Comparison Of Two Encoding Methods in RAID6 Storage Architecture and Analysis of Improvement of EVENODD Encodings

Pengyang Wu*

Communication engineering, Beijing Jiaotong university, Weihai, Shandong, 264400, China

* Corresponding Author Email: 20721031@bjtu.edu.cn

Abstract. This paper provides a comparative analysis of two widely used error correction codes: RS (Reed-Solomon) and EVENODD. The analysis covers principles, algebraic forms, computational complexities, minimum Hamming distances, decoding processes, and specific application scenarios. RS encoding is known for its versatility, allowing for the flexible addition of parity symbols. However, it comes with a high computational cost due to polynomial evaluations. On the other hand, EVENODD encoding offers simplicity and computational efficiency through mere XOR operations but has limitations on the number of parity symbols it can generate. The paper explores the algebraic representations of both codes, discusses their minimum Hamming distances, and evaluates their decoding processes. Additionally, it analyzes the computational requirements of EVENODD under small write operations and provides a comprehensive comparison of the computational complexities of RS and EVENODD, aiding practitioners in choosing the most suitable error correction scheme for specific applications. A seeming limitation in our building process is because of the number of information disks must fall into a prime number. Nevertheless, if the preferred quantity of data sectors isn't a prime number, we can easily posit the existence of additional disks filled entirely with zeros, and this won't impact the encoding and decoding procedures.

Keywords: RS code; EVENODD; Encoding.

1. Introduction

Reed-Solomon (RS) encoding is a powerful error correction technique that plays a pivotal role in data communication and storage systems. At its core, RS encoding operates on symbols rather than individual bits, making it robust and versatile. RS encoding relies on the mathematics of finite fields to generate redundancy that can be utilized to detect as well as correct errors in information data disks. It achieves this by creating a set of equations based on the data symbols and then calculating parity symbols from these equations. The key advantage of RS encoding lies in its ability to customize the number of parity symbols according to the specific requirements of the system. This flexibility allows RS codes to adapt to different error correction needs, making them widely applicable.

One notable advantage of RS encoding is its flexibility in adjusting the number of parity symbols to suit different scenarios. However, this flexibility comes at the cost of increased computational complexity, as RS encoding involves polynomial evaluations, making it resource intensive.

EVENODD encoding is an alternative error correction technique known for its simplicity and efficiency. It offers a different approach compared to RS encoding, with distinct strengths and weaknesses. EVENODD encoding simplifies error correction by employing XOR (exclusive OR) operations exclusively. It divides data into blocks and calculates parity for each block, with one block's parity being XORed with the next block's data. EVENODD code is a data storage and redundancy handling method developed to improve the reliability of storage systems, particularly in RAID configurations.

Its fundamental function involves distributing parity information across multiple disks within a RAID array. Unlike RAID 5, which protects against a single disk failure with one parity disk, EVENODD extends this protection to multiple disk failures. It achieves this by establishing a mathematical relationship between data and parity, facilitating data recovery even when 2 disks fail simultaneously. Additionally, EVENODD supports both software and hardware implementations and

can be easily parallelized for faster processing. However, it has a limitation on the maximum number of parities checking symbols it can generate.

EVENODD's primary advantage lies in its computational simplicity, thanks to its reliance on XOR operations. It can be efficiently implemented in both software and hardware and supports parallel processing. Nevertheless, EVENODD is constrained by an upper limit of 2 on the number of parity symbols it can generate, which may restrict its application in situations requiring extensive error correction.

2. Algebraic Form Analysis of EVENODD and RS Encoding Schemes

We can use the polynomial M(x) to rewrite the parity check matrix like the Reed Solomon code by adding one column full of 0s. For a (m - 1) * (m + 2) data disk, we shall assume that each column in the data matrix is a polynomial modulo M(x):

$$a(\beta) = a_{m-2}\beta^{m-2} + \cdots + a_1\beta + a_0 \tag{1}$$

So, the H matrix of EVENODD can be derived:

$$H = \begin{bmatrix} 1 & 1 & \cdots & 1 & 1 & 0 \\ 1 & \beta & \ldots & \beta^{m-1} & 0 & 1 \end{bmatrix} \tag{2}$$

As codeword multiplies adverse of H is zero vector, it can be seen that that the minimum hamming distance of EVENODD is 3. If there exist two codewords and their hamming distance is 2, since

$$c \cdot H^T = [0 \quad 0] \tag{3}$$

and nonzero codeword weight is 2, there must be 2 columns of H that their molulus sum is 0, which is impossible when m is prime.

The quantity of iterations demanded for error correction is a critical performance metric for these codes. RS encoding generally demands a higher number of iterations during decoding due to its structured algebraic form. It involves polynomial evaluations and matrix operations, which can be computationally intensive, especially for longer codewords. EVENODD encoding, on the other hand, excels in terms of iteration count. Its decoding process is notably efficient, typically involving fewer iterations. This efficiency stems from the simplicity of XOR operations, which are inherently faster to execute, as is shown in Fig 1. Consequently, EVENODD is well-suited for scenarios where low-latency error correction is crucial.

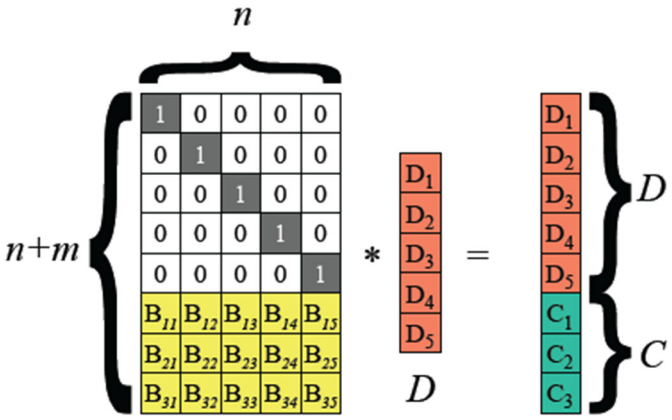

Fig 1. Generation of RS code

However, it's important to note that the efficiency of decoding also depends on factors like the code's minimum Hamming distance and the specific decoding algorithm used. RS codes may still outperform EVENODD in certain scenarios, particularly when dealing with more complex error patterns or when a high level of error correction is required.

3. Minimum Hamming Distance Analysis of EVENODD & RS on Encoding and Decoding

3.1. Decoding Analysis for EVENODD

EVENODD can theoretically correct up to 2 disk failures. It can detect up to 2 erasures and correct 1 error because the dmin=3. In decoding process, for the two failures we have 4 cases shown below:

Either 2 parity disks failed, or

2 information disks failed, or

1 information and parity 1 failed, or

1 information and parity 2 failed.

(1) If 2 parity disks failed, the decoding would certainly be successful because we can calculate the exact value of two parity disks using the principles discussed above.

(2) 2 information disks failed:

Assuming the data in position row i and column j is a(i,j) . Supposing i and j columns encounter the failures (i<j<m, 2 information disks). S (0) denotes the parity 1 and S (1) parity 2.

$$a_{\langle -(j-i)-1\rangle_m,j} \Leftarrow S^{(1)}{}_{\langle i-1\rangle m} \tag{4}$$

After calculating S, we need to find the first element that is isolated in an equation of parity 2. It's obvious in the regulation that $t \neq L+1$, so we let t be i. If L is equal to i-1, the element in parity 2 $a_{i-1,m+1}$ can be used to work out the block in j column $a_{\langle i-1-j\rangle m,j}$.

Afterwards we use parity 1 in that row and other known data blocks in that row to calculate the other missing data block in column i:

$$a_{\langle -(j-i)-1\rangle_m,i} \Leftarrow S^{(0)}{}_{\langle -(j-i)-1\rangle_m} \oplus a_{\langle -(j-i)-1\rangle_m,j} . \tag{5}$$

Similarly, apply this rule to fix the next L and element in parity 2 $a_{L,m+1}$:

$$L-i=-(j-i)-1. \tag{6}$$

So, we can use this block in parity 2 to calculate the next element in column j:

$$a_{\langle L-t\rangle m,t} \overset{t=j}{=} a_{\langle -j+2i-1-j\rangle m,t} = a_{\langle -2(j-i)-1\rangle m,t} . \tag{7}$$

Then extend to the general form, a((-l(j-i)-1) mod m),j) equals S XOR element in parity2 a(l(i-j)+j-1,m+1).

The row number x is obtained by solving the equation.

$$\langle x-j\rangle_m = \langle -l(j-i)-i-1\rangle_m \text{, where t=j} \tag{8}$$

Use this to obtain the next element that is isolated in parity 2:

$$a_{\langle -l(j-i)-1\rangle_m,j} \Leftarrow S^{(1)}{}_{\langle -l(j-i)+j-1\rangle_m} \oplus a_{\langle -(l-1)(j-i)-1\rangle_m,i} \tag{9}$$

Repeat this cycle, let L be 2 to m - 1. It's only when m and j - i are mutually prime so that the row number $-l(j-i)-1$ when L ranges from 1 to m can be distinct when modulus m, so that the cycle won't stop. That's why m is prime is indispensable. (3) and (4): There is 1 variable and 1 equation, which resembles the situation in (2) but simpler.

3.2. Analysis into RS Code's Robustness

For the RS code, a comparison of performances involving a fundamental coding method is given in the figure below. In the implementation, codeword disks from the (255,144,113) Reed–Solomon

code undergo modulation by virtue of a 256-QAM phase multiplication constellation and are then delivered through a Gaussian white noise (AWGN) channel [1]. In terms of the output signal of the channel, states are quantized into 8 bits. Three distinct performance figures are depicted, representing the outcomes of two tough decoding methods and one standing soft decoding methods.

Due to the illustration graph of EVENODD's data disks, what' s worth notice is that if 2 information disks failed, there are 8 missing blocks, and they should contain 5 blocks of distinct colors, so due to the drawer principle, there must be at least 1 color which has only 1 block, which means there is only 1 unknown variable in the corresponding parity, and that' s the block we want to find first when decoding, which is regarded as the breakthrough opening the decoding chain of wholesale recovery of data disks.

On the other hand, the two plain decoding methods encompass RDP decoding and an algebraic form of plain-decision decoding algorithm emerged specially for this study. One of the figures indicates the asymptotic decoding efficiency for a substantial number of intersection points, implying its large capacity of data size. By contrast, Fig 2 shows another curve demonstrates that it is feasible to closely approach asymptotic performance using a finite list size, which is predicted to contain 32 codewords to the most.

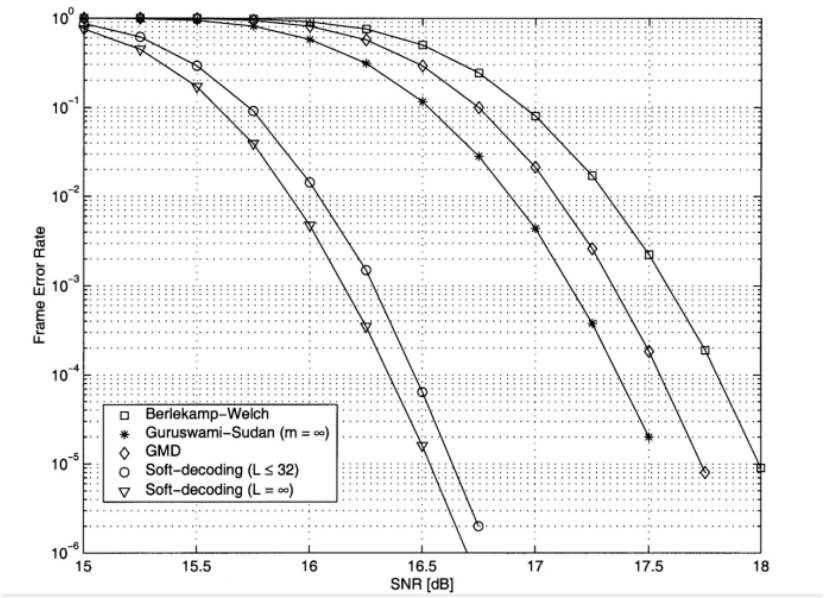

Fig. 2 Performance comparison for a simple coding scheme [2]

4. EVENODD Code Computation Analysis in Small Write Operation

In scenarios where multiple disks are employed within a system, it is common to face situations requiring numerous tiny write operations. A tiny write operation is characterized by updating a single data sector, effectively altering just one symbol within the system. The EVENODD system provides a high degree of versatility in handling such operations due to its capacity to work with symbols of varying sizes. In practice, a symbol is often implemented as a disk sector, aligning with the typical approach for managing these small write operations.

4.1. Rewrite Blocks Other Than Diagonal

Whenever an information symbol undergoes a rewrite operation, the parity element isn't qualified in the diagonal positions (m-3, 2), (m-4, 3), ..., (1, m-2), it has an impact on only two redundant symbols. Consequently, the process necessitates just tribal R and W operations to complete the encoding process. By considering an element as equivalent to a data sector, when we update a disk sector, the typical scenario requires reading only three data sectors: the element needed to be updated

plus two additional redundant data disks including parity disks. Subsequently, three disk sectors need to be written to complete the update process efficiently.

Undoubtedly, if the element $a_{i,j}$ ($0 \le i \le m-2, 0 \le j \le m-1, \langle i+j \rangle_m \ne m-1$), which means it isn't located in diagonal elements(S), is supplanted by element R($r \leftarrow a_{i,j}$), such modifications should be implemented for other symbols [3]:

$$a_{i,m} \leftarrow a_{i,m} \oplus a_{i,j} \oplus r \tag{10}$$

This formula operates on the row parity of row i.

4.2. Rewrite Blocks of Diagonal

For the diagonal parity, if the information symbol being rewritten is situated within the diagonal positions (m-4, 3), (m-5, 4), ..., (1, m-2), a distinct scenario arises. In this case, all the elements within the (m+1) parity 2 column are involved by this operation, and subsequently, the corresponding symbol in parity 1 is also affected as part of the process [4]. If element ($a_{i,j}$ $0 \le i \le m-2, 0 \le j \le m-1, \langle i+j \rangle_m = m-1$), which means it is located in diagonal elements(S).

Such modifications should be implemented for other symbols:

$$a_{i,m} \leftarrow a_{i,m} \oplus a_{i,j} \oplus r \tag{11}$$

However, at the same time, all symbols in parity 2 should be updated:

$$a_{t,m+1} = a_{t,m+1} \oplus a_{i,j} \oplus r\,. \tag{12}$$

Let's make the assumption that we use the following encoded data disks which are table 1 and table 2:

Table 1. Array needing small write operation [5]

Array needing small write operation						
0	1	1	0	0	0	0
0	1	0	0	0	1	1
0	0	1	0	0	1	1
0	0	1	1	0	0	1

Suppose that we want to overwrite data element (1, 2) with a number 0. Given it doesn't lie in diagonal (4, 2), (3, 3), (2, 4), (1, 5), due to formulas illustrated above, we need to update elements (1, 6) and (2, 7). The new data disk is.

Table 2. Array after small write operation

Array after small write operation						
0	**0**	1	0	0	**1**	0
0	1	0	0	0	1	**0**
0	0	1	0	0	1	1
0	0	1	1	0	0	1

4.3. Tradeoffs & Complexity Computation in Encoding of RS and EVENODD

In this section, we conduct a comparison of the complexity between EVENODD and a conventional error detection and correction code methods, specifically a Reed-Solomon code algorithm. Both RS and an EVENODD codes require an expected limitation of number of redundant data disks, which is exactly 2. Nonetheless, a main advantage of EVENODD code is that it fully relies on parity checking devices in hardware, a component commonly found in standard RAID-5 controllers [6]. As a result, EVENODD code can be performed on formal RAID-5 controllers,

independent of any hardware modifications [7]. In contrast, the method targeted at RS codes mandates specialized hardware to correspond with computations in finite algebraic fields, making it incompatible with standard RAID-5 controllers.

Assume size of information matrix is (n-1) and n, row parity is that there are (n-1) times of XOR in one row, so there are (n-1) ^2 times of XORs. Diagonal parity contains calculating S: (n-2) times of XORs and calculating parities: (n-2) *(n-1) as well as modulus sum: (n-1) times of XORs [8]. Totally there are (n-1) (n-2) +(n-1) +(n-2) =n^2-n-1 times of XORs in diagonal parity encoding.

In summary, protecting n(n-1) data blocks use (2n-3) n XORs.

Additionally, if we compare the method above with the encoding process of RS code, we can derive that parity 1 encodings are interchangeable when computing. As a result, the distinction of two coding schemes in complexity of the encoding is attributed to the different processes in working out the second redundancy parity check disk [9]. Consider a RS scheme of codeword length 8 specifically. Each time of multiplying by quantity a is equivalent to being multiplied the companion matrix A in (13) [10]:

$$A=\begin{pmatrix}0&1&0&0&0&0&0&0\\0&0&1&0&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&1\\1&0&1&1&1&0&0&0\end{pmatrix}. \tag{13}$$

In terms of multiplying the byte (c1, c2, c3, etc., c8) by quantity a, the process requires 3 times of XOR operations. Actually, the result matrix of multiplying the data matrix above by the matrix A will generate the byte (c8, c1, c2 XOR c8, c3 XOR c8, c4 XOR c8, c5, c6, c7). Summing them up, the implementation of encoding would need 3x XOR operations. So, implementing on the bytes b0, b1, ⋯, b(m-1) requires such times of XORs [11].

$$8(m-1)+\sum_{i=1}^{m-1}3i=\frac{3m^2+13m-16}{2} \tag{14}$$

Table 3 below compares the number of XOR operations needed to encode (m-1) bytes per disk in a disk array with m information disks:

Table 3. Comparison of RS and EVENODD's complexity of encoding [12]

Number of information disks	EVENODD	Reed-Solomon	improvement factor
5	312	376	1.21
7	664	954	1.44
11	1752	3250	1.86
13	2488	5112	2.05
17	4344	10624	2.45
23	8088	24442	3.02
29	12948	46648	3.59
31	14872	56250	3.78

5. Conclusion

In our paper, we have introduced an innovative approach called EVENODD to do the task of overcoming double disk failures in RAID architectures. Our EVENODD method offers several significant advantages when compared to other existing recovery methods for handling two disk failures:

Efficient Redundancy: EVENODD requires the addition of only two redundant disks to tolerate two disk failures, which is an optimal configuration. Simplicity and Compatibility: It relies on straightforward exclusive-OR computations and only necessitates parity hardware, commonly found

in formal RAID-5 processors. This means EVENODD can ubiquitously integrate into formal RAID-5 processors without need for additional hardware modifications.

Integration Flexibility: EVENODD can be seamlessly incorporated into established RAID techniques. For instance, it allows for the distribution of parity across all disks, mitigating potential bottlenecks during repeated write operations, as often seen in RAID-5 setups. Symbol Size Versatility: Our method accommodates symbols of varying sizes, ranging from bits to multiple data disks, without imposing constraints on size of bits.

Decoding of EVENODD in program: Every consecutive two columns in EVENODD represent a parity disk's involving elements. Then the decoding operator uses a two-tier loop to recover the data disks, where the outer loop is related to number of parity check element L, which ranges from 1 to m-1 to consider all the elements respectively in redundant disk parity 2. The inner loop is operated on the four elements that are involved in the parity 2 element. By calculating how many disks in the 4 disks are missing, we can select the data disk element with only 1 disk lost. Then we fix the element and thus working out the other row element by parity 1.

References

[1] M. Blaum, J. Brady, J. Bruck, Jai Menon. EVENODD: an efficient scheme for tolerating double disk failures in RAID architectures. IEEE Transactions on Computers, 1995, 44(2): 192-202.

[2] R. Koetter, A. Vardy. Algebraic soft-decision decoding of Reed-Solomon codes. IEEE Transactions on Information Theory, 2003, 49(11): 2809-2825.

[3] W. Chen, T. Wang, C. Han, J. Yang. Erasure-correction-enhanced iterative decoding for LDPC-RS product codes. China Communications, 2021, 18(1): 49-60.

[4] R. Koetter, F. R. Kschischang. Coding for errors and erasures in random network coding. IEEE Transactions on Information Theory, 2008, 54(8): 3579-3591.

[5] R. Koetter, A. Vardy. Algebraic soft-decision decoding of Reed-Solomon codes. IEEE Transactions on Information Theory, 2003, 49(11): 2809-2825.

[6] D. V. Sarwate, N. R. Shanbhag. High-speed architectures for Reed-Solomon decoders. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2001, 9(5): 641-655.

[7] D. S. Papailiopoulos, A. G. Dimakis. Locally repairable codes. IEEE Transactions on Information Theory, 2014, 60(10): 5843-5855.

[8] I. Djurdjevic, Jun Xu, K. Abdel-Ghaffar, et al. A class of low-density parity-check codes constructed based on Reed-Solomon codes with two information symbols. IEEE Communications Letters, 2003, 7(7): 317-319.

[9] H. Hou, P. P. C. Lee. A new construction of EVENODD codes with lower computational complexity. IEEE Communications Letters, 2018, 22(6): 1120-1123.

[10] H. Fu, H. Hou, L. Zhang. Extended EVENODD+ codes with asymptotically optimal updates and efficient encoding/decoding, 2021 XVII international symposium "problems of redundancy in information and control systems" (REDUNDANCY). Moscow, Russian Federation, 2021: 1-6.

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

[12] H. Hou, Y. S. Han, K. W. Shum, et al. A unified form of EVENODD and RDP codes and their efficient decoding. IEEE Transactions on Communications, 2018, 66(11): 5053-5066.