IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received March 9, 2017, accepted April 18, 2017, date of publication April 27, 2017, date of current version June 7, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2699170

Bandwidth Overhead-Free Data Reconstruction Scheme for Distributed Storage Code With Low Decoding Complexity

MINGJUN DAI, XIA WANG, HUI WANG, XIAOHUI LIN, AND BIN CHEN
Shenzhen Key Lab of Advanced Communication and Information Processing, Shenzhen Key Lab of Media Security, College of Information Engineering, Shenzhen University, Guangdong 518060, China
State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China

Corresponding author: Hui Wang (wanghsz@szu.edu.cn)

This work was supported in part by the Natural Science Foundation of China under Grant 61301182, 61171071, 61575126, in part by the Specialized Research Fund for the Doctoral Program of Higher Education from the Ministry of Education under Grant 20134408120004, in part by the Key Project of Department of Education of Guangdong Province under Grant 2015KTSCX121, from Guangdong Natural Science Foundation under the Project number 2015A030313552, in part by the Foundation of Shenzhen City under Grant KQCX20140509172609163, JCYJ20160422112909302 and in part by the Natural Science Foundation of Shenzhen University under Grant 00002501.

ABSTRACT The (n, k) combination property (CP) is defined as follows: k source packets are mapped into $n \geq k$ packets and any k out of these n packets are able to recover the information of the original k packets. This (n, k) CP is extensively needed by cloud storage service providers. Reed–Solomon (RS) codes possess CP at the cost of high encoding and decoding complexity for two reasons: operation over a large-size finite field and time-consuming matrix inversion operation. By operating within the binary field and by allowing only zigzag decoding at the decoder, binary zigzag decoding that possesses CP lowers the decoding complexity significantly. The drawback is that storage room overhead is needed. Corresponding to this storage room overhead, in the data reconstruction process, intuitively fetching k whole stored packets will consume overhead bandwidth. In this paper, a data reconstruction scheme that is optimal in terms of bandwidth consumption is designed, where optimal means the bandwidth consumption is equal to the volume of data to be reconstructed, namely, no overhead bandwidth is needed. To do that, a universal method of fetching sub-packet is proposed, and its corresponding decoding method is also designed.

INDEX TERMS Distributed storage, network code, zigzag decoding, data reconstruction bandwidth.

I. INTRODUCTION

In distributed storage (DS) systems, network coding (NC) [1] has been adopted to improve the reliability of data storage [2]–[4]. Normally, the combination property (CP) [5] is required, whereby k source packets are mapped into $n \geq k$ packets and any k out of these n packets are able to recover the information of the original k packets. Reed-Solomon (RS) codes [6] possess CP and hence have been widely adopted in the design of DS systems [7]. However, the encoding and decoding are operated within a large size finite field and the decoding is non-zigzag decodable, which has high decoding complexity [8].

To lower decoding complexity, especially for mobile applications [9], [10], the idea of binary zigzag decoding (BZD) is proposed. In BZD, the operation is within the binary field, and the decoding is zigzag decoding (ZD) [11] whose decoding steps resembles a zigzag. Researchers have proposed CP-BZD storage codes that possess CP and BZD simultaneously, where some additional storage room is introduced. A series of studies have tried to reduce the storage room overhead [12]–[14].

In this work, instead of focusing on reducing storage room, we now consider decreasing the consumption of communication bandwidth for data reconstruction based on the CP-BZD storage code [14]. An intuitive method is to fetch arbitrary k whole packets. Due to storage room overhead, the consumed bandwidth is larger than the volume of information to be reconstructed. To reduce bandwidth consumption, we propose a sub-packets fetching algorithm whose consumption of bandwidth is equal to the volume of data information to be reconstructed. Therefore, this method is optimal in terms of bandwidth consumption. The corresponding decoding algorithm is also designed.

The paper is organized as follows: In Sec. II, CP-BZD designed in [14] is briefly reviewed. In Sec. III, attention is transferred from storage room to communication bandwidth.

In Sec. IV and Sec. V, we propose a data reconstruction scheme that is optimal in terms of bandwidth consumption for cases when $k \leq 4$ and $k > 4$, respectively. Finally, in Sec. VI, we draw conclusions.

II. PRELIMINARIES ON CP-BZD CODES

A. ENCODING

Let s_1 through s_k denote the original k source packets, each with length L. The j-th bit of the i-th source packet is denoted by $s_{i,j} \in \{0, 1\}$. Consider (n, k) systematic code, with the n encoded packets denoted by $c_1, c_2, \ldots, c_n$, respectively. The first k encoded packets are actually the source packets, namely $c_i = s_i$ for all $i \in \mathcal{K} \triangleq \{1, 2, \ldots, k\}$. These k packets are called *systematic packets*. The last $m \triangleq n-k > 0$ packets are encoded by linearly combining the source packets. For $i \in \mathcal{M} \triangleq \{1, 2, \ldots, m\}$, the packets c_{k+i}'s are called *parity packets*. They are generated by shifting the source packets by different number of bits and then adding them over the binary field in a bit-wise manner. Let $\boldsymbol{T}$ be the $m \times k$ matrix that represents the numbers of bits shifted by the source packets in order to form the parity packets. Its (i, j)-th element, denoted by T_{ij}, represents the number of bits shifted by source packet s_j when forming the parity packet c_{k+i}, $i \in \mathcal{M}, j \in \mathcal{K}$.

As assumed in [14], we consider only the cases where $n \leq 2k$, or equivalently, $m \leq k$. Consider a $(n = 2k, k)$ systematic code that is CP-BZD. It is clear that removing some of the parity packets results in a $(k + m, k)$ systematic code, where $m < k$, which is also CP-BZD. Therefore, it suffices to consider the single case when $m = k$ or equivalently $n = 2k$.

The $\boldsymbol{T}$ designed in [14] is divided into two cases, including $k \leq 4$ and $k > 4$. Detailed $\boldsymbol{T}$ are listed as follows:

When $k \leq 4$,

$$\boldsymbol{T} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad k = 2, \tag{1}$$

$$\boldsymbol{T} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}, \quad k = 3, \tag{2}$$

$$\boldsymbol{T} = \begin{bmatrix} 0 & 1 & 3 & 2 \\ 2 & 0 & 1 & 3 \\ 3 & 2 & 0 & 1 \\ 1 & 3 & 2 & 0 \end{bmatrix}, \quad k = 4. \tag{3}$$

When $k > 4$,

$$\boldsymbol{T} = \begin{bmatrix} 0 & 1 & 3 & 6 & 10 & \ldots & \frac{k(k-1)}{2} \\ \frac{k(k-1)}{2} & 0 & 1 & 3 & 6 & \ldots & \frac{(k-1)(k-2)}{2} \\ \frac{(k-1)(k-2)}{2} & \frac{k(k-1)}{2} & 0 & 1 & 3 & \ldots & \frac{(k-2)(k-3)}{2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 3 & 6 & 10 & 15 & \ldots & 0 \end{bmatrix}. \tag{4}$$

A graphical illustration of the packets is shown in Fig. 1, where each column in a parity packet denotes that the corresponding bits are summed up. For example, the first four bits of c_5 are $s_{1,1}$, $s_{1,2} + s_{2,1}$, $s_{1,3} + s_{2,2} + s_{4,1}$, and $s_{1,4} + s_{2,3} + s_{3,1} + s_{4,2}$, respectively.

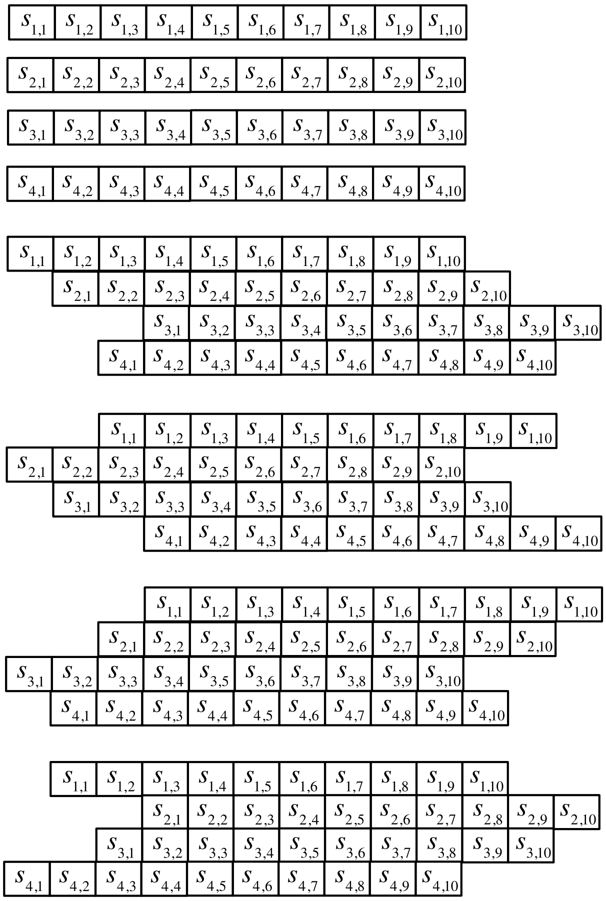

$s_{1,1}$ $s_{1,2}$ $s_{1,3}$ $s_{1,4}$ $s_{1,5}$ $s_{1,6}$ $s_{1,7}$ $s_{1,8}$ $s_{1,9}$ $s_{1,10}$
$s_{2,1}$ $s_{2,2}$ $s_{2,3}$ $s_{2,4}$ $s_{2,5}$ $s_{2,6}$ $s_{2,7}$ $s_{2,8}$ $s_{2,9}$ $s_{2,10}$
$s_{3,1}$ $s_{3,2}$ $s_{3,3}$ $s_{3,4}$ $s_{3,5}$ $s_{3,6}$ $s_{3,7}$ $s_{3,8}$ $s_{3,9}$ $s_{3,10}$
$s_{4,1}$ $s_{4,2}$ $s_{4,3}$ $s_{4,4}$ $s_{4,5}$ $s_{4,6}$ $s_{4,7}$ $s_{4,8}$ $s_{4,9}$ $s_{4,10}$

FIGURE 1. Illustration of (8, 4) CP-BZD code.

We repeat the following theorem from [14] for completeness:

Theorem 1: Given any k out of the n encoded packets, the k original packets can be recovered by the zigzag decoding algorithm.

B. PROPERTIES

We first give one basic definition, then re-state the distinct relative difference (DRD) property from [14], and finally investigate the storage room overhead (SRO) and set up the bit indexing framework within each packet.

For $p, i, j \in \mathcal{K}$, $i \neq j$, define relative difference

$$\Delta^{p}_{i,j} \triangleq T_{pj} - T_{pi}. \tag{5}$$

1) DRD [14]

Given any $i, j, m, n \in \mathcal{K}$, where $i \neq j$ and $m \neq n$, we have

(a) $\Delta^{i}_{m,n} \neq 0$;

(b) $\Delta^{i}_{m,n} \neq \Delta^{j}_{m,n}$.

2) SRO

Systematic packets have no storage room overhead, namely, SRO $= 0$. Parity packets have the following storage room overhead:

$$\begin{cases} \text{if } k = 2, 3, & \text{then SRO} = 1, \\ \text{if } k = 4, & \text{then SRO} = 3, \\ \text{if } k > 4, & \text{then SRO} = \dfrac{k(k-1)}{2}. \end{cases}$$

3) BIT INDEXING

Within each parity packet, by starting from the left towards the right, we set the bit index increase from 0. For example, for cases $k \geq 4$, the bit indices within those parity packets are from 0 to $L - 1 + \frac{k(k-1)}{2}$.

C. PROPOSED EQUALITY-DECODING (ED) ALGORITHM

Dedicated for CP-BZD code, beside the zigzag decoding method as described in [12] and [14], in this work we propose the Equality-Decoding (ED) algorithm, which is later used as a basic building block. We first describe the operations needed before the decoding. Afterwards, we describe the proposed decoding method.

1) PRE-PROCESSING

Suppose among the k coded packets provided for data reconstruction, $J \leq k$ of them are parity packets. Since the $k - J$ systematic packets do not require any decoding, the corresponding source packets can be directly recovered. Furthermore, they can be subtracted from the J parity packets. We call the resultant parity packets as degenerated parity packets. Let c_i^- denote the degenerated packet of c_i. Let the source packets remaining unknown be indexed by $\mathcal{J} \subset \mathcal{K}$. Subtract each index of the J parity packets by k and put all of them into the set $\mathcal{I}$. Note that $|\mathcal{I}| = |\mathcal{J}| = J$. The decoding task is then reduced to the case of decoding J source packets from J degenerated parity packets. Let $\boldsymbol{T}^-$ denote the $J \times J$ submatrix obtained from $\boldsymbol{T}$ by retaining the rows indexed by $\mathcal{I}$ and the columns indexed by $\mathcal{J}$.

2) DECODING

The decoding can initiate either from the left or from the right. Without loss of generality, we consider initiating from the left. Recall that each row of $\boldsymbol{T}^-$ represents the numbers of bits shifted by the corresponding components, respectively, to compose the corresponding degenerated parity packet. Therefore, in an intermediate decoding stage, in a degenerated parity packet, the number of bits that can be recovered is equal to the difference between the second smallest element and the smallest element within the row of $\boldsymbol{T}^-$ that is associated withl this parity packet. We hence propose the *ED Algorithm* whose pseudo-code is illustrated in Algorithm 1.

Each element of $\boldsymbol{T}^-$ will be updated by adding a certain number, and we define this updated element as *accumulated element*.

During an intermediate stage, we need to find one row in $\boldsymbol{T}^-$ (suppose the i-th row) that contains a single smallest accumulated element. Let s_j denote the source packet that is associated with this element. The number of bits that can be recovered within s_j is the difference between the second smallest accumulated element and the smallest accumulated element in the i-th row of $\boldsymbol{T}^-$. Let δ denote such a difference. Note that the recently recovered δ bits in s_j can now be viewed as known in other degenerated parity packets. Therefore, the column in $\boldsymbol{T}^-$ that is associated with s_j is increased by δ, and an updated $\boldsymbol{T}^-$ is obtained.

Algorithm 1 ED Algorithm

```
Require: $k$ packets, $J \times J$ matrix $\boldsymbol{T}^-$;
do
  for $i = 1$ to $J$ //row loop
    Let $r_i$ be the $i$-th row of $\boldsymbol{T}^-$;
    Find the two smallest accumulated elements of $r_i$;
    Let $T_{i,j}$ and $T_{i,j'}$ be the smallest and the second
    smallest accumulated elements, respectively;
    Let $\delta = T_{i,j'} - T_{i,j}$;
    if $\delta \neq 0$
    then
      for $i' = 1$ to $J$ do
        $T_{i',j} = T_{i',j} + \delta$;
        recover $\delta$ bits in $s_j$;
      end for
    else
      continue;
    end if
  end for
while there exist bits to be recovered
```

We can repeat the above process for the newly updated $\boldsymbol{T}^-$, until either lall information bits are successfully recovered or the process cannot continue.

III. ATTENTION FROM STORAGE ROOM TO COMMUNICATION BANDWIDTH

We now transfer focus from storage room to communication bandwidth. We investigate the bandwidth consumption for data reconstruction. An intuitive method is to collect arbitrary k whole packets, which may be bandwidth-inefficient due to the nature of storage room overhead of the CP-BZD storage codes.

We claim that it is possible to collect arbitrary k packets or sub-packets with each having a length of L. In other words, the amount of bits fetched or delivered is equal to the amount of information to be reconstructed, which indicates optimal bandwidth consumption. As a systematic packet has a length of L, we only need to consider how to fetch length-L subpacket from each parity packet. Given arbitrary number of degenerated parity packets and the corresponding $\boldsymbol{T}^-$, we will propose a method that fetches L consecutive bits from each of these degenerated parity packets. Due to a slight difference in storage code construction as described in subsec. II-A, the corresponding data reconstruction implementation is also divided into two cases: $k \leq 4$ and $k > 4$, which are dealt with in the following two sections, respectively.

IV. OPTIMAL BANDWIDTH RECONSTRUCTION SCHEME FOR CASES WHEN $k \leq 4$

As cases $k = 2, 3$ are straightforward, we describe the proposed scheme for case $k = 4$ only. Although this case has already been designed in [15], we repeat the proposed scheme for the sake of being thorough.

Since those fetched bits within a degenerated parity packet are consecutive, we only need to consider how to fetch the leftmost bit (LMB). To find the LMB, we propose the *Minimum Sum Rule (MSR)* as follows: "Within matrix $\boldsymbol{T}^-$, define one combination of J elements that collectively form a minus one slope (MOS) as follows: In the top row, arbitrarily choose one element, say the one with the column index $a \in \{1, 2, \ldots, J\}$. The other $J-1$ elements in the other $J-1$ rows are formed in the following manner: In row $i \in \{2, 3, \ldots, J\}$, the element in the $(a + (i-1)) \mod J$-th position is chosen. Since in total there are J options for choosing the first element, there are also J options for choosing the MOS combination elements. Among these J options, we exhaustively find the MOS combination with minimum sum. We name the found MOS combination as MSR-MOS combination." Afterwards, within each degenerated parity packet that is associated with a row in $\boldsymbol{T}^-$, the bit index of the LMB we fetched is exactly the intersection of the row and the MSR-MOS combination. We call this sub-packets fetching rule as MSR-MOS-F and denote it by putting squares around those MSR-MOS combination elements in $\boldsymbol{T}^-$.

Example: Set $L = 10$. Without the loss of generality, we assume packets c_6, c_7, c_8, and c_3 are utilized for data reconstruction. Detailed sub-packets fetching is illustrated in Fig. 2. In (a), how to obtain $\boldsymbol{T}^-$ from $\boldsymbol{T}$ is shown. In (b), the found MSR-MOS combination in $\boldsymbol{T}^-$ is shown in squares. In (c), those sub-packets (within the two dashed vertical lines) are fetched by the MSR-MOS-F method. More specifically, in packets c_6 through c_8, we choose bits from 0 through 9, from 1 through 10, and from 1 through 10, respectively. Besides, those numbers in (c) show one decoding order of information bits based on degenerated packets c_6^-, c_7^-, c_8^-.

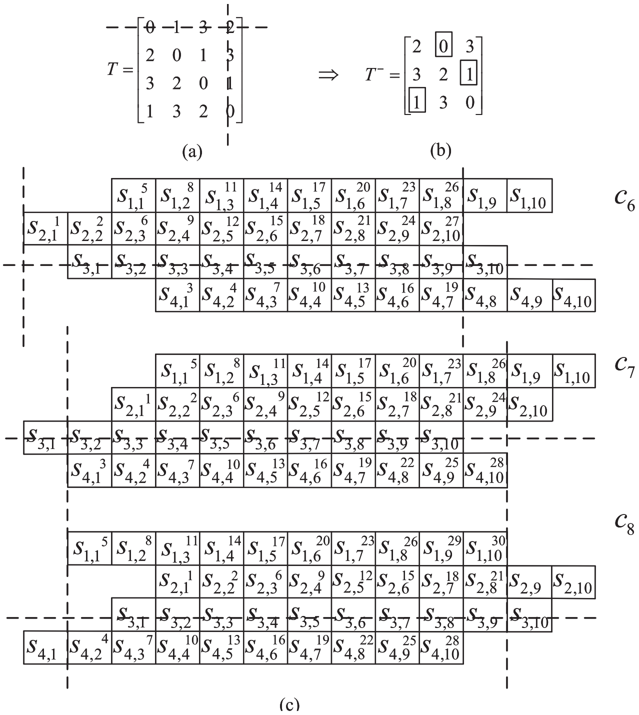


FIGURE 2. An example for MSR-MOS-F and the corresponding decoding order.

V. OPTIMAL BANDWIDTH RECONSTRUCTION SCHEME FOR CASES WHEN $k > 4$

We first describe the proposed scheme in subsec. V-A. We then provide a proof to show that the proposed scheme is able to recover all the original information in subsec. V-B. Finally, in subsec. V-C, we illustrate with an example.

A. DATA RECONSTRUCTION SCHEME

We first illustrate the proposed sub-packets fetching scheme and name it as optimal bandwidth reconstruction fetching (OBR-F). Afterwards, with the resultant packets obtained by OBR-F, we illustrate the proposed decoding method based on ED algorithm and name it as OBR-ED.

1) OBR-F

The main idea is first about cyclically transforming $\boldsymbol{T}^-$ into a form that satisfies DRD property. *Cyclic transformation* is defined as: Cut a block into two blocks, denoted by $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$, respectively. If the cut is vertical, move the left block, say $\boldsymbol{D}_1$, to the right of $\boldsymbol{D}_2$. Otherwise, if the cut is horizontal, move the top block, say $\boldsymbol{D}_1$, to below $\boldsymbol{D}_2$. Note that the components in corresponding degenerated parity packets should accompany the cyclic transformation of $\boldsymbol{T}^-$ accordingly. Afterwards, within each degenerated parity packet obtained by the above cyclic transformation, fetch a length-L sub-packet.

Detailed implementation of OBR-F includes four steps. In Step I, transform $\boldsymbol{T}^-$ into a form, denoted by $\boldsymbol{A}$, whose elements in the top row increase monotonically. In Step II, extend matrix $\boldsymbol{A}$ to a big matrix which will be defined below. In Step III, within such a big matrix, find a $J \times J$ submatrix that are formed by consecutive rows and columns, denoted by $\boldsymbol{C}$, that satisfies the following condition:

$$\Delta^i_{m-1,m} > \Delta^j_{m-1,m} \text{ for all } i < j \leq m, \quad m \in \{2, 3, \ldots, J\}, \tag{6}$$

where definition of Δ simulates (5) and is applied on matrix $\boldsymbol{C}$ as follows:

$$\Delta^p_{i,j} \triangleq C_{pj} - C_{pi}, \quad p, i, j \in \mathcal{J}, \ i \neq j. \tag{7}$$

In Step IV, let matrix $\boldsymbol{C}$ serve as the degenerated matrix $\boldsymbol{T}^-$. Within each degenerated parity packet, fetch a sub-packet that consists of L consecutive bits.

Step I: If those elements in the top row of $\boldsymbol{T}^-$ increase monotonically, simply let $\boldsymbol{A} = \boldsymbol{T}^-$. Otherwise, let i_s denote the column index of the smallest element in the top row of $\boldsymbol{T}^-$. Cyclically move the block that consists of the first

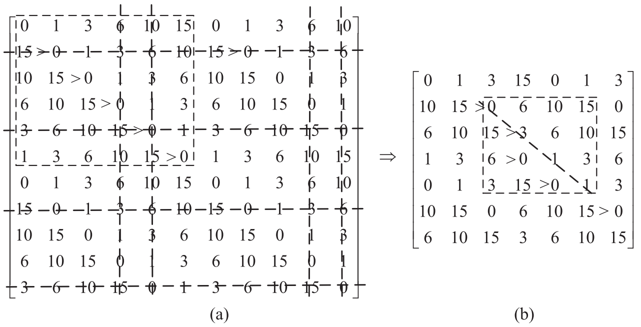


FIGURE 3. An example for a big matrix.

$i_s - 1$ columns of $\boldsymbol{T}^-$ to the right of the J-th column and let $\boldsymbol{A}$ be the resultant matrix.

Step II: Copy the block of the first $J - 1$ columns of $\boldsymbol{A}$ after the J-th column. In the resultant matrix, copy the block of the first $J-1$ rows after the J-th row. We call such a matrix a *big matrix*. A big matrix example for setting $k = 6, J = 4$ is shown in Fig. 3. For the sake of easy understanding, the big matrix that corresponds to $\boldsymbol{A} = \boldsymbol{T}$ (surrounded by the dashed square) is shown in (a), where those removed rows and columns that correspond to intact systematic packets and failed parity packets, respectively, are crossed out by dashed lines. In (b), only the $\boldsymbol{A}$ that corresponds to $\boldsymbol{T}^-$ is extended to the big matrix.

Step III: Within the big matrix, start by testing whether the submatrix in the top-left corner, namely $\boldsymbol{C} = \boldsymbol{A}$, satisfies the condition in (6). In submatrix $\boldsymbol{A}$, according to DRD (a), there are two types of signs that denote the relationship between two horizontally adjacent elements: "larger than" and "smaller than". We only show those "larger than" signs in the figure and let a *dotted curve* connect all those adjacent signs. If the dotted curve is below the diagonal line of $\boldsymbol{A}$, then submatrix $\boldsymbol{C} = \boldsymbol{A}$ automatically satisfies the condition in (6). Otherwise, with slope -1, draw the tangent line of the dotted curve and let P_A denote the point of tangency. Shift $\boldsymbol{A}$ along its diagonal line to $\boldsymbol{C}$ with P_A serving as the top-left vertex of $\boldsymbol{C}$. Submatrix $\boldsymbol{C}$ satisfies the condition in (6) and the proof is deferred to a later subsection. An illustrative example of finding $\boldsymbol{C}$ may refer to Fig. 4.

Step IV: Within submatrix $\boldsymbol{C}$, we call the elements that are in the diagonal line *pivotal elements*. Each pivotal element denotes the bit index of the LMB in the associated degenerated parity packet. Starting from such a bit, we fetch L consecutive bits to compose one sub-packet. We denote such a bits-chosen pattern by placing a square around each pivotal element of matrix $\boldsymbol{C}$. We draw a *dashed diagonal line* closely below the squares as shown in Fig. 3 (b).

Remark: Note that $\boldsymbol{C}$, that satisfies (6), may not be unique. The above steps just provide one way of finding an option of $\boldsymbol{C}$.

2) OBR-ED

Dedicated for the resultant sub-packets obtained by OBR-F, we propose a corresponding decoding method based on ED algorithm, and we name it as OBR-ED. Note that during the decoding process, elements in $\boldsymbol{C}$ are updated as described in the ED algorithm. Therefore, to differentiate with the original $\boldsymbol{C}$, we let $\boldsymbol{C}^u$ denote the updated $\boldsymbol{C}$. OBR-ED contains two steps:

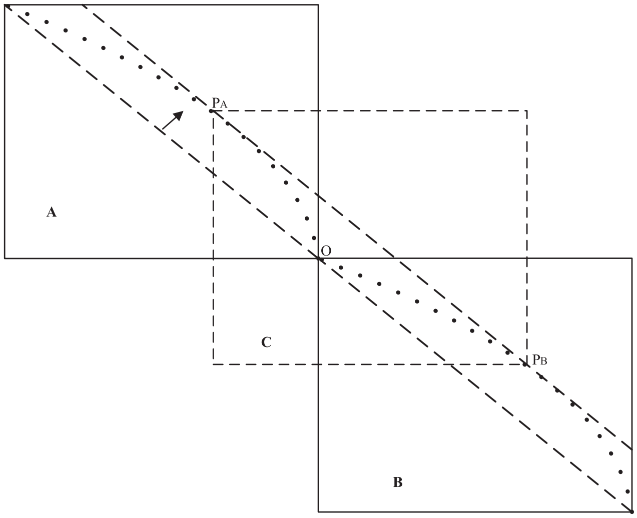


FIGURE 4. Illustration of finding a cyclic transformation submatrix in the big matrix.

Step I: Apply the ED algorithm to the top row of matrix $\boldsymbol{C}$.

Step II: For each t that increases from 2 to J and subject to initial constraint that the sign in the $(t-1, t-1)$-th position of $\boldsymbol{C}^u$ is an equal sign, namely,

$$C^u_{t-1,t-1} = C^u_{t-1,t}, \tag{8}$$

we apply the ED algorithm to the $t \times t$ submatrix of $\boldsymbol{C}^u$ in the top-left corner. During the above t-increasing step, the condition for transition from $t-1$ to t is (8).

The ED algorithm applied on a $t \times t$ submatrix of $\boldsymbol{C}^u$ is as follows:

Initially, due to condition (8), the $(t-1, t-1)$-th sign is an equal sign.

Substep I: Apply the ED algorithm to the t-th row.

Substep II: Apply the ED algorithm to the first row, followed by the second row, the third row, ..., until the t-th row.

Substep III: Repeat Substep II until condition

$$C^u_{t,t} = C^u_{t,t+1} \tag{9}$$

for transition from t to $t+1$ is satisfied.

B. PROOF OF ABILITY TO RECOVER ALL THE ORIGINAL INFORMATION

We first give the properties possessed by the OBR-F resultant. Afterwards, based on such properties, we prove that OBR-F combined with OBR-ED is able to reconstruct all the original information.

1) PROPERTIES

There are four properties for matrix $\boldsymbol{C}$ or $\boldsymbol{C}^u$ (we use $\boldsymbol{C}$ to represent both):

(a) *Distinct Relative Difference (DRD) Property*: DRD as described in Sec. II is preserved for matrix $\boldsymbol{C}$.

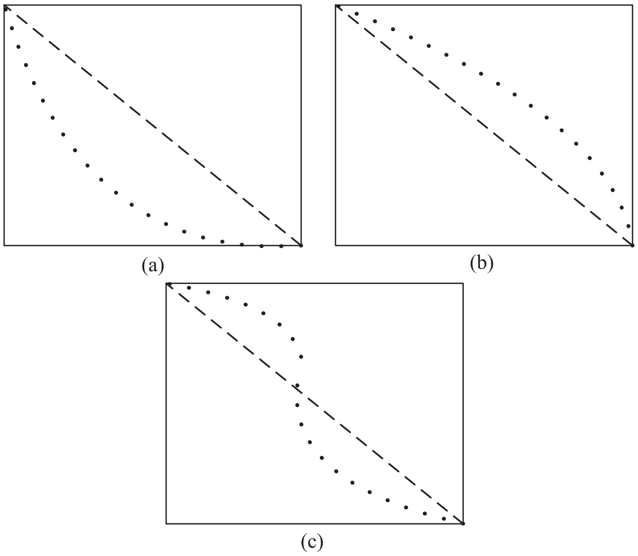


FIGURE 5. Location relationships between dotted curve and dashed diagonal line.

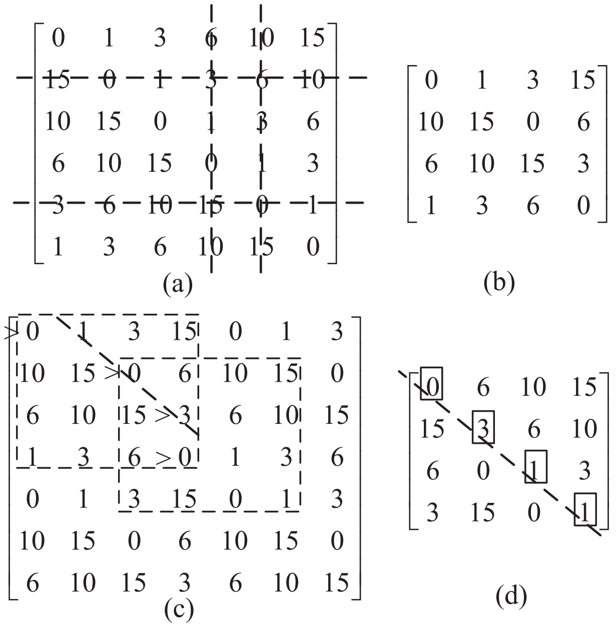
- (b) *Monotonically Decreasing (MD) Property*: Within $\boldsymbol{C}$, the slope at each point of the dotted curve is in region $[-\pi/2, 0)$.
- (c) *No Cut-off Bits (NCB) Property*: Within the degenerated parity packet that is associated with row $j \in \{1, 2, \ldots, J\}$ of $\boldsymbol{C}$, there is no cut-off bit in the component that corresponds to column j of $\boldsymbol{C}$.
- (d) *Equal-Larger-Smaller (ELS) Property*: During the OBR-ED process, if the equality sign occurs within the region as described by condition (6) in $\boldsymbol{C}^u$, the sign closely below the equality is "larger than", and the signs above the equality are all "smaller than".

All the above follow purely or in part from the cyclic nature of $\boldsymbol{T}$, which is a phenomenon that automatically applies to its submatrices and the corresponding cyclic transformation resultant. Besides, (b) follows in part from the fact that the dotted curve in $\boldsymbol{T}$ is monotonically non-increasing and removing certain rows and certain columns of $\boldsymbol{T}$ preserves the non-increasing phenomenon, which also applies to the submatrix of $\boldsymbol{T}$ and the corresponding cyclic transformation resultant, (c) follows from the sub-packet fetching method OBR-F, and (d) follows from (6).

2) OBR-F COMBINED WITH OBR-ED CAN RECONSTRUCT ALL THE ORIGINAL INFORMATION

Theorem 2: For the (n, k) CP-BZD code, based on the resultant sub-packets obtained by applying OBR-F on arbitrary k packets, OBR-ED is able to reconstruct all the original information.

Proof: We divide the proof into two steps. In Step I, we show that by applying OBR-F, we can always transform matrix $\boldsymbol{T}^-$ into form $\boldsymbol{C}$ that satisfies the condition in (6), which is dealt with in Lemma 3. In Step II, based on the sub-packets obtained by OBR-F, we prove that the OBR-ED method can recover all the original information. In particular, we prove that those cut-off bits in the left hand side and right hand side do not prevent from recovering all the original information in Lemma 4 and Lemma 5, respectively. ■



$$\begin{bmatrix} 0 & 1 & 3 & 6 & 10 & 15 \\ 15 & 0 & 1 & 3 & 6 & 10 \\ 10 & 15 & 0 & 1 & 3 & 6 \\ 6 & 10 & 15 & 0 & 1 & 3 \\ 3 & 6 & 10 & 15 & 0 & 1 \\ 1 & 3 & 6 & 10 & 15 & 0 \end{bmatrix}$$

(a)

$$\begin{bmatrix} 0 & 1 & 3 & 15 \\ 10 & 15 & 0 & 6 \\ 6 & 10 & 15 & 3 \\ 1 & 3 & 6 & 0 \end{bmatrix}$$

(b)

$$\begin{bmatrix} 0 & 1 & 3 & 15 & 0 & 1 & 3 \\ 10 & 15 & 0 & 6 & 10 & 15 & 0 \\ 6 & 10 & 15 & 3 & 6 & 10 & 15 \\ 1 & 3 & 6 & 0 & 1 & 3 & 6 \\ 0 & 1 & 3 & 15 & 0 & 1 & 3 \\ 10 & 15 & 0 & 6 & 10 & 15 & 0 \\ 6 & 10 & 15 & 3 & 6 & 10 & 15 \end{bmatrix}$$

(c)

$$\begin{bmatrix} \boxed{0} & 6 & 10 & 15 \\ 15 & \boxed{3} & 6 & 10 \\ 6 & 0 & \boxed{1} & 3 \\ 3 & 15 & 0 & \boxed{1} \end{bmatrix}$$

(d)

FIGURE 6. Illustrative example for cyclic transformation in OBR-F.

Lemma 3: Given a $J \times J$ submatrix $\boldsymbol{T}^-$ obtained from the $k \times k$ matrix $\boldsymbol{T}$ by taking its arbitrary J rows and J columns, OBR-F can transform $\boldsymbol{T}^-$ into matrix $\boldsymbol{C}$ which satisfies condition (6).

Proof: According to the first three steps of OBR-F, cyclic transformation on a matrix is implemented by finding a submatrix/block that is formed by consecutive rows and columns within the corresponding big matrix. Within such a big matrix, if we can draw a dashed diagonal line of length J, such that the $J \times J$ submatrix that uses this line as diagonal line satisfies conditions c1 and c2 as shown below, then the condition in (6) can be satisfied.

c1) The top "larger than" sign in the submatrix is closely below the diagonal line.

c2) Within this submatrix, no "larger than" sign occurs above the diagonal line.

The former condition is to ensure that those elements in the top row of the submatrix increase monotonically, and the latter condition follows from the fact that the DRD holds for $\boldsymbol{T}$ and its submatrices and the corresponding cyclic transformation resultant. Note that it is also possible that there is no "larger than" sign within such a submatrix. In this case, condition (6) is automatically satisfied.

We claim that we are able to find or draw such a diagonal line and find the corresponding submatrix $\boldsymbol{C}$. We start by testing the $J \times J$ submatrix in the top-left corner, namely $\boldsymbol{A}$. According to Step I of OBR-F, $\boldsymbol{A}$ satisfies condition c1, namely the start points of the dotted curve and the dashed diagonal line in the top-left corner of $\boldsymbol{A}$ overlap. As a result, according to the MD property, there are in total three possibilities for the location relationship between the dotted

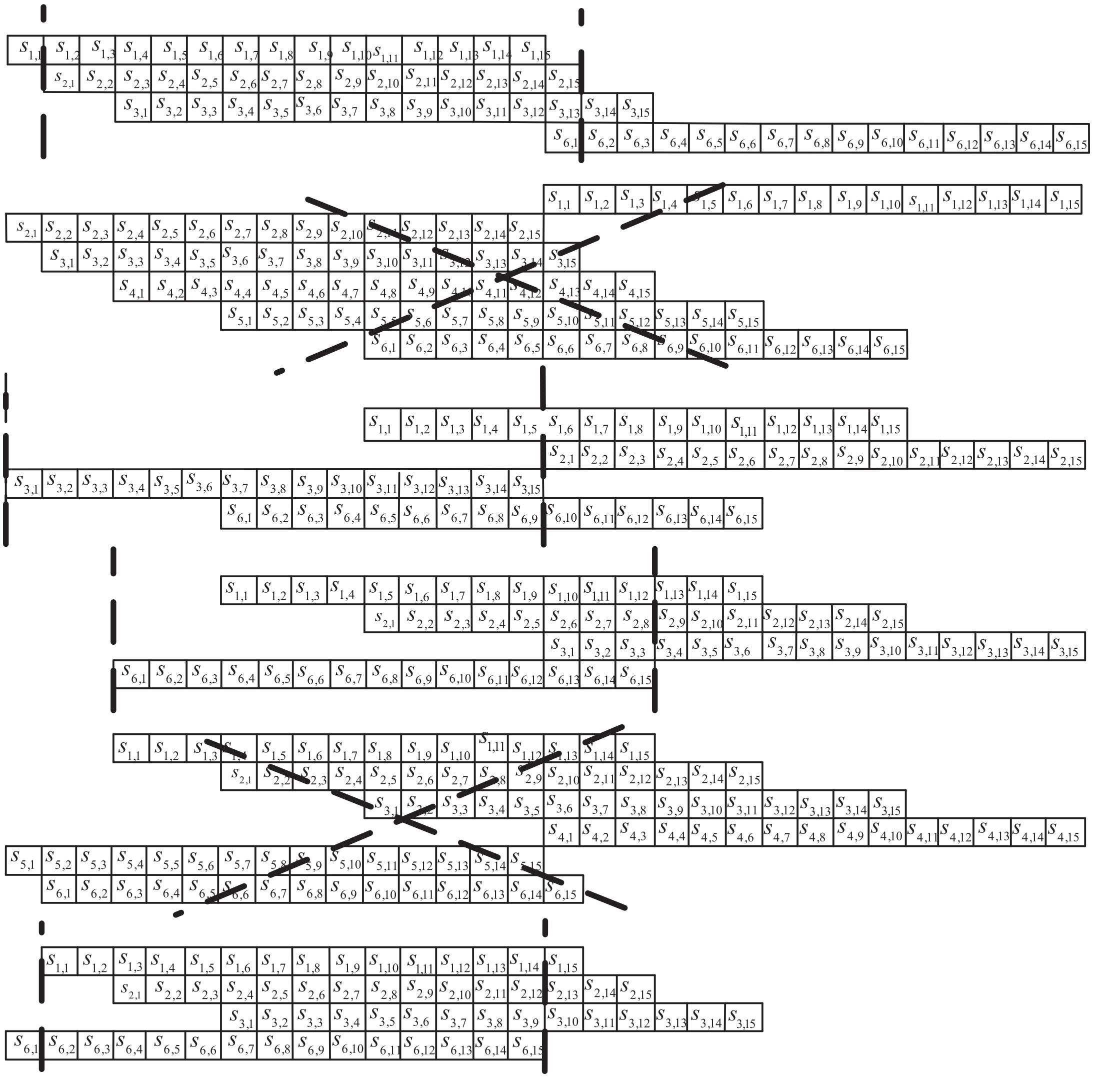


FIGURE 7. **An illustrative example for sub-packets fetching under OBR-F.**

curve and the dashed diagonal line, which are illustrated to be cases (a), (b), and (c) as shown in Fig. 5. The first two cases represent scenarios where the dotted curve is below or above the dashed diagonal line, respectively. The third case represents the scenario where there exist points of the dotted curve that are sometimes below and sometimes above the dashed diagonal line. For case (a), the condition in (6) is automatically satisfied. For cases (b) and (c), without loss of generality, we only illustrate the case for (b). As shown in Fig. 4, we shift $\boldsymbol{A}$ along its diagonal to $\boldsymbol{B}$ where $\boldsymbol{A}$ and $\boldsymbol{B}$ have a common vertex O. We draw the tangent lines for the dotted curve in $\boldsymbol{A}$ and $\boldsymbol{B}$, and let P_A and P_B denote the point of tangency, respectively. Due to simple geometry knowledge, these two tangent lines must overlap. We thus shift $\boldsymbol{A}$ along its diagonal line to $\boldsymbol{C}$ with P_A serving as the top-left vertex of $\boldsymbol{C}$. It is straightforward that submatrix $\boldsymbol{C}$ satisfies the condition in (6), which can be observed visually from Fig. 4. ■

Based on matrix form $\boldsymbol{C}$ and corresponding sub-packets, we investigate the decoding process from the left hand side to the right hand side.

Lemma 4: Based on the sub-packets fetched by the OBR-F method, OBR-ED can initiate the decoding process from the left hand side and propagate to the state as if those cut-off bits on the left hand side are not removed.

Proof: Consider the top row of $\boldsymbol{C}$. According to construction of $\boldsymbol{C}$, within the degenerated parity packet that is associated with this row, there is no cut-off bit on the left hand side. The ED rule can be directly applied to this row, which leads to the equality sign in the $(1, 1)$-th position of $\boldsymbol{C}^u = \boldsymbol{C}$.

We proceed to consider the 2×2 submatrix in the top-left corner of $\boldsymbol{C}^u$. According to the ELS property, the sign in the $(2, 1)$-th position is "larger than" which indicates that all those cut-off bits on the left hand side of the component that are associated with column 1 have been recovered. Besides, due to the NCB property, there is no cut-off bit on the left

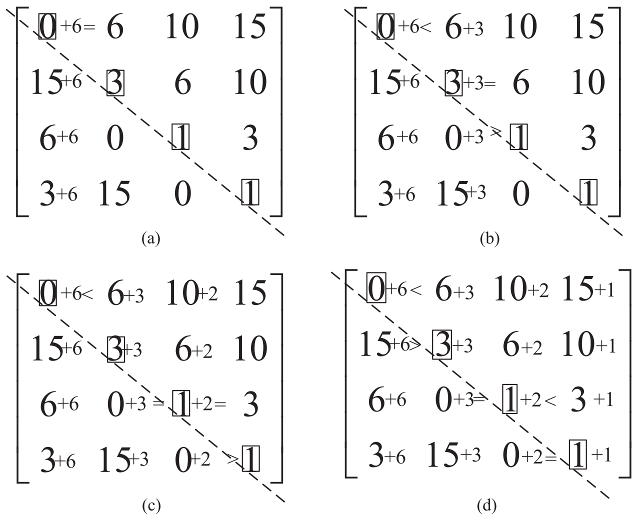


FIGURE 8. An illustrative example for decoding process in OBR-ED.

hand side of the component that is associated with column 2. Therefore, the decoding process within this 2×2 submatrix is equivalent to the case where the bits on the left hand side are not removed. According to Theorem 1, the above decoding process within the 2×2 submatrix can propagate towards the right ceaselessly until an external condition is satisfied, where the external condition is when one accumulated element in either the first column or the second column within the 2×2 submatrix is equal to a certain element to its right. According to (6), the above process terminates when $C^u_{2,2} = C_{2,3}$, where these two elements are updated and original, respectively.

Now, consider the 3×3 submatrix in the top-left corner. According to the ELS property, the (3, 2)-th sign is "larger than", which indicates that those cut-off bits on the left hand side of the component that are associated with column 2 have been recovered. Similar to the previous case, the decoding process within the 3×3 submatrix is equivalent to the case where those cut-off bits on the left hand side of the components that are associated with the 3×3 submatrix are not removed and hence can propagate until when $C^u_{3,3} = C_{3,4}$.

The above process continues until the $(J-1, J-1)$-th accumulated element is equal to the $(J-1, J)$-th element, namely when

$$C^u_{J-1,J-1} = C^u_{J-1,J}. \qquad (10)$$

This fact combined with the NCB property indicates that all those cut-off bits on the left hand side of those J degenerated parity packets have been recovered. ■

As all those cut-off bits on the left hand side have been recovered, the decoding process is equivalent to the decoding process in [14]. Therefore, according to Theorem 1, the decoding process continues to the right hand side. We will show that those cut-off bits on the right hand side do not affect the decodability as well.

Lemma 5: Based on those sub-packets obtained by OBR-F, we can use OBR-ED until all those cut-off bits on the left hand side are recovered, namely until (10) is satisfied. Afterwards, during the process of applying OBR-ED,

(a) All bits are recovered at pivotal elements of $\boldsymbol{C}^u$ in the sense that the bits are recovered at the component that is associated with the pivotal element's column.

(b) All the original information can be recovered, namely those cut-off bits on the right hand side can be deemed as if not removed at all.

Proof: In substep I of OBR-ED, according to the ELS property, (10) indicates the ED algorithm is applied on the J-th row which results in the equality sign in the (J, J)-th position. Those bits are recovered at the J-th pivotal element.

Now, since those elements in the top row of $\boldsymbol{C}^u$ increase monotonically, we can apply substep II of OBR-ED on $\boldsymbol{C}^u$ iteratively. During such an ED application, it is straightforward that all bits are recovered at those pivotal elements as the following shows: Initially, applying ED to the top row results in the equality sign at position (1, 1). Equality sign at position (t, t) indicates a "larger than" sign at position $(t+1, t)$ and applying ED results in an equality sign at position $(t+1, t+1)$. This completes the proof for statement (a).

Statement (a) indicates that those bits are recovered at those pivotal elements instead of at the position of cut-off bits. Since those bits in the component that is associated with pivotal element are not cut off, we conclude that recovering all the information does not need the cut-off bits on the right hand side. ■

C. AN INTUITIVE EXAMPLE

1) EXAMPLE SETTING

Set $L = 15$ and $k = 6$. Use packets c_4, c_5, c_7, c_9, c_{10}, and c_{12} to recover the original information. Systematic packets c_4 and c_5 have length L, while parity packets c_7, c_9, c_{10}, and c_{12} all have length $L + \frac{k(k-1)}{2} = L + 15$. As described in Subsec. II-C, systematic packets c_4 and c_5 do not require any decoding and hence can be directly recovered. Furthermore, they can be substituted into the parity packets, which is equivalent to removing the corresponding columns (columns 4 and 5) in the matrix. This removal of rows and columns is illustrated in Fig. 6 (a), where the removal of rows 2 and 5 corresponds to the fact that parity packets c_{k+2} and c_{k+5} are not fetched. We rewrite those remaining rows and columns into a 4×4 matrix as shown in Fig. 6 (b).

2) SUB-PACKET FETCHING AND DECODING

The process of cyclic transformation on the matrix is shown in Fig. 6 (c), with the resultant matrix $\boldsymbol{C}$ together with those pivotal elements as shown in Fig. 6 (d), where the bits-chosen pattern is denoted by squares around each pivotal element. The corresponding packets are shown in Fig. 7, where the two dashed crossings on two parity packets indicate that parity packets c_8 and c_{11} are not fetched. Note that within each of the remaining degenerated packets, there are two vertical dashed lines with relative distance L, which means the sub-packet within these two vertical lines are fetched. More specifically, in packets c_7, c_9, c_{10}, and c_{12}, we choose L consecutive bits starting from 1, 0, 3, and 1, respectively. A detailed decoding process in the first few steps, actually the updating of each element in corresponding $\boldsymbol{C}^u$, is illustrated in Fig. 8.

VI. CONCLUSION

A data reconstruction scheme for the storage code that possesses the combination property (CP) and zigzag decodable (ZD) is proposed for distributed storage systems. Our proposed scheme contacts k packets and fetches sub-packets of length L bits from each packet. It is optimal in terms of bandwidth usage in the sense that the amount of bits downloaded is equal to the amount of information to be reconstructed.

REFERENCES

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, Jul. 2000.

[2] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4539–4551, Sep. 2010.

[3] B. Dong, Q. Zheng, F. Tian, K.-M. Chao, R. Ma, and R. Anane, "An optimized approach for storing and accessing small files on cloud storage," *J. Netw. Comput. Appl.*, vol. 35, no. 6, pp. 1847–1862, Nov. 2012.

[4] X. Guan and B. Y. Choi, "Push or pull? Toward optimal content delivery using cloud storage," *J. Netw. Comput. Appl.*, vol. 40, pp. 234–243, Apr. 2014.

[5] M. Xiao, M. Medard, and T. Aulin, "A binary coding approach for combination networks and general erasure networks," in *Proc. IEEE ISIT*, Nice, France, Jun. 2007, pp. 786–790.

[6] S. B. Wicker and V. K. Bhargava, *Reed-Solomon Codes and Their Applications*. Hoboken, NJ, USA: Wiley, 1994.

[7] A. G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh, "A survey on network codes for distributed storage," *Proc. IEEE*, vol. 99, no. 3, pp. 476–489, Mar. 2011.

[8] F. D. Rossi, M. G. Xavier, C. A. F. De Rose, "E-ECO: Performance-aware energy-efficient cloud data center orchestrration," *J. Netw. Comput. Appl.*, vol. 75, pp. 83–96, Jan. 2017.

[9] H. S. Yeo, X. S. Phang, H. Lee, "Leveraging client-side storage techniques for enhanced use of multiple consumer cloud storage services on resource-constrained mobile devices," *J. Netw. Comput. Appl.*, vol. 43, pp. 142–156, Aug. 2014.

[10] Z. Yan and W. Shi, "CloudFile: A cloud data access control system based on mobile social trust," *J. Netw. Comput. Appl.*, vol. 86, pp. 46–58, May 2017.

[11] S. Gollakota and D. Katabi, "ZigZag decoding: Combating hidden terminals in wireless networks," in *Proc. SIGCOMM*, Aug. 2008, pp. 159–170.

[12] C. W. Sung and X. Gong, "A zigzag-decodable code with the MDS property for distributed storage systems," in *Proc. IEEE ISIT*, Istanbul, Turkey, Jul. 2013, pp. 341–345.

[13] M. Dai *et al.*, "Design of (4, 8) binary code with MDS and zigzag-decodable property," *Wireless Pers. Commun.*, vol. 89, no. 1, pp. 1–13, Jul. 2016.

[14] M. Dai, C. W. Sung, H. Wang, X. Gong, and Z. Lu, "A new zigzag-decodable code with efficient repair in wireless distributed storage," *IEEE Trans. Mobile Comput.*, vol. 16, no. 5, pp. 1218–1230, May 2017.

[15] M. Dai, B. Mao, Y. Fan, X. Lin, H. Wang, and B. Chen "Optimal reconstruction bandwidth scheme for zigzag-decodable code with combination property in cloud storage system," in *Proc. Int. Conf. Connected Vehicles Expo (ICCVE)*, Shenzhen, China, Oct. 2015, pp. 180–184.

[16] W. M. P. van der Aalst and E. Damiani, "Processes meet big data: Connecting data science with process science," *IEEE Trans. Services Comput.*, vol. 8, no. 6, pp. 810–819, Jun. 2015.

[17] H. Iwamoto *et al.*, "Designing carrier's online storage 'family cloud' for enhancing telecom home services," in *Proc. IEEE 17th Int. Conf. Intell. Next Generat. Netw. (ICIN)*, Venice, Italy, Oct. 2013, pp. 75–85.

[18] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4539–4551, Jul. 2000.

[19] D. S. Papailiopoulos, J. Luo, A. G. Dimakis, C. Huang, and J. Li, "Simple regenerating codes: Network coding for cloud storage," in *Proc. IEEE INFOCOM*, Mar. 2012, pp. 2801–2805.

[20] R. C. Chiang, A. J. Uppal, and H. H. Huang, "An adaptive IO prefetching approach for virtualized data centers," *IEEE Trans. Services Comput.*, to be published.

[21] V. R. Cadambe, C. Huang, J. Li, and S. Mehrotra, "Polynomial length MDS codes with optimal repair in distributed storage," in *Proc. IEEE ASILOMAR*, Pacific Grove, CA, USA, Nov. 2011, pp. 1850–1854.

[22] A. Thangaraj and C. Sankar, "Quasicyclic MDS codes for distributed storage with efficient exact repair," in *Proc. IEEE Inf. Theory Workshop*, Paraty, Brazil, Oct. 2011, pp. 45–49.

[23] I. Tamo, Z. Wang, and J. Bruck, "Zigzag codes: MDS array codes with optimal rebuilding," *IEEE Trans. Inf. Theory*, vol. 59, no. 3, pp. 1597–1619, Mar. 2013.

[24] S. S. Arslan, "Redundancy and aging of efficient multidimensional MDS parity-protected distributed storage systems," *IEEE Trans. Device Mater. Rel.*, vol. 14, no. 1, pp. 275–285, Jan. 2014.

[25] V. R. Cadambe, S. A. Jafar, H. Maleki, K. Ramchandran, and C. Suh, "Asymptotic interference alignment for optimal repair of MDS codes in distributed storage," *IEEE Trans. Inf. Theory*, vol. 59, no. 5, pp. 2974–2987, May 2013.

[26] S. S. Yau and Y. C. Liu, "On decoding of maximum-distrance spearable linear codes," *IEEE Trans. Inf. Theory*, vol. 17, no. 4, pp. 487–491, Apr. 1971.

[27] M. Dai, H. Y. Kwan, and C. W. Sung, "Linear network coding strategies for the multiple access relay channel with packet erasures," *IEEE Trans. Wireless Commun.*, vol. 12, no. 1, pp. 218–227, Jan. 2013.

[28] J. Heide, M. V. Pedersen, F. H. P. Fitzek, and M. Medard, "On the code parameters and coding vector representation for practical RLNC," in *Proc. IEEE Int. Conf. Commun.*, Kyoto, Japan, May 2009, pp. 1–5.

[29] P. Vingelmann, M. V. Pedersen, F. H. P. Fitzek, and J. Heide, "Multimedia distribution using network coding on the iphone platform," in *Proc. ACM Multimedia Workshop Mobile Cloud Media Comput.*, Firenze, Italy, Oct. 2010, pp. 1–5.

[30] M. Sathiamoorthy *et al.*, "XORing elephants: Novel erasure codes for big data," *VLDB Endowment*, vol. 6, no. 5, pp. 325–336, Mar. 2013.

[31] S. Li, S. Wan, D. Chen, X. He, Y. Guo, and P. Huang, "Exploiting decoding computational locality to improve the I/O performance of an XOR-coded storage cluster under concurrent failures," in *Proc. Int. Symp. Reliable Distrib. Syst.*, Nara, Japan, Oct. 2014, pp. 125–135.

[32] M.-S. Hwang and C. H. Lee, "Secure access schemes in mobile database systems," *Trans. Emerg. Telecommun. Technol.*, vol. 12, no. 4, pp. 303–310, Apr. 2001.

[33] M. Gerami and M. Xiao, "Repair for distributed storage systems with erasure channels," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Budapest, Hungary, Jun. 2013, pp. 4058–4062.

[34] X. Yang, X. Tao, E. Dutkiewicz, X. Huang, Y. J. Guo, and Q. Cui, "Energy-efficient distributed data storage for wireless sensor networks based on compressed sensing and network coding," *IEEE Trans. Wireless Commun.*, vol. 12, no. 10, pp. 5087–5099, Oct. 2013.

[35] J. Pääkkönen, C. Hollanti, and O. Tirkkonen, "Device-to-device data storage for mobile cellular systems," in *Proc. IEEE Globecom Workshops*, Atlanta, GA, USA, Dec. 2013, pp. 671–676.

MINGJUN DAI received the Ph.D. degree in electronic engineering from the City University of Hong Kong in 2012. He joined the faculty with the College of Information Engineering, Shenzhen University, Shenzhen, China, and is currently an Associate Professor. His research interests include cooperative relay network, network coding design, distributed storage, and visible light communication.

XIA WANG is with the College of Information Engineering, Shenzhen University, Shenzhen, China. Her research interests include distributed storage.

HUI WANG is with Shenzhen University, Shenzhen, China. His research interests include distributed storage.

XIAOHUI LIN is with Shenzhen University, Shenzhen, China. His research interests include distributed storage.

BIN CHEN is with Shenzhen University, Shenzhen, China. His research interests include distributed storage.