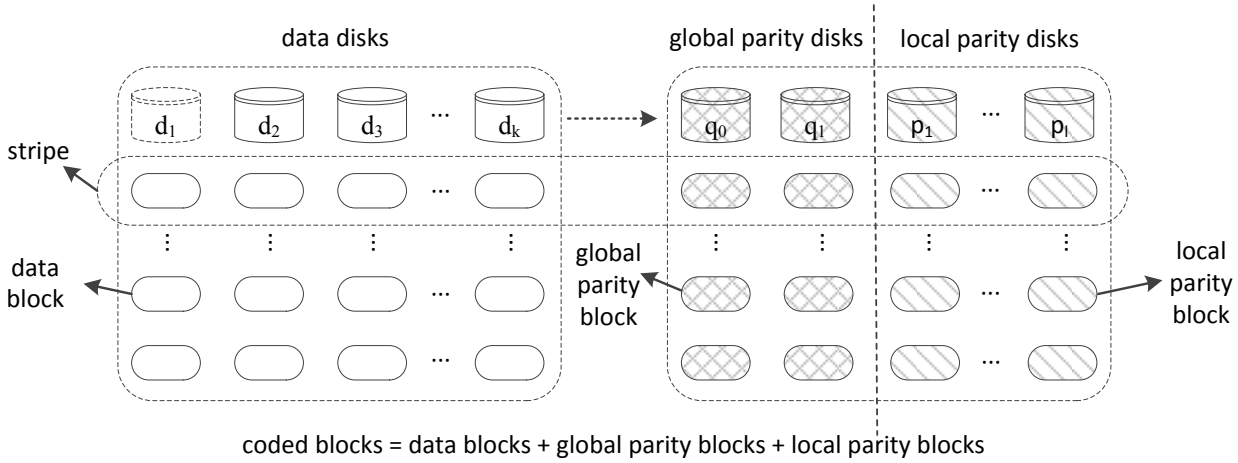

Figure 1. Relationship between disks, stripes, and coded blocks in DLRC

2 Encoding Structure of DLRC

2.1 Relationship between Disks, Stripes, and Coded Blocks in DLRC

Before introducing the encoding structure of DLRC, we will briefly introduce the relationship between disks, stripes and coded blocks in DLRC. As shown in Figure 1, since DLRC adopts the horizontal coding mode, the data blocks and the parity blocks are respectively stored on different disks. The data blocks on the same stripe form a fault-tolerant group, and encode to the corresponding global parity blocks and local parity blocks, then stored in the corresponding parity disks. Data blocks and parity blocks are collectively referred to as coded blocks.

2.2 Encoding Structure of DLRC

DLRC includes four parameters k, m, n, l, recorded as DLRC(k, m, n, l). That, k represents the number of data blocks, m represents the number of global parity blocks, n represents the number of coded blocks involved in calculating each local parity block and l represents the number of local parity blocks. The four parameters need to satisfy the condition: (n×l) can be divisible by (k+m). The following describes the encoding structure of DLRC by two specific samples DLRC(10, 2, 4, 3) and DLRC(10, 2, 6, 4) as examples.

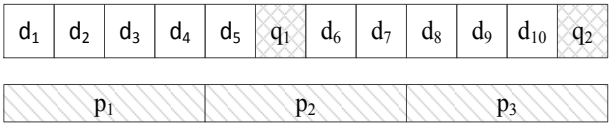

Figure 2. Encoding structure of DLRC(10 , 2 , 4, 3)

As shown in Figure 2, DLRC(10, 2, 4, 3) has a total of 10 data blocks d_1 d_{10}. Two global parity blocks q_1 and q_2 are generated by these 10 data blocks. Then they are placed respectively after the fifth data block and the last data block to participate the subsequent operation in group. In addition, every 4 coded blocks generate a local parity block. So DLRC(10, 2, 4, 3) has a total of 3 local parity blocks. Each coded block participates in the calculation of one local parity block.

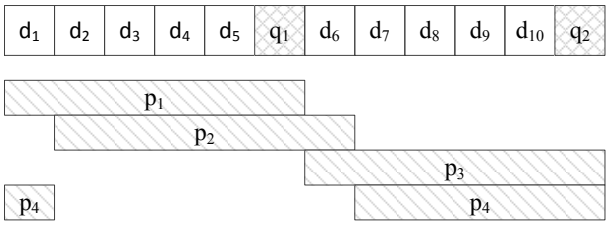

Figure 3. Encoding structure of DLRC(10, 2, 6, 4)

DLRC(10, 2, 6, 4) in Figure 3 is similar to DLRC(10, 2, 4, 3). There are also 10 data blocks and 2 global parity blocks in the code. The difference is that every 6 coded blocks are selected to generate a local parity block in DLRC(10, 2, 6, 4). It has a total of 4 local parity blocks, and each coded block participates in the calculation of 2 local parity blocks.

3 Encoding Algorithm of DLRC

Based on the configuration parameters and encoding structure of DLRC, this section gives the specific encoding formula and the coefficient selection conditions when DLRC is theoretically able to be reconstructed. Taking DLRC(6, 2, 4, 2) as an example, which is shown in Figure 4, DLRC(6, 2, 4, 2) contains a total of 4 parity blocks, so that at most 4 blocks fail simultaneously can be repaired. However, since DLRC does not satisfy the MDS [11] property, not all 4-failure cases can be reconstructed. For example, when d_1, d_2, d_3 and q_1 fail at the same time, it cannot be reconstructed in theory. The reason is that only the parity blocks p_1 and q_2 associate with the three missing data blocks. That

is equivalent to computing three unknown parameters using two linearly independent equations. As we know, that is impossible to find a unique solution. For the theoretically reconfigurable failure mode, it is still necessary to ensure the coefficients of the encoding matrix to achieve true reconstruction. In the following, we use DLRC(6, 2, 4, 2) as an example to discuss the encoding coefficients in the theoretically reconfigurable 4-failure modes.

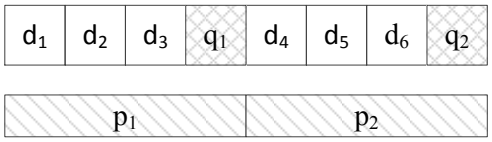

Figure 4. Encoding structure of DLRC(6, 2, 4, 2)

For DLRC(6, 2, 4, 2) shown in Figure 4, the encoding equations are constructed as follows. Through these formulas, DLRC(6, 2, 4, 2) can tolerate any modes of 3-failure simultaneously:

$$
\begin{aligned}
q_1 &= \alpha_1 d_1 + \alpha_2 d_2 + \alpha_3 d_3 + \beta_1 d_4 + \beta_2 d_5 + \beta_3 d_6 \\
q_2 &= \alpha_1^2 d_1 + \alpha_2^2 d_2 + \alpha_3^2 d_3 + \beta_1^2 d_4 + \beta_2^2 d_5 + \beta_3^2 d_6 \\
p_1 &= d_1 + d_2 + d_3 + q_1 \\
p_2 &= d_4 + d_5 + d_6 + q_2
\end{aligned}
$$

The above encoding equations can be converted into the following forms:

$$
\begin{aligned}
q_1 &= \alpha_1 d_1 + \alpha_2 d_2 + \alpha_3 d_3 + \beta_1 d_4 + \beta_2 d_5 + \beta_3 d_6 \\
q_2 &= \alpha_1^2 d_1 + \alpha_2^2 d_2 + \alpha_3^2 d_3 + \beta_1^2 d_4 + \beta_2^2 d_5 + \beta_3^2 d_6 \\
p_1 &= (1+\alpha_1) d_1 + (1+\alpha_2) d_2 + (1+\alpha_3) d_3 + \beta_1 d_4 + \\
&\quad \beta_2 d_5 + \beta_3 d_6 \\
q_2 &= \alpha_1^2 d_1 + \alpha_2^2 d_2 + \alpha_3^2 d_3 + \left(1+\beta_1^2\right) d_4 + \\
&\quad \left(1+\beta_2^2\right) d_5 + \left(1+\beta_3^2\right) d_6
\end{aligned}
$$

The matrix form of the above equations is:

$$
\begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \\ q_1 \\ q_2 \\ p_1 \\ p_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \alpha_1 & \alpha_2 & \alpha_3 & \beta_1 & \beta_2 & \beta_3 \\ \alpha_1^2 & \alpha_2^2 & \alpha_3^2 & \beta_1^2 & \beta_2^2 & \beta_3^2 \\ 1+\alpha_1 & 1+\alpha_2 & 1+\alpha_3 & \beta_1 & \beta_2 & \beta_2 \\ \alpha_1^2 & \alpha_2^2 & \alpha_3^2 & 1+\beta_1^2 & 1+\beta_2^2 & 1+\beta_3^2 \end{bmatrix} * \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \end{bmatrix}
$$

In the following, we will take 4 data blocks failure as an example to discuss the con-ditions that the coding coefficients need to satisfy.

(1)3 of the data blocks are in the same group, and 1 data block belongs to the oth-er group. Assuming d_1, d_2, d_3 and d_4 failed, the encoding matrix can be expressed as:

$$
\begin{bmatrix} d_5 \\ d_6 \\ q_1 \\ q_2 \\ p_1 \\ p_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \alpha_1 & \alpha_2 & \alpha_3 & \beta_1 & \beta_2 & \beta_3 \\ \alpha_1^2 & \alpha_2^2 & \alpha_3^2 & \beta_1^2 & \beta_2^2 & \beta_3^2 \\ 1+\alpha_1 & 1+\alpha_2 & 1+\alpha_3 & \beta_1 & \beta_2 & \beta_3 \\ \alpha_1^2 & \alpha_2^2 & \alpha_3^2 & 1+\beta_1^2 & 1+\beta_2^2 & 1+\beta_3^2 \end{bmatrix} * \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \end{bmatrix}
$$

If we want this failure mode to be reconstructed, it is equivalent to that the column vector has a solution where the data blocks locate. The column vector has a solution equivalent to the determinant's value of the coding matrix is not zero. That is,

$$
\begin{vmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \alpha_1 & \alpha_2 & \alpha_3 & \beta_1 & \beta_2 & \beta_3 \\ \alpha_1^2 & \alpha_2^2 & \alpha_3^2 & \beta_1^2 & \beta_2^2 & \beta_3^2 \\ 1+\alpha_1 & 1+\alpha_2 & 1+\alpha_3 & \beta_1 & \beta_2 & \beta_3 \\ \alpha_1^2 & \alpha_2^2 & \alpha_3^2 & 1+\beta_1^2 & 1+\beta_2^2 & 1+\beta_3^2 \end{vmatrix} \neq 0
$$

It is simplified to: $(\alpha_2-\alpha_1)(\alpha_3-\alpha_1)(\alpha_3-\alpha_2) \neq 0$.

That is, $\alpha_1 \neq \alpha_2 \neq \alpha_3$.

Other possible 4 data blocks failure modes are calculated in the same way, and the conditions that the coefficients need to satisfy are:$\alpha_i \neq \alpha_j, \beta_i \neq \beta_j\ (i, j = 1,2,3; i \neq j)$

(2)2 data blocks belong to one group, and the other 2 data blocks belong to another group. The calculation method is the same as above, so the specific calculation pro-cess is omitted. The conditions for obtaining the coefficient need to be:$\alpha_i \neq \alpha_j, \beta_m \neq \beta_n, \alpha_i + \alpha_j \neq \beta_m + \beta_n\ (i, j, m, n = 1,2,3; i \neq j, m \neq n)$

In addition to the case of four data blocks failure, the 4-failure modes includes nine cases that are 3 data blocks and 1 local parity block failure, 3 data blocks and 1 glob-al parity block failure and so on. Using the above methods to discuss and calculate, the conditions for the overall coding matrix coefficients to be obtained are as follows:

$$
\begin{gathered}
\alpha_i \neq \alpha_j \neq \beta_i \neq \beta_j \neq 0, \alpha_i + \alpha_j + \alpha_i * \alpha_j \neq 0, \beta_i + \beta_j \neq \alpha_k \\
\beta_i * \beta_j \neq 1, \alpha_i + \alpha_j \neq \beta_k, \beta_k^2 \neq -1, \alpha_i + \alpha_j \neq \beta_m + \beta_n \\
\frac{\alpha_k^2}{1+\alpha_k} \neq \beta_m \neq \beta_i + \beta_j, \frac{1+\beta_k^2}{\beta_k} \neq \alpha_i + \alpha_j \neq \frac{\alpha_m^2}{1+\alpha_m} \\
(i, j, m, n, k = 1,2,3; i \neq j, m \neq n)
\end{gathered}
$$

4 Experiments

This section will analyze the performance of DLRC by two experiments: (1)the relationship between DLRC's parameters and its performance; (2)comparison of DLRC with other common erasure codes.

4.1 Experiment of DLRC's Parameters

To implement the analysis of DLRC's performance, the parameters k and m in the code are fixed. In the case of setting 10 data blocks and 2 global parity blocks, we analyze the storage overhead, 4-failure tolerance rate, fault tolerance ability, and average reconstruction overhead of single block with different combinations of n and l. The experimental comparison data is shown in Table 1.

It can be seen in the table that when n is fixed, as the l increases, the storage overhead, 4-failure tolerance rate and fault tolerance ability increase while average reconstruction overhead of single block does not change. When l is fixed, the value of n is proportional to the average reconstruction overhead of single block, and other properties are unchanged. In order to reduce

Table 1. The Data of DLRC's Encoding Performance

n	l	Storage Overhead (block)	Fault tolerance Ability (block)	4-failure tolerance rate (%)	average reconstruction overhead of single block(block)
2	6	18	8	100	2
3	4	16	6	99.8	3
4	3	15	5	98.7	4
4	6	18	8	100	4
4	9	21	11	100	4
6	2	14	4	93	6
6	4	16	6	100	6
6	6	18	8	100	6

Table 2. The Data of DLRC's Encoding Performance

Coding scheme	Storage Overhead (block)	Fault tolerance Ability(block)	4-failure tolerance rate(%)	average reconstruction overhead of single block(block)
RS(10, 4)	14	4	100	10
LRC(10, 2, 2)	14	4	86	6
ESRC(10, 2, 2)	14	4	93	6
SHEC(10, 5, 4)	15	5	83	4
DLRC(10, 2, 4, 3)	15	5	98.7	4
DLRC(10, 2, 6, 4)	16	6	100	6

the amount of data need to download, it is necessary to make the number of coded blocks included in each group as small as possible, which leads to an increase in the number of groups and an increase in storage overhead. Therefore, when the parameters k and m are fixed, the dynamic balance of storage overhead, reconstruction overhead, fault tolerance ability and fault tolerance rate can be achieved by adjusting the values of n and l. Thus, the values of n and l can be selected according to the actual performance requirements of the storage system.

4.2 Comparative Experiment of DLRC and Common Erasure Codes

In this section, we will compare DLRC with the classic RS code and three kinds of block codes to analyze the performance. Table 2 lists the storage overhead, 4-fault tolerance rate, fault tolerance ability and average reconstruction overhead of single block of RS(10, 4), LRC(10, 2, 2), ESRC(10, 2, 2), SHEC(10, 5, 4), DLRC(10, 2, 4, 3) and DLRC(10, 2, 6, 4).

The coding schemes in the table all have 10 original data blocks, which ensures the fairness of the comparison experiment. Compare the first row and the sixth row in the table. DLRC has a 50% improvement in fault tolerance ability and a 40% reduction in reconstruction overhead compared with RS at the expense of a small extra amount of storage space. Compared with LRC, DLRC(10, 2, 4, 3) has a 12.7% improvement in 4-fault tolerance rate, a 25% improvement in fault tolerance ability and a 33.3% reduction in average reconstruction overhead of single block. Compared with LRC, DLRC(10, 2, 6, 4) has a 14% improvement in 4-fault tolerance rate, a 25% improvement in fault tolerance ability and no change in average reconstruction overhead of single block. Compared with ESRC(10, 2, 2) and DLRC(10, 2, 4, 3) in the table, DLRC has a 5.7% improvement in 4-fault tolerance rate, a 25% improvement in fault tolerance ability and a 33.3% reduction in average reconstruction overhead of single block. Compared with SHRC(10, 5 ,4), DLRC(10, 2, 4, 3) has a 15.7% improvement in 4-fault tolerance rate and no change in other performance. Overall, DLRC achieves high fault tolerance rate and fault tolerance ability and low reconstruction overhead at the expense of a small amount of extra storage overhead, so the performance of DLRC is superior.

5 Conclusions

Compared with most other erasure codes, DLRC reduces the number of data blocks need to download when a single node fails. More than 90% of disk failures in real systems are single disk failure [12], so DLRC is effective to improve the overall system performance. In addition, DLRC retains the global parity blocks, which ensures the fault tolerance ability of the code. Moreover, the global parity blocks participate in the intra-group calculation, which greatly reduces the overall reconstruction overhead. The flexibility of DLRC is that the parameters n and l can be dynamically selected according to the balance of storage overhead, fault-tolerance ability, fault-tolerance rate, and reconstruction overhead. Therefore, the performance can meet the actual requirement of the distributed storage systems. Moreover, when the requirements of the distributed storage system change, there is no need to reselect other coding schemes, but only need to adjust the parameters of DLRC to achieve a new performance ratio.

6 References

[1] Y.Wen. IDC:The total global data is expected to reach 44 ZB in 2020, and China ac-counts for 18% of the total

data. http://www.cctime.com/html/2018-4-25/1377609.htm. Apr, 2018.

[2] Y.Wang and S.Li. Research and Performance Evaluation of Data Replication Technology in Distributed Storage Systems. Computers & Mathematics with Applications, 51(11), 1625-1632, May, 2006.

[3] J.S.Plank. Erasure Codes for Storage Applications. Proceedings of the 4th USENIX Conference on File and Storage Technologies. San Francisco, USA, 2005.

[4] S.L.Yang and G.Y.Zhang. Review of Data Recovery in Storage Systems Based on Erasure Codes. Journal of Frontiers of Computer Science and Technology, 11(10), 1531-1544. October, 2017.

[5] W.K.Lin and D.M.Chiu and Y.B.Lee. Erasure Code Replication Revisited. Proceedings of the 4th International Conference on Peer-To-Peer Computing, 90-97. Washington, USA, August, 2004.

[6] H.Weatherspoon and J.Kubiatowicz. Erasure Coding vs. Replication: A Quantitative Comparison. Proceedings of the 1st International Workshop on Peer-To-Peer Systems (IPTPS), 328-338. London, UK, March, 2002.

[7] B.Calder and J.Wang and A.Ogus, et al. Windows Azure Storage: A Highly Available Cloud Storage Service with Strong Consistency. Proceedings of the 23rd ACM Symposium on Operating Systems Principles, 143-157. Portland, Oregon, October, 2011.

[8] C.Huang and H.Simitci and Y.Xu, et al. Erasure Coding in Windows Azure Storage. Proceedings of the 2012 USENIX Conference on Annual Technical Conference, 2-13. Cascais, Portugal, June, 2012.

[9] C.Huang and M.Chen and J.Li. Pyramid Codes: Flexible Schemes to Trade Space for Access Efficiency in Reliable Data Storage Systems. Proceedings of the 6th IEEE International Symposium on Network Computing and Applications, 79-86, Cambridge, USA, 2007.

[10] T.Miyamae and T.Nakao and K.Shiozawa. Erasure Code with Shingled Local Parity Groups for Efficient Recovery from Multiple Disk Failures. Proceedings of the 10th USENIX Conference on Hot Topics in System Dependability, 5-5. Broomfield, USA, October, 2014.

[11] Y.X.Fu and S.L.Wen and L.Ma, et al. Summary of Single Disk Error Reconstruction Optimization Methods for Erasure Code Storage Systems. Journal of Computer Research and Development, 55(1), 1-13, 2018.

[12] X.H.Luo and J.W.Shu. Summary of Research for Erasure Code in Storage System. Journal of Computer Research and Development, 49(1), 1-11, 2012.