Comparison with EVENODD Code and Reed Solomon Code based on Implementation of RAID 6

Wanzhen Wei *

Southern University of Science and Technology, Shenzhen Guangdong 518067, China

* Corresponding author: 12012237@mail.sustech.edu.cn

Abstract. In the era of big data, the precision of information is paramount and any loss or corruption is intolerable for the entire system. To ensure robust data resilience and redundancy, some advancements encompass techniques like data replication, erasure coding, and fault-tolerant architectures. Reed-Solomon (RS) code and EVENODD code are utilized as error-correcting code in data storage and various applications. Additionally, the RAID 6 technology is a storage technology that implements block-level striping and dual parity in order to withstand the simultaneous failure of up to two drives. In a comparative analysis of EVENODD and RS code, it is observed that EVENODD code entails fewer XOR operations, indicating a higher level of efficiency in encoding operations. However, RS code outperforms EVENODD in terms of I/O operation required for decoding. Moreover, it summarizes the application performance of RS code and EVENODD code in the framework of RAID 6, providing valuable insights for the next generation of storage systems.

Keywords: Reed-Solomon code, EVENODD code, RAID 6.

1. Introduction

The increasing demand for reliable data storage, driven by the evolution of cloud storage technology and 5G technology, has long relied on the implementation of RAID (Redundant Arrays of Inexpensive Disks). RAID systems are integral to data storage, with each RAID level denoting a specific data distribution method across multiple hard drives. Essentially, there are two categories of RAID: standard RAID and hybrid RAID. Standard RAID level ranges from 0 to 6. However, RAID2,3,4 with limited practical utility is primarily confined to research applications, as the functionalities required for most applications are already covered by RAID0,1,5,6 and hybrid RAID. Among the variable RAID types, RAID 6 stands as a noteworthy example of an error-correcting code strategy, capable of tolerating up to two errors and having high robustness [1].

In digital communications and storage systems, Reed-Solomon (RS) code is the most widely employed channel code. It is constructed and decoded on finite field or Galois field, where each code word corresponds to a system of q linear equations in k variables. RS code's widespread application stems from its unique suitability for error correction across a diverse array of applications [2]. Furthermore, RS code excels in burst error correction, making it suitable for applications in fading channels and recording systems [3].

In addition to RS code, EVENODD represents another crucial code for correcting double disk failures [4]. EVENODD employs just two redundant disks and simple exclusive-OR computations (XOR). An important benefit of EVENODD is its compatibility with parity hardware, facilitating seamless implementation on standard RAID5 controllers without necessitating hardware modifications [5].

Prior research has infrequently delved into the contrast between the mathematical principles and application performance of RS code and EVENODD code. Consequently, this article conducts a detailed comparison of them within the framework of RAID 6 implementation, encompassing factors such as encoding complexity, I/O requirements for data reading and writing operations, and their distinct performance advantages and disadvantages in real-world applications.

2. RAID 6 and Reed Solomon (RS) code

2.1. Overview of RAID 6 technology

The RAID 6 method exemplifies the use of error-correcting codes, enabling it to withstand two errors and maintain a high level of reliability. Commonly employed RAID 6 algorithms like EVENODD and RDP are favored for their capacity to exclusively utilize XOR operations, making them optimal for redundancy. This feature is widely adopted in storage systems [1].

Table 1. Performance Comparison between Different RAID

RAID level number	Minimum disk number	Maximum fault-tolerance	Available capacity	Space utilization rate	Read Performance	Write Performance
Single disk	1	0	1	100%	1	1
0	2	0	n	100%	n	n
1	2	n-1	1	50%	n	1
5	3	1	n-1	$\frac{n-1}{n}$ (68%-94%)	n-1	n-1
6	4	2	n-2	$\frac{n-2}{n}$ (50%-88%)	n-2	n-2
10	4	1 disk damaged in any RAID 1	\	50%	\	\
50	6	1 disk damaged in any RAID 5	\	$\frac{n-1}{n}$ (67%-94%)	\	\
60	8	2 disks damaged in any RAID 6	\	$\frac{n-2}{n}$ (50%-88%)	\	\

RAID 0 parallels two or more disks where data are stored after being segmented. However, RAID 0 has neither redundancy nor fault tolerance, which results in full space utilization. RAID1 mirrors two or more sets of n disks to each other. RAID 1 has the best data security but the lowest disk utilization 1/n among all RAID levels from Table 1 because RAID 1 mirrors two or more sets of N disks to each other. RAID 1 does not have a verification mechanism which leads to the situation that RAID 1 cannot distinguish the correct one when there is a difference between two hard drives. This is called brain splitting.

RAID 5 separates parity information from the actual data and stores them on separate disks. This configuration enables the recovery of lost data when a disk becomes corrupted by utilizing the remaining data and the parity information linked to the damaged disk. In contrast, RAID 6 enhances fault tolerance by introducing a second, independent parity information block. This additional parity block ensures that even if two disks fail simultaneously, data integrity remains intact. Moreover, RAID 6 utilizes (n-2)/n of the available space for this purpose. RAID 6 needs to allocate more disk space and additional verification calculations for parity information, resulting in greater IO operations and computation compared to RAID 5 [1].

Hybrid RAID is mixed with two types of data distributing methods, which makes it inherit characteristics such as the fault-tolerance and space utilization. For example, the space utilization rate is the same for RAID10 and RAID 1 from Table 1.

RAID 60 is a combination of RAID 6 and RAID 0, which allows for Stripe access to two or more sets of RAIDS 6. Compared to pure RAID 6, RAID 60 has higher performance. The usage threshold of RAID 60 is high while the capacity utilization rate is low.

2.2. Introduction to Reed Solomon (RS) Code

Reed-Solomon code is established and decoded within a finite field, also known as a Galois field. There is a set of k information symbols denoted as $\{m_0, m_1, ..., m_{k-2}, m_{k-1}\}$, all derived from the finite field GF(q) Each Reed-Solomon code word can be expressed through equation (1) as a system of q linear equations involving k variables, as illustrated below [2].

$$c = [\, P(0), P(\alpha), P(\alpha^2), \ldots, P(\, \alpha^{q-1})] \tag{1}$$

Each Reed-Solomon code word can be expressed through equation (1) as a system of q linear equations involving k variables, as illustrated below.

$$\begin{aligned} \mathrm{P}\,(0) &= m_0 + m_1 + m_2 + \ldots + m_{k-1} \\ \mathrm{P}(\alpha) &= m_0\alpha_0 + m_1\alpha_1 + m_2\alpha_2 + \ldots + m_{k-1}\alpha_{k-1} \\ \mathrm{P}(\alpha^2) &= m_0\alpha_0^2 + m_1\alpha_1^2 + m_2\alpha_2^2 + \ldots + m_{k-1}\alpha_{k-1}^2 \\ &\ldots \\ \mathrm{P}(\alpha^{q-1}) &= m_0\alpha_0^{k-1} + m_1\alpha_1^{k-1} + m_2\alpha_2^{k-1} + \ldots + m_{k-1}\alpha_{k-1}^{k-1} \end{aligned} \tag{2}$$

A system can be generated from code words and information symbols:

$$\begin{bmatrix} 1 & 1 & 1 & \ldots & 1 \\ \alpha_0 & \alpha_1 & \alpha_2 & \ldots & \alpha_{k-1} \\ \alpha_0^2 & \alpha_1^2 & \alpha_2^2 & \ldots & \alpha_{k-1}^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \alpha_0^{k-1} & \alpha_1^{k-1} & \alpha_2^{k-1} & \ldots & \alpha_{k-1}^{k-1} \end{bmatrix} \begin{bmatrix} m_0 \\ m_1 \\ m_2 \\ \vdots \\ m_{k-1} \end{bmatrix} = \begin{bmatrix} P(0) \\ P(\alpha) \\ P(\alpha^2) \\ \vdots \\ P(\alpha^{q-1}) \end{bmatrix} \tag{3}$$

The generation matrix of RS code is G.

$$G = \begin{bmatrix} 1 & 1 & 1 & \ldots & 1 \\ \alpha_0 & \alpha_1 & \alpha_2 & \ldots & \alpha_{k-1} \\ \alpha_0^2 & \alpha_1^2 & \alpha_2^2 & \ldots & \alpha_{k-1}^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \alpha_0^{k-1} & \alpha_1^{k-1} & \alpha_2^{k-1} & \ldots & \alpha_{k-1}^{k-1} \end{bmatrix} \tag{4}$$

Decoding of RS code is solving the incorrect value and incorrect position from the equation. Then, based on the Syndrome value calculated from the received data, the coefficient Λ will be solved, and then X will be solved:

$$(1+X_1\mathrm{x})\,(1+X_2\mathrm{x})\ldots(1+X_v\mathrm{x}) = 1+\Lambda_1\mathrm{x} + \Lambda_2 x^2 + \ldots + \Lambda_{\mathrm{v}} x^{\mathrm{v}} \tag{5}$$

Then, calculate Y based on the 'Syndrome' formula. That is the incorrect value S_i

$$S_i = Y_1X_1^i + Y_2X_2^i + \ldots + Y_vX_v^i \tag{6}$$

After knowing the location and value of the error, they can remove the error from the received data to recover the encoded data sent:

$$\mathrm{C}\,(\mathrm{x}) = \mathrm{R}\,(\mathrm{x}) - \mathrm{E}\,(\mathrm{x}) \tag{7}$$

After recovering the encoded data, the verification data is directly discarded, leaving only the original information data sent.

2.3. Reed Solomon (RS) code in RAID 6

In a RAID 6 coding scheme, Reed-Solomon coding can be implemented using a matrix of dimensions (m-1) ×(m+2). The initial m columns store the unprocessed data, while the final two columns store the parity information. The diagram below illustrates the application of Reed-Solomon encoding in a RAID 6 setup with 5 data disks (m is prime).

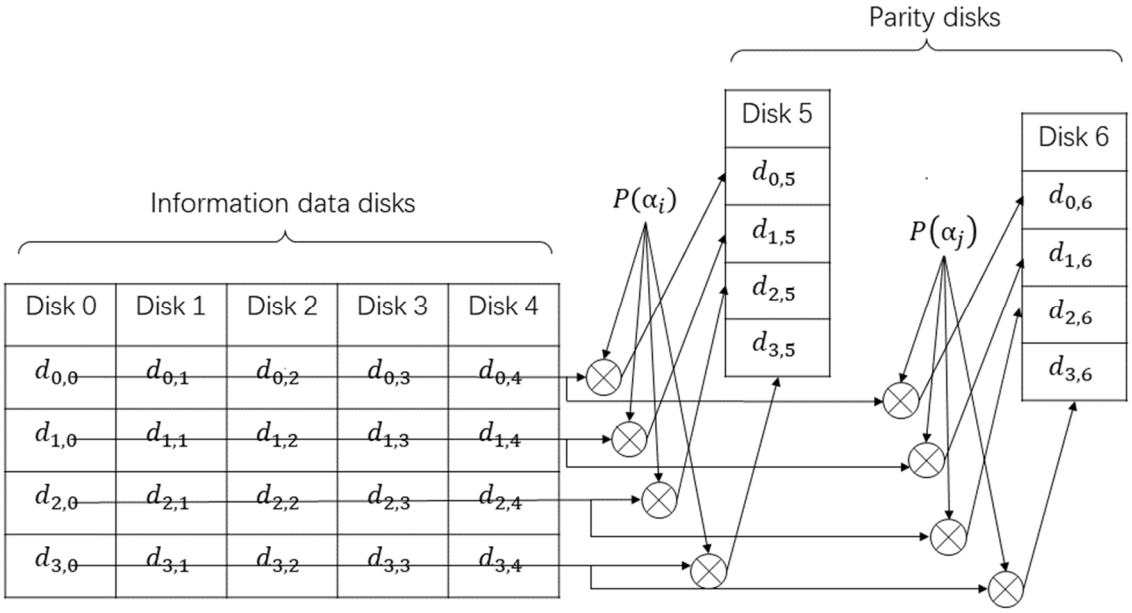

Figure 1. Encode procedure for Reed Solomon code implemented in RAID 6 with 5 data disks (Original)

Each row of information data disks in Figure 1 multiples a column, such as $P(\alpha_i)$. The result is the parity $d_{0,5}$, which is stored in the parity disk 5. $P(\alpha_i)$ and $P(\alpha_j)$ represent any column in the generation matrix G, shown in (4). Every column in G is linearly independence which helps parity decode the lost data.

The procedure of decoding in storage systems often refers to recover the information data disk from physical damage by remaining data disks, including information data disks and parity disks. Decoding in RS code in the framework of RAID 6 technology is divided into several situations which are listed in Table 2:

Table 2. Situation analysis required decoding for RS code in RAID 6

Disk damaged situations	Number of disks lost as parity disks	Number of disks lost as information data disks	Total number of disks lost	Whether can be solved
Lost no more than 2 parity disks	1 or 2	0	1 or 2	Yes
Lost no more than 2 information data disks	0	1 or 2	1 or 2	Yes
Lost 1 parity disk and 1 information data disk	1	1	2	Yes
Lost more than 2 disks for any circumstances			≥2	No

Situations list:

1. Lost no more than 2 parity disks;
2. Lost no more than 2 information data disks;
3. Lost 1 parity disk and 1 information data disk;
4. Lost more than 2 disks for any circumstances, including more than 2 lost in either parity disks and information disks or failure in both of them.

For the top 3 situations, the destroyed data disks can be retrieved because of the characteristic of RAID 6. For situation 1, parity disks can be recovered by encoding them again. For situation 2 and 3, two equations will be listed according to $P(\alpha_i)$ and $P(\alpha_j)$. Since two equations can definitely solve for two variables, it can obtain the missing disks. However, two equations cannot solve for three variables. Situation 4 can be decoded. More parity disks are required to solve problems of losing more than 2 disks.

3. EVENODD code in RAID 6

EVENODD coding in a RAID 6 scheme utilizes a (m-1) ×(m+2) array. Figure 1 illustrates the encoding structure of the EVENODD scheme (m = 5).

The EVENODD parity is set by S [6] by the diagonal part of the matrix shown in Figure 2:

$$\boldsymbol{S} = \oplus_{t=1}^{m-1} d_{m-1-t,t} \tag{8}$$

For each l (l is under m-2), the parity disks are attained by:

$$d_{l,m} = \oplus_{t=0}^{m-1} d_{l,t} \tag{9}$$

$$d_{l,\mathrm{m+1}} = \boldsymbol{S} \oplus (\oplus_{t=0}^{m-1} d_{<l-t>_m,t}) \tag{10}$$

It is considered that the notation $< y >_i = j$ is for j = y (mod i) only when 0 ≤ j ≤ i-1. Disk m is essentially the result of disks in the same row shown in (9). Disk (m + 1) indicates the redundancy of the diagonal in the center, which is demonstrated as two connected oblique lines in Figure 2, XOR with S as described in (10).

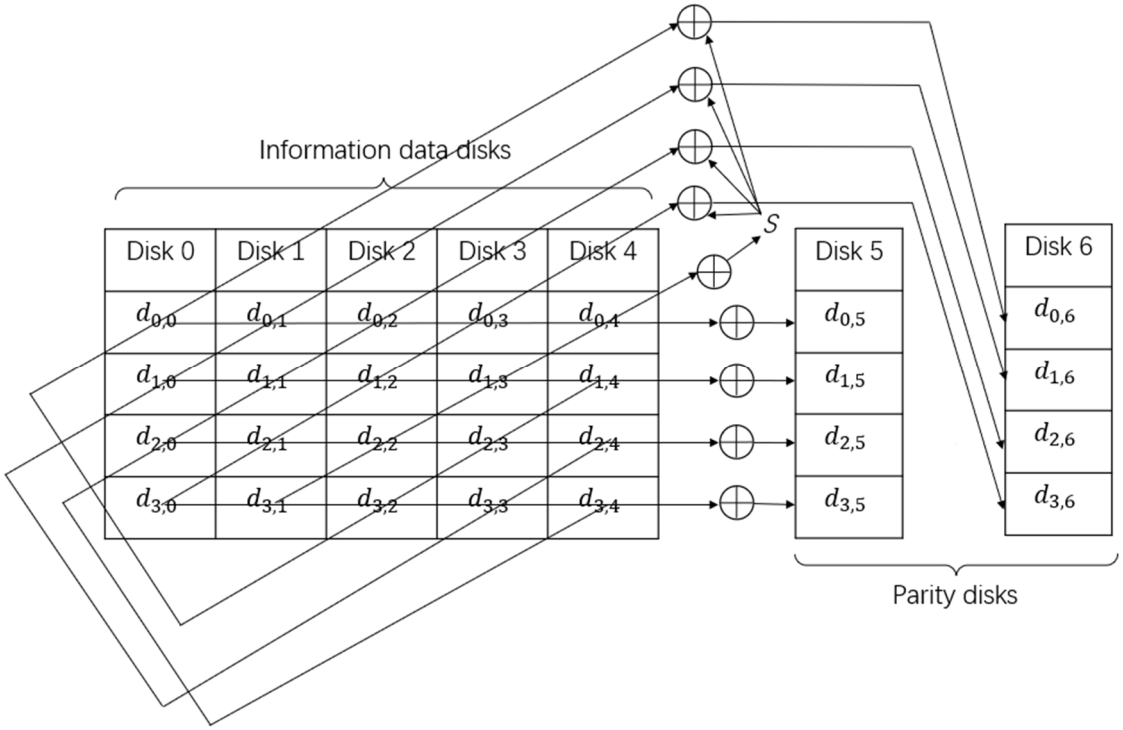

Figure 2. Encode procedure for EVENODD code implemented in RAID 6 with 5 data disks [7]

Decoding situations in EVENODD code in the framework of RAID 6 technology is the same as that of RS code which is listed in Table 2. However, the decoding procedure is different for the top 3 situations. For situation 1, parity disks can be recovered by encoding them which is also the same as RS code.

The first parity is included in an equation of XOR operations with the information disks in the same row shown in equation (9). The second parity is included in an equation of XOR operations with S (including data disks in the diagonal of the entire data disks matrix) and corresponding data disks in equation (10).

4. Comparison of hardware I/O required when decoding in EVENODD code and RS code implemented RAID 6 technology

Compared the hardware I/O required when decoding in EVENODD code and RS code implemented RAID 6 technology in Table 3, the disk array will be set to have 2 disks for parity check and m disks for original data storage, which is a typical RAID6 structure of a (m-1) ×(m+2) matrix. Number of disks read when rebuilding from disk damage is not including failed disks.

Table 3. Hardware I/O required when decoding in EVENODD code and RS code implemented RAID 6 technology

Types of operation	Number of disk RS code read	Number of disks EVENODD code read
Recover1 failed disk	m	m
Recover 2 failed disks	m	(m-1) × m

From Table 3, the RS code provides benefits for dual failure recovery, resulting in reduced I/O. Compared to RS code, the EVENODD code decoding process not only relates the data disks of the row m, but also can relate to data disks in the diagonal and from different rows. In consequence, EVENODD code have to read all the data (m-1) × m when decoding on some occasions, which is much more than RS code is required to read.

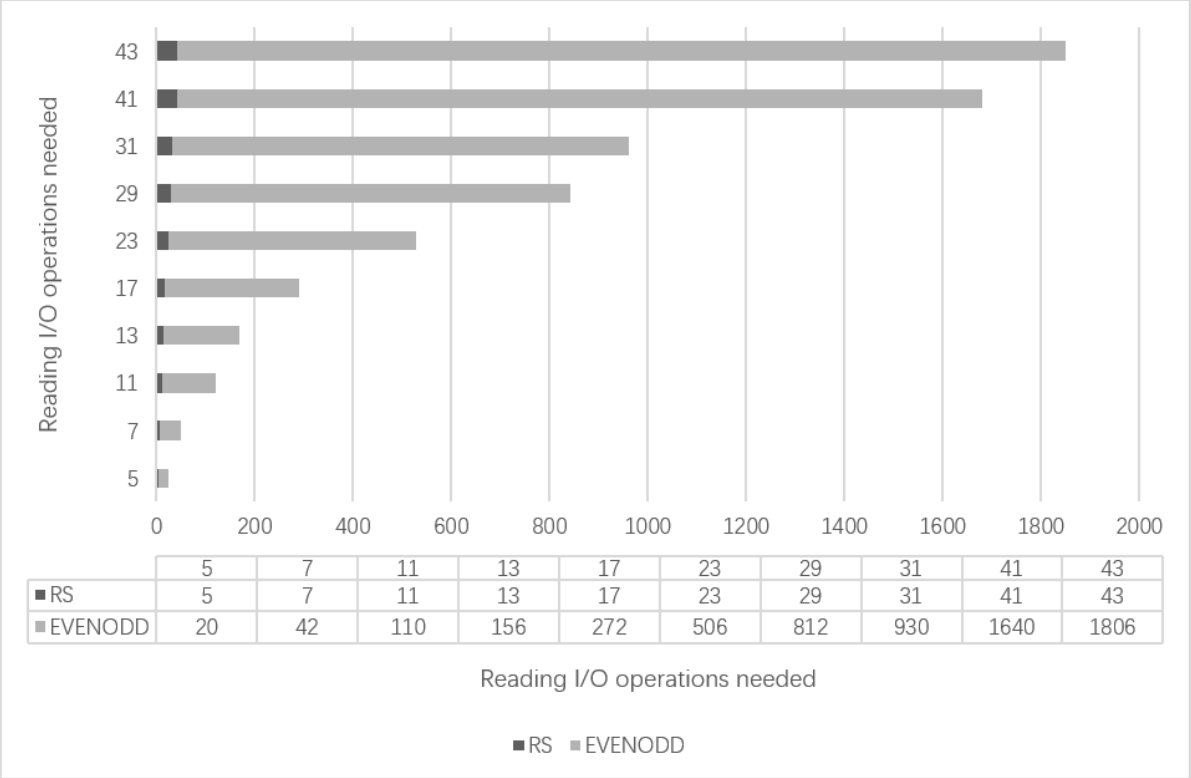

Figure 3. Reading I/O operations needed when rebuilding from 2 failed disks (Original)

It is demonstrated in Figure 3 that RS code needs less reading I/O operations than EVENODD. When m equals 7, EVENODD needs 5 times reading I/O operations more than that of RS code. When m reaches 43, it is 41 times larger.

5. Comparison of complexity of EVENODD code and RS code implemented in RAID 6

First, a detailed comparison between encoding complexity in RS and EVENODD is made when original data constructs an $(\alpha - 1) \times \alpha$ matrix. The analysis of encoding complexity for both RS and EVENODD scheme involves tallying the count of XOR [5].

It is concluded EVENODD code requires a number of XOR to encode:

$$8(2\alpha^2 - 2\alpha - 1) \tag{11}$$

And RS code needs a total of XOR operations:

$$1.5\alpha^3 + 13\alpha^2 - 30.5\alpha + 16 \tag{12}$$

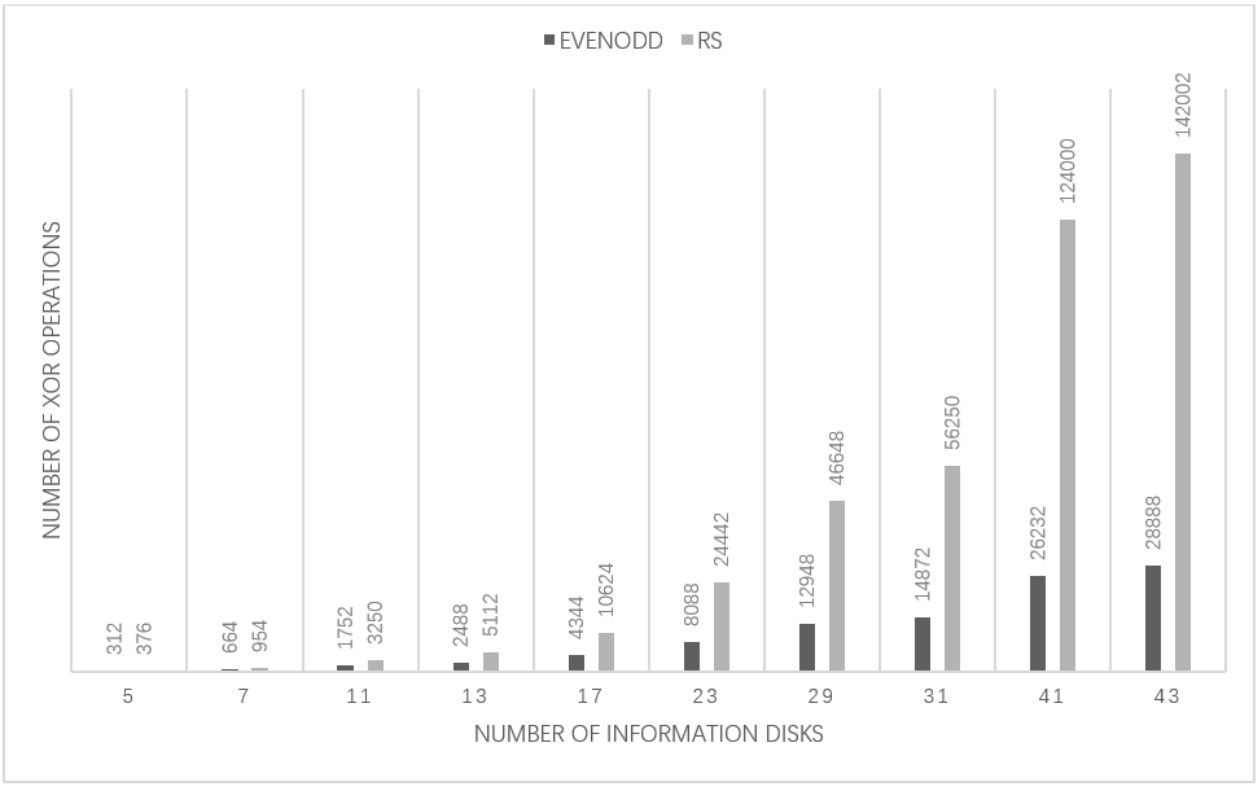

Figure 4. The count of XOR operations required to encode in RAID 6 [6]

The horizontal axe of Figure 4 is information disks number α and the vertical axe of it is the count of XOR. From Figure 4, it shows the count of XOR needed to encode for α information disks for RS code and EVENODD code is about the same at the α equals 5. The count of XOR for encoding RS code rises at α^3 level while that of EVENODD code grows up at α^2 level. When α reaches 13, the count of XOR for RS code is nearly twice as much as that of EVENODD code. In the calculation, the maximum value of α is 43, where it becomes evident that the count of XOR for encoding RS code becomes four times more intricate than that for EVENODD code.

As α continues to increase, the disparity in XOR operation complexity between RS code and EVENODD code becomes increasingly pronounced. This discrepancy primarily stems from the distinct approaches to computing the second redundancy disk. While the encoding complexity of EVENODD code may not be readily apparent when α is small, it becomes significantly more efficient than RS code in encoding when α surpasses 13. This efficiency translates into substantial savings in computational resources.

Additionally, a comparison is conducted between the encoding complexity of RS and EVENODD code versus basic parity scheme. It is assessed that the count of XOR necessary for applying the parity scheme with the same framework is

$$8(\alpha - 1)^2 \tag{13}$$

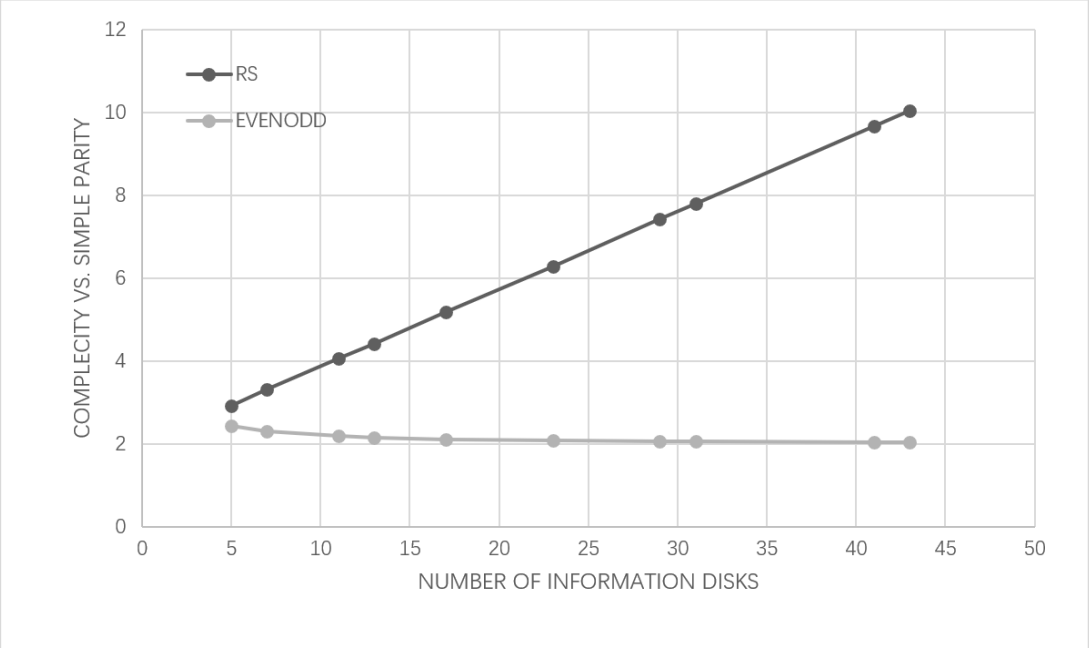

Figure 5. The ratio of XOR operations in the EVENODD or RS scheme compared to those in the simple parity scheme [6]

In Figure 5, the horizontal axis represents information disks α while the vertical axis displays the ratio of XOR operations in the EVENODD or RS scheme compared to those in the simple parity scheme. Figure 5 clearly illustrates that EVENODD is approximately twice as complex as simple parity in asymptotic terms. Moreover, as 'm' increases, the complexity of the RS code relative to simple parity also increases, reaching an asymptotic level of approximately 0.1875 times greater [6].

6. Application

Reed-Solomon (RS) code is widely employed as channel code in the domains of digital communications and digital storage systems. It plays a crucial role in error correction across diverse systems, including storage devices (such as tapes, optical discs, DVDs, and bar code), wireless and mobile communication (encompassing cellular phones and microwave links), space and satellite communication [8], DTV/DVB/HDTV [3], high-speed modems (like ADSL and xDSL). More recently, it has been implemented in ATM networks [9], a core technology for implementing B-ISDN services. The broad acceptance of RS code is primarily attributed to its inherent properties that render itself exceptionally suited for error correction across a wide range of applications. Of particular significance is its proficiency in correcting burst errors, making it particularly attractive for communication channels [3].

EVENODD code serves as a notable example as an erasure coding. EVENODD code typically requires fewer disks or storage nodes to withstand an equivalent number of disk failures or storage node failures. EVENODD code finds applications in various storage systems, including data grids and distributed storage (e.g., Facebook File System) [10]. Damage is common in data storage, and data loss can have catastrophic consequences for the entire system. Storage industry, academic projects as well as major technology corporations routinely employ erasure coding [9]. Furthermore, to attain higher fault tolerance, designs like EVENODD (p, 3) [11] and EVENODD (p, 4) [10] have

been introduced. In contrast, RS code finds broader application across various scenarios and industries, whereas EVENODD code is primarily confined to its common usage within storage systems.

7. Conclusion

In terms of complexity, the EVENODD code involves fewer XOR operations compared to the RS code. As a result, in encoding operations, the generalized EVENODD code proves to be significantly more efficient and valuable than the RS code. RAID6 algorithms with EVENODD have the advantage of exclusively using XOR operations to achieve optimal redundancy, which is widely adopted in storage systems.

However, the RS code offers advantages when it comes to double failure recovery, as it results in less I/O. This is because the uneven distribution of redundant parity bits in EVENODD code can easily lead to I/O bottlenecks. In contrast, the RS code finds application in a wider range of scenarios and industries, while EVENODD code is primarily prevalent within storage systems.

These comparisons and analyses can provide better insights into the future designs about optimal fault tolerance coding strategy. There are many improvement designs, with the general direction of reducing network transmission volume, reducing communication nodes, reducing disk read data, repairing multiple lost symbols at once, better dispersing repair load, concurrent repairs, and reducing repair time.

References

[1] Z. Jiang. Performance Analysis of Utilizing Reed Solomon Code in Redundant Array of Independent Disk. 2022 International Symposium on Advances in Informatics, Electronics and Education (ISAIEE), 2022, 69 - 72.

[2] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. Journal of the Society for Industrial & Applied Mathematics,1960, 8 (2): 300 - 304.

[3] H. Benjamin and B. Kamali. Selection of the most efficient Reed-Solomon code for a specific application using neural networks. 1999 IEEE Communications Theory Mini-Conference, 1999, 58 - 61.

[4] Y. Lan, H. Hou and P. Zhang. Efficient Repair Algorithm for Information Column of EVENODD(p,4) Codes. 2020 IEEE 20th International Conference on Communication Technology (ICCT), 2020, 447 - 453.

[5] M. Blaum, J. Brady, J. Bruck and Jai Menon. EVENODD: an efficient scheme for tolerating double disk failures in RAID architectures. in IEEE Transactions on Computers, 1995, 4 (2): 192 - 202.

[6] Rajkumar Buyya, Toni Cortes and Hai Jin. The EVENODD Code and its Generalization: An Efficient Scheme for Tolerating Multiple Disk Failures in RAID Architectures. in High Performance Mass Storage and Parallel I/O: Technologies and Applications, IEEE, 2002, 187 - 208.

[7] Y. Chen. Efficiency Comparison of Row-Diagonal Parity and EVENODD Encoded Check Disk Repair Algorithms. 2022 International Symposium on Advances in Informatics, Electronics and Education (ISAIEE), 2022, 55 - 58.

[8] Stephen B. Wicker; Vijay K. Bhargava. Reed-Solomon Codes and the Exploration of the Solar System. IEEE, 1994, 25 - 40.

[9] S. J. Li, K. F. Pan, J. S. Yuan, A. J. Vigil and A. Berg. Adaptive Reed-Solomon coding for wireless ATM communication. Proceedings of the IEEE Southeast on 2000. 'Preparing for The New Millennium', 2000, 27 - 30.

[10] S. Huang, H. Hou and X. Yu. A Lower Bound on Disk Reads for Single Information Disk Failure Recovery and One Recovery Scheme for EVENODD (p, 3). 2019 Ninth International Workshop on Signal Design and its Applications in Communications (IWSDA), 2019, pp. 1 - 5.

[11] J. Haibo, F. Mingyu, X. Yilong, W. Xiaojing and W. Yu. Improved decoding algorithm for the generalized EVENODD array code. Proceedings of 2012 2nd International Conference on Computer Science and Network Technology, 2012, 2216 - 2219.