

# ***IET Communications***

## **Special issue Call for Papers**

---

**Be Seen. Be Cited.  
Submit your work to a new  
IET special issue**

Connect with researchers and experts in your field and share knowledge.

Be part of the latest research trends, faster.


**Read more**



The Institution of  
Engineering and Technology

## ORIGINAL RESEARCH

# Weighted zigzag decodable fountain codes for unequal error protection

Yin Zhang<sup>1</sup> | Yuli Zhao<sup>1</sup>  | Francis C. M. Lau<sup>2</sup> | Bin Zhang<sup>1</sup> | Zhiliang Zhu<sup>1</sup> | Hai Yu<sup>1</sup>

<sup>1</sup>Software College, Northeastern University, Shenyang, China

<sup>2</sup>Department of Electronic & Information Engineering, Hong Kong Polytechnic University, Hong Kong

## Correspondence

Yuli Zhao, Software College, Northeastern University, Shenyang, China.  
Email: zhaoyl@swc.neu.edu.cn

## Funding information

National Key Research and Development Program of China, Grant/Award Number: 2019YFB1405803; Fundamental Research Funds for the Central Universities, Grant/Award Numbers: N2017016, N2017011; National Natural Science Foundation of China, Grant/Award Numbers: 61603082, 61977014, 61902056, 61902057

## Abstract

By combining bit-shift and exclusive-or operations, a weighted zigzag decodable fountain code is proposed to achieve an unequal error protection property. In the proposed scheme, the input symbols of different importance levels are first pre-coded into variable nodes using low-density parity-check codes. Then, bit-shift operations prior to exclusive-or are performed on the non-uniformly selected variable nodes to generate encoded symbols. An analysis of the erasure probabilities based on the and-or tree is further introduced. Simulation results show that by appropriately choosing the maximum bit-shift amount, the proposed weighted zigzag decodable fountain codes can successfully recover the more important input symbols prior to the less important ones.

## 1 | INTRODUCTION

In general, messages to be transmitted over the Internet are first divided into packets. A packet is discarded and considered as lost if the destination cannot receive it correctly. Hence, the Internet is usually modeled as a packet erasure channel (PEC). Similar to data transmission over a binary erasure channel (BEC), forward error correction codes can be used for achieving reliable data transmission over a PEC. However, some forward error correction codes with fixed code rates such as low-density parity-check (LDPC) codes and turbo codes are not designed for transmitting data over a BEC/PEC with a time-varying property. Given equally important messages of finite length, fountain codes enable the transmitter to generate an infinite number of encoded symbols randomly [1]. Thus, fountain codes, such as Luby transform codes [2, 3], raptor codes [4], online fountain codes [5, 6], zigzag decodable fountain codes [7], are suitable for protecting transmitted data over the Internet where the channel erasure probabilities are time-varying or even unknown.

Initially, fountain codes consider transmitted data requiring equal error protection (EEP). Fountain codes with EEP property mainly concentrate on the distribution and storage applications of bulk data [8]. However, in some applications, a portion of data may require more protection than other data [9]. Considering applications such as the transmission of video or images using layered coders (H. 264/SVC, H. 265/SHVC, SVT/AV1), some parts of data are deemed more important than others [10–12]. Moreover, the heterogeneity of communication networks in the emerging Internet of Things applications brings the need for prioritizing different types of data [13, 14]. Hence, the more important portion of the original data needs to be recovered with priority. Channel codes with unequal error protection (UEP) property were first studied in [15]. Since then, different UEP codes have been designed with LDPC codes [16, 17], Reed-Muller codes [18] and Polar codes [19, 20].

Rateless codes with the UEP property proposed in [21] enlarge the selection probability of the more important symbols in the encoding process. Thus, the more important input symbols achieve more protection than the less important part

This is an open access article under the terms of the [Creative Commons Attribution](https://creativecommons.org/licenses/by/4.0/) License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

© 2022 The Authors. *IET Communications* published by John Wiley & Sons Ltd on behalf of The Institution of Engineering and Technology.

and are recovered earliest. Extended window fountain (EWF) codes proposed in [22] group the input symbols into overlapped windows, and the more important symbols exist in more windows. The encoded symbols are generated by first selecting a window and choosing input symbols from the selected window. Moreover, in [23], a duplication strategy is used to enhance the UEP property. Generally, most current UEP rateless codes [24–28] are based on weighted UEP codes in [21], EWF codes in [22] or the duplication strategy in [23]. In [29], considering the structure of the stopping set in the BP decoding process, the authors introduce some fixed symbols to generate encoded symbols together with the original input symbols. By using a weighted strategy, the UEP fountain code in [29] achieves a lower BER compared with the method in [23]. In [30], duplication strategy and pre-coding by low-density parity-check (LDPC) codes are used to enhance the UEP property. It has been found that the lower-degree encoded symbols are most likely to crop incident edges and recover their neighbored input symbols. Hence, the UEP fountain codes proposed in [31] provide a degree-dependent strategy to increase the selection probability that lower-degree encoded symbols select the input symbols of higher importance. The UEP fountain codes mentioned above improve the performance for recovery of the more important input symbols at the expense of performance deterioration of the less important ones.

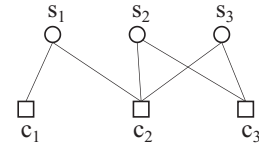
Zigzag decodable fountain codes proposed in [7, 32] combine the bit-shift operation with symbol exclusive-or operation to generate encoded symbols and achieve a low overhead. These codes are further designed to improve intermediate recovery [33] and achieve efficient correction in wireless distributed storage [34].

Motivated by the advantage of the zigzag decodable fountain codes, by shifting the variable nodes and further using exclusive-or operation to generate encoded symbols, we propose a weighted zigzag decodable fountain code with UEP property (WZ-UEP codes for short). Variable nodes of different importance classes are generated by pre-coding different classes of input symbols using LDPC codes of different code rates. By increasing the bit-shift amount, the WZ-UEP codes improve the symbol overhead performance of higher priority data without performance deterioration of lower priority data.

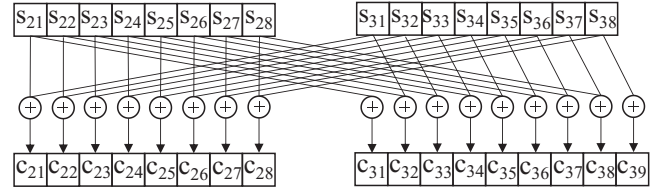
The rest of this paper is organized as follows. In Section 3, we present the proposed WZ-UEP codes. The asymptotic analysis by using the AND-OR tree to find the optimal code parameter of two important classes is given in Section 4. Simulation results are shown in Section 5. Finally, Section 6 concludes the paper.

## 2 | ZIGZAG DECODABLE FOUNTAIN CODES

In this section, we briefly review the fundamental of zigzag decodable fountain codes. For each encoded symbol, the encoder generates the number of input symbols according to a given degree-distribution  $\Omega(x)$ . Each input symbol of size  $l$



**FIGURE 1** Symbol tanner graph of the zigzag decodable fountain codes with 3 input symbols and 3 encoded symbols



**FIGURE 2** The bitwise tanner graph induced by residual encoded symbols  $e_2$  and  $e_3$

is randomly selected. The zigzag decodable fountain codes shift the selected input symbols for some bits. An encoded symbol is assigned with the exclusive-or of the selected input symbols after bit-shift.

Although bit-shift operation potentially increases the bit-length of encoded symbols, the advantage is straightforward. As a toy example, Figure 1 illustrates a zigzag decodable fountain code with 3 input symbols and 3 encoded symbols. Each input symbol has  $l = 8$  bits. Given input symbols  $\{s_1, s_2, s_3\}$ , 3 encoded symbols  $c_1, c_2, c_3$  are generated. The input symbol  $s_1$  is copied to value  $c_1$ . Both  $s_2$  and  $s_3$  are selected to generate encoded symbol  $c_2$  and  $c_3$ . The exclusive-or of  $s_2$  and  $s_3$  is assigned to  $c_3$ . But, the encoder assigns the  $c_3$  with the exclusive-or of  $s_2$  and  $s_3$  with 1 bit right shift.

By using symbol level belief propagation (BP) algorithm, we have  $s_1 = c_1$ . The edges incident from  $s_1$  are eliminated. The residual encoded symbols  $c_2, c_3$  all connect with two input symbols  $s_2$  and  $s_3$ . Hence, the decoder stops and cannot recover  $s_2$  and  $s_3$ . Further, we take each bit in input/encoded symbols as a node and construct the bitwise tanner graph included by the residual encoded symbols, as shown in Figure 2. It can be seen that the bits  $c_{31}$  and  $c_{39}$  are copied from  $s_{21}$  and  $s_{38}$ , respectively. Thus, we have  $s_{21} = c_{31}, s_{31} = s_{21} + c_{21}, s_{22} = s_{31} + c_{32}, \dots, s_{28} = s_{37} + c_{38}, s_{38} = s_{28} + c_{28}$ . The input symbols  $c_2$  and  $c_3$  can be recovered by using bitwise BP algorithm.

In general, the zigzag decodable fountain code achieves a low number of encoded symbols required to recover the original input symbols at the expense of a slightly larger size of encoded symbols. In [7], the bit-shift number of each input symbol follows a predefined bit-shift distribution given as  $\Lambda(x) = \sum_{\theta=0}^{\theta_{max}} \frac{1}{1+\theta_{max}} x^\theta$ , where,  $\theta_{max}$  is the optional maximum bit-shift number. It has been proven that the larger the maximum bit-shift number is, the lower symbol overhead the zigzag fountain code achieves. But, to some extent, the bitwise overhead would increase.



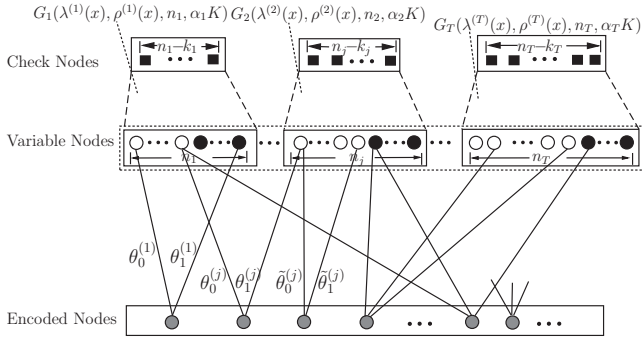


FIGURE 3 The two-layer-bipartite graph of the proposed codes

### 3 | PROPOSED METHOD

In our scheme, the transmitting file is divided into several packets. Each packet is composed of  $K$  input symbols each of  $l$  bits ( $l \geq 1$ ). According to the importance of the input symbols, they are regrouped into  $T$  sets  $S_1, S_2, \dots, S_T$ . Level- $j$  input symbols, also known as input symbols of the  $j$ -th importance, are contained in set  $S_j$ . We have  $|S_j| = \alpha_j K$  ( $\alpha_j \geq 0$ ) and  $\sum_{j=1}^T \alpha_j = 1$ . Moreover, for the index  $j < \tilde{j}$ , the input symbols in set  $S_j$  are more important than those in set  $S_{\tilde{j}}$ . The encoding process of our proposed WZ-UEP codes includes two phases. In the first phase, to provide more protection on the high-priority data, we pre-code the input symbols of different importance levels using LDPC codes of different code rates. Afterwards, to achieve a low overhead for recovering the high priority data, we combine the bit-shift operation and the unbalanced selection strategy in the encoding process of LT codes. Then, the UEP property can be achieved.

As shown in Figure 3, the input symbols of level- $j$  is pre-coded into Type- $j$  variable nodes using LDPC codes with code rate  $R_j$ . For all  $0 < j < \tilde{j} \leq T$ , to provide more protection for the more important input symbols, we have  $R_j < R_{\tilde{j}}$ . Let  $\Phi_{d_v}^{(j)}$  denote the fraction of degree- $d_v$  variable nodes generated from the input symbols in  $S_j$ . Thus, the degree distribution of Type- $j$  variable node is given by  $\Phi^{(j)}(x) = \sum_{d_v} \Phi_{d_v}^{(j)} x^{d_v}$ . We can then denote the edge-degree distribution [30] by  $\lambda^{(j)}(x) = \sum_{d_v} \lambda_{d_v}^{(j)} x^{d_v-1} = \Phi^{(j)}(x)/\Phi^{(j)}(1)$ , where  $\lambda_{d_v}^{(j)}$  is the fraction of edges connected to degree- $d_v$  variable nodes of Type- $j$ . Similarly, let  $\Gamma_{d_c}^{(j)}$  denote the fraction of degree- $d_c$  check nodes of Type- $j$ . Then, we have the degree distribution of the Type- $j$  check nodes as  $\Gamma^{(j)}(x) = \sum_{d_c} \Gamma_{d_c}^{(j)} x^{d_c}$ . Denote  $\rho_{d_c}^{(j)}$  as the fraction of edges which are related to the degree- $d_c$  check nodes. The edge-degree of check nodes can be expressed as  $\rho^{(j)}(x) = \sum_{d_c} \rho_{d_c}^{(j)} x^{d_c-1} = \Gamma^{(j)}(x)/\Gamma^{(j)}(1)$ . The bipartite graph corresponding to the LDPC code for the  $j$ -th importance level input symbols is denoted by  $G_j(\lambda^{(j)}(x), \rho^{(j)}(x), n_j, \alpha_j K)$ , where,  $n_j$  is the number of variable nodes generated from  $\alpha_j K$  input symbols of level- $j$ . Denote  $n = \sum_{j=1}^T n_j$ .

Let  $\Omega(x) = \sum_{\kappa=1}^n \Omega_{\kappa} x^{\kappa}$  be the degree distribution of the encoded symbols in Figure 3, where  $\Omega_{\kappa}$  is the fraction of

degree- $\kappa$  encoded nodes generated. Thus, the edge-degree distribution of an encoded node [21] is given as  $\omega(x) = \sum_{t=1}^{n-1} \omega_t x^t = \frac{\Omega'(x)}{\Omega'(1)}$ , where  $\omega_t$  represents the fraction of edges connected to degree- $t$  encoded symbols.

To provide more protection to the variable nodes of higher importance levels, these variable nodes are selected with a higher probability to generate encoded symbols. Let  $P_j$  be the probability that a particular Type- $j$  variable node is selected ( $j = 1, 2, \dots, T$ ) to generate an encoded node. Then, we have  $\sum_{j=1}^T P_j n_j = 1$ . Moreover, for all  $0 < j < \tilde{j} \leq T$ , we have  $P_j \geq P_{\tilde{j}}$ . Let  $\theta_{max}^{(j)}$  be the maximum bit-shift amount of Type- $j$  variable nodes ( $j = 1, 2, \dots, T$ ). Then, the generator polynomial for the shift probability of the Type- $j$  variable nodes is given as

$$\Lambda^{(j)}(x) = \sum_{\theta=0}^{\theta_{max}^{(j)}} \frac{1}{1+\theta_{max}^{(j)}} x^{\theta}.$$

The encoding process for generating an encoded node from  $T$  classes of variable symbols is as follows.

- (i) Randomly select a degree  $\kappa$  according to the degree-distribution  $\Omega(x)$ .
- (ii) Select each variable node of level- $j$  with a probability  $P_j$  ( $j = 1, 2, \dots, T$ ). Denote the selected  $\kappa$  variable nodes as  $c_1, c_2, \dots, c_{\kappa}$ .
- (iii) Find out the least importance level of  $\{c_1, c_2, \dots, c_{\kappa}\}$  and denote it as  $r$ .
- (iv) If  $r < T$ , we choose the  $\kappa$  shift numbers according to the shift polynomial  $\Lambda^{(r)}(x)$  and denote them in vector  $\vec{\theta} = [\theta_0^{(r)}, \theta_1^{(r)}, \dots, \theta_{\kappa-1}^{(r)}]$ . Let  $\theta_{min} = \min(\{\theta_0^{(r)}, \dots, \theta_{\kappa-1}^{(r)}\})$ , the encoded symbol is given by  $\sum_{v=1}^{\kappa} c_v z^{\theta_{v-1}^{(r)} - \theta_{min}^{(r)}}$ .
- (v) If  $r = T$ , the encoded symbol is assigned with  $\sum_{v=1}^{\kappa} c_v$ .

Thus, this process can be represented by another bipartite graph  $H$  connecting the encoded symbols and the variable nodes. The weight of each edge in  $H$  represents the bit-shift number of the connected variable node.

Note: Although the larger the maximum bit-shift number is, the less encoded symbols are required for decoding. To achieve a low symbol overhead and minimize the average bit-number in an encoded symbol, we select the bit-shift number according to the distribution of the selected input symbols of the least important level in step (iv). If the least-importance level of the selected input symbols is  $T$ , none of the selected input symbols shift at all.

**Theorem 1.** Let  $\gamma$  denote the ratio of the number of received encoded symbols to the number of input symbols. The average node degree of the encoded symbols is given by  $\mu = \sum_{\kappa=1}^n \kappa \Omega_{\kappa} = \Omega'(1)$ . The degree distribution of Type- $j$  variable nodes in  $H$  is given by

$$\Psi_j(x) = (1 + P_j(x-1))^{\mu \gamma K}. \quad (1)$$

*Proof.* Given  $\gamma$  and  $\mu$ , the total number of connections between the variable nodes and the received encoded symbols is approximately  $\gamma \mu K$ . The probability that a Type- $j$  variable node having  $b$  connections with the received encoded symbols can be

expressed as  $\Psi_{j,b} = \binom{\mu\gamma K}{b} P_j^b (1 - P_j)^{\mu\gamma K - b}$ . Then, the degree distribution of Type- $j$  variable nodes in  $H$  is given by

$$\Psi_j(x) = \sum_{b=0}^{\mu\gamma K} \Psi_{j,b} x^b = \sum_{b=0}^{\mu\gamma K} \binom{\mu\gamma K}{b} (P_j x)^b (1 - P_j)^{\mu\gamma K - b}.$$

Applying the binomial theorem, we can obtain (1).  $\square$

For simplicity, we re-write the probability  $P_j$  as  $\frac{b_j}{n}$ . As  $K \rightarrow \infty$ , we have  $\Psi_j(x) \approx \exp\left(\frac{\mu\gamma b_j(x-1)}{\sum_{j=1}^T \frac{\alpha_j}{R_j}}\right) \triangleq \tilde{\Psi}_j(x)$ . Let  $\zeta_{j,b}$  denote the probability that a randomly selected edge connects to a Type- $j$  variable node of degree- $b$ , we have  $\zeta_{j,b} = \frac{b\Psi_{j,b}}{P_j\mu\gamma K}$ . Thus, as  $K \rightarrow \infty$ , the edge degree distribution of Type- $j$  variable nodes in  $H$  can be calculated by  $\zeta_j(x) = \sum \zeta_{j,b} x^{b-1} = \frac{\tilde{\Psi}'_j(x)}{\tilde{\Psi}'_j(1)} = \tilde{\Psi}_j(x)$ .

## 4 | ASYMPTOTIC ANALYSIS

### 4.1 | And-or tree analysis

In this section, we analyze the performance of our proposed WZ-UEP codes using AND-OR tree over an erasure channel [21]. We derive the asymptotic expression of the erasure probability of the variable nodes at each iteration of the BP decoding process. Let

- (i)  $x_{j,i}^{(\tau)}$  be the erasure probability of the  $i$ -th bit ( $i = 1, 2, \dots, l$ ) in the message from a Type- $j$  check node to a Type- $j$  ( $j = 1, 2, \dots, T$ ) variable node at the  $\tau$ -th iteration.
- (ii)  $y_{j,i}^{(\tau)}$  denote the erasure probability of the  $i$ -th bit in the message received by a Type- $j$  check node from a Type- $j$  variable node at the  $\tau$ -th iteration.
- (iii)  $u_{j,i}^{(\tau)}$  be the erasure probability of the  $i$ -th bit in the message from a Type- $j$  variable node to an encoded symbol at the  $\tau$ -th iteration.
- (iv)  $v_{j,i}^{(\tau)}$  represent the erasure probability of the  $i$ -th bit of the message from an encoded symbol to a Type- $j$  variable node at the  $\tau$ -th iteration.
- (v)  $\mathcal{Q}_j^{(\tau)} = (\mathcal{Q}_{j,1}^{(\tau)}, \mathcal{Q}_{j,2}^{(\tau)}, \dots, \mathcal{Q}_{j,l}^{(\tau)})$  be the erasure probability vector of a Type- $j$  variable node at the  $\tau$ -th iteration, where  $\mathcal{Q}_{j,i}^{(\tau)}$  denotes the probability that the  $i$ -th bit of the Type- $j$  variable node is erased at the  $\tau$ -th iteration.

Initially, all the messages from the variable nodes are erased, that is, for all  $j \in \{1, 2, \dots, T\}$ ,  $i \in \{1, 2, \dots, l\}$ ,  $y_{j,i}^{(0)} = u_{j,i}^{(0)} = 1$  holds.

Next, we derive the density evolution equation for  $x_{j,i}^{(\tau)}$  and  $y_{j,i}^{(\tau)}$ . The erasure probability of the  $i$ -th bit of the message along a randomly selected edge from a degree- $d_c$  check node to the Type- $j$  variable node at the  $\tau$ -th iteration is  $1 - \{1 - y_{j,i}^{(\tau-1)}\}^{d_c-1}$ .

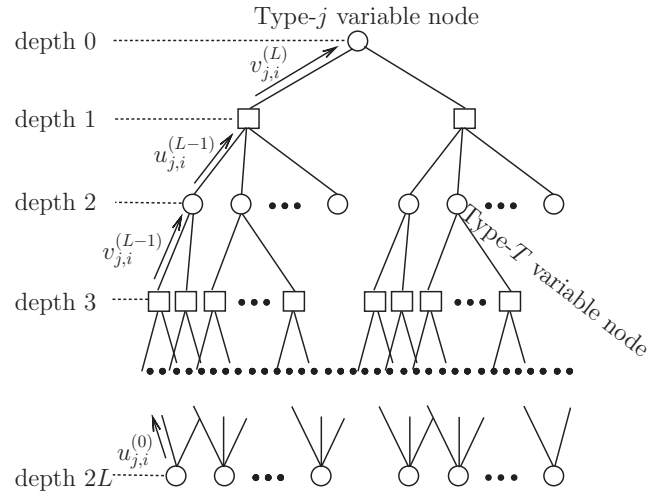


FIGURE 4 An and-or tree whose root node is a Type- $j$  variable node

Hence, we have

$$x_{j,i}^{(\tau)} = 1 - \sum \rho_{d_c}^{(j)} \left[1 - y_{j,i}^{(\tau-1)}\right]^{d_c-1} = 1 - \rho^{(j)} \left(1 - y_{j,i}^{(\tau-1)}\right). \quad (2)$$

Suppose a Type- $j$  variable node connects with  $d_1$  check nodes in  $G_j$  and  $d_2$  encoded symbols. It is straightforward that the  $i$ -th bit of the message from a variable node to a check node is erased only if the  $i$ -th bit of all the incoming message are erased. Hence, the erasure probability of the  $i$ -th bit of a message from a Type- $j$  variable node to a Type- $j$  check node at the  $\tau$ -th iteration is  $\{x_{j,i}^{(\tau)}\}^{d_1-1} \{y_{j,i}^{(\tau)}\}^{d_2}$ . We therefore get

$$\begin{aligned} y_{j,i}^{(\tau)} &= \sum_{d_1} \sum_{d_2} \lambda_{d_1}^{(j)} \Psi_{j,d_2} \left[x_{j,i}^{(\tau)}\right]^{d_1-1} \left[y_{j,i}^{(\tau)}\right]^{d_2} \\ &= \lambda^{(j)} \left(x_{j,i}^{(\tau)}\right) \tilde{\Psi}_j \left(y_{j,i}^{(\tau)}\right). \end{aligned} \quad (3)$$

Similarly, we have  $u_{j,i}^{(\tau)} = \Phi^{(j)}(x_{j,i}^{(\tau)}) \zeta_j(v_{j,i}^{(\tau)})$ .

Thirdly, we derive the value of  $v_{j,i}^{(\tau)}$  using and-or tree analysis ( $j = 1, 2, \dots, T$ ). According to the bipartite graph  $H$ , we can construct an and-or tree of depth  $2L$  ( $L > 0$ ). As shown in Figure 4, the root of the tree is a Type- $j$  variable node at depth 0. Each variable node at depth 0, 2, 4, ...,  $2L-2$  is regarded as an OR-node, and each encoded node at depth 1, 3, 5, ...,  $2L-1$  is labeled as an AND-node. The nodes at depth  $t_p + 1$  are deemed as children of nodes at depth  $t_p$  ( $t_p = 0, 1, 2, \dots, 2L-1$ ).

According to the encoding process presented in Section 3, it is straightforward that an encoded symbol of degree- $\kappa$  is constructed by  $\kappa$  variable nodes of up to  $\kappa$  different importance levels. Define  $q_t$  as the probability that a child of an AND-node independently be a Type- $t$  variable node. We have  $q_t = \frac{P_t \alpha_t K}{R_t}$ . We discuss the erasure probability of the  $i$ -th bit in the message from an encoded symbol to a Type- $j$  variable

node ( $j = 1, 2, \dots, T-1$ ) from 2 aspects. Considering the first scenario that an encoded node is generated by  $\kappa$  non-Type- $T$  variable nodes. Let  $\eta_{\kappa, \tilde{\theta}}^{(j)}$  denote the probability that a chosen edge  $\tilde{\theta}$  satisfies the following conditions: (i) the connected variable node is of level- $j$  importance; (ii) the edge  $\tilde{\theta}$  is labeled with  $\theta_0$ ; (iii) the degree of the connected encoded symbol is  $\kappa$ ; (iv) except the variable node connected by  $\tilde{\theta}$ , the set of the variable nodes neighbored with the encoded symbol contains  $a_t$  variable nodes of type- $t$  ( $t = 1, 2, \dots, T-1$ ); (v) except for edge  $\tilde{\theta}$ , other  $\kappa-1$  edges are labeled  $\theta_i$  ( $i = 1, 2, \dots, \kappa-1$ ). Suppose  $\hat{t} = \max\{t | a_t > 0, t = 1, 2, \dots, T-1\}$  and  $r = \max(j, \hat{t})$ , we get the probability

$$\eta_{\kappa, \tilde{\theta}}^{(j)} = \omega_{\kappa} \Lambda_{\theta_0}^{(r)} \left\{ \Lambda_{\theta_1}^{(r)} \cdots \Lambda_{\theta_{a_1}}^{(r)} \right\} \cdots \left\{ \Lambda_{\theta_{\sum_{\beta=1}^{r-1} a_{\beta}+1}}^{(r)} \cdots \Lambda_{\theta_{\kappa-1}}^{(r)} \right\}, \quad (4)$$

where  $\sum_{\beta=1}^r a_{\beta} = \kappa-1$ . Suppose  $a_0 = 0$ , the probability that the  $i$ -th bit in the message from an encoded symbol to an Type- $j$  variable node in edge  $\tilde{\theta}$  is not erased at the  $\tau$ -th iteration is

$$\prod_{t=1}^r q_t^{a_t} \prod_{m=\sum_{\delta=0}^{t-1} a_{\delta}+1}^{\sum_{\delta=1}^t a_{\delta}} (1 - u_{t,i+\theta_0-\theta_m}^{(\tau-1)}).$$

However, if a given encoded symbol is generated by at least one type- $T$  input symbol, the children of the AND-node would not shift at all. Thus, the probability can be expressed by  $\prod_{t=1}^T q_t^{a_t} (1 - u_{t,i}^{(\tau-1)})^{a_t}$ . Thus, we obtain the erasure probability of the  $i$ -th bit of a message from an encoded symbol to a Type- $j$  variable nodes ( $j = 1, 2, \dots, T-1$ ) after  $\tau$  iterations as in (5).

$$\begin{aligned} v_{j,i}^{(\tau)} &= \sum_{\kappa} \omega_{\kappa} \sum_{a_1+\dots+a_{T-1}=\kappa-1} \binom{a_1}{\kappa-1} \binom{a_2}{\kappa-1-a_1} \\ &\quad \cdots \binom{a_{T-2}}{\kappa-1-\dots-a_{T-3}} \sum_{\theta_0, \theta_1, \dots, \theta_{a_{T-1}}} \eta_{\kappa, \tilde{\theta}}^{(j)} \\ &\quad \times \left\{ 1 - \prod_{t=1}^{T-1} q_t^{a_t} \prod_{m=\sum_{\delta=0}^{t-1} a_{\delta}+1}^{\sum_{\delta=1}^t a_{\delta}} (1 - u_{t,i+\theta_0-\theta_m}^{(\tau-1)}) \right\} \\ &\quad + 1 - \sum_{\kappa} \omega_{\kappa} \left\{ \sum_{a_1+\dots+a_{T-1}=\kappa-1} \binom{a_1}{\kappa-1} \binom{a_2}{\kappa-1-a_1} \right. \\ &\quad \cdots \binom{a_{T-1}}{\kappa-1-\dots-a_{T-2}} \prod_{t=1}^T q_t^{a_t} (1 - u_{t,i}^{(\tau-1)})^{a_t} \\ &\quad - \sum_{a_1+\dots+a_{T-1}=\kappa-1} \binom{a_1}{\kappa-1} \binom{a_2}{\kappa-1-a_1} \\ &\quad \left. \cdots \binom{a_{T-2}}{\kappa-1-\dots-a_{T-3}} \prod_{t=1}^{T-1} q_t^{a_t} (1 - u_{t,i}^{(\tau-1)})^{a_t} \right\}. \quad (5) \end{aligned}$$

To simplify the notation, we denote  $\hat{u}_{t,i}^{(\tau)} = \sum_{\theta_m=0}^{\theta_{max}^{(r)}} \Lambda_{\theta_m}^{(t)} u_{t,i-\theta_m}^{(\tau-1)}$ . Substituting (4) into (5), we have

$$\begin{aligned} v_{j,i}^{(\tau)} &= 1 - \sum_{\theta_0=0}^{\theta_{max}^{(r)}} \Lambda_{\theta_0}^{(j)} \omega \left( \sum_{t=1}^r \left\{ q_t (1 - \hat{u}_{t,i+\theta_0}^{(\tau-1)}) \right\} \right) \\ &\quad + \omega \left( \sum_{t=1}^{T-1} q_t (1 - u_{t,i}^{(\tau-1)}) \right) - \omega \left( 1 - \sum_{t=1}^T q_t u_{t,i}^{(\tau-1)} \right). \quad (6) \end{aligned}$$

Furthermore, we consider the erasure probability that the  $i$ -th bit of a message from an encoded symbol to a Type- $T$  variable node. We take a Type- $T$  variable node as a root node in the and-or tree. It is straightforward that the child AND-node of the OR-node does not shift at all. Hence, we have

$$\begin{aligned} v_{T,i}^{(\tau)} &= 1 - \sum_{\kappa} \omega_{\kappa} \prod_{i=1}^T q_i^{a_i} (1 - u_{t,i}^{(\tau-1)})^{a_i} \\ &= 1 - \omega \left( 1 - \sum_{i=1}^T q_i u_{t,i}^{(\tau-1)} \right). \quad (7) \end{aligned}$$

Finally, the erasure probability of the  $i$ -th bit of the message received by the Type- $j$  variable node at the  $\tau$ -th iteration is obtained as

$$\mathcal{Q}_{j,i}^{(\tau)} = \Phi^{(j)}(x_{j,i}^{(\tau)}) \Psi_j(v_{j,i}^{(\tau)}). \quad (8)$$

## 4.2 | WZ-UEP codes with two importance classes

In this section, we investigate the WZ-UEP codes with  $T = 2$  different importance levels. The fraction of the more important input symbols (MIS) is  $\alpha_1 = 0.1$ . The remains are considered as the less important symbols (LIS). LDPC codes of rates  $R_1 = 0.5$  and  $R_2 = 0.9$  are used for pre-coding the more and less important input symbols, respectively. The parameters related to the LDPC codes are given as (i)  $\Phi^{(1)}(x) = \Phi^{(2)}(x) = x^3$ ; (ii)  $\Gamma^{(1)}(x) = x^6$ ; (iii)  $\Gamma^{(2)}(x) = x^{30}$ . The degree distribution function of the encoded symbols in [4] is considered and given by

$$\begin{aligned} \Omega(x) &= 0.0080x + 0.4936x^2 + 0.1662x^3 \\ &\quad + 0.0726x^4 + 0.0826x^5 \\ &\quad + 0.0561x^8 + 0.0372x^9 + 0.0556x^{19} \\ &\quad + 0.0250x^{64} + 0.0031x^{66}. \quad (9) \end{aligned}$$

Assume the maximum bit-shift amount of the Type-1 variable nodes  $\theta_{max}^{(1)} = 0, 3, 6$ . Given  $P_1 = \frac{b_1}{n} = \frac{b_1}{\frac{\alpha_1 K}{R_1} + \frac{(1-\alpha_1)K}{R_2}}$  and

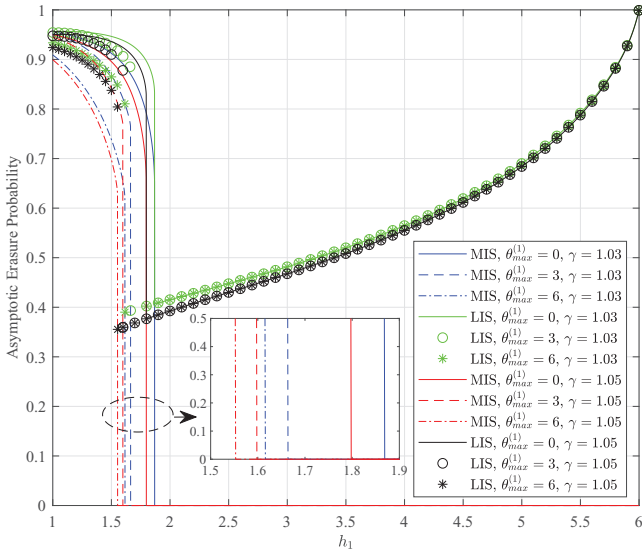


FIGURE 5 The asymptotic erasure probability versus  $b_1$

$P_1 \times \frac{\alpha_1 K}{R_1} + P_2 \times \frac{(1-\alpha_1)K}{R_2} = 1$ , we get  $b_2 = 1 + \frac{\alpha_1 R_2 (1-b_1)}{(1-\alpha_1)R_1}$ . As  $b_1 \geq b_2 > 0$ , we have  $b_1 \in [1, 6]$ . Figure 5 shows the erasure probability versus the value of  $b_1$ . As can be seen in this figure, the erasure probability of the MIS part continually decreases with the increasing of  $b_1$ <sup>1</sup>. The erasure probability of the LIS part also decreases with  $b_1$ , when  $b_1$  is relatively small. However, when  $b_1$  increases to a certain point ( $b_1 = 1.869$  when  $\gamma = 1.03$ ,  $\theta_{max}^{(1)} = 0$ ), the erasure probability of the LIS part begins to increase with the increasing of  $b_1$ . The reason is that with the increasing of  $b_1$  (hence decreasing of  $b_2$ ), LISs have a lower opportunity to be selected when generating encoded symbols. Moreover, given  $\gamma = 1.03$  ( $\gamma = 1.05$ ), the codes with  $\theta_{max}^{(1)} = 6$  achieve minimum erasure probability at the turning point  $b_1 = 1.616$  ( $b_1 = 1.553$ ).

## 5 | SIMULATION RESULTS

In this section, files of size 2.841 MB and 8.79 MB are divided into input symbols of  $l = 8$  bits. The input symbols are further grouped into several parts. Each part contains  $K = 1000$  ( $K = 4000$ ) input symbols. In each part, the first  $\alpha_1 \times K = 0.1K$  ones are more important than others. We use the following codes to encode each group of input symbols.

- (i) WZ-UEP1 codes: proposed fountain codes using  $b_1 = 1.869$ ,  $R_1 = 0.5$ ,  $R_2 = 0.9$ ;
- (ii) WZ-UEP2 codes: proposed fountain codes using  $b_1 = 1.54$ ,  $R_1 = 0.4$ ,  $R_2 = 0.8$ ;
- (iii) ZD-EEP1 codes: the zigzag decodable codes in [7] using  $R = 0.9$  generate each encoded symbol independently adopting shift operation with  $P = 0.0380$ ;

<sup>1</sup> One of the purposes of Figure 5 is to illustrate and to find an optimal point where the LIS achieves the lowest asymptotic erasure probability. To locate this point more easily, we use a linear scale (instead of log scale) at the  $y$  axis.

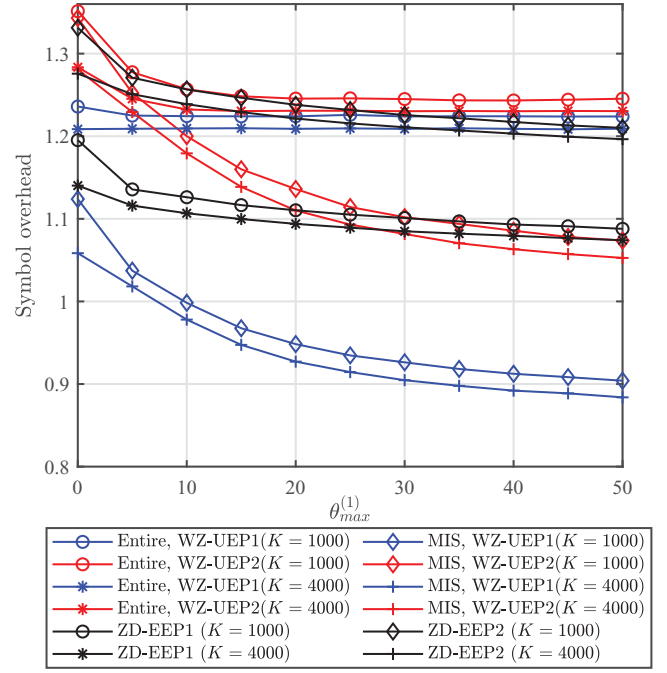


FIGURE 6 The symbol overhead of WZ-UEP codes and ZD-EEP codes versus the maximum bit-shift amount  $\theta_{max}^{(1)}$ .  $\theta_{max}^{(1)} \in \{0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50\}$ ,  $K \in \{1000, 4000\}$

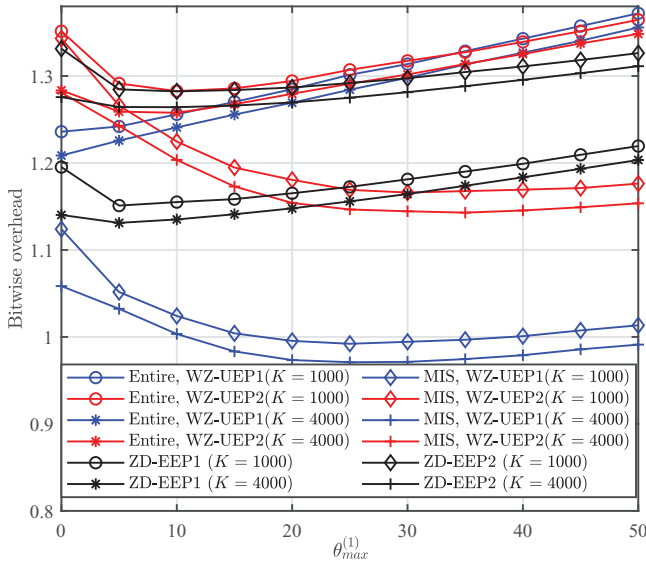
- (iv) ZD-EEP2 codes: the zigzag decodable codes in [7] using  $R = 0.8$  generate each encoded symbol adopting shift operation with probability  $P = 0.0303$ .

We find that the WZ-UEP1 code and ZD-EEP1 code achieve similar average bit numbers for each encoded symbol, for example, 8.968 and 8.967 when  $K = 1000$ ,  $\theta_{max}^{(1)} = 50$ ; 8.97 and 8.967 when  $K = 4000$ ,  $\theta_{max}^{(1)} = 50$ . Moreover, similar results can be achieved using WZ-UEP2 codes and ZD-EEP2 codes.

Figure 6 shows the symbol overhead versus the maximum bit-shift amount. As can be seen in this figure, the symbol overhead of ZD-EEP codes and WZ-UEP codes for the MIS part continually decreases with the increase of the maximum bit-shift amount. However, the symbol overhead for recovery of the entire input symbols using WZ-UEP1 codes ( $K = 1000$ ) and WZ-UEP2 codes ( $K = 1000$  and  $K = 4000$ ) decreases at the beginning. As the maximum bit-shift amount  $\theta_{max}^{(1)}$  increases to a certain point, the symbol overhead becomes stable. Especially, for the WZ-UEP1 codes ( $K = 4000$ ), the symbol overhead does not change with the increase of the maximum bit-shift amount  $\theta_{max}^{(1)}$ . Moreover, given the maximum bit-shift amount, the WZ-UEP codes achieve better performance for MIS part than for entire input symbols with respect to the symbol overhead. In contrast to the codes with no bit shift operation, a slightly bit shift leads to comparative improvement on symbol overhead.

Further, we record the received bit numbers and define bitwise overhead as the ratio of the number of received bits to the number of input bits ( $K \times l$ ). Figure 7 plots the bitwise overhead versus the maximum shift amount. The bitwise overhead for MIS part using WZ-UEP1 codes and WZ-UEP2 codes

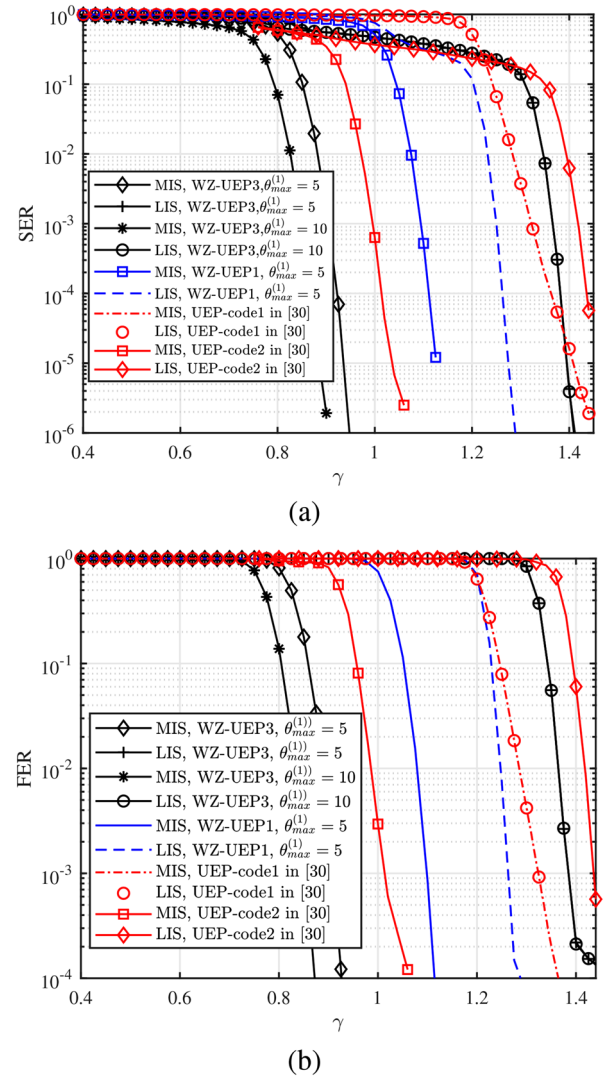




**FIGURE 7** The bitwise overhead of WZ-UEP codes and ZD-EEP codes versus the maximum bit-shift amount  $\theta_{max}^{(1)}$ .  $\theta_{max}^{(1)} \in \{0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50\}$ ,  $K \in \{1000, 4000\}$

decreases with the increasing  $\theta_{max}^{(1)}$ , when  $\theta_{max}^{(1)}$  is relatively small. When the bitwise overhead for MIS part using WZ-UEP1 codes approaches the minimum point ( $\theta_{max}^{(1)} = 25$ ), it begins to increase with  $\theta_{max}^{(1)}$  slowly. Similar results can be seen by using WZ-UEP2 codes. The minimum bitwise overhead for recovering the MIS part is achieved at the point  $\theta_{max}^{(1)} = 30$  when  $K = 1000$  ( $\theta_{max}^{(1)} = 35$  when  $K = 4000$ ). The bitwise overhead for recovery of the entire input symbols using WZ-UEP1 codes monotonically increases with  $\theta_{max}^{(1)}$ . However, the WZ-UEP2, ZD-EEP1, and ZD-EEP2 codes, respectively, achieve the minimum bitwise overhead for recovery of the entire input symbols at the points  $\theta_{max}^{(1)} = 10$ ,  $\theta_{max}^{(1)} = 5$  and  $\theta_{max}^{(1)} = 10$ . Afterwards, the bitwise overhead for recovery of the entire input symbols increases slowly.

From the results discussed above, it can be concluded that our proposed WZ-UEP code possesses an UEP property and that a better symbol overhead of the MIS can be achieved with a larger maximum bit-shift number. However, in the encoding process, we select bit-shift number according to a predefined bit-shift distribution. To reduce the number of bits in the encoded symbols, we further subtract the minimum one from the selected bit-shift numbers. The selection and subtraction operations induce extra delay compared with the codes without bit-shift operations. Moreover, the bit-shift operation increases the number of bits in the encoded symbols. While the codes without the bit-shift operations are decoded iteratively based on one simple symbol tanner graph, our proposed WZ-UEP code needs to be decoded based on a number of interconnected bitwise tanner graphs. As a result, the overall structure of the bitwise tanner graph used for the decoding of our proposed WZ-UEP code is more complex and hence requires a longer decoding delay.



**FIGURE 8** (a) SER and (b) FER versus the ratio of the number of received encoded symbols to the number of input symbols  $\gamma$ . MIS: SER/FER of the MIS. LIS: SER/FER of the LIS

We further simulate the symbol error rate (SER) and frame error rate (FER) of WZ-UEP1 code using  $K = 4000$ ,  $\theta_{max}^{(1)} = 5$ . For comparison, we also simulate the UEP fountain code in [30] which precodes the MIS with a (3,6)-regular LDPC code of  $R_1 = 0.5$  and the LIS with a (3,30)-regular LDPC of  $R_2 = 0.9$ . We name it as “UEP-code1”. The code uses  $\Omega(x)$  as the degree distribution of the encoded symbols. As shown in Figure 8, due to the avalanche effect caused by the degree distribution  $\Omega(x)$ , the UEP-code1 loses its UEP property. Also, our WZ-UEP1 code outperforms the UEP-code1 in terms of SER and FER performance. In Figure 8, we redraw the SER/FER curves of another UEP fountain code in [30] (denoted here as “UEP-code2”) which uses  $\Omega_1(x) = 0.1448x + (1.0 - 0.1448)x^2$  as the degree distribution of the encoded symbols. The UEP-code2 pre-codes the MIS with a  $R_1 = 0.5(\lambda_1, \rho_1)$ -irregular LDPC code and the LIS with a  $R_2 = 0.9(3, 30)$ -regular LDPC code, where  $\lambda^{(1)}(x) = 0.409x + 0.202x^2 + 0.0768x^3 + 0.1971x^6 + 0.1151x^7$  and  $\rho^{(1)}(x) = x^5$ . Compared with our



proposed WZ-UEP1 code and the UEP-code1, the UEP-code2 achieves the best SER/FER performance for the MIS part but the worst performance for LIS part. Next, we increase the value of  $b_1$  in WZ-UEP1 code from 1.869 to 2.437 and name the code as “WZ-UEP3 code”. The aim is to increase the selection probability of MIS and hence to improve the SER/FER performance of MIS. Figure 8 shows that the WZ-UEP3 codes with  $\theta_{max}^{(1)} = 5$  and 10 possess UEP properties and outperform UEP-code2 in terms the SER/FER performance. Compared to the WZ-UEP3 code with  $\theta_{max}^{(1)} = 5$ , the WZ-UEP3 code with  $\theta_{max}^{(1)} = 10$  achieves better MIS SER/FER without degrading the error performance of the LIS.

## 6 | CONCLUSION

In this paper, we proposed a novel weighted zigzag decodable fountain code to recover information of different importance levels gradually. This is achieved by (i) pre-coding input symbols of different importance levels using LDPC codes of different code rates; (ii) performing an unequal selection of variable nodes with different importance levels; (iii) allowing the selected variable nodes to shift certain bits according to their importance levels before performing exclusive-or operation to form encoded symbols. Moreover, we analyzed the proposed WZ-UEP codes by using AND-OR tree evaluation and formulated the equations for calculating erasure probabilities. We further obtained the erasure probability of WZ-UEP codes with two importance classes and the optimal parameter of  $b_1$ . Simulation results showed that the proposed scheme possessed unequal error protection properties. The WZ-UEP code is capable of improving the performance for recovering the MIS part and the entire input symbols by using bit-shift operations with respect to the symbol overhead.

However, compared with traditional LT-like UEP codes, our proposed WZ-UEP codes make use of LDPC codes of different code rates. Such a requirement induces extra complexity and delay in the encoding and decoding processes. Moreover, the bit-shift operation increases the average bit-length of the encoded symbols and the overall size of the bitwise tanner graph. It also indirectly increases the number of iterations in the decoding process. In the future, we will attempt to improve the encoding process by, for example, considering polar codes with UEP properties. We will also investigate novel ways to reduce the decoding delay.

## ACKNOWLEDGEMENTS

This research was supported by the National Key Research and Development Program of China (Grant No. 2019YFB1405803), the Fundamental Research Funds for the Central Universities (Grant Nos. N2017016, N2017011), and the National Natural Science Foundation of China (Grant Nos. 61603082, 61977014, 61902056 and 61902057).

## CONFLICT OF INTEREST

The authors have declared no conflict of interest.

## DATA AVAILABILITY STATEMENT

Data available on request from the authors.

## ORCID

Yuli Zhao  <https://orcid.org/0000-0001-7298-7463>

## REFERENCES

- Mackay, D.J.C.: Fountain codes. *IEE Proc.: Commun.* 152(6), 1062–1068 (2005)
- Luby, M.: LT codes. In: 43rd Annual IEEE Symposium on Foundations of Computer Science, pp. 271–282. IEEE, Piscataway (2002)
- Zhao, Y., Lau, F.C.M., Zhu, Z., Yu, H.: Scale-free Luby transform codes. *Int. J. Bifurcation Chaos* 22(4), 1250094 (2012)
- Shokrollahi, A.: Raptor codes. *IEEE Trans. Inf. Theory* 52(6), 1250094 (2012)
- Cassuto, Y., Shokrollahi, A.: Online fountain codes with low overhead. *IEEE Trans. Inf. Theory* 61(6), 3137–3149 (2015)
- Zhao, Y., Zhang, Y., Lau, F.C.M., Yu, H., Zhu, Z.: Improved online fountain codes. *IET Commun.* 12(18), 2297–2304 (2018)
- Takayuki, N.: Zigzag decodable fountain codes. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* E100A(8), 1693–1704 (2017)
- Anglano, C., Gaeta, R., Grangetto, M.: Exploiting rateless codes in cloud storage systems. *IEEE Trans. Parallel Distrib. Syst.* 26(5), 1313–1322 (2015)
- Zhang, L., Li, Y., Cheng, P., Li, M.: A layer-mixed FEC scheme for scalable media transmission over mobile TV services. *IEEE Trans. Broadcast.* 63(2), 309–320 (2017)
- Schwarz, H., Marpe, D., Wiegand, T.: Overview of the scalable video coding extension of the H.264/AVC standard. *IEEE Trans. Circuits Syst. Video Technol.* 17(9), 1103–1120 (2007)
- Boyce, J.M., Ye, Y., Chen, J., Ramasubramanian, A.K.: Overview of SHVC: Scalable extensions of the high efficiency video coding (HEVC) standard. *IEEE Trans. Circuits Syst. Video Technol.* 26(1), 20–34 (2016)
- Gavrovskaya, A.M., Milivojevic, M.S., Zajic, G.: Analysis of SVT-AV1 format for 4k video delivery. In: 28th Telecommunications Forum (TELFOR 2020), pp. 1–4. IEEE, Piscataway (2020)
- Karimzadeh, M., Vu, M.: Metrics and algorithms for designing convolutional codes with unequal error protection. *IEEE Trans. Veh. Technol.* 70(11), 11169–11183 (2021)
- Karimzadeh, M., Vu, M.: Short blocklength priority-based coding for unequal error protection in the AWGN channel. In: IEEE Global Communications Conference (IEEE GLOBECOM). IEEE, Piscataway (2019)
- Masnick, B., Wolf, J.: On linear unequal error protection codes. *IEEE Trans. Inf. Theory* 13(4), 600–607 (1967)
- Rahnavard, N., Pishro-Nik, H., Fekri, F.: Unequal error protection using partially regular LDPC codes. *IEEE Trans. Commun.* 55(3), 387–391 (2007)
- Rahnavard, N., Fekri, F.: New results on unequal error protection using LDPC codes. *IEEE Commun. Lett.* 10(1), 43–45 (2006)
- Buch, G., Burkert, F.: Concatenated Reed-Muller codes for unequal error protection. *IEEE Commun. Lett.* 3(7), 202–204 (1999)
- Liang, H., Liu, A., Zhang, Y., Cheng, F.: Rateless transmission of polar codes with information unequal error protection. *IET Commun.* 13(12), 1721–1727 (2019)
- Hadi, A., Alsusa, E., Al-Dweik, A.: Information unequal error protection using polar codes. *IET Commun.* 12(8), 956–961 (2018)
- Rahnavard, N., Vellambi, B.N., Fekri, F.: Rateless codes with unequal error protection property. *IEEE Trans. Inf. Theory* 53(4), 1521–1532 (2007)
- Sejtinović, D., Vukobratović, D., Doufexi, A., Senk, V., Piechocki, R.J.: Expanding window fountain codes for unequal error protection. *IEEE Trans. Commun.* 57(9), 2510–2516 (2009)
- Ahmad, S., Hamzaoui, R., Al-Akaidi, M.M.: Unequal error protection using fountain codes with applications to video communication. *IEEE Trans. Multimedia* 13(1), 92–101 (2011)

24. Deng, K., Yuan, L., Wan, Y., Pan, J.: Expanding window fountain codes with intermediate feedback over BIAWGN channels. *IET Commun.* 12(8), 914–921 (2018)
25. Huang, J., Fei, Z., Cao, C., Xiao, M., Jia, D.: On-line fountain codes with unequal error protection. *IEEE Commun. Lett.* 21(6), 1225–1228 (2017)
26. Xu, X., Zeng, Y., Guan, Y., Yuan, L.: Expanding-window BATS code for scalable video multicasting over erasure networks. *IEEE Trans. Multimed.* 20(2), 271–281 (2018)
27. Duan, Y., Ding, L., Yang, F., Qian, L., Zhi, C.: UEP online fountain codes with sequential window strategy. In: *IEEE International Conference on Communications in China (ICCC)*, pp. 899–904. IEEE, Piscataway (2020)
28. Wu, S., Guan, Q., Miao, Z.: A new class of LT-based UEP rateless codes for satellite image data transmission. *AEU Int. J. Electron. Commun.* 122, 153256 (2020)
29. Wang, Y., Liu, R.: Enhanced unequal error protection coding scheme of luby transform codes. *IET Commun.* 9(1), 33–41 (2015)
30. Yuan, L., Li, H., Wan, Y.: A novel UEP fountain coding scheme for scalable multimedia transmission. *IEEE Trans. Multimed.* 18(7), 1389–1400 (2016)
31. Arslan, S.S., Cosman, P.C., Milstein, L.B.: Generalized unequal error protection LT codes for progressive data transmission. *IEEE Trans. Image Process.* 21(8), 3586–3597 (2012)
32. Sung, C.W., Gong, X.: A zigzag-decodable code with the MDS property for distributed storage systems. In: *2013 IEEE International Symposium on Information Theory*, pp. 341–345. IEEE, Piscataway (2013)
33. Jun, B., Yang, P., No, J.S., Park, H.: New fountain codes with improved intermediate recovery based on batched zigzag coding. *IEEE Trans. Commun.* 65(1), 23–36 (2017)
34. Dai, M., Sung, C.W., Wang, H., Gong, X., Lu, Z.: A new zigzag-decodable code with efficient repair in wireless distributed storage. *IEEE Trans. Mob. Comput.* 16(5), 1216–1230 (2017)

**How to cite this article:** Zhang, Y., Zhao, Y., Lau, F.C.M., Zhang, B., Zhu, Z., Yu, H.: Weighted zigzag decodable fountain codes for unequal error protection. *IET Commun.* 16, 2082–2090 (2022).  
<https://doi.org/10.1049/cmu2.12462>