A Lightweight Dynamic Storage Algorithm with Adaptive Encoding for Energy Internet

Song Deng, *Member, IEEE*, Yujia Zhai, Di Wu, *Member, IEEE*, Dong Yue, *Fellow, IEEE*, Xiong Fu, Yi He, *Member, IEEE*

This is the supplementary file for the paper entitled A Lightweight Dynamic Storage Algorithm with Adaptive Encoding for Energy Internet in IEEE Transactions on Services Computing. Additional sections, figures, and tables are put into this file and cited by the paper.

S1. The summary of the comparisons among different code families

Table 1 summarizes the comparisons among different code families.

S2. Preliminary

S2.1 EC Storage System

The concept of EC is often encountered in distributed storage: if the redundancy level is $k+m$, m parity blocks are calculated out of k source data blocks, and the $k+m$ data blocks are stored on $k+m$ disks, each disk contains stripes divided into exactly w strips. Therefore, any m disk failures can be tolerated. When a disk fails, you only need to randomly select m normal data blocks to calculate all source data. In the event of disk failures, all the source data can be calculated by randomly selecting m normal data blocks. If $k+m$ data blocks are spread across different storage nodes, then m node failures can be tolerated [1]. Such a typical system is shown in Fig. 1.

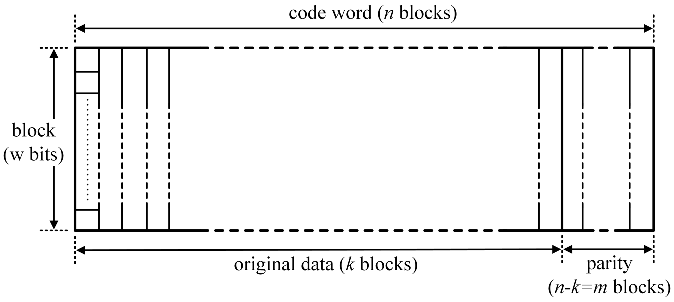


Fig. 1. A typical storage system encoding the stream data in block of w bits representation for code word of n blocks, original data of k blocks and parity of m blocks.

In computer systems, data blocks and check blocks are usually applied in the Galois field $GF\left(2^w\right)$, and the ECs that are skillfully operated by the Galois field are usually called $[n,k]$ ECs. Let's use the formula to reason.

Lemma 1. *If the redundancy level in Galois Field $GF\left(2^w\right)$ is $k+m$, m parity blocks are calculated from k source data blocks, and the $k+m$ data blocks are stored on $k+m$ hard disks respectively, it can tolerate any m hard disk failures.*

Proof. Let k linearly independent vectors of length m be represented as $P=(p_0,p_1,\cdots,p_{k-1})$, whose all elements are in the Galois field $GF\left(2^w\right)$. Denote the source data as $Q=(q_0,q_1,\cdots,q_{k-1})$, whose all components are also represented as Galois field elements in $GF\left(2^w\right)$. The code word of data Q is then $T=p_0q_0+p_1q_1+\cdots+p_{k-1}q_{k-1}$. This encoding process can also be expressed by using the $generator\ matrix$ P of dimension $k\times n$ as $T=P\cdot Q$, where the generator matrix P satisfies the following equation.

$$P=\begin{bmatrix} p_0 \\ p_1 \\ \vdots \\ p_{k-1} \end{bmatrix}^T=\begin{bmatrix} p_{0,0} & p_{0,1} & \cdots & p_{0,n-1} \\ p_{1,0} & p_{1,1} & \cdots & p_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ p_{k-1,0} & p_{k-1,1} & \cdots & p_{k-1,n-1} \end{bmatrix}^T. \quad (1)$$

The proof immediately follows. □

In most data storage application examples, ECs have the maximum distance separable (MDS) property, which means that any m encoded blocks in the vector P can be tolerated to recover the data. The MDS property is guaranteed to remain unchanged as long as the square matrix created by removing any m rows from P is guaranteed to be invertible.

S2.2 Reed-Solomon Code

The RS code [2] relies on the Vandermonde matrix to ensure the invertibility of the matrix, and the $generator\ matrix\ P$ should satisfy

$$P=\begin{bmatrix} 1 & \cdots & 1 & \cdots & 1 \\ x_0 & \cdots & x_i & \cdots & x_{n-1} \\ x_0^2 & \cdots & x_i^2 & \cdots & x_{n-1}^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_0^{k-1} & \cdots & x_i^{k-1} & \cdots & x_{n-1}^{k-1} \end{bmatrix}^T, \quad (2)$$

where x_i are all different elements in $GF\left(2^w\right)$.

Lemma 2. *RS code uses Vandermonde matrix in $GF\left(2^w\right)$ to satisfy the invertible condition, only need to generate matrix P to satisfy Eq. (2).*

TABLE 1
COMPARISONS AMONG DIFFERENT CODE FAMILIES

Different algorithms	Storage cost	Constraints in choosing parameters	Recovery latency	A RAID-6 scenario
Replication	High	No	Lowest	No
RS code	Lowest	w=8,16,32	High	No
CRS code	Low	3≤w≤32	Not high	No
MDR code	Not high	Blaum-Roth code : w+1 is prime Liberation code : w is prime The Liber8tion code : w=8	Not low	Yes

Proof. For a generator matrix of the Vandermonde form would yield a non-systematic form of the data, then the data Q is not an explicit part of the code word T. An equivalent generator matrix P' can be obtained by performing elementary row transformation on P, and the generator matrix P' satisfies the following equation.

$$P'=\left[\,I,V\,\right]^T=\begin{bmatrix} 1 & 0 & \cdots & 0 & v_{0,0} & \cdots & v_{0,m-1} \\ 0 & 1 & \cdots & 0 & v_{,0} & \cdots & v_{1,m-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & v_{k-1,0} & \cdots & v_{k-1,m-1} \end{bmatrix}^T, \quad (3)$$

where the left side is the identity matrix I_k of dimension k, and the right side is the parity coding matrix V. Therefore, it can be obtained

$$T = P' \cdot Q = (q_0, q_1, \cdots, q_{k-1}, v_0, v_1, \cdots, v_{m-1})^T, \quad (4)$$

where $(q_0, q_1, \cdots, q_{k-1})^T \cdot V = (v_0, v_1, \cdots, v_{m-1})$.

The proof is completed. □

Lemma 3. *When any one of the data blocks q_i is damaged, data recovery needs to be performed through the decoding process. The data decoding process first selects the remaining valid code words to form a decoding column vector.*

Proof. Assuming $k = 4$, $m = 3$, the generator matrix P satisfies Eq. (2) in Lemma 2, where the blocks $\{q_1, q_3, v_1\}$ are damaged, the remaining data $\{q_0, q_2, v_0, v_2\}$ can be selected as decoding column vectors. Then the decoded data Q^* can achieve data recovery Q by $Decode\,(Q^*) = (P^*)^{-1} \cdot Q^*$, and $Q^* = Q$. The decoding process can be formulated as follows.

$$\text{Decode}\,(Q^*)=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ x_0^2 & x_1^2 & x_2^2 & x_3^2 \end{bmatrix}^{-1} \cdot \begin{bmatrix} q_0 \\ q_2 \\ v_0 \\ v_2 \end{bmatrix} = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix}. \quad (5)$$

where, P^*, Q^* respectively represent the generation matrix and decoding column vector of the remaining block reorganization after destruction. The proof is completed. □

S2.3 Cauchy Reed-Solomon Code

In addition to the transformation from the Vandermonde generation matrix, we directly define the matrix V, which also satisfies the invertible condition. Then directly define the matrix V as a Cauchy matrix, and the corresponding EC is usually called a CRS code [3].

Lemma 4. *RS code using Cauchy matrix in $GF\,(2^w)$ to satisfy the reversible condition.*

Proof. In the CRS code, define $X = (x_1, x_2, \cdots, x_k)$ and $Y = (y_1, y_2, \cdots, y_m)$, where x_i's and y_i's are different elements of $GF\,(2^w)$. Then the element of row i column j in the Cauchy matrix is $1/\,(x_i + y_j)$, where $x_i \neq y_j, i = 1, 2, \cdots, k, j = 1, 2, \cdots, m$ and $x_i \neq x_j, y_i \neq y_j (i \neq j)$. The parity coding matrix V can be represented by a Cauchy matrix of dimension $k \times m$ as $V = (v_{ij})_{(k\times m)} = \left(\frac{1}{x_i+y_j}\right)_{(k\times m)}$, where the parity coding matrix V satisfies the following equation.

$$V = \begin{bmatrix} \frac{1}{x_1+y_1} & \frac{1}{x_1+y_2} & \cdots & \frac{1}{x_1+y_m} \\ \frac{1}{x_2+y_1} & \frac{1}{x_2+y_2} & \cdots & \frac{1}{x_2+y_m} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{x_k+y_1} & \frac{1}{x_k+y_2} & \cdots & \frac{1}{x_k+y_m} \end{bmatrix}. \quad (6)$$

The proof is completed. □

Asymptoticity: We can easily verify that each submatrix of the Cauchy matrix is a non-singular matrix, of which its inverse exists. Since the complexity of the Vandermonde matrix inversion operation is $\Theta\,(n^3)$, and the complexity of the Cauchy matrix inversion operation is only $\Theta\,(n^2)$. Thus, the Cauchy matrix can be applied to substitute the Vandermonde matrix to reduce the computational complexity of decoding. The Galois field binary matrix is used to improve the operation efficiency, and the multiplication is directly converted into an Exclusive OR (XOR) logical operation, which greatly reduces the operational complexity.

S2.4 Minimal Density RAID-6 Code

The minimum density bit matrix is a coding distribution matrix (CDM) [4], which reaches a lower bound of $2kw+k-1$ non-zero entries [5]. However, it defines the MDS RAID-6 code, which can be done when one of X_i matrices has exactly w ones and the remaining $k-1$ matrices have exactly $w+1$ ones. These matrices defining codes lend us an excellent combination of properties, stated as follows:

- Their coding performance is comparable to the performance of $\frac{k-1}{2w} + k - 1$ XOR operations per encoded element, and the best coding performance is equivalent to the performance of $k-1$ XOR operations per encoded element, so the absence of $\frac{1}{2w} + 1$ is a significant factor in a performance penalty. The loss of the only advantage is its independence in k, which is suitable for dynamically adding or removing data disks, and its modification performance is optimal.
- Their decoding performance is locally optimal, by applying the intermediate results during decoding with an enhanced standard matrix inversion technique [6].

- Their properties are well suited to algorithms for the reconstruction of uncorrelated sector faults.

If we encode RAID-6 using $generator\ matrix\ P$, as matrix will be affected by a lot of constraints.

Lemma 5. *The generator matrix P_R of the MDR code can be directly checked for parity.*

Proof. For $k > w$, the first kw rows of P_R form an identity matrix, and an identity matrix must be included in the next w rows, then the composition of the last w rows is the only flexibility in the RAID-6 specification. When $k \leq w$, there must be at least $kw + k - 1$ 1s in these remaining w rows to ensure the reversibility of P_R. The lower bound for the MDS matrix to achieve minimum density codes arrives. □

There are three different constructions of Minimal Density codes for different values of w:

- When $w + 1$ is prime, **Blaum-Roth** codes [5].
- When w is prime, **Liberation** codes [6].
- When $w = 8$, the **Liber8tion** code [7].

These encoded codes all share the same characteristics of the performance. In the operation of each code word, $(k-1)/2w+(k-1)$ XOR operations are performed. Therefore, the computational performance of these codes becomes stronger with the increase of w and reaches asymptotic optimality when $w \to \infty$. At the same time, the decoding performance of these codes is slightly worse, and near-optimal performance [6] is achieved with the help of a technique called *Code-Specific Hybrid Reconstruction* [8].

S3.EXPERIMENTAL ENVIRONMENT

The experimental environment is shown in Table 2.

TABLE 2
CONFIGURATION

Name	CPU	RAM	OS	HDFS
Server_1	Intel Core i7-10700	16G	Ubuntu 20.04	Hadoop 3.0.0-alpha2
Server_2	Intel Core i7-10700	16G	Ubuntu 20.04	Hadoop 3.0.0-alpha2
Server_3	Intel Core i7-10700	16G	Ubuntu 20.04	Hadoop 3.0.0-alpha2
Server_4	Intel Core i7-10700	16G	Ubuntu 20.04	Hadoop 3.0.0-alpha2

S4.Representativeness of the Synthetic Workload

In order to achieve high performance and high capacity efficiency of the workload, we have taken the following two measures.

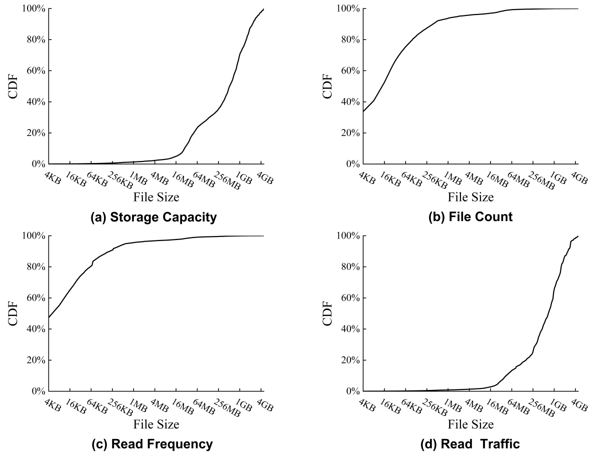
First, in order to solve the inevitable performance overhead caused by HDFS remote access to metadata. In Section 3.2 of the regular paper, we propose that LFSRS enables local FS to store user data and metadata, so as to reduce network communication in data access processing, and realize data locality while utilizing Replication strategy and EC capabilities. In Section 3.4 of the regular paper, the distributed storage strategy of LDSA-AE effectively reduces the amount of redundant data, and the configuration information storage of other servers is in the metadata of local FS. Therefore, remote access to metadata is not required during data access processing, enabling lightweight dynamic redundancy control for workloads under EI scenarios.

However, this is not enough, as small stochastic accesses with access skew dominate in these workloads. Therefore, in Section 3.3 of the regular paper, we introduced DCUACA to use finer-grained data chunk-based access frequency monitoring to accurately identify hot and cold data, thereby addressing differences in target workloads. If the access frequency is low enough, more data chunks can be coded with less access frequency, enabling higher capacity efficiency.

In the end, we believe that a good option is to use data features and task runtime to identify categories of common jobs in the workload. Then, by synthesizing the representation of the workload [9], the distribution of storage capacity, file number, access frequency, and access traffic is reproduced from the original trace, which can form a fast performance evaluation, that is, a fast design cycle.

S5.Production Tracing Survey in Synthetic Workload Based on 1-Hour Samples

The production tracing survey in a synthetic workload based on 1-hour samples is shown in Fig. 2.



(a) Storage Capacity

(b) File Count

(c) Read Frequency

(d) Read Traffic

Fig. 2. Production Tracing Survey in Synthetic Workload Based on 1-Hour Samples.

S6.Description of Workloads

The detailed description of workloads is shown in Table 3. W_1 represents the workload of active data, and W_2 represents the workload of inactive data.

TABLE 3
DESCRIPTION OF WORKLOADS

Workload	Avg Object Size	# Object	Avg Request Size	Total Capacity
W_1	158.31MB	13000	216.88MB	2.87TB
W_2	30.67MB	210000	25.79MB	8.97GB

S7.The performance evaluation of the DCUACA algorithm

In order to evaluate the performance of our proposed DCUACA algorithm, for the experimental data set, we divide the data set [training set, test set] into [0.7, 0.3] in proportion to conduct experiments. We select four evaluation indicators of accuracy, precision, recall, and F1 score to prove the performance of the above evaluation method.

Definition 1. *Assuming that T (True) stands for correct, F (False) stands for error, P (Positive) stands for 1, and N (Negative) stands for 0. First look at the prediction result as P or N, and then compare the prediction result according to the actual result, and give the judgment result as T or F. The confusion matrix can be constructed according to the output of the model. Table 4 shows the confusion matrix.*

TABLE 4
Confusion Matrix

Predicted Label **True Label**	**0**	**1**
0	True Negative (TN)	False Positive (FP)
1	False Negative (FN)	True Positive (TP)

Wherein TP, FP, FN, TN can be understood as:

- *TP: The prediction is active, the actual is active, and the prediction is correct.*
- *FP: The prediction is active, but it is actually inactive. The prediction is wrong.*
- *FN: The prediction is inactive, but it is actually active. The prediction is wrong.*
- *TN: The prediction is inactive, but it is actually inactive. The prediction is correct.*

Accuracy is the percentage of the total sample that predicts the correct result. According to definition 1, we can get an expression of accuracy.

$$\text{Accuracy } = \frac{TP+TN}{TP+TN+FP+FN}. \tag{7}$$

Precision is for prediction results, which means the probability that all samples predicted to be positive are actually positive samples. According to definition 1, we can get an expression of precision.

$$\text{Precision } = \frac{TP}{TP+FP} \tag{8}$$

Recall is for the original sample, which means the probability of being predicted to be a positive sample in the actual positive sample. According to definition 1, we can get an expression of the recall.

$$\text{Recall } = \frac{TP}{TP+FN} \tag{9}$$

Precision and Recall are a pair of contradictory measures, so in order to be able to comprehensively consider these two indicators, we can define a new indicator according to the break-even point between them: F1 score, which is the harmonic mean number of Precision and Recall, its expression is as follows.

$$\text{F1 score } = \frac{2 * \text{ Precision } * \text{ Recal}}{\text{Precision } + \text{ Recall}} \tag{10}$$

The evaluation performance is listed in Table 5.

TABLE 5
Evaluate Performance

Accuracy	Precision	Precision	F1 score
0.9199	0.9213	0.8962	0.9086

References

[1] J. S. Plank, “T1: erasure codes for storage applications,” in *Proc. of the 4th USENIX Conference on File and Storage Technologies*, 2005, pp. 1–74.

[2] I. S. Reed and G. Solomon, “Polynomial codes over certain finite fields,” *Journal of the society for industrial and applied mathematics*, vol. 8, no. 2, pp. 300–304, 1960.

[3] J. Bloemer, M. Kalfane, R. Karp, M. Karpinski, M. Luby, and D. Zuckerman, “An xor-based erasure-resilient coding scheme,” 1995.

[4] J. S. Plank, A. L. Buchsbaum, and B. T. Vander Zanden, “Minimum density raid-6 codes,” *ACM Transactions on Storage (TOS)*, vol. 6, no. 4, pp. 1–22, 2011.

[5] M. Blaum and R. M. Roth, “On lowest density mds codes,” *IEEE Transactions on Information Theory*, vol. 45, no. 1, pp. 46–59, 1999.

[6] J. S. Plank, “The raid-6 liber8tion code,” *The International Journal of High Performance Computing Applications*, vol. 23, no. 3, pp. 242–251, 2009.

[7] J. S. Plank, “A new minimum density raid-6 code with a word size of eight,” in *2008 Seventh IEEE International Symposium on Network Computing and Applications*. IEEE, 2008, pp. 85–92.

[8] J. L. Hafner, V. Deenadhayalan, K. Rao, and J. A. Tomlin, “Matrix methods for lost data reconstruction in erasure codes.” in *FAST*, vol. 5, no. December, 2005, pp. 15–30.

[9] Y. Chen, A. Ganapathi, R. Griffith, and R. Katz, “The case for evaluating mapreduce performance using workload suites,” in *2011 IEEE 19th annual international symposium on modelling, analysis, and simulation of computer and telecommunication systems*. IEEE, 2011, pp. 390–399.