Journal of Engineering, Project, and Production Management
2026, 16(2), 2025-172

Efficient Management Technology of Big Data in Distributed Storage Economy Management Field Using Erasure Code

Xin Liu
Lecturer, School of Economics and Management, Zhengzhou Normal University, Zhengzhou, 450044, China, E-mail: liuxin199010@outlook.com

Project Management
Received August 20, 2025; revised October 9, 2025; accepted October 11, 2025
Available online March 7, 2026

Abstract: With the growth of big data, the application of distributed storage is becoming increasingly widespread. Traditional Erasure Codes (ECs) are computationally intensive in distributed storage systems and inefficient for managing large-scale economic data. To address this issue, a Reed-Solomon code storage method based on the Reed-Muller transform is proposed. The encoding and decoding algorithm of Reed-Solomon (RS) codes is verified through the Reed-Muller (RM) transform, and a cyclic shift is added to the Reed-Muller transform to reduce the computational complexity of ECs. The distributed storage file system is expanded to ensure smooth operation across the system's modules. The experimental results showed that in the extended file system, the proposed Reed-Muller Reed-Solomon code achieved fast encoding and decoding speeds across different numbers of data blocks, with maximum encoding and decoding speeds of 981MB/s and 915MB/s, respectively. The EC proposed by the research was not affected by file size, and its encoding and decoding speed increased with increased file size. Moreover, the Reed-Muller and Reed-Solomon codes can efficiently handle economic data with lower memory usage than other ECs. Overall, the Reed-Muller Reed-Solomon code proposed by the research has good performance and can efficiently handle big data.

Keywords: Erasure code (EC), distributed storage, data management, reed-solomon (RS) code, reed-muller (RM) transformation.

DOI 10.32738/JEPPM-2025-172

1. Introduction

As technologies such as big data and the Internet of Things rapidly grow, there are various technical means for data collection, and the required storage capacity of data has also increased sharply (Liu et al., 2022). The storage method of data has also shifted from centralized storage to distributed storage. The construction and storage costs of centralized systems are relatively high, so distributed storage systems have emerged (Li et al., 2023). In distributed storage systems, Erasure Codes (ECs) are increasingly being adopted by commercial management systems to ensure data security and reduce data management costs (Xu, 2024). The current EC is broadly utilized in distributed storage systems, reducing management costs and improving data security. Commonly used ECs include RS and Low-Density Parity-Check (LDPC) codes (Facenda et al., 2023). The RS code has good encoding and decoding performance, can tolerate multiple data failures, and has strong error-correction capabilities. However, RS is computationally complex, and the cost of repairing damaged data is relatively high. In addition, although the RS code improves the storage efficiency of the storage system, it correspondingly increases network traffic usage and network control input and output of the system (Sun et al., 2024). The Hadoop Distributed File System (HDFS) is the most commonly used storage method and is often used in distributed storage systems for economic management storage. HDFS has a high fault tolerance for data management and can simultaneously manage the storage of multiple sets of data. However, existing RS codes suffer from high computational complexity and high repair costs in large-scale economic data storage and recovery scenarios. Therefore, the key research question in this paper is how to retain the fault-tolerance advantages of RS codes while reducing computational overhead and improving the scalability of big data management. To address this issue, this paper proposes an RS code data storage method based on the RM transform. This method is based on RM variation to cyclically shift the data storage of RS codes to reduce computational complexity. At the same time, RM transformation is used to improve the encoding and decoding information of RS codes, expand the HDFS system, and enhance the efficient management capability of RM-RS codes for economic

big data.

The main theoretical developments and innovative contributions of this study are as follows:

(1) At the theoretical level, a new EC structure, RM-RS, that combines RM transformation and RS coding is proposed. By introducing a cyclic shift strategy and a redundant vector grouping mechanism, the computational path of traditional RS codes under high-density data blocks is optimized, reducing the complexity of XOR operations.

(2) A distributed storage system architecture for large-scale economic data is constructed, and the HDFS file system is extended to support the modular deployment of RM-RS coding in the writing, reading and recovery processes.

(3) From the system dimension, a correlation measurement model between encoding rate, decoding efficiency, resource usage, and recovery success rate is established to systematically evaluate the adaptability and scalability of RM-RS under different parameters.

(4) RM-RS can achieve higher throughput performance and lower resource consumption without relying on hardware acceleration, which is significant for the intelligent development of economic big data management.

2. Related Work

As big data advances, the application field of distributed storage has become increasingly widespread. As ECs are commonly used in distributed storage systems, they have received widespread attention from scholars at home and abroad. Ma et al. (2022) proposed a cross-platform data storage library PyRS based on RS EC to address the issue that existing EC can only be used on Linux platforms. The Vandermonde matrix was utilized as the encoding matrix, and the Numba library was applied to accelerate and optimize the program. The outcomes showed that, compared to a Linux platform, the Central Processing Unit (CPU) usage of PyRS code increased by 15% and the memory usage decreased by 5%. Gao et al. (2021) proposed a method based on partial re-encoding to address the issue of errors in the user memory of RS EC. By adding check bits to the generation matrix to improve fault localization performance, the reliability verification of the RS-EC decoder was carried out in a field programmable gate array. The experimental results showed that the decoder could tolerate 90% of user storage failures. Chen et al. (2023a) proposed an RS code based on advanced elasticity to address the problem that traditional RS codes cannot meet practical needs in redundant conversion. By establishing flexible coding matrices and layout strategies, different types of redundant conversions could be served to reduce conversion traffic. The experimental results showed that Advanced Elastic Reed-Solomon (AERS) codes reduced network traffic by 50% to 85.7%. Wu et al. (2023) proposed an elastic-based RS EC to address the issue of high network Input/Output (I/O) and reduced storage performance caused by redundant conversion. By eliminating data block relocation, limiting network I/O for parity block updates, and implementing collaborative design for encoding matrix construction and data placement, network gain could be achieved in both forward and backward directions. Kim (2023) proposed an orthogonal frequency division multiplexing system based on power-controlled RS coding to address the issue of codeword error rate, which was caused by control code character power. By deriving the signal-to-noise ratio function and proposing the system's word error rate (WER), the channel inversion at the transmitter was combined with the erasure generated by the receiver's sorting attenuation to provide a lower WER for the system.

In addition, the widespread use of distributed storage systems in data management has also attracted widespread attention from scholars both domestically and internationally. Yin et al. (2023) proposed an EC encoding and decoding method based on liberation codes to address the issue of reduced storage system performance during encoding and decoding of EC. By stripping and redesigning the data, the data nodes were divided into multiple virtual nodes, a symmetric decentralized hash ring was constructed, and then the virtual nodes were dispersed into the hash ring. Xu et al. (2021a) proposed a partition-based design-based EC encoding and decoding data method to address the issue of uneven traffic in distributed storage system racks caused by random data. The balanced cross-rack traffic distribution was provided by selecting replacement nodes and searching for lost data blocks. The results indicated that the proposed NewLib reduced data read latency by 62.83%. Xu et al. (2021b) proposed an EC-based recovery task scheduling module to address the problem of reduced fault recovery speed of data nodes in traditional EC storage systems. Traffic recovery between nodes was simulated through bipartite graphs, and tasks were intelligently selected and processed. The results showed that the proposed scheduling module improved the recovery speed by 30.68%. Jiang et al. (2023) proposed a coding system based on a programmable data plane to address the significant computational resources required for encoding traditional EC in distributed storage systems. By aggregating multiple data streams, the CPU and broadband consumption of the host encoder could be reduced, and the reliability of data and repair costs were balanced. Chen et al. (2023b) raised a comprehensive fault data repair scheme with EC to address the issue of isolated design in data repair technology for distributed storage systems. Two coefficients were calculated for the device, and an EC strip was created. The system hierarchy was utilized to perform repair operations and minimize cross-rack repair bandwidth.

According to the research of domestic and foreign scholars, traditional EC has a high traffic occupancy rate in distributed storage systems. This study proposes the encoding and decoding of RS codes based on RM transformation, aiming to reduce the traffic occupancy of RS codes in distributed storage systems and lower the computational complexity of storage systems.

3. Data Management of Distributed Storage System Based on RM-RS Code

3.1. Improved RS Code Design Incorporating an RM Transformation

At present, most distributed storage management adopts EC. EC divides the initial data into n data blocks and encodes them over a finite field to obtain m check blocks. The verification block and data block are stored on different nodes to

complete the fault-tolerant mechanism construction of EC (Liu et al., 2020). RS encoding is the most commonly used EC. Taking RS (6,4) as an example, the basic principle of EC encoding is shown in Fig. 1.

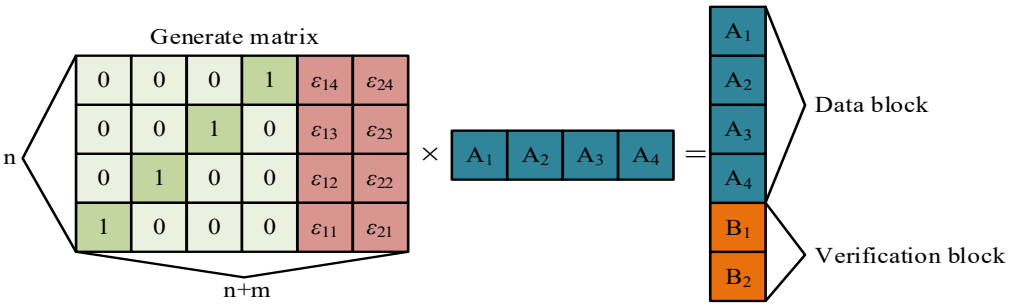


Fig. 1. Basic principle of EC coding

In Fig. 1, RS (6,4) divides the initial data into 4 data blocks and stores them in the storage system. The data blocks are encoded to obtain 2 check blocks, where ε_{ij} represents the encoding coefficient of the corresponding data block in the generation matrix. In a storage system, when a data block changes, due to the fact that the EC encodes the data block in a finite field to obtain a checksum block, the checksum block also changes accordingly. The checksum block updates with the update of the data block, which can be expressed by Eq. (1).

$$B_i^{(t)} = \sum_{j=1}^{k} \varepsilon_{ij} \times A_j^{(t)} \tag{1}$$

In Eq. (1), B_i is the number of test blocks, is the number of data blocks, and t represents the t-th change (for $t \geq 0$) of the data block. When EC performs data repair, if a data block is damaged, the internal generation matrix of EC is reversible, so the damaged data can be retrieved through reversible matrix operations (Zhang et al., 2022). RS code, as a classic EC of EC encoding, has good storage efficiency and is widely applied in distributed storage systems. The process of writing RS code into a distributed storage system is shown in Fig. 2.

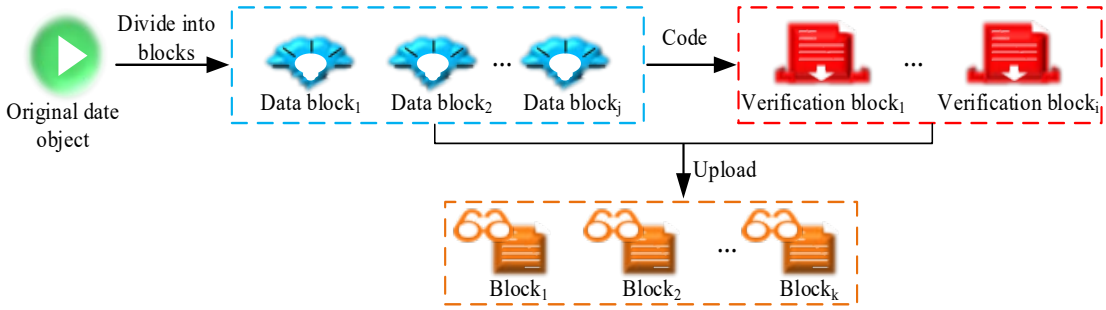


Fig. 2. RS code writing process to distributed storage systems

In Fig. 2, the RS EC decomposes the stored data into different data blocks, verifies the data blocks into different checksum codes, and reads the data blocks and checksum blocks accordingly. In the process of repairing data blocks, the efficiency of EC greatly affects the efficiency of repairing lost data blocks in distributed storage systems (Yao et al., 2022). In the traditional RS code for data block repair tasks, data blocks need to be obtained from other nodes internally, where the obtained data blocks are re-transmitted to the encoder for decoding work, and the lost data blocks are recovered. However, this process is affected by network speed, and network congestion will reduce the efficiency of data block repair. To reduce the computational workload of RS codes during the encoding process, an RM transformation is used for optimization. RM transformations can convert the input of data into output, and the transformation expression is shown in Eq. (2).

$$y^T = B_N \cdot x^T \tag{2}$$

In Eq. (2), y represents the output of RM variation, x represents the system input vector, and B_N represents the recursive matrix. The definition and expression of recursive matrix are shown in Eq. (3).

$$B_{2^{i+1}} = \begin{bmatrix} B_{2^i} & B_{2^i} \\ 0 & B_{2^i} \end{bmatrix}, B_1 = [1] \tag{3}$$

In Eq. (3), $i \geq 0$. At $N = 6$, RM changes as shown in Fig. 3.

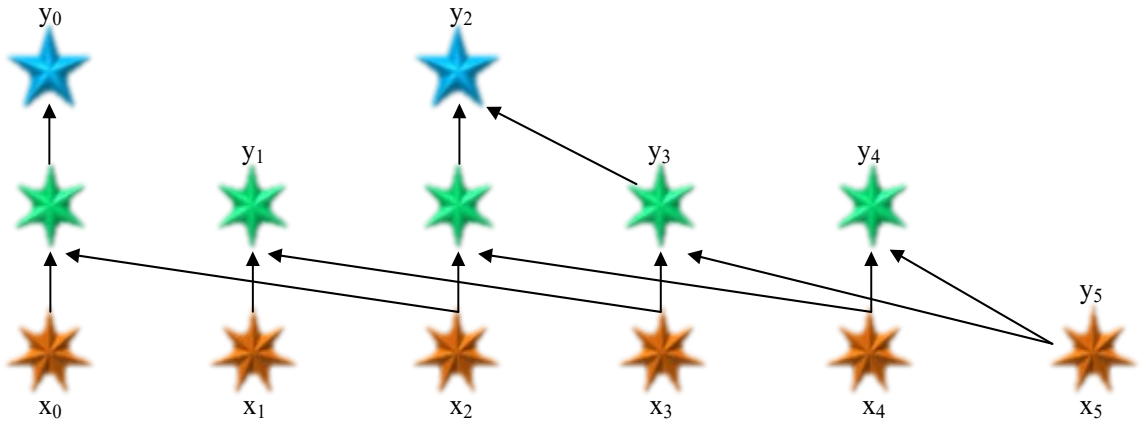


Fig. 3. RM Transformation diagram with N=6

In Fig. 3, in $N = 6$, the RM transformation is manifested as iterative calculation. The system inputs the vector $x = x_0, x_1, \cdots, x_5$ and performs repeated calculations on each intermediate layer node to obtain the output vector $y = y_0, y_1, \cdots, y_5$ of the RM transformation, reducing the number of calculations. To verify the RS code, the checksum term is expressed using a formula, as shown in Eq. (4).

$$p_i(x) = \sum_{j=0}^{k-1} \lambda_{j+1}^{i} c_j(x), i = 0,1,2 \tag{4}$$

In Eq. (4), $p_i(x)$ represents the check polynomial, and k represents the number of information polynomials. In $i = 0$, the first checksum expression is shown in Eq. (5).

$$c_k(x) = \sum_{j=0}^{\frac{R}{2}-1} c'(x) \tag{5}$$

In Eq. (5), R represents the power of the vector, and in $i = 1,2$, the checksum expression is shown in Eq. (6).

$$c_k(x) = \sum_{j=0}^{\frac{R}{2}-1} \lambda_j^i c_j(x) + (\lambda_j^i + v_{r-1}^i) c_{R/2+j}(x) \tag{6}$$

The output set $\{c_i(x)\}_{i=0}^{R-1}$ can be obtained from Eq. (5) and Eq. (6), with a value range of R. A new output set $\{c_i'(x)\}_{i=0}^{R-1}$ can be obtained through RM transformation, with a value range of $R/2$. The encoding algorithm can be used to repeat the transformation until the value of the set reaches 2. At $R = 8$, the RS encoding algorithm data is shown in Fig. 4.

In Fig. 4, in $R = 8$, the input set is $\{c_i(x)\}_{i=0}^{7}$. The solid arrows in the figure represent the data stream of the encoding algorithm, and the dashed arrows represent the multiplication of the sets during integration, which involves 16 XOR operations and 6 cyclic shifts. In the decoding algorithm of Binary-Matrix optimized Reed-Solomon (BM-RS) codes, by calculating and searching for the inverse element matrix of polynomials in advance, when the ring is represented as a prime number l greater than the number of information polynomials, to optimize storage, an additional inverse element smaller than $0.5l$ is stored in the data storage table, which can effectively reduce the computation time in the RM-RS decoding process. The expression of the XOR operation frequency during the encoding process of the BM-RS code is shown in Eq. (7).

$$XOR(l) = (2l - 2^s) + 2(s-1) + \sum_{r=1}^{s-1} 2^r - 1 \tag{7}$$

In Eq. (7), l represents a prime number. The RM transform verifies the results of polynomials by calculating the intermediate node layer data. A cyclic shift in the RM transformation is introduced to reduce its complexity in the encoding

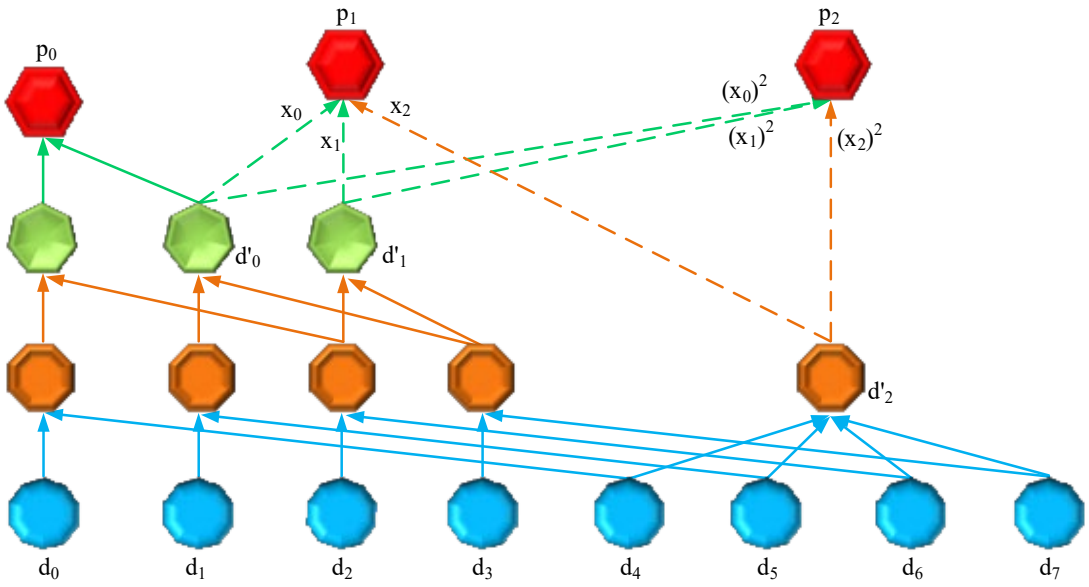


Fig. 4. Data graph of R=8 RS encoding algorithm

and decoding calculation process, but it will generate more iterative calculations. By moving the structure on the ring t positions in the opposite direction, node and shift operations can be achieved. The expression for the number of cyclic shifts is shown in Eq. (8).

$$CYC(l) = 2[\log_2 k] \tag{8}$$

In Eq. (8), k denotes the number of information polynomials.

3.2. Application of RM-RS Code in Efficient Management of Economic Data

To enable the efficient management of economic data using RM-RS codes, research is being conducted on expanding distributed storage systems to improve the efficiency of RM-RS codes in managing economic data. A distributed storage system composed of a group of information-exchange and computing nodes using message passing can enable ordinary computers to perform large-scale storage and computing tasks (Alkhabet and Ismail, 2023). The centralized metadata structure and P2P distributed structure are two typical structures of distributed storage systems. The composition of the distributed storage system is shown in Fig. 5.

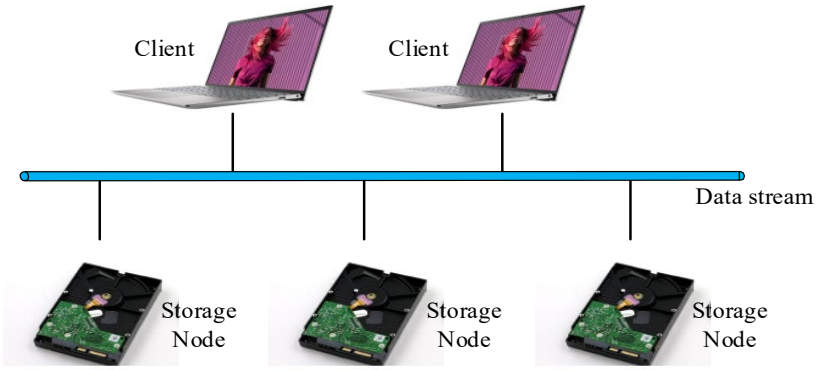


Fig. 5. Composition of a distributed storage system

In Fig. 5, the operation of a distributed storage system mainly involves users storing data streams in the system through clients, and the distributed storage system then stores the received data streams separately in storage nodes. HDFS and Google GFS storage systems are commonly used in distributed storage systems. HDFS has high fault tolerance, can store a large amount of data, and can repeatedly read data. Therefore, research has adopted HDFS for economic data management (Zhang et al., 2024). In traditional HDFS, data nodes cannot be expanded, and raw data is stored in a single data node. Therefore, to store a large amount of data, HDFS is extended on the basis of HDFS. The schematic diagram of the extended HDFS is denoted in Fig. 6.

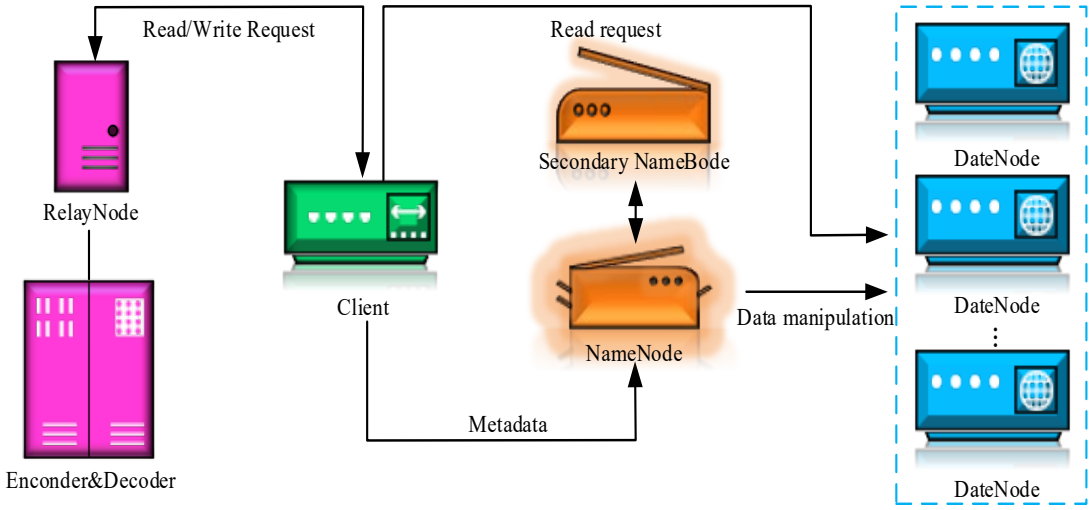


Fig. 6. Schematic diagram of an extended HDFS operation

In Fig. 5, the operation of a distributed storage system mainly involves users storing data streams in the system through clients, and the distributed storage system then stores the received data streams separately in storage nodes. HDFS and Google GFS storage systems are commonly used in distributed storage systems. HDFS has high fault tolerance, can store a large amount of data, and can repeatedly read data. Therefore, research has adopted HDFS for economic data management (Zhang et al., 2024). In traditional HDFS, data nodes cannot be expanded, and raw data is stored in a single data node. Therefore, to store a large amount of data, HDFS is extended on the basis of HDFS. The schematic diagram of the extended HDFS is denoted in Fig. 6.

In Fig. 6, the file data is first encoded by the intermediate node during transmission. The encoded data is passed from the client to the name node, and a secondary name node is added to store the received data in the data node through the name node. This working principle ensures smooth operation between modules and avoids interdependence between them. After expansion, HDFS can divide the file operation process into two parts for data management: read and write. In the process of expanding HDFS for data management, the RM-RS code is added to perform load calculation on the data within the system, as expressed in Eq. (9).

$$load_i = a_i(t) + \beta_i(t) \tag{9}$$

In Eq. (9), a_i represents the resource utilization rate of the storage system, and β_i represents the storage time of data in the system. The load balancing of storage systems can be divided into two categories: one is when the system itself stores data and manages it, and the other is when the system passively accepts data storage and hands it over to the system for management. Load balancing is performed by adding RM-RS codes, and its balancing expression is shown in Eq. (10).

$$d(x_j, s_i) = \frac{-(\ln(1-(\left|h_1(x_j) - h_2(s_i)\right|))}{w_i} \tag{10}$$

In Eq. (10), s_i represents the number of servers, x_j represents metadata, and w_i represents weights. The state of a distributed storage system environment can represent the total computing power of data management, as expressed in Eq. (11).

$$S_t = C(t) = \{c_1(t), c_2(t), \cdots c_n(t)\} \tag{11}$$

In Eq. (11), t denotes the set of data at a certain time. Due to the fact that HDFS extensions use initial data management for data management, there will be replica management for data stored in distributed storage systems. The stored data is encoded using RM-RS code and divided into several data blocks, which are then decoded to become data check blocks. When using RM-RS codes for data storage in distributed storage systems, if the data is damaged, RM-RS codes use block coding to repair the damaged data. In distributed storage systems, RM-RS codes can be used for data repair in a serial manner, and the repair time consumed during the repair process can be expressed mathematically, as shown in Eq. (12).

$$t_s = \sum_{i=0}^{r-1} t_i = \sum_{i=0}^{r-1} \frac{B}{\beta_i} \tag{12}$$

In Eq. (13), r represents the number of data block failures, B represents the size of the data block, and β represents the data bandwidth in the system. In the repairing failed data blocks, as the repair time of the data block increases, the success rate of data repair corresponding to the data block will also decrease with the increase of repair time, as expressed in Eq. (13).

$$t'_p = \max\{\frac{sB}{\beta}, \frac{B}{\beta_s}, \frac{B}{\beta_{s+1}}, \cdots, \frac{B}{\beta_{r-1}}\} \tag{13}$$

In Eq. (13), s represents the s-th repair topology in the data bandwidth. When a data block is lost, the encoding matrix of the RM-RS code can be used to identify the factors causing the loss of the data block. Using the encoding matrix, the content of the data block can be recalculated, as shown in Eq. (14).

$$D = (F')^{-1} F'D = (F')^{-1} P' \tag{14}$$

In Eq. (14), F' represents the full rank matrix, $(F')^{-1}$ represents the inverse matrix of the full rank matrix, and P' represents the encoded vector of the data block. Using the RM-RS code to decode data blocks, the calculation expression is shown in Eq. (15).

$$F = \begin{bmatrix} E \\ X \\ M \end{bmatrix}, P = \begin{bmatrix} D \\ C \\ G \end{bmatrix} \tag{15}$$

In Eq. (15), M represents the number of data blocks, and C represents the number of check blocks. By using RM-RS codes to store economic data in extended HDFS, efficient data management can be achieved.

4. Empirical Analysis of Economic Data Management based on RM-RS Code

4.1. Performance Verification Analysis of RM-RS Code

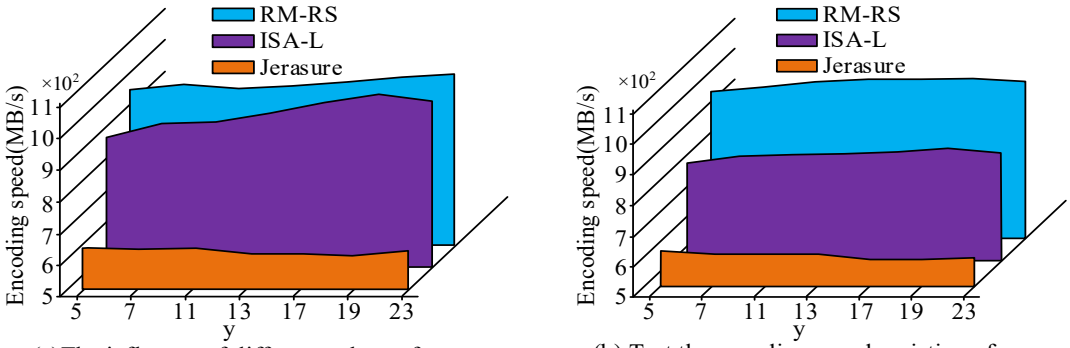
To verify the encoding performance of RM-RS, experimental analysis was conducted in an HDFS distributed storage system with the experimental parameters indicated in Table 1. The encoding and decoding performance of BM-RS was compared with other classic ECs by adjusting different parameters, and the impact of different parameters on the encoding performance of EC is shown in Fig. 7.

From Fig. 7(a), regardless of the value of y in data block y, the encoding speed of RM-RS was the fastest among the three different ECs, with the fastest speed reaching 981MB/s. However, the encoding speed of the Jerasure code was slow under different y values, and as the y value increased, the encoding speed decreased continuously. The RM-RS code increased in encoding speed and performance as the value of y value continually increased. Therefore, RM-RS codes could improve encoding speed by increasing the value of y. From Fig. 7(b), under the condition of different values of the prime number o, all three different ECs could have relatively stable encoding speeds. However, the decoding speed of the Jerasure and Intel Storage Acceleration Library (ISA-L) codes was lower than that of the BM-RS code. As the value of o increased, the encoding speed of the Jerasure code decreased, indicating a continuous decrease in encoding performance. As the value of o increased, the encoding speed of ISA-L code also kept rising, with a maximum speed of 872MB/s. When the value of o was set to 5, the RM-RS code had the lowest encoding speed, maintaining at 943MB/s, and as the value of o increased, the encoding speed also increased continuously. From this, the coding performance of the BM-RS code proposed in the study is better than the other two commonly used EC codes. The impact of different parameters on the decoding performance of EC is shown in Fig. 8.

Table 1. Experimental parameter variables

Parameter	Explanation
y	Number of data blocks
o	Prime numbers greater than 2, $y \le o$
File size	Original file size (Default 128MB)

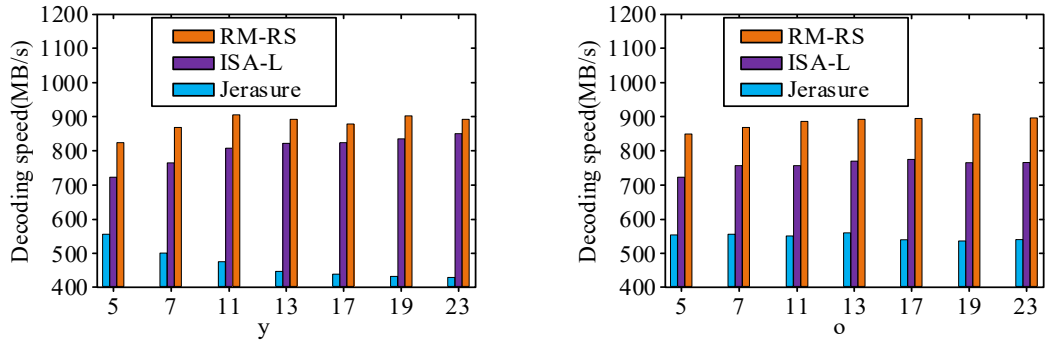
In Fig. 8(a), regardless of the value of y, the decoding speed of the RM-RS code was relatively high, with a maximum of 915MB/s. When the y value of the Jerasure code was set to 5, the decoding speed was the highest, at 560MB/s. As the value of y continued to increase, the decoding speed of the Jerasure code decreased, resulting in a continuous decline in the decoding performance of the Jerasure code. As the y value increased, the decoding speed of ISA-L code continued to rise, with a maximum decoding speed of 843MB/s, but the decoding speed still failed to exceed that of RM-RS code. This is because the study added cyclic shift to RM-RS codes, which can improve their decoding performance. From Fig. 8(b), the decoding speed of the three different ECs varied when o was taken at different values. During the process of increasing the value of o, the decoding speed of RM-RS code continuously improved with the increase of o value, until the o value reached 13. After the value of o reached 13, the trend of decoding speed tended to stabilize, with a maximum of 910MB/s. As the value of o increased, the decoding speed of the Jerasure code showed a slight downward trend. When the value of o was 13, the decoding speed was the highest, at 576MB/s. It can be inferred that the value of prime number o is extremely important for the decoding performance of EC. The impact of different file sizes on the performance of EC encoding and decoding is shown in Fig. 9.



(a)The influence of different values of data blocks on encoding speed

(b) Test the encoding speed variation of prime numbers with different values

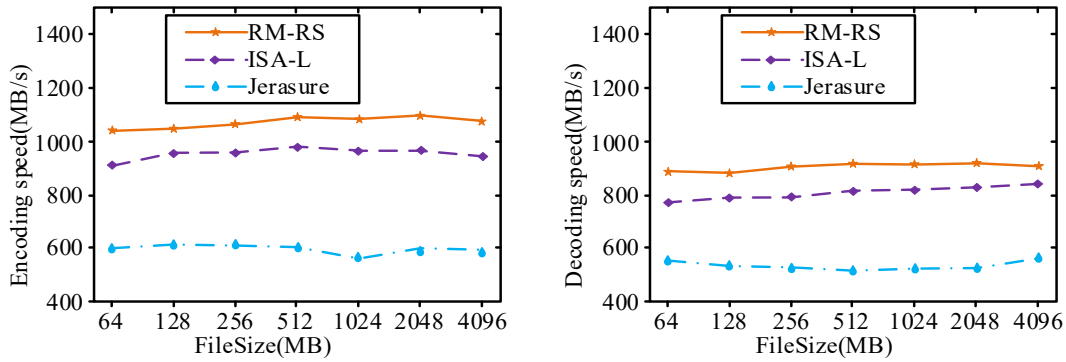
Fig. 7. Influence of different parameters on encoding performance



(a)Changes in decoding speed under different data block values

(b)The variation of decoding speed under different prime values

Fig. 8. Influence of different parameters on decoding performance



(a) Changes in encoding speed for different file sizes

(b) The decoding speed variation of different file sizes

Fig. 9. Influence of different file sizes on encoding and decoding performance

From Fig. 9(a), when the initial value of FileSize was 64MB, the FileSize value increased by a factor of 1 each time. As the FileSize value increased, the encoding speed curve of RM-RS code became relatively stable and was not greatly affected by file size. When the FileSize value was 2048MB, the encoding speed reached its maximum of 1125MB/s. As the FileSize value increased, the encoding speed of the Jerasure code decreased. When the file size reached 1024MB, the encoding speed dropped to a minimum of 587MB/s. From Fig. 9(b), as the FileSize value increased exponentially, the decoding speed of RM-RS code and ISA-L code gradually accelerated. However, the decoding speed of ISA-L code was lower than that of RM-RS code, regardless of the file size, indicating that the decoding performance of ISA-L code was not as good as that of RM-RS code. As the FileSize value increased, the decoding speed of the Jerasure code showed a continuous downward trend. From this, the encoding and decoding performance of the RM-RS code is not affected by FileSize. Therefore, applying the RM-RS code to the storage of large files can greatly improve the storage performance of the system.

4.2. Practical Effect Analysis of Economic Data Management based on RM-RS Code

To verify the practical effectiveness of the RM-RS code with economic data management, the study collected 100,000 economic data points from a large enterprise, which was supplemented by a self-built dataset of 20,000 points. The storage performance of the RM-RS code was compared with the classical EC code, and the comparison results are denoted in Fig. 10.

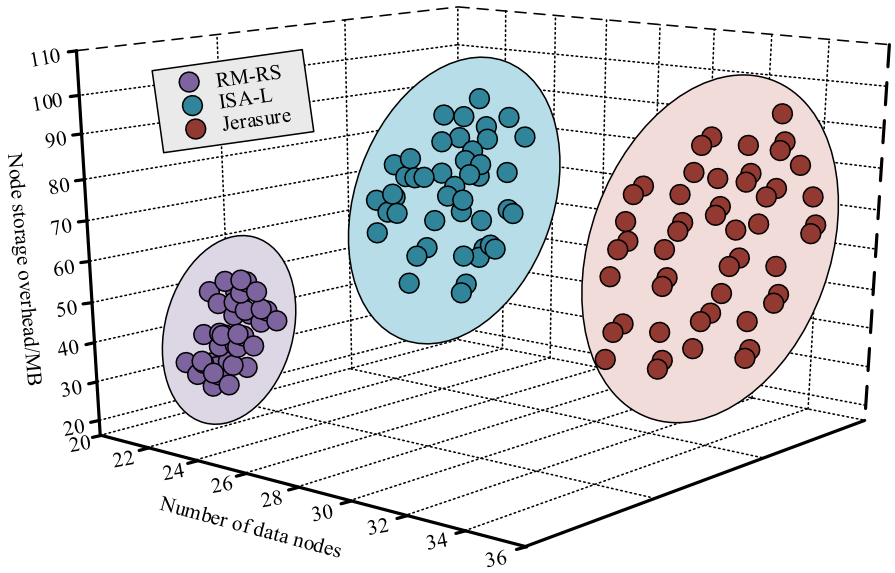


Fig. 10. Relationship between storage overhead and number of nodes for different ECs

From Fig. 10, in the self-built economic database, as the storage of economic data increased, the storage range of the RM-RS code continued to converge. With the increase in data storage, the occupancy of the storage system did not increase significantly, and the input data could be well stored and managed. As the storage of economic data continued to increase, ISA-L codes were more dispersed for data storage. The continuous increase in data storage led to a steady rise in the occupancy of the storage system, and the data storage could not be clustered together. Consequently, the internal occupancy of the storage system was relatively high. When the RM-RS code was used for economic data management, the average storage system internal occupancy size was 44MB, while the storage system internal occupancy sizes of erasure code and ISA-L code were 94MB and 76MB, respectively. Therefore, it can be concluded that the RM-RS code can utilize smaller amounts of memory for more efficient economic data management. The experiment of using a self-built economic data database to recover destroyed economic data from RM-RS codes is shown in Fig. 11.

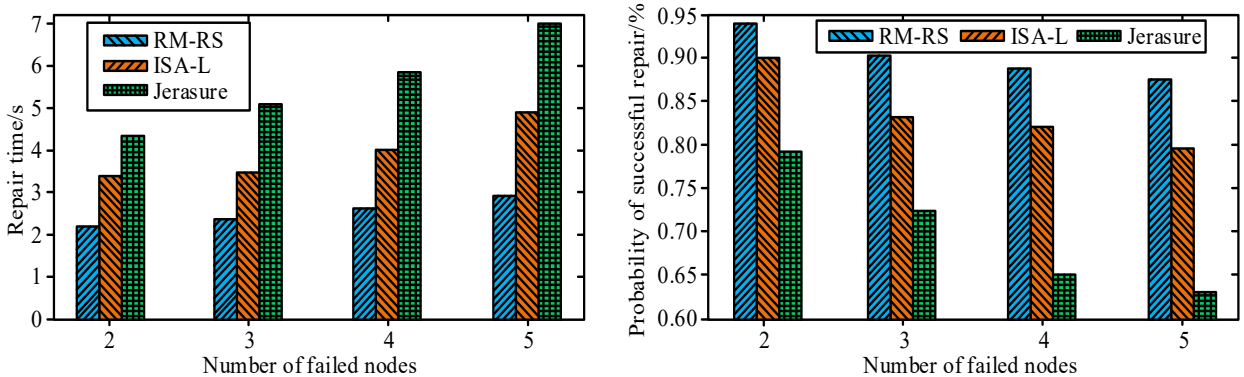


Fig. 11. Comparison of different ECs for repairing damaged nodes

From Fig. 11(a), when the number of failed nodes was 2, the BM-RS code repair time was 2.1 seconds. When the number of failed nodes increased to 5, the repair time required for the BM-RS code increased to 2.9 seconds. The Jerasure code required a repair time of 4.2 seconds when the number of failed nodes was 2, and when the number of failed nodes increased to 5, the repair time increased to 7 seconds. As the number of failed nodes increased, the repair time required for ISA-L code increased from 3.4 seconds to 4.9 seconds. From this, the RM-RS code has a shorter repair time for data failure nodes. From Fig. 11(b), the RM-RS code had a high success rate in repairing data failure nodes, with a repair success rate of 94% when the number of failure nodes was 2. But as the number of failed nodes increased, the success rate of repairs gradually decreased. When the number of failed nodes increased to 5, the repair success rate decreased to 89%, which was

not significant. The success rate of ISA-L code repair was highest at 90% when the number of failed nodes was 2. When the number of failed nodes was 5, the lowest repair success rate was 79%. The success rate of Jerasure code repair decreased the most, gradually decreasing from 78% to 63%. From this, it can be seen that using RM-RS codes for the recovery of destroyed data requires a shorter time and has a higher success rate for data destruction, which can effectively manage data recovery.

5. Discussion

The proposed RM-RS coding scheme demonstrated excellent performance across multiple experimental dimensions, with outstanding performance in encoding and decoding speed, file size adaptability, data block scalability, and storage resource utilization. First, in terms of coding structure design, the RM-RS algorithm introduced a cyclic shift strategy based on the RM transform, reducing the repeated calculation of polynomial coefficients and the frequency of XOR operations. Compared to traditional RS coding schemes, it optimized the operations of the generator matrix in a layered manner and introduced a redundant vector grouping mechanism, making the redundant block generation process more structured and improving parallelization efficiency (Alappat et al., 2022). Furthermore, the RM-RS scheme employed a pre-calculated inverse matrix strategy during the decoding phase, effectively reducing the latency overhead of high-order matrix inversion and making it suitable for high-density data block recovery tasks, which is also supported by Pham et al. (2025). Regarding performance trends, experimental results showed that the speed of RM-RS coding increased with increased data blocks or file size. This phenomenon can be attributed to two factors: First, the increased granularity of parallel computing makes multi-thread scheduling more efficient, fully leveraging the advantages of thread concurrency within the available CPU resources; second, large-scale data input improves the locality of system I/O access, reducing the frequency of cache swaps required for data in and out, thereby optimizing overall throughput (Wan et al., 2021).

In file size expansion experiments, the RM-RS scheme demonstrated significant stability and adaptability. Compared with traditional Jerasure and ISA-L encoding, RM-RS maintained a sustained increase in encoding and decoding rates with minimal fluctuation as file sizes increased from 64MB to 2048MB. This demonstrates the scheme's improved scalability for large file scenarios and its suitability for applications such as economic management that prioritize both high throughput and high stability. This trend is consistent with the throughput improvements demonstrated by the Esetstore system in large-scale data recovery (Liu et al., 2020). From a system resource utilization perspective, the RM-RS scheme, at comparable node scale and data density, consumes relatively low internal storage resources, averaging 44MB, significantly lower than the 94MB and 76MB of Jerasure and ISA-L, respectively. This demonstrates that RM-RS coding not only offers high performance but also possesses excellent resource scheduling capabilities and energy-saving potential, providing an engineering foundation for its deployment in large-scale distributed economic data storage and recovery.

Based on the above analysis, the RM-RS coding scheme, without relying on GPUs or specialized hardware instruction sets, achieves performance levels approaching or even exceeding those of some hardware acceleration libraries through structural optimization and system scheduling mechanisms, demonstrating excellent versatility and portability. Its synergistic advantages in performance, efficiency, and resource utilization lay the foundation for its future widespread application in heterogeneous distributed environments.

6. Conclusion

This study proposed an RM-RS coding scheme by introducing an RM transformation to improve traditional RS codes for distributed storage systems. The empirical results demonstrate that RM-RS achieves significantly higher encoding and decoding efficiency than classical erasure coding libraries such as Jerasure and ISA-L, while maintaining stable performance under varying file sizes and data block configurations. In addition, RM-RS reduces computational complexity and storage overhead, achieving efficient management of large-scale economic data with shorter repair times. These findings highlight the advantages of RM-RS in balancing performance, resource utilization, and reliability for distributed storage applications. Nevertheless, the present work does not extensively address the energy consumption costs of the storage system. Future research will investigate the energy efficiency of RM-RS coding and extend its applicability to heterogeneous and resource-constrained environments.

Funding

The research is supported by 2024 Ministry of Education Supply and Demand Matching Employment and Education Project: “Practice Exploration of school-enterprise Cooperation Focusing on Employment and Education” (project number: 2023122844667); Innovation and Entrepreneurship Training Program for College Students in Henan Province in 2024: Research on the practice path of the Integrated Development of Culture and Tourism in Henan Province under the Digital Background - Research from the perspective of Tourism and Performing Arts (Project No. : S202412949019).

Institutional Review Board Statement

Not applicable.

Declaration of Artificial Intelligence (AI) Tools

The author confirmed that no AI tools were used in the preparation of this manuscript.

Reference

Alappat, C., Hager, G., Schenk, O., and Wellein, G. (2022). Level-based blocking for sparse matrices: Sparse matrix-power-vector multiplication. *IEEE Transactions on Parallel and Distributed Systems,* 34(2), 581-597. doi: 10.1109/TPDS.2022.3223512

Alkhabet, M., and Ismail, M. (2023). Security algorithms for distributed storage system for E-health application over wireless body area network. *Journal of Ambient Intelligence and Humanized Computing,* 14(12), 15781-15792. doi: 10.1007/s12652-020-02733-1

Chen, J., Li, Z., Fang, G., Hou, Y., and Li, X. (2023b). A comprehensive repair scheme for distributed storage systems. *Computer Networks, 235*(5), 954-967. doi: 10.1016/j.comnet.2023.109954

Chen, J., Li, Z., Zhou, R., Su, L., and Wang, N. (2023a). Advanced elastic Reed-Solomon codes for erasure-coded key value stores. *IEEE Internet of Things Journal,* 11(3), 4747-4762. doi: 10.1109/JIOT.2023.3299574

Facenda, K., Krishnan, N., and Domanovitz, E. (2023). Adaptive relaying for streaming erasure codes in a three node relay network. *IEEE Transactions on Information Theory,* 69(7), 4345-4360. doi: 10.1109/TIT.2023.3254464

Gao, Z., Zhang, L., Cheng, Y., Guo, K., Ullah, A., and Reviriego, P. (2021). Design of FPGA-implemented Reed–Solomon erasure code (RS-EC) decoders with fault detection and location on user memory. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* 29(6), 1073-1082. doi: 10.1109/TVLSI.2021.3066804

Jiang, W., Jiang, H., Wu, J., and Chen, Q. (2023). Accelerating Distributed Cloud Storage Systems with In-Network Computing. *IEEE Network,* 37(4), 64-70. doi: 10.1109/MNET.002.2200631

Kim, Y. (2023). Power Control of Reed–Solomon-Coded OFDM Systems in Rayleigh Fading Channels. *Information,* 14(4), 247-261. doi: 10.3390/info14040247

Li, S., Cao, Q., Wan, S., Xia, W., and Xie, C. (2023). gPPM: a generalized matrix operation and parallel algorithm to accelerate the encoding/decoding process of erasure codes. *ACM Transactions on Architecture and Code Optimization,* 20(4), 1-25. doi:10.1145/3625005

Liu, C., Wang, Q., Chu, X., Leung, W., and Liu, H. (2020). Esetstore: An erasure-coded storage system with fast data recovery. *IEEE transactions on parallel and distributed systems*, 31(9), 2001-2016. doi: 10.1109/TPDS.2020.2983411

Liu, K., Peng, J., Wang, J., Huang, Z., and Pan, J. (2022). Adaptive and scalable caching with erasure codes in distributed cloud-edge storage systems. *IEEE Transactions on Cloud Computing,* 11(2), 1840-1853. doi:

Ma J, Yan W, Zhang X, Huang M., and Wang J. (2022). Pyrs: Cross-platform data fault-tolerant storage library based on rs erasure code. *Journal of Internet Technology,* 23(7), 1597-1611. doi: https://jit.ndhu.edu.tw/article/view/2826

Pham, V., Ghattas, O., Clemens, N. T., and Willcox, K. E. (2025). Real-time aerodynamic load estimation for hypersonics via strain-based inverse maps. *AIAA Journal,* 63(1), 91-101. doi: 10.2514/1.J064375

Sun, M., Tang, D., Li, Y., Wang, X., Cai, H., and Zeng, Q. (2024). An Erasure Code with Low Repair-Cost Based on A Combined-stripe Encoding Structure. *International Journal of Network Security,* 26(2), 206-216. doi: 10.6633/IJNS.202403 26(2).06

Wan, L., Huebl, A., Gu, J., Poeschel, F., Gainaru, A., Wang, R., Chen, J., Liang, X., Ganyushin, D., Munson, T. and Foster, I. (2021). Improving I/O performance for exascale applications through online data layout reorganization. *IEEE Transactions on Parallel and Distributed Systems,* 33(4), 878-890. doi: 10.1109/TPDS.2021.3100784

Wu, S., Shen, Z., Lee, C., Bai, Z., and Xu, Y. (2023). Elastic Reed-Solomon Codes for Efficient Redundancy Transitioning in Distributed Key-Value Stores. *IEEE/ACM Transactions on Networking,* 32(1), 670-685. doi: 10.1109/TNET.2023.3303865

Xu, L., Lyu, M., Li, Q., Xie, L., Li, C., and Xu, Y. (2021b). SelectiveEC: Towards balanced recovery load on erasure-coded storage systems. *IEEE Transactions on Parallel and Distributed Systems,* 33(10), 2386-2400. doi: 10.1109/TPDS.2021.312997

Xu, L., Lyu, M., Li, Z., Li, C., and Xu, Y. (2021a). A data layout and fast failure recovery scheme for distributed storage systems with mixed erasure codes. *IEEE Transactions on Computers,* 71(8), 1740-1754.

Xu, Y. (2024). Comparative Study of Erasure Code in Distributed Systems and Blockchain. Highlights in Science, *Engineering and Technology*, 81(2), 512-518.

Yao, W., Hao, M., Hou, Y., and Li, X. (2022). FASR: an efficient feature-aware deduplication method in distributed storage systems. *IEEE Access,* 10, 15311-15321. doi: 10.1109/ACCESS.2022.3147545

Yin, C., Xu, Z., Li, W., Li, T., Yuan, S., and Liu, Y. (2023). Erasure codes for cold data in distributed storage systems. *Applied Sciences,* 13(4), 2170-2183. doi: 10.3390/app13042170

Zhang, M., Kang, Q., and Lee, C. (2024). FlexRaft: Exploiting Flexible Erasure Coding for Minimum-Cost Consensus and Fast Recovery. *IEEE Transactions on Parallel and Distributed Systems,* 35(10),1826-1840. doi: 10.1109/TPDS.2024.3443424

Zhan,g X., Liang, N., Liu, Y., Zhang, C., and Li, Y. (2022). SA-RSR: A read-optimal data recovery strategy for XOR-coded distributed storage systems. *Frontiers of Information Technology & Electronic Engineering,* 23(6), 858-875. doi: 10.1631/FITEE.2100242

Professor Xin Liu is a dedicated lecturer at the School of Economics and Management, Zhengzhou Normal University. He earned his degree in Economics from the Business School at Zhengzhou University. Professor Liu teaches a range of courses, including digital economy, investment, management science (intelligent decision making), marketing (tourism performance marketing), and financial mathematics. His dedication to teaching has been recognized with several awards, such as the 2023 Henan Province Outstanding Bachelor's Thesis Excellent Advisor Award, the 2024, 2023, and 2020 Henan Province Teacher Ethics and Conduct Theme Education Activity Second Prize, and the 2025 Zhengzhou Normal University Curriculum Ideology and Politics Teaching Skills Competition First Prize. He was also honored as one of the university's "Student's Most Beloved Teachers." In research, Professor Liu has led seven provincial-level projects, including the 2026 Henan Province University humanities and social sciences research general project, and has

participated in five provincial/ministerial-level research initiatives. He also successfully guided students through the 2024 Henan Province College Student Innovation and Entrepreneurship Training Program. Professor Liu's ongoing work focuses on integrating ideological and political education into core curricula, as seen in his leadership of the 2025 Undergraduate Teaching Quality and Teaching Reform Project for the Microeconomics demonstration course.