Performance Comparison and Analysis of RAID-6 Encoding and Scaling in Edge Computing Environments

Zehao Liu *

Pennsylvania State University, State College PA 16803, USA

* Corresponding Author Email: zql5426@psu.edu

Abstract. In today's world where cloud computing is widely used, edge computing is starting to be used as a new type of data computing and management service. This paper first describes the importance of data reliability and storage fault tolerance for edge computing operating in multiple environments, and then focuses on RAID-6 to address the data storage requirements of edge computing. In this paper, the feasibility of RAID-6 for edge computing is discussed and two different RAID-6 coding algorithms are introduced. The better option for edge computing is explored through performance comparison. Among the two RAID-6 coding algorithms, it is considered that Row-Diagonal parity will be more suitable than EVENODD code for edge computing scenarios through comparison. But perhaps there are better coding algorithms with lower complexity that can again improve the performance of edge computing storage systems. An edge computing storage system using RAID-6 architecture would be very beneficial for a wide range of real-world application scenarios.

Keywords: Edge Computing, RAID-6, Storage System.

1. Introduction

Edge computing, as a new data computing and management service, has been widely popularized in recent years. It is different from cloud computing in that edge computing can support data storage and processing at the local user end. However, Edge computing can work together with Cloud computing [1]. After being processed locally through edge computing, the data is sent to cloud servers. Therefore, using edge computing as an adjunct to cloud computing can solve the latency as well as network bandwidth problems of fully using cloud computing and provides great help in data storage and processing in many fields. For example, in today's rapidly evolving automotive self-driving technology, edge computing can quickly respond to changing road and vehicle conditions by processing data locally and uploading it to cloud servers later. If data is processed in the cloud, the time and risk involved in data transfer could lead to serious driving safety issues when the vehicle encounters unexpected conditions.

However, edge computing still has some challenges. For example, the efficiency of data processing, in most cases, using edge computing locally requires extremely high data processing speeds to react to unexpected situations. In some IoT (Internet of Things) application scenarios, data is generated at a very fast rate [2]. Therefore, the algorithms as well as the structure of data processing need to have high performance and efficiency. Another example, data security is also an issue. Edge computing devices are often located far from cloud servers and in large numbers. Without regular maintenance and management, the devices are vulnerable to outside cyberattacks, which can lead to data corruption and tampering. On top of that, edge computing devices tend to be easier to damage, which can still lead to data loss or corruption. Therefore, data transfer and backup between edge computing devices and cloud storage servers are very important for data security.

For the above challenges, this article will focus on the research on the storage aspect of edge computing. The RAID (Redundant Arrays of Inexpensive Disks) system will be used as a solution in this research, and the concept of RAID is introduced in [3]. The use of RAID-6 architecture for edge computing storage will be explored as the topic of this article.

In the next section, this article will first introduce the RAID-6 structure and concepts. Section 3 will analyze the storage requirements in edge computing environments. Coding algorithms and

extensions of RAID-6 will be discussed in section 4. The performance comparison and analysis of different coding algorithms will be done in section 5. Section 6 includes some practical applications of RAID-6 coding in edge computing and discusses potential future directions. Finally, the paper will be summarized in section 7.

2. RAID-6 Architecture and Concepts

The RAID (Redundant Array of Independent Disks) architecture is a widely used architecture for storage systems. RAID-6 is one of the RAID architectures with high performance and good error correction [4]. It differs from other RAID structures in that it has two parity disks. Figure 1 below is a simple example structure of RAID-6.

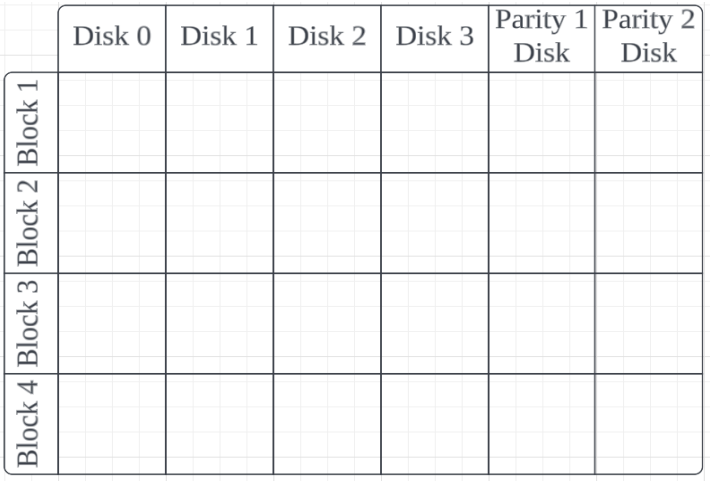


Fig 1. RAID-6 Architecture (Photo/Picture credit: Original).

In RAID-6, there are various algorithms for calculating the parity disk. Often it is necessary to calculate the exclusive or sum of the data disks to derive the value of the blocks in the parity disk. To check whether the data disk is damaged or not, comparing the data in the parity disk. Through the algorithms, the damaged disk can be reconstructed by using different combinations of other disks.

With these two parity disks, it can be recovered from any two simultaneous disk failures. This makes it have better data security in real-world applications. Since RAID-6 can correct any two simultaneous disk failures, it has a better reconstruction performance than other architectures [5].

However, having two parity disks also leads to some bottlenecks. More disks result in RAID-6 being more expensive compared to other structured storage systems. The amount of computation required to calculate the data for both parity disks is also much higher, resulting in slower write speeds [6].

3. Storage Requirements in Edge Computing Environments

Edge computing is used today in a variety of environments. Edge computing has similar data storage requirements in multiple environments. For some IoT applications, the amount of locally generated data is huge [7]. One of the advantages of edge computing over cloud computing is that it allows a large amount of data to be processed at the edge and sent to the cloud, rather than passing large amounts of data directly to the cloud for computation. For example, according to Stephan's presentation [8], the amount of data generated by various sensors in a car will reach terabytes per hour. This requires edge computing to be able to process large amounts of data in a short period, so a low-complexity and efficient algorithm becomes important.

In addition, the environment in which the edge segments are located is often uncertain. As in the example of self-driving cars mentioned above, the reliability of the data is very important for the operation of the car. But the car may operate in any possible scenario, whether it's a safe one or one with a harsh environment. This means that the edge can receive damage at any time, and damage also

means corrupted or lost data. Therefore, in this case, the storage system for edge computing needs to have data recovery capability and fault tolerance.

In summary, edge computing storage systems operating in various environments have very high requirements for data reliability and fault tolerance.

4. RAID-6 Encoding Algorithm and Extension

Nowadays there are many different encoding algorithms for RAID-6, such as Reed-Solomon code, EVENODD code, and Row-Diagonal Parity (RDP) algorithm. This article will start by describing the most widely used EVENODD code.

The EVENODD code, as mentioned above, is one of the RAID-6 codes that can correct any two disk failures. The EVENODD code has m data disks where m is a prime number and 2 parity disks. Each disk has m-1 data blocks. Figure 2 below is an example of the EVENODD code disk structure.

	Disk 0	Disk 1	Disk 2	Disk 3	Disk 4	Parity 1	Parity 2
Block 1	1	0	1	1	0		
Block 2	0	1	1	0	0		
Block 3	1	1	0	0	0		
Block 4	0	1	0	1	1		

Fig 2. Example of EVENODD Code (Photo/Picture credit: Original).

For encoding, it is easy to get the values of parity 1 disk. Adding up the values in the data blocks for each row using the XOR operation to get the data for parity 1, as shown in Figure 3. For example, for the first row of our input matrix, the parity 1 block of the first row will be$1 \oplus 0 \oplus 1 \oplus 1 \oplus 0 = 1$.

	Disk 0	Disk 1	Disk 2	Disk 3	Disk 4	Parity 1	Parity 2
Block 1	1 XOR	0 XOR	1 XOR	1 XOR	0	1	
Block 2	0 XOR	1 XOR	1 XOR	0 XOR	0	0	
Block 3	1 XOR	1 XOR	0 XOR	0 XOR	0	0	
Block 4	0 XOR	1 XOR	0 XOR	1 XOR	1	1	

Fig 3. Calculate for Parity 1 (Photo/Picture credit: Original).

As for the parity 2 disk, there's a complex formula for its calculation. Since the parity can be even or odd, a sum of the parity diagonal S needs to be calculated to check if the parity is even or odd [9]. For calculating the S sum, use the formula with XOR sum:

$$S = \sum_{t=1}^{m-1} a_{m-1-t,t}. \quad (1)$$

In the formula, a indicates the data block. In this example, which is graphically shown in Figure 4,$1 \oplus 0 \oplus 0 \oplus 0 = 1$

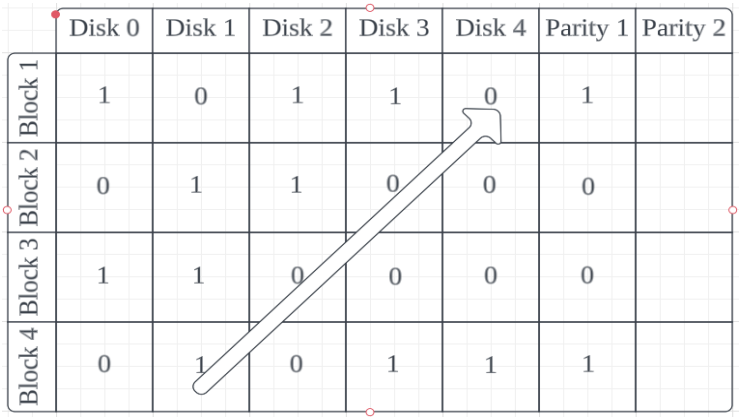


Fig 4. Example for Sum S (Photo/Picture credit: Original).

After getting S, the formula below is used to calculate the parity 2 disk data:

$$a_{l,m+1} = S + \sum_{t=0,t\neq l+1}^{m-1} a_{<l-t>m,t}. \quad (2)$$

Here, the l indicates the row number. The parity 2 disk data is the summation of different diagonal values and sum S. In Figure 5, by using the formula above, the data connected with the same color are the different diagonals. Doing XOR sums for the data in the same color and Sum S will give the value of a Parity 2 block. For example, block 3 of parity 2 disk is the sum of blue blocks and Sum S which is:$1 \oplus 1 \oplus 1 \oplus 1 \oplus 1 = 1$ After the calculation, the encoded codeword comes out, which the same as the data is shown in Figure 5.

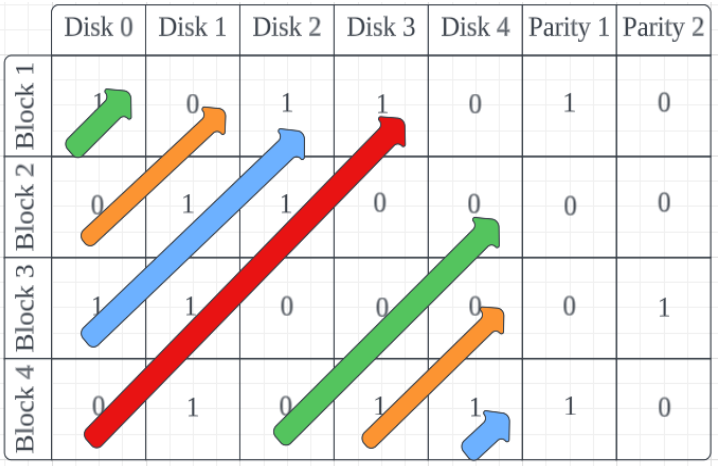


Fig 5. Calculate for Parity 2 (Photo/Picture credit: Original).

For decoding, the EVENODD code algorithm can easily correct any single disk failure. With any possible two simultaneous disk failures, there are different helper disk choices available for recovering failure disks. Four main failure situations might happen: 1. two information disk failures. 2. Two parity disk failures. 3. One information disk failure and parity 1 disk failure. 4. One information disk failure and one parity disk failure. All those four situations were covered in the class. For our implementation, an example of the 4th situation was implemented.

In addition to EVENODD code, RDP code is also a RAID-6 structure algorithm. Compared to EVENODD code, RDP has an easier-to-calculate parity disk. It is also a good choice for edge computing which requires low complexity computation.

For RDP's encoding, it is similar to the EVENODD code algorithm, it also has a parameter p. This parameter p must be a prime number greater than 2. There are p-1 data disks and 2 parity disks, and each data disk has p-1 blocks. An example is shown in Figure 6 below. The Row parity has the same calculation method as the EVENODD code. For the Diagonal parity disk, just make an XOR sum of the data in the same color to get the Diagonal parity value. Here, the RDP algorithm does not need to compute a Sum S as the EVENODD code needs to do.

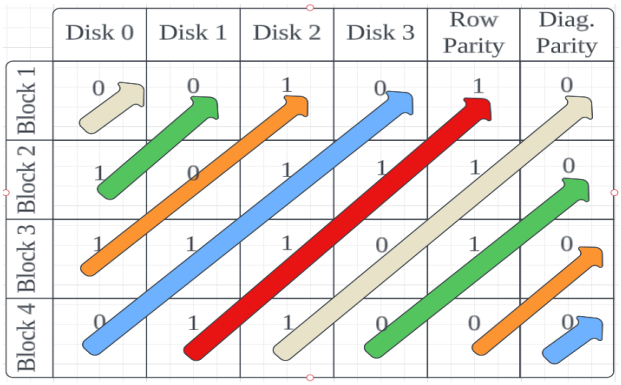


Fig 6. Example of RDP algorithm (Photo/Picture credit: Original).

5. Performance Comparison and Analysis

For performance comparisons of different RAID-6 encoding algorithms, this is usually done by comparing the number of XOR sums they require. The article [5] compares the number of XOR operations required for RDP and EVENODD code on several different-sized disks.

Table 1. Per Row XOR Counts for Parity Construction [5].

Data disks	Number of XORs in RDP	Number of XORs in EVENODD	Difference
4	6	6.67	11.1%
6	10	10.8	8.0%
8	14	14.86	6.1%
12	22	22.91	4.1%
16	30	30.93	3.1%

The above Table 1 compares the performance of RDP as well as EVENODD code in terms of Encoding. Several samples of different-sized disks were used to study the influence of disk size on its performance. For the data in Table 1, it can be observed that the number of XOR calculations used by RDP is smaller than the number of XOR calculations required by EVENODD code in general. When there are only 4 data disks, they have a very large difference in the number of XORs. But as the number of data disks increases, the gap between them becomes smaller.

Table 2. Per Row XOR Counts for Data Reconstruction [5]

Data disks	Number of XORs in RDP	Number of XORs in EVENODD	Difference
4	6	9.67	61.2%
6	10	13.80	83.0%
8	14	17.86	27.6%
12	22	25.91	17.8%
16	30	33.93	13.1%

Table 2 is used to compare the number of XOR computations used in the reconstruction for the two RAID-6 coding algorithms. In the same way that the encoding performance of the algorithms is compared, a variety of different disk sizes are used. It can be seen in this Table 2 that the difference in the number of XOR calculations required for their reconstruction becomes much larger compared to encoding. RDP requires less XOR computation operation for reconstruction than EVENODD code.

In summary, RDP is a better option for edge computing since it has a faster encoding. Most importantly it has better performance and low complexity as compared to EVENODD code.

6. RAID-6 Coding Applications and Perspectives

Edge computing with RAID-6 architecture can have different applications in reality. For example, in the self-driving car example mentioned in the article, edge computing with RAID-6 architecture

can better ensure the data security of the car and quickly react to unexpected situations with local edge computing and disk coding architecture.

Drones can also use edge computing with RAID-6 structured storage systems. Drones have the potential to work in a variety of different environments. Inevitably, there will be disk failures. This is where the fault tolerance of the RAID-6 structure becomes especially important. And like the autopilot example, a RAID-6 structured storage system can better secure the data.

As Edge computing is becoming more widely used in the future, edge computing with RAID-6 storage architecture will bring higher fault tolerance and reliability for data storage systems. However, RAID-6 might not be the architecture with the best efficiency of data processing. A better RAID-6 encoding method with less complexity may again improve the efficiency of data storage for edge computing.

7. Conclusion

This paper focuses on the data processing requirements of edge computing and tries to address the requirements with the RAID-6 coding structure, which has high fault tolerance and data reliability. Edge computing in various environments has very high requirements for data storage and security. Therefore, RAID-6 architecture is very suitable to meet the data storage needs of edge computing. With the help of illustration, this paper explains two different RAID-6 coding algorithms including EVENODD code and RDP algorithm and they are compared in terms of performance. Based on the performance evaluation it is concluded that RDP with low complexity and better performance is more suitable for edge computing storage systems. Edge computing with RAID-6 architecture is beneficial for many real-world application scenarios. However, better RAID-6 coding algorithms may also have the opportunity to improve the storage efficiency and performance of edge computing storage systems.

References

[1] Cao, K., Liu, Y., Meng, G., Sun, Q.: An Overview on Edge Computing Research. IEEE Access. 8, pp. 85714–85728 (2020).

[2] Jiang, L., Xu, D., Cai, H., Jiang, Z., Bu, F., Xu, B.: An IoT-Oriented Data Storage Framework in Cloud Computing Platform. IEEE Transactions on Industrial Informatics. vol. 10, no. 2, pp. 1443-1451 (2014).

[3] Patterson, D.A., Gibson, G., Katz, R.H.: A case for redundant arrays of inexpensive disks (RAID). ACM SIGMOD Record. 17, 109–116 (1988).

[4] Klein, H., Keller, J.: "RAID Architecture with Correction of Corrupted Data in Faulty Disk Blocks," 22th International Conference on Architecture of Computing Systems 2009, Delft, Netherlands, pp. 1-4. (2009).

[5] Jin, C., Feng, D., Jiang, H., Tian, L.: A Comprehensive Study on RAID-6 Codes: Horizontal vs. Vertical. 2011 IEEE Sixth International Conference on Networking, Architecture, and Storage, Dalian, China, pp. 102-111 (2011).

[6] Hefner, K.: RAID 6, https://www.techtarget.com/searchstorage/definition/RAID-6-redundant-array-of-independent-disks#:~:text=What%20are%20the%20disadvantages%20of,extra%20disks%20required%20for%20parity, last accessed 2023/09/30.

[7] Cai, H., Xu, B., Jiang, L., Vasilakos, A.V.: IoT-Based Big Data Storage Systems in Cloud Computing: Perspectives and Challenges. IEEE Internet of Things Journal. 4, pp. 75–87 (2017).

[8] Heinrich, S.: Lucid Motors Flash Memory Summit. (2017).

[9] Blaum, M., Brady, J., Bruck, J., Menon, J: EVENODD: an efficient scheme for tolerating double disk failures in RAID architectures. IEEE Transactions on Computers. 44, pp. 192–202 (1995).

[10] Corbett, P., English, B., Goel, A., Grcanac, T., Kleiman, S., Leong, J., Sankar, S.: USENIX Association Proceedings of the Third USENIX Conference on File and Storage Technologies Row-Diagonal Parity for Double Disk Failure Correction. (2004).