A Class of Rateless Reed-Solomon Codes with Near-Linear Computational Complexities

Leilei Yu, Sian-Jheng Lin, *Member, IEEE,* and Yunghsiang S. Han, *Fellow, IEEE*

***Abstract*—This paper proposes a class of rateless Reed-Solomon (RLRS) codes with near-linear encoding/decoding complexities. Like fountain codes, the RLRS codes can generate a reasonably large number of encoded packets in packet-level transmissions. Furthermore, the RLRS codes are maximum distance separable (MDS) codes that always maintain zero reception overhead. In the proposed RLRS codes, the preservative field extensions are realized through Cantor's field tower, which avoids searching some quadratic irreducible polynomials as in the prior RLRS codes based on Cauchy generator matrices. Additionally, the proposed RLRS codes are based on Vandermonde generator matrices, whereby the LCH transforms, a variant of fast Fourier transforms (FFTs) over binary extension fields, can be employed to reduce the encoding/decoding complexity. To further improve computational efficiency, this paper also proposes a scheduling scheme for the LCH transforms to generate encoded packets on demand, instead of generating packets whose number must be a power of two. Analysis shows that compared to the prior approach, the used field tower leads to a lower speed of computational complexity growth caused by field extensions. In addition, with the total number of source packets to be transmitted being k, analysis shows that the proposed RLRS codes have the encoding/decoding complexity $\mathcal{O}(\log_2 k)$ per source packet, superior to $\mathcal{O}(k)$ in the prior approach.**

***Index Terms*—Fountain codes, rateless codes, Reed-Solomon codes, zero reception overhead, computational complexities.**

I. Introduction

INTERNET is a real-world model of the binary erasure channel (BEC), in which data is specified to be transmitted in the form of packets [1], [2]. Ensuring reliable transmission of packets over the Internet has been a hot research topic in recent decades. Among various topics, Automatic Repeat Request (ARQ), which sends a re-transmission request once an error has been detected, is a typical strategy used in communication protocols, such as the ubiquitous TCP/IP [3]. However, ARQ is unsuitable for many scenarios, such as data transmission over heavily impaired channels or multicasting in which data are transmitted to multiple destinations. To solve these issues, coding-based schemes are developed [4]–[11].

The coding-based schemes encode k source packets into $n = k + t$ packets, and then all source packets can be decoded (recovered) from any $\bar{n}$ $(\geq k)$ out of n encoded packets. Note that any decoding mentioned in this paper defaults to erasure decoding. Conventionally, k/n is called the code rate and $\bar{n} - k$ is called the reception overhead. To ensure reliable data transmissions, the code rate is generally determined according to prior knowledge of channel quality. In 2002, a digital fountain approach was introduced [8] to automatically adapt the channel without any channel knowledge. Based on this approach, the sender can continuously send encoded packets until receiving an acknowledgment message from the receiver, i.e., n can be theoretically infinite. The coding scheme based on this approach is called a fountain code or rateless code, and it is particularly suitable for channels with unstable quality. LT codes were the first practical realization of rateless codes [7], and Raptor codes were their significant theoretical and practical improvement [9]. In particular, Raptor codes have high computational efficiency due to their per-source packet encoding and decoding complexities of constant orders. Nevertheless, it is well known that the above codes cannot guarantee zero reception overhead [12], [13]. For several applications, such as P2P file sharing, the bandwidth is typically the most constrained resource [14], making codes with zero reception overhead more preferable. To design rateless codes with zero reception overhead, Reed-Solomon (RS) codes were considered a potential candidate [15].

RS codes are a well-known class of maximum distance separable (MDS) codes, which always offer zero reception overhead [16]. Despite their advantage in terms of reception overhead, they are constructed over finite fields so that the total number of encoded packets cannot exceed the size of the chosen finite field [17]. In other words, the code rate of RS codes is bounded by the field size, resulting in their inability to be used as rateless codes like the ones mentioned above. To address this issue, [18], [19] suggested cycling the encoded packets generated by RS codes, but the advantage of zero reception overhead is lost. In [15], by introducing preservative field extensions, the authors proposed rateless RS (RLRS) codes to produce some encoded packets exceeding the size of the chosen finite field while maintaining zero reception overhead. The RLRS codes proposed in [15] are based on Cauchy generator matrices, leading to per-source packet encoding and decoding complexities of both $\mathcal{O}(k)$, which are much higher than those of the above rateless codes.

To reduce the computational complexities of RLRS codes, some fast algorithms for RS codes could be utilized. Indeed, the conventional RS encoding and decoding algorithms have the total computational complexities of quadratic orders. By using fast polynomial arithmetic [20], one can encode and decode RS codes in $\mathcal{O}(n \log_2^2 n \log_2 \log_2 n)$ [21]. In [22], based on fast Walsh-Hadamard transforms, Didier proposed

This work was supported by the National Key Research and Development Program of China under Grant 2022YFA1004902, the National Natural Science Foundation of China under Grant 62071446. (Corresponding author: Sian-Jheng Lin)

L. Yu and Y. S. Han are with the Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, Shenzhen, China (e-mail: yuleilei@uestc.edu.cn, yunghsiangh@gmail.com). S.-J. Lin is an independent researcher (e-mail: sjhenglin@gmail.com).

RS encoding and decoding algorithms with the computational complexities of $\mathcal{O}(n\log_2 n)$ and $\mathcal{O}(n\log_2^2 n)$, respectively. Furthermore, Lin et al. in [23], [24] proposed a variant of fast Fourier transforms (FFTs) over binary extension fields, termed LCH transforms, and then extended Didier's work based on them to develop $\mathcal{O}(n\log_2 k)$ encoding and $\mathcal{O}(n\log_2 n)$ decoding algorithms [25]. The above algorithms form the lowest computational complexities for RS codes by far. All of the above motivates us to propose a class of RLRS codes that offers low computational complexities.

In this paper, the proposed RLRS codes are based on the following ideas. To begin with, a total of k source packets are encoded using an RS code over the finite field $\mathbb{F}_{2^M}$, capable of generating up to 2^M encoded packets. When the channel is bad, such that the received packets are not enough for decoding, a conventional way is that the sender uses RS codes over a larger finite field. As the new RS codes use a different finite field, the previously encoded packets cannot be used in decoding. However, according to [15], the packets generated by the old RS code, i.e., $(2^M, k)$ RS code over $\mathbb{F}_{2^M}$, is equivalent to the first 2^M packets generated by the new RS code, i.e., $(2^{2M}, k)$ RS code over $\mathbb{F}_{2^{2M}}$, if $\mathbb{F}_{2^{2M}}$ is a preservative field extension of $\mathbb{F}_{2^M}$. Based on this property, the previously encoded packets are still valid in decoding the $(2^{2M}, k)$ RS code. In packet-level transmissions, the preservative field extension can be used several times until a sufficient number of encoded packets are generated. In contrast to [15], the RLRS codes proposed in this paper use Vandermonde generator matrices to construct RS codes rather than the Cauchy generator matrices [15]. This leads to the result that fast polynomial arithmetic can be used in RLRS encoding and decoding algorithms to reduce computational complexities.

In this paper, simulations were also conducted to demonstrate the performance of different rateless codes, such as the proposed RLRS codes, Cauchy-based RLRS codes [15], and RaptorQ [26]. Note that the RaptorQ is the most advanced Raptor code, and it is implemented by an open-source library on Github [27]. The results show that the proposed RLRS codes perform significantly better than the Cauchy-based RLRS codes as the total number of source packets k or the packet loss probability ϵ increases. Particularly, the proposed RLRS code performs better than the RaptorQ when the number of source packets is small. This offers the possibility for the proposed RLRS codes to replace RaptorQ in short codes, as the former can always successfully decode while maintaining zero reception overhead, while the latter cannot. The main contributions of this paper are summarized as follows.

1) This paper proposes a new construction of the RLRS codes based on Vandermonde generator matrices. In particular, Cantor's field tower realizes the preservative field extensions that are the core of constructing RLRS codes. This avoids finding quadratic irreducible polynomials as in [15] and leads to a lower speed of computational complexity growth caused by field extensions than that in [15].
2) This paper presents an LCH-based encoding/decoding algorithm for the proposed RLRS codes.
3) This paper proposes a scheduling scheme for the LCH transform to generate encoded packets on demand, avoiding unexpected computation costs of the LCH transform that always produce burst outputs.
4) This paper analyzes the computational complexities of the proposed RLRS codes. The analysis results show that both of the per-source packet encoding and decoding complexities of the proposed RLRS codes are $\mathcal{O}(\log_2 k)$, improving the prior result $\mathcal{O}(k)$ based on the Cauchy-based RLRS codes. Simulations show that the performance of the proposed RLRS codes is competitive compared to that of other rateless codes.

The remainder of this paper is organized as follows: The necessary backgrounds, including finite fields and RS codes, are introduced in Sec. II. Section III proposes a new class of RLRS codes and presents the corresponding fast encoding/decoding algorithm. In Sec. IV, a scheduling scheme is presented to enable encoded packets to be generated on demand. Complexity analysis is given in Sec. V. Simulations and comparisons are given in Sec. VI. Finally, Sec. VII concludes this paper.

II. Preliminaries: Finite fields & Reed-Solomon codes

Throughout this paper, $\mathbb{N}$ denotes the set of whole numbers and $(\cdots i_2 i_1 i_0)_2$ denotes the binary representation of i, where $i = \sum_{j\in\mathbb{N}} i_j \cdot 2^j$. Additionally, any bold lowercase letter is considered a row vector by default. For any two integers $i < j$, $i, j \in \mathbb{N}$, we use $[i, j)$ to denote the set $\{i, i+1, \cdots, j-1\}$, and use $\mathbf{a}[i, j) = (a_i, a_{i+1}, \cdots, a_{j-1})$ to denote the sub-vector of $\mathbf{a} = (a_0, a_1, a_2, \cdots)$. Due to the focus on erasure channels, the decoding mentioned in this paper defaults to erasure decoding rather than error-correction decoding [28].

A. Binary extension fields

This section introduces the binary extension fields used in the proposed RLRS codes. Precisely, Cantor in [29] introduced a specific sequence $u_0, u_1, u_2, \cdots$ of elements from algebraic closure of binary field $\mathbb{F}_2$ to construct the field tower

$$\begin{aligned}
\mathbb{F}_{2^2} &:= \mathbb{F}_2[u_0]/(u_0^2 + u_0 + 1),\\
\mathbb{F}_{2^4} &:= \mathbb{F}_{2^2}[u_1]/(u_1^2 + u_1 + u_0),\\
\mathbb{F}_{2^8} &:= \mathbb{F}_{2^4}[u_2]/(u_2^2 + u_2 + u_1 u_0),\\
&\vdots
\end{aligned} \tag{1}$$

Unless otherwise stated, suppose that m is a positive integer and $M = 2^m$, then the field tower above leads to

$$\mathbb{F}_{2^M} = \mathbb{F}_2(u_0, u_1, \cdots, u_{m-1}). \tag{2}$$

The Cantor basis of the above $\mathbb{F}_{2^M}$ can be denoted by $\boldsymbol{v}_M = (v_0, v_1, \cdots, v_{M-1})$, where

$$v_i = u_{m-1}^{i_{m-1}} \cdots u_1^{i_1} u_0^{i_0} \quad \text{with } i = (i_{m-1}\cdots i_1 i_0)_2. \tag{3}$$

For instance, the Cantor basis of $\mathbb{F}_{2^8}$ is

$$\boldsymbol{v}_8 = (1, u_0, u_1, u_1u_0, u_2, u_2u_0, u_2u_1, u_2u_1u_0). \tag{4}$$

After that, each element in $\mathbb{F}_{2^M}$ can be denoted by $\omega_i = \sum_{j=0}^{M-1} i_j \cdot v_j$, where $i = (i_{M-1} \cdots i_1 i_0)_2$ and $0 \leq i < 2^M$. Subsequently, Lemma 1 and Lemma 2 can be easily obtained.

Lemma 1. *For any $i \in \mathbb{N}$, $0 = u_i^2 + u_i + v_{2^i-1}$.*

Proof. It is easy to derive from (1) and (3). □

Lemma 2. *For any $0 \leq i < 2^\ell$, $\omega_{i+2^\ell} = \omega_i + \omega_{2^\ell}$, where $i \in \mathbb{N}, \ell \in \mathbb{N}$.*

Proof. Let $i = (i_{\ell-1} \cdots i_1 i_0)_2$, then $\omega_{i+2^\ell} = \sum_{j=0}^{\ell-1} i_j \cdot v_j + v_\ell = \omega_i + \omega_{2^\ell}$. □

The above provides some basic information about Cantor's field tower. The following describes the definition of preservative field extension [15], which is the key to constructing RLRS codes and will be used in Sec. III.

Definition 1. *Assume that $a, b \in \mathbb{F}_{2^q}$ with $z = a \cdot b \in \mathbb{F}_{2^q}$, and*

$$\begin{aligned} \mathbf{a} &= (a_0, a_1, \cdots, a_{q-1}) \in \mathbb{F}_2^q \\ \mathbf{b} &= (b_0, b_1, \cdots, b_{q-1}) \in \mathbb{F}_2^q \\ \mathbf{z} &= (z_0, z_1, \cdots, z_{q-1}) \in \mathbb{F}_2^q \end{aligned} \tag{5}$$

denote the corresponding binary vectors of a, b, z, respectively. Let $Q = 2q$ and $a', b', b'' \in \mathbb{F}_{2^Q}$ that keep the binary vectors of a, b unchanged subject to a zero-padding

$$\begin{aligned} \mathbf{a}' &= (0, 0, \cdots, 0, a_0, a_1, \cdots, a_{q-1}) \in \mathbb{F}_2^Q \\ \mathbf{b}' &= (0, 0, \cdots, 0, b_0, b_1, \cdots, b_{q-1}) \in \mathbb{F}_2^Q \\ \mathbf{b}'' &= (b_0, b_1, \cdots, b_{q-1}, 0, 0, \cdots, 0) \in \mathbb{F}_2^Q, \end{aligned} \tag{6}$$

where $\mathbf{a}', \mathbf{b}', \mathbf{b}''$ respectively denote the binary vectors of a', b', b''. Then $\mathbb{F}_{2^Q}$ is said to be a preservative field extension of $\mathbb{F}_{2^q}$ if and only if the resultant binary vectors from $a' \cdot b'$ and $a' \cdot b''$ are respectively $(0, 0, \cdots, 0, z_0, z_1, \cdots, z_{q-1}) \in \mathbb{F}_2^Q$ and $(z_0, z_1, \cdots, z_{q-1}, 0, 0, \cdots, 0) \in \mathbb{F}_2^Q$.

B. Reed-Solomon codes

The conventional viewpoint of RS codes is to map the coefficient vector of a polynomial to the corresponding evaluation vector [16]. More precisely, given a k-element information vector $\mathbf{c} = (c_0, c_1, \cdots, c_{k-1}) \in \mathbb{F}_{2^M}^k$, where $k < 2^M$, the corresponding polynomial is defined as $c(x) = \sum_{i=0}^{k-1} c_i x^i$. Then the (n, k) RS codeword for $\mathbf{c}$ is the evaluation vector of $c(x)$ at n distinct points, i.e.,

$$\mathbf{r} = (c(\omega_0), c(\omega_1), \cdots, c(\omega_{n-1})). \tag{7}$$

Clearly, $k < n \leq 2^M$ due to the fact that $\mathbb{F}_{2^M}$ has at most 2^M distinct points. For convenience, (7) can be rewritten as the following matrix form

$$\mathbf{r}^{\mathrm{T}} = G \cdot \mathbf{c}^{\mathrm{T}}, \tag{8}$$

where G is a generator matrix and has the Vandermonde form

$$G = \begin{pmatrix} 1 & \omega_0 & \omega_0^2 & \cdots & \omega_0^{k-1} \\ 1 & \omega_1 & \omega_1^2 & \cdots & \omega_1^{k-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_{n-1} & \omega_{n-1}^2 & \cdots & \omega_{n-1}^{k-1} \end{pmatrix}. \tag{9}$$

For the systematic RS codes, the information vector $\mathbf{c}$ to codeword vector $\mathbf{r}$ is reorganized as follows.

(i) Determine the coefficient vector of $c(x)$ by making $\{c(\omega_i) = c_i\}_{i=0}^{k-1}$, where $c(x)$ is of degree less than k.
(ii) Calculate the codeword $\mathbf{r}$ according to $\mathbf{r}^{\mathrm{T}} = G \cdot \mathbf{f}^{\mathrm{T}}$, where $\mathbf{f}$ is the coefficient vector of $c(x)$ determined in the previous step.

The RS code above is systematic since the first step ensures that $\mathbf{r}[0, k) = \mathbf{c}$. In Step (i), the coefficient vector of $c(x)$ can be uniquely determined, and various methods can be used to achieve this. For instance, a plain method is to solve the Vandermonde linear system whose coefficient matrix consists of the first k rows in G. The plain method results in the computational complexity of $\mathcal{O}(k^2)$ if the lower-upper (LU) decomposition proposed in [30] is used. Lagrange interpolation is another commonly used method that can complete Step (i) with the complexity of $\mathcal{O}(k \log_2^2 k)$ if the base field supports FFTs [31].

It can be known from [31] that the FFT is an efficient tool to convert the coefficient vector of a polynomial into the corresponding evaluation vector, and inverse FFT (IFFT) performs the inverse process. In 2014, based on a special polynomial basis $\mathbb{X}$, termed Lin-Chung-Han (LCH) basis, the authors proposed a variant of FFT/IFFT over binary extension fields [23]. Similar to conventional FFT/IFFTs, k-point LCH/ILCH transforms achieves $\mathcal{O}(k \log_2 k)$ in both additive and multiplicative complexities [23]–[25]. To improve the computational complexities, this paper uses the LCH/ILCH transforms to complete Steps (i) and (ii) in the encoding process. In such, generating n output symbols with the k-point LCH/ILCH transform requires only the total computational complexity of $\mathcal{O}(n \log_2 k)$ [23]–[25].

The decoding process of systematic (n, k) RS codes essentially obtains the first k point-value pairs from any k point-value pairs of $c(x)$. The details are as follows.

(i) Determine the coefficient vector of $c(x)$ through the known k point-value pairs.
(ii) Calculate $\mathbf{r}[0, k) = (c(\omega_0), c(\omega_1), \cdots, c(\omega_{k-1}))$.

Similarly, the first step can be accomplished by solving a Vandermonde linear system or Lagrange interpolation, both of which have the same complexities as before. To reduce the computational complexities, Lin et al. in [23], [25] introduced a fast algorithm that employs the LCH/ILCH transforms mentioned above to complete the entire decoding process. This algorithm possesses the total computational complexity of $\mathcal{O}(n \log_2 n)$ and will be utilized in this paper. For convenience, some notations about LCH/ILCH transforms are defined below for later use.

Generally, if $\mathbf{f}_{\mathbb{X}}$ denotes the coefficient vector of $f(x) \in \mathbb{F}_{2^d}[x]$ in the LCH basis $\mathbb{X}$, where $\deg(f(x)) < k$ and d, k are both power of two, then we use $\mathrm{LCH}_{2^d}(\mathbf{f}_{\mathbb{X}}, k, \beta)$ to denote the LCH transform for computing the evaluation vector

$$\mathbf{r} = (f(\omega_0 + \beta), \cdots, f(\omega_{k-1} + \beta)), \tag{10}$$

where $\beta \in \mathbb{F}_{2^d}$. Briefly, $\mathbf{r} = \mathrm{LCH}_{2^d}(\mathbf{f}_{\mathbb{X}}, k, \beta)$. Furthermore, we use $\mathrm{ILCH}_{2^d}(\mathbf{r}, k, \beta)$ to denote the corresponding ILCH transform such that $\mathbf{f}_{\mathbb{X}} = \mathrm{ILCH}_{2^d}(\mathbf{r}, k, \beta)$.

TABLE I
DEFINITIONS OF SOME IMPORTANT NOTATIONS

Symbol	Definition
k	the total number of source packets.
B	individual packet size (in bits), and $B = 2^b$.
$\mathbb{F}_{2^M}$	the initial field, where $M = 2^m < B$.
$\mathbb{F}_{2^{2M}}$	the preservative field extension of $\mathbb{F}_{2^M}$.
$f_i(x)$	$f_i(x) \in \mathbb{F}_{2^M}[x]$ is the i-th initial evaluated polynomial with degree less than k, where $i \in [0, B/M)$.
$g_i(x)$	$g_i(x) = u_m \cdot f_{2i}(x) + f_{2i+1}(x) \in \mathbb{F}_{2^{2M}}[x]$ is the i-th new evaluated polynomial with degree less than k, where $i \in [0, B/(2M))$.
$\mathbf{f}_i$	$\mathbf{f}_i \in \mathbb{F}_{2^M}^k$ is the coefficient vector of $f_i(x)$, where $i \in [0, B/M)$.
$\mathbf{g}_i$	$\mathbf{g}_i = u_m \cdot \mathbf{f}_{2i} + \mathbf{f}_{2i+1} \in \mathbb{F}_{2^{2M}}^k$ is the coefficient vector of $g_i(x)$, where $i \in [0, B/(2M))$.
F	the matrix consisting of all initial coefficient vectors, i.e., $F = (\mathbf{f}_0^{\mathrm{T}}, \cdots, \mathbf{f}_{B/M-1}^{\mathrm{T}})$.
G	the generator matrix of (n, k) RS code given in (9).
R	the matrix consisting of polynomial evaluations, i.e., $R = G \cdot F$.

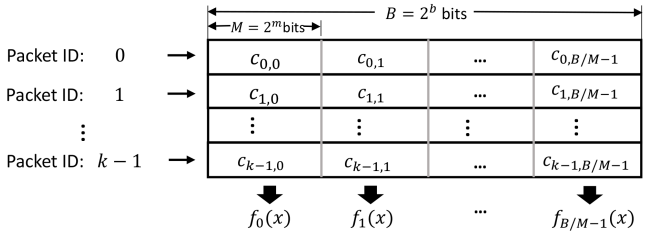

Fig. 1. Schematic diagram of k source packets.

III. PROPOSED RATELESS REED-SOLOMON CODES

Based on the field tower given in (1), this section proposes a new construction of the RLRS codes. The corresponding fast encoding and decoding algorithms are then presented. Some important notations used in this section are listed in TABLE I.

A. Proposed construction

To begin with, assume that there are k source packets to be transmitted, and k is a power of two (if not, it is easy to align by appending empty packets). Suppose that each packet has $B = 2^b > M$ bits; it can be considered as the packet has B/M symbols in $\mathbb{F}_{2^M}$. Recall the RS codes described in Sec. II-B, B/M evaluated polynomials of a systematic (n, k) RS code over $\mathbb{F}_{2^M}$, denoted by $f_0(x), f_1(x), \cdots, f_{B/M-1}(x)$, can be determined from the k source packets. Fig. 1 provides a schematic diagram where each column forms an information vector and generates an evaluated polynomial according to the first step of RS encoding. Formally, let the coefficient vector of $f_i(x)$ be denoted by $\mathbf{f}_i \in \mathbb{F}_{2^M}^k$, where $i \in [0, B/M)$. The following formula can be obtained according to the RS encoding described in Sec. II-B:

$$R = G \cdot F, \tag{11}$$

where G is shown in (9), $F = (\mathbf{f}_0^{\mathrm{T}}, \mathbf{f}_1^{\mathrm{T}}, \cdots, \mathbf{f}_{B/M-1}^{\mathrm{T}}) \in \mathbb{F}_{2^M}^{k \times \frac{B}{M}}$, and

$$R = \begin{pmatrix} f_0(\omega_0) & \cdots & f_{B/M-1}(\omega_0) \\ f_0(\omega_1) & \cdots & f_{B/M-1}(\omega_1) \\ \vdots & \ddots & \vdots \\ f_0(\omega_{n-1}) & \cdots & f_{B/M-1}(\omega_{n-1}) \end{pmatrix} \in \mathbb{F}_{2^M}^{n \times \frac{B}{M}}. \tag{12}$$

In such, n encoded packets in the (n, k) RS codes correspond to n rows in R. The upper part of Fig. 2 shows an example of $n = 2^M$. As mentioned earlier, the total number of encoded packets generated by the traditional RS code is limited since n cannot exceed the size of $\mathbb{F}_{2^M}$.

To increase the number of encoded packets beyond the size of the initial (current) finite field, one can generate $B/(2M)$ new evaluated polynomials with degree less than k according to the following formula

$$g_i(x) = u_m \cdot f_{2i}(x) + f_{2i+1}(x) \in \mathbb{F}_{2^{2M}}[x], \quad 0 \le i < B/(2M), \tag{13}$$

where $\mathbb{F}_{2^{2M}}$ comes from the field tower in (1). It is easy to see that the coefficient vector of each new polynomial $g_i(x)$, denoted by $\mathbf{g}_i$, can be calculated via

$$\mathbf{g}_i = u_m \cdot \mathbf{f}_{2i} + \mathbf{f}_{2i+1} \in \mathbb{F}_{2^{2M}}^k, \tag{14}$$

where $u_m \cdot \mathbf{f}_{2i}$ means shifting the binary representation of each element in $\mathbf{f}_{2i}$ to the high order M bits (zeros being padded at the low order M bits). Now, more polynomial evaluations shown in the lower part of Fig. 2 can be obtained because each new evaluated polynomial $g_i(x)$ is in $\mathbb{F}_{2^{2M}}[x]$. We next demonstrate that the $B/(2M)$ new polynomials evaluating at $x = \omega_0, \omega_1, \cdots, \omega_{2^M-1}$ yield the same results as the 2^M encoded packets shown in the upper part of Fig. 2. This results in all encoded packets being considered as constructed solely from $(2^{2M}, k)$ RS codes over $\mathbb{F}_{2^{2M}}$, such that all source packets can be reconstructed using only conventional $(2^{2M}, k)$ RS decoding.

Specifically, for $0 \le i < B/(2M)$ and $0 \le j < 2^M$, we have

$$g_i(\omega_j) = u_m \cdot f_{2i}(\omega_j) + f_{2i+1}(\omega_j) \in \mathbb{F}_{2^{2M}}. \tag{15}$$

Lemma 3 shows $\mathbb{F}_{2^{2M}}$ is a preservative field extension of $\mathbb{F}_{2^M}$. This indicates that in (15), the result of $f_{2i}(\omega_j)$ in $\mathbb{F}_{2^{2M}}$ is the same as that in $\mathbb{F}_{2^M}$, $f_{2i+1}(\omega_j)$ as well. The binary representation of each $g_i(\omega_j)$ in (15) is formed by simply concatenating the binary representations of $f_{2i}(\omega_j)$ and $f_{2i+1}(\omega_j)$ in $\mathbb{F}_{2^M}$. From the above, all packets in the upper part of Fig. 2 can be regarded as composed of the evaluation vectors of $g_0(x), g_1(x), \cdots, g_{B/(2M)-1}(x)$ at $x = \omega_0, \omega_1, \cdots, \omega_{2^M-1}$.

Lemma 3. *For any $i \in \mathbb{N}$, $\mathbb{F}_{2^{2^{i+1}}}$ is the preservative field extension of $\mathbb{F}_{2^{2^i}}$ when $\mathbb{F}_{2^{2^i}}$ and $\mathbb{F}_{2^{2^{i+1}}}$ are both from the field tower in* (1).

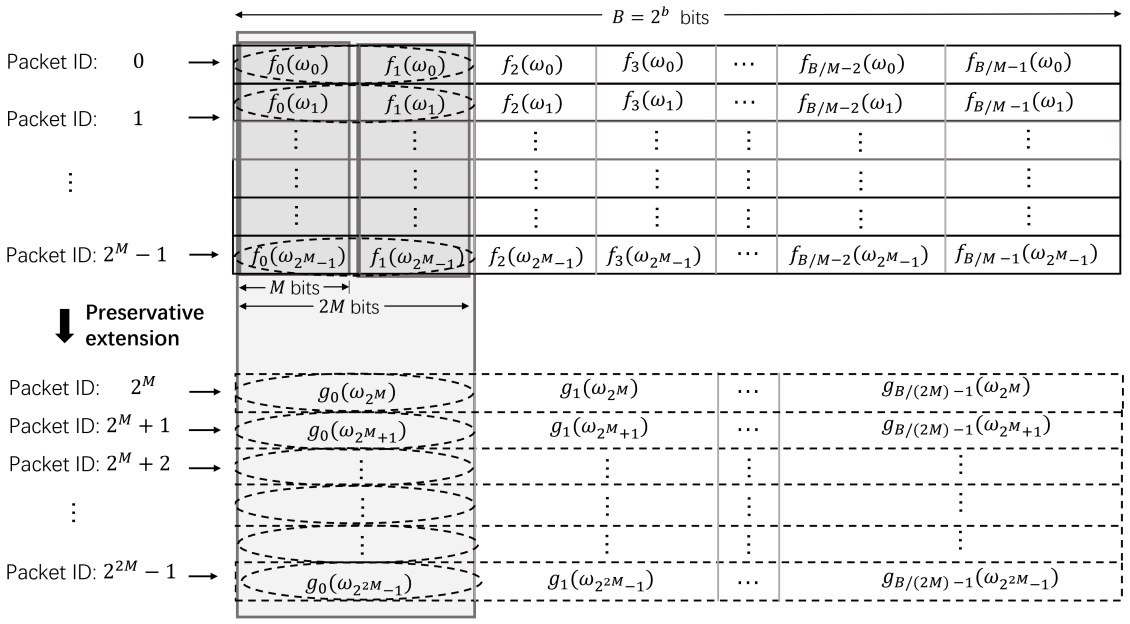

Fig. 2. Schematic diagram of proposed RLRS codes in packet-level transmission.

Proof. Consider that $a \in \mathbb{F}_{2^{2^i}}$ and $b \in \mathbb{F}_{2^{2^i}}$ with $z = a \cdot b \in \mathbb{F}_{2^{2^i}}$. As shown in Definition 1, let $a', b', b'' \in \mathbb{F}_{2^{2^{i+1}}}$ that keep the binary vectors of a, b unchanged subject to a zero-padding. From the field tower in (1), we have that

$$a' = 0 \cdot u_i + a, \quad b' = 0 \cdot u_i + b. \tag{16}$$

Let $b = b_0 v_0 + b_1 v_1 + \cdots + b_{2^i-1} v_{2^i-1}$. Then we have $b'' = b_0 v_{2^i} + b_1 v_{2^i+1} + \cdots + b_{2^i-1} v_{2^{i+1}-1}$. From (3), $b'' = b_0 u_i v_0 + b_1 u_i v_1 + \cdots + b_{2^i-1} u_i v_{2^i-1} = b \cdot u_i + 0$. It is easy to know that $a' \cdot b' = 0 \cdot u_i + z$ and $a' \cdot b'' = z \cdot u_i + 0$. Hence, the conditions in Definition 1 are met. This completes the proof. □

Essentially, the above method is to convert $R \in \mathbb{F}_{2^M}^{n \times \frac{B}{M}}, G \in \mathbb{F}_{2^M}^{n \times k}, F \in \mathbb{F}_{2^M}^{k \times \frac{B}{M}}$ in (11) into $R \in \mathbb{F}_{2^{2M}}^{n \times \frac{B}{2M}}, G \in \mathbb{F}_{2^{2M}}^{n \times k}, F \in \mathbb{F}_{2^{2M}}^{k \times \frac{B}{2M}}$ while maintaining the results of (11) in $\mathbb{F}_{2^M}$. After applying the field extension $b-m$ times, the matrices R, G, F in (11) can finally be converted to the finite field $\mathbb{F}_{2^B}$, such that the total number of encoded packets achieves 2^B. Here, B is the size of a single packet. Typically, B is much larger than M. For instance, the initial finite field of RS code is often set to $\mathbb{F}_{2^8}$ (i.e., $M = 8$, initial field symbols are bytes), and the size of encoded packet B is at least a few hundred bytes [32]. This means that the initial RS code can be extended many times. Importantly, the work [15] has demonstrated that it is enough to extend the initial RS code only once in practice. As stated in [15], when $M = 8$ and $k = 100$, the probability of decoding failure after one extension is less than 10^{-160}, even if the packet loss probability is 0.99 in BEC. This shows that the field expansion by more than twice is meaningless for practical applications.

B. Fast algorithms

This subsection presents fast encoding and decoding algorithms for the proposed RLRS codes by using the LCH/ILCH transforms introduced in Sec. II-B.

1) Encoding: As mentioned earlier, the LCH/ILCH transforms require that the input polynomial is represented over a special basis $\mathbb{X}$. Let $F_{\mathbb{X}} = (\mathbf{f}_{\mathbb{X},0}^{\mathrm{T}}, \mathbf{f}_{\mathbb{X},1}^{\mathrm{T}}, \cdots, \mathbf{f}_{\mathbb{X},B/M-1}^{\mathrm{T}}) \in \mathbb{F}_{2^M}^{k \times \frac{B}{M}}$ denote the matrix consisting of all initial coefficient vectors over the basis $\mathbb{X}$. The evaluation matrix R in (11) can be calculated through the LCH transforms with the input $F_{\mathbb{X}}$. Given k source packets of size $B = 2^b$ bits each, the encoding steps of the proposed RLRS code are as follows:

(i) Sender determines the coefficient vectors $\mathbf{f}_{\mathbb{X},0}^{\mathrm{T}}, \mathbf{f}_{\mathbb{X},1}^{\mathrm{T}}, \cdots, \mathbf{f}_{\mathbb{X},B/M-1}^{\mathrm{T}}$ from the k source packets. Precisely, let the k source packets consist of B/M information vectors $\mathbf{c}_0, \mathbf{c}_1, \cdots \mathbf{c}_{B/M-1}$, where each $\mathbf{c}_i$ is a column in Fig. 1. Then calculate

$$\mathbf{f}_{\mathbb{X},i} = \mathrm{ILCH}_{2^M}(\mathbf{c}_i, k, 0), \quad \text{for } i \in [0, B/M). \tag{17}$$

(ii) Let ℓ take $1, 2, \cdots, \frac{2^M}{k} - 1$ in turn, the sender calculates

$$\mathbf{r}_i[\ell k, (\ell+1)k) = \mathrm{LCH}_{2^M}(\mathbf{f}_{\mathbb{X},i}, k, \omega_{\ell k}), \quad i \in [0, B/M),$$

and then sends each generated encoded packet to the receiver. The process is terminated once the acknowledgment message from the receiver is obtained.

(iii) If the sender cannot receive the acknowledgment message in the previous step, let $\mathbf{r}'_i = u_m \cdot \mathbf{r}_{2i} + \mathbf{r}_{2i+1}$ and $\mathbf{g}_{\mathbb{X},i} = u_m \cdot \mathbf{f}_{\mathbb{X},2i} + \mathbf{f}_{\mathbb{X},2i+1}$ for any $i \in [0, \frac{B}{2M})$. The sender calculates

$$\mathbf{r}'_i[\ell k, (\ell+1)k) = \mathrm{LCH}_{2^{2M}}(\mathbf{g}_{\mathbb{X},i}, k, \omega_{\ell k}), i \in [0, \frac{B}{2M}) \tag{18}$$

by letting ℓ take $\frac{2^M}{k}, \cdots, \frac{2^{2M}}{k} - 1$ in turn. The sender sends the generated encoded packet to the receiver one at a time until it receives the acknowledgment message from the receiver.

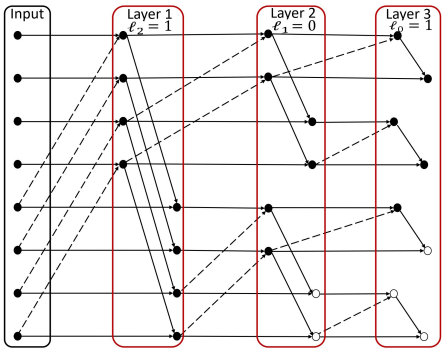

Fig. 3. Flow graph of 8-point LCH transform with five output symbols (all solid dots are the symbols that need to be calculated, and the hollow dots are the symbols that do not need to be calculated).

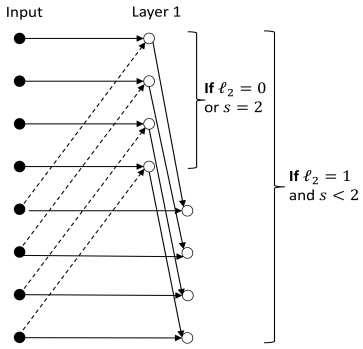

Fig. 4. First layer of 8-point LCH transform with $\ell = (\ell_2\ell_1\ell_0)_2$ output symbols (s is defined in (19)).

In Step (iii), the new coefficient vector $\mathbf{g}_{\mathbb{X},i}$ can be extended to a larger finite field if the receiver still cannot successfully receive k encoded packets. However, it is unlikely that more than one extension is required in real applications.

2) Decoding: Let the ID of the last successfully received encoded packet be $n-1$ when the receiver collects exactly k encoded packets. One can easily derive that all final evaluated polynomials with degrees less than k are in $\mathbb{F}_{2^{\widehat{n}}}[x]$, where $\widehat{n} = 2^{\lceil \log_2 \log_2 n \rceil}$. After that, all encoded packets can be considered as generated by (n,k) RS codes over $\mathbb{F}_{2^{\widehat{n}}}$ and a total of $B/\widehat{n}$ codewords need to be decoded. The decoding of the RLRS code is no different from that of the traditional RS code over $\mathbb{F}_{2^{\widehat{n}}}$, so we directly adopt the decoding algorithm given in [23], [25].

IV. Scheduling Scheme for LCH Transforms

Like other rateless codes, the RLRS encoder should produce encoded packets one by one. However, the encoding algorithm proposed in the previous section calculates polynomial evaluations via k-point LCH transforms. Thus, this generates k encoded packets in each encoding round. When the total number of generated packets n is not a multiple of k, the proposed encoding contains many unnecessary operations (inversely proportional to $n \mod k$). To improve the computation efficiency, this section proposes a scheduling scheme that enables the output symbols of k-point LCH transforms to be generated as needed. As a result, the output symbols of k-point LCH transforms are generated in order.

To begin with, let $k = 2^{k_0}$, where $k_0 \in \mathbb{N}$. From [23], the k-point LCH transform flow graph consists of k_0 layers. More precisely, each layer of k-point LCH transform has k nodes, and all output symbols are in layer k_0 (e.g. 8-point LCH transform is shown in Fig. 3). Let ℓ be the number of output symbols the k-point LCH transform needs to output in order. Here, $0 \le \ell \le k$. We use $H_k^i(\ell)$ to denote the number of nodes calculated in layer i of the k-point LCH transform with ℓ output symbols. Obviously, we have $H_k^i(k) = k$ and $H_k^i(0) = 0$ for any $i = 1, 2, \cdots, k_0$. In the following, we first give the specific formula of $H_k^i(\ell)$, where $0 < \ell < k$ and $1 \le i \le k_0$, and then show the relationship between $H_k^i(\ell)$ and $H_k^i(\ell+1)$. This relationship helps us to produce required output symbols via a k-point LCH transform.

Let the binary representation of ℓ be $(\ell_{k_0-1} \cdots \ell_0)_2$, and

$$s = \min\{i \mid \ell_i \neq 0, \forall i \in [0, k_0)\}, \tag{19}$$

where $0 < \ell < k$. The form of $H_k^i(\ell)$ can be deduced as follows. For the first layer, at least the first 2^{k_0-1} nodes must be calculated. Whether the last 2^{k_0-1} nodes in the first layer need to be calculated depends on the values of s and ℓ_{k_0-1}. An example of $k = 8$ is shown in Fig. 4. Obviously, if $\ell_2 = 0$ or $s = 2$, then $H_8^1(\ell) = 4$; Otherwise, $H_8^1(\ell) = 8$. It is not difficult to check that

$$H_k^1(\ell) = \begin{cases} 2^{k_0-1} + \ell_{k_0-1} \cdot 2^{k_0-1}, & \text{if } s < k_0 - 1, \\ 2^{k_0-1}, & \text{if } s = k_0 - 1. \end{cases} \tag{20}$$

Another example of 8-point LCH transform with five output symbols is shown in Fig. 3. As $k_0 = 3$ and $5 = (101)_2$, then $s = 0$. Immediately, one can see from (20) that $H_8^1(5) = 8$.

For the second layer, the number of nodes that need to be calculated is determined by ℓ_{k_0-1} and $H_{\frac{k}{2}}^1(\ell')$, where $\ell' = (\ell_{k_0-2} \cdots \ell_0)_2$. More precisely, $H_k^2(\ell)$ is the sum of $\ell_{k_0-1} 2^{k_0-1}$ and $H_{\frac{k}{2}}^1(\ell')$. Since $H_{\frac{k}{2}}^1(\ell')$ can be obtained from (20), then

$$H_k^2(\ell) = \begin{cases} \ell_{k_0-1} \cdot 2^{k_0-1} + (2^{k_0-2} + \ell_{k_0-2} \cdot 2^{k_0-2}), & \text{if } s < k_0 - 2, \\ \ell_{k_0-1} \cdot 2^{k_0-1} + 2^{k_0-2}, & \text{if } s = k_0 - 2. \end{cases} \tag{21}$$

As shown in the example of Fig. 3, the number of nodes to be calculated in layer 2 is the sum of four and $H_4^1(1)$, where $H_4^1(1) = 2$. Thus, we have $H_8^2(5) = 6$.

By analogy, for $i \in \{2, \cdots, k_0 - s\}$, the number of nodes that need to be calculated in layer i is determined by $\{\ell_{k_0-j}\}_{j=1}^{i-1}$ and $H_{2^{k_0-i+1}}^1(\ell'')$, where $\ell'' = (\ell_{k_0-i} \cdots \ell_0)_2$. Then,

$$\begin{aligned} H_k^i(\ell) = {} & \ell_{k_0-1} \cdot 2^{k_0-1} + \ell_{k_0-2} \cdot 2^{k_0-2} \\ & + \cdots + \ell_{k_0-i+1} \cdot 2^{k_0-i+1} + H_{2^{k_0-i+1}}^1(\ell''), \end{aligned} \tag{22}$$

where $\ell'' = (\ell_{k_0-i} \cdots \ell_0)_2$. Using (20), the above formula can be reformulated as

$$H_k^i(\ell) = \begin{cases} \left(\sum_{j=k_0-i+1}^{k_0-1} \ell_j \cdot 2^j\right) + \left(2^{k_0-i} + \ell_{k_0-i} \cdot 2^{k_0-i}\right), & \\ & 2 \le i < k_0 - s, \\ \left(\sum_{j=k_0-i+1}^{k_0-1} \ell_j \cdot 2^j\right) + 2^{k_0-i}, & i = k_0 - s. \end{cases}$$
$$= \begin{cases} 2^{k_0-i} + \sum_{j=k_0-i}^{k_0-1} \ell_j \cdot 2^j, & 2 \le i < k_0 - s, \\ \ell, & i = k_0 - s. \end{cases} \tag{23}$$

Formula (23) leads to $H_8^3(5) = 5$ in Fig. 3. The above shows the number of nodes needed to be calculated for the first $k_0 - s$ layers. For each of the remaining s layers, the number of nodes to be calculated is always ℓ because $\{\ell_i = 0\}_{i=0}^{s-1}$. In summary, $H_k^i(\ell)$ can be organized as the following

$$H_k^i(\ell) = \begin{cases} 2^{k_0-i} + \sum_{j=k_0-i}^{k_0-1} \ell_j \cdot 2^j, & i = 1, \cdots, k_0 - s - 1, \\ \ell, & i = k_0 - s, \cdots, k_0 \ . \end{cases} \tag{24}$$

The following considers the case of increasing the number of output symbols of k-point LCH transform one by one. When ℓ is an odd number, adding one output symbol to the k-point LCH transform with ℓ output changes the number of nodes needed to be calculated in the last layer. In this case,

$$H_k^i(\ell+1) = \begin{cases} H_i^k(\ell), & \text{if } i = 1, 2, \cdots, k_0 - 1, \\ H_k^k(\ell) + 1, & \text{if } i = k_0. \end{cases} \tag{25}$$

We consider below the case where ℓ is an even number. When $\ell = 0$, $H_k^i(\ell+1)$ can be obtained directly from (24) since $H_k^i(0) = 0$ for any $i = 1, 2, \cdots, k_0$. When ℓ is an even positive number, let $\ell = (\ell_{k_0-1} \cdots \ell_s \cdots 00)_2$ and $\ell + 1 = (\ell_{k_0-1} \cdots \ell_s \cdots 01)_2$, where $s \ge 1$. From (24), one can know

$$H_k^i(\ell+1) = H_k^i(\ell), \quad i = 1, 2, \cdots, k_0 - s - 1. \tag{26}$$

In the next $s+1$ layers, (24) indicates $H_k^i(\ell+1) = 2^{k_0-i} + \ell = 2^{k_0-i} + H_k^i(\ell)$, where $i = k_0 - s, \cdots, k_0$. Therefore, for any $0 < \ell < k$,

$$H_k^i(\ell+1) = \begin{cases} H_k^i(\ell), & i = 1, \cdots, k_0 - s - 1 \\ H_k^i(\ell) + 2^{k_0-i}, & i = k_0 - s, \cdots, k_0 \ . \end{cases} \tag{27}$$

It should be noted that (27) also holds if ℓ is an odd number due to $s = 0$ in that case. Algorithm 1 shows the process of the k-point LCH transform producing the next symbol while ℓ symbols have been output in order. From line 2 to line 4, the first output symbol of the k-point LCH transform can be produced via (24). The number of nodes calculated in lines 7 to 9 are from (27).

Algorithm 1 sch-LCH($\mathbf{f}_{\mathbb{X}}, k, \beta, n$)

Input: $\mathbf{f}_{\mathbb{X}} \in \mathbb{F}_{2^M}^k$, $k = 2^{k_0} \le 2^M$, $\beta \in \mathbb{F}_{2^M}$, $0 \le \ell < k$
Output: $f(\omega_\ell + \beta)$
Require: $\{\text{sch-LCH}(\mathbf{f}_{\mathbb{X}}, k, \beta, i)\}_{i=0}^{\ell-1}$ have been executed.
1: **if** $\ell = 0$ **then**
2: **for** $i = 1, 2, \cdots, k_0$ **do**
3: Calculate the first 2^{k_0-i} unknown nodes in layer i.
4: **end for**
5: **else**
6: Obtain s via (19).
7: **for** $i = k_0 - s, k_0 - s + 1 \cdots, k_0$ **do**
8: Calculate the first 2^{k_0-i} unknown nodes in layer i.
9: **end for**
10: **end if**

V. Complexity analysis

This section analyzes the theoretical computational complexity of the proposed RLRS codes. To begin with, according to Sec. III, one can know that two basic operations are only required, i.e., packet-by-packet addition and packet-by-symbol multiplication. Considering a single packet size B, let $Add(M)$ denote the complexity of one packet-by-packet addition over $\mathbb{F}_{2^M}$, and $Mul(M)$ denote the complexity of one packet-by-symbol multiplication over $\mathbb{F}_{2^M}$. Generally, we have

$$Add(M) = \eta \cdot Mul(M) \tag{28}$$

for some $\eta \le 1$. The value of η highly depends on finite field arithmetic implementations [15].

In the proposed encoding, k-point LCH transforms must first be performed to compute the original B/M evaluated polynomials. This part can be considered as calculating all systematically encoded packets. From [23], k-point LCH/ILCH requires $k \log_2 k$ packet-by-packet additions and $\frac{k}{2} \log_2 k$ packet-by-symbol multiplications. Hence, the average encoding complexity of generating one systematic encoded packet over $\mathbb{F}_{2^M}$ can be seen as

$$\mathcal{X}(M) = \frac{Add(M) \cdot k \log_2 k + Mul(M) \cdot \frac{k}{2} \log_2 k}{k}$$
$$= (\frac{1}{2} + \eta) \cdot \log_2 k \cdot Mul(M). \tag{29}$$

The above is also the average encoding complexity of generating one non-systematic encoded packet over $\mathbb{F}_{2^M}$. In fact, the average encoding complexity of generating one non-systematic encoded packet is affected by field extensions. The following will demonstrate that the complexity growth caused by field extensions is smaller compared with that in [15]. This is due to the special structure of the field tower (1), and it can also be applied to the construction of RLRS codes in [15].

A. Complexity growth

After one field extension, each symbol in $\mathbb{F}_{2^{2M}}$ is represented by a binary sequence of length $2M$. Note that the packet size does not change as the field changes, as shown in Fig. 2. Then, $Add(2M)$ is still the complexity of performing bit-wise XOR on B bits, and we have $Add(2M) = Add(M)$. For packet-by-symbol multiplication in $\mathbb{F}_{2^{2M}}$, we first consider the case of single symbol. Let a symbol be $a \cdot u_m + b \in \mathbb{F}_{2^{2M}}$ and a constant symbol $c \cdot u_m + d \in \mathbb{F}_{2^{2M}}$, where $a, b, c, d \in \mathbb{F}_{2^M}$. Then

$$(a \cdot u_m + b) \cdot (c \cdot u_m + d)$$
$$= ac \cdot u_m^2 + (ad + bc) \cdot u_m + bd. \tag{30}$$

TABLE II
COMPARISON OF OPERATION COMPLEXITIES

Operation	Complexity	
	[15]	Proposed
Corr. to $Add(2M)$	$Add(M)$	$Add(M)$
Corr. to $Mul(2M)$	$\frac{4+3\eta}{2}Mul(M)$	$\frac{3(1+\eta)}{2}Mul(M)$
Corr. to $\mathcal{X}(M)$	$(1+\eta)kMul(M)$	$(\frac{1}{2}+\eta)\log_2 kMul(M)$
Corr. to $\mathcal{X}(2M)$	$\frac{(4+5\eta)}{2}kMul(M)$	$\frac{(3+7\eta)}{4}\log_2 kMul(M)$
$\xi=\mathcal{X}(2M)/\mathcal{X}(M)$	$2+\frac{\eta}{2(1+\eta)}>2$	$\frac{7}{4}-\frac{1}{4(2\eta+1)}<1.67$

By applying Lemma 1 and Karatsuba algorithm [33], the above formula can be converted into

$$\begin{aligned}&(a\cdot u_m+b)\cdot(c\cdot u_m+d)\\&=((a+b)\cdot(c+d)-b\cdot d)\cdot u_m+(b\cdot d+a\cdot cv_{2^m-1}).\end{aligned}\tag{31}$$

In (31), c, d and v_{2^m-1} are all constants, then $c+d, c\cdot v_{2^m-1}$ can be calculated in pre-processing. Consequently, (31) can be done by 3 multiplications and 3 additions in $\mathbb{F}_{2^M}$. Note that the packet-by-symbol multiplication in $\mathbb{F}_{2^M}$ at this time has only half the size. From the above, one can get

$$Mul(2M)=3\cdot\frac{Mul(M)+Add(M)}{2}=\frac{3(1+\eta)}{2}Mul(M).\tag{32}$$

From the above, over $\mathbb{F}_{2^{2M}}$, the encoding complexity of generating one non-systematic encoded packet is

$$\begin{aligned}\mathcal{X}(2M)=&\frac{Add(2M)\cdot k\log_2 k+Mul(2M)\cdot\frac{k}{2}\log_2 k}{k}\\=&\frac{3+7\eta}{4}\log_2 k\cdot Mul(M).\end{aligned}\tag{33}$$

Then according to (29) and (33), the complexity growth speed caused by the field extension is

$$\xi=\frac{\mathcal{X}(2M)}{\mathcal{X}(M)}=\frac{7\eta+3}{4\eta+2}=\frac{7}{4}-\frac{1}{4(2\eta+1)}.\tag{34}$$

where $1.5<\xi<1.67$.

The above result is useful for analyzing the complexity of the proposed encoding. Clearly, the complexity growth speed obtained in this paper is lower compared with $\xi=2+0.5\frac{\eta}{1+\eta}>2$ presented in [15]. TABLE II shows the comparison of operation complexities between [15] and this paper. It is worth mentioning that if the realization of the field extension in [15] is replaced by using the field tower, one can check that the corresponding complexity growth speed ξ can be converted from $2+0.5\frac{\eta}{1+\eta}$ to $1.5+\frac{\eta}{1+\eta}$. The improvement is

$$\frac{0.5-0.5\frac{\eta}{1+\eta}}{2+0.5\frac{\eta}{1+\eta}}=\frac{1}{4+5\eta},\quad\text{where } 0<\eta\leq 1.\tag{35}$$

The above formula indicates that the field tower in this paper can be used in [15] to reduce the corresponding computational complexity growth speed by 11% to 25%.

B. Encoding complexity

As more than one field extension has no practical meaning, this subsection only considers the case where field extension occurs at most once. [15] gives the upper bounds of the average number of non-systematic encoded packets if field extension does not occur or only occurs once. Specifically, let $\epsilon<1$ be the packet loss probability in data transmission and p the probability that the receiver cannot successfully receive k packets without field extension. The above two upper bounds are, respectively,

$$\mathcal{K}_0=\frac{\epsilon k}{1-\epsilon},\qquad\mathcal{K}_1=\frac{p\cdot\epsilon k}{1-\epsilon},\tag{36}$$

where $\mathcal{K}_0$ is the result when no filed extension occurs, and $\mathcal{K}_1$ is the result that occurs only once. Now, it is straightforward to show that the per-source packet encoding complexity of the proposed RLRS code is upper-bounded by

$$C_E\leq\frac{k\cdot\mathcal{X}(M)+\mathcal{X}(M)\cdot\mathcal{K}_0+\mathcal{X}(2M)\cdot\mathcal{K}_1}{k},\tag{37}$$

where the first term of the numerator can be seen as the complexity of generating all systematic encoded packets. Further, the above formula can be reformulated as

$$\begin{aligned}C_E&\leq\mathcal{X}(M)+\frac{\mathcal{X}(M)}{k}\cdot\mathcal{K}_0+\frac{\xi\cdot\mathcal{X}(M)}{k}\cdot\mathcal{K}_1\\&=\mathcal{X}(M)\cdot\frac{1+\xi p\epsilon}{1-\epsilon}\\&=(\eta+\frac{1}{2})\cdot\frac{B}{M}\cdot\log_2 k\cdot Mul(M)\cdot\frac{1+\xi p\epsilon}{1-\epsilon}.\end{aligned}\tag{38}$$

Given a fixed packet loss probability ϵ, the above formula indicates the per-source packet encoding complexity is $\mathcal{O}(\log_2 k)$. This is an improvement compared to $\mathcal{O}(k)$ in [15].

C. Decoding complexity

As shown in Sec. III-B, all source packets can be decoded according to the (n,k) LCH-based RS decoding algorithm over $\mathbb{F}_{2^{\widehat{n}}}$, where $\widehat{n}=2^{\lceil\log_2\log_2 n\rceil}$. When decoding a single codeword, [25] shows that the decoding algorithm has the packet-by-packet addition and packet-by-symbol multiplication complexities of both $\mathcal{O}(n\log_2 n)$. The average size of n is $k/(1-\epsilon)$. If the total number of encoded packets $n<2^M$, then $\widehat{n}=M$ and the order of overall decoding complexity is $\mathcal{O}(\frac{k}{1-\epsilon}\cdot\log_2\frac{k}{1-\epsilon}\cdot(Add(M)+Mul(M)))$. Hence, the per-source packet decoding complexity is

$$\mathcal{O}\left(\frac{1+\eta}{1-\epsilon}\cdot\log_2\frac{k}{1-\epsilon}\cdot Mul(M)\right).\tag{39}$$

Alternatively, if the total number of encoded packets $n>2^M$, then $\widehat{n}=2M$ and the per-source packet decoding complexity is

$$\mathcal{O}\left(\frac{3+5\eta}{2(1-\epsilon)}\cdot\log_2\frac{k}{1-\epsilon}\cdot Mul(M)\right).\tag{40}$$

Given a fixed packet loss probability ϵ, both (39) and (40) indicate the per-source packet decoding complexity is $\mathcal{O}(\log_2 k)$.

VI. SIMULATIONS AND COMPARISONS

This section demonstrates the performance of the proposed RLRS codes and compares them with other rateless codes, such as the Cauchy-based RLRS codes [15] and the well-known RaptorQ [26]. The RaptorQ is the most advanced

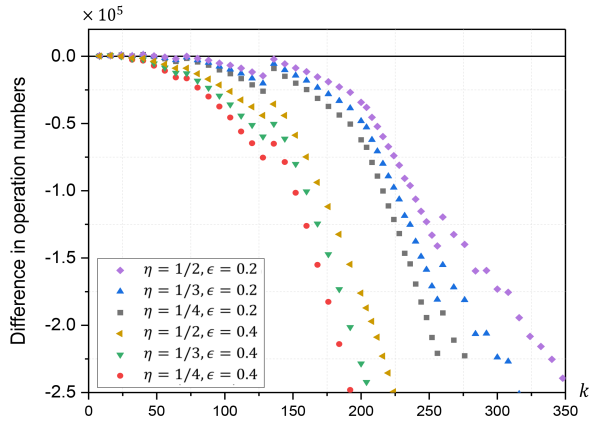

Fig. 5. Result of subtracting the number of packet-by-packet additions required by the Cauchy-based RLRS codes [15] from that of the proposed RLRS codes (all involved operations are reduced to packet-by-packet additions over $\mathbb{F}_{2^8}$, η is shown in (28), and ϵ is the packet loss probability).

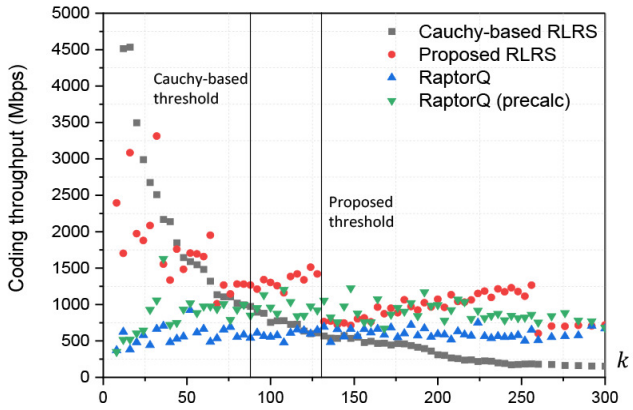

Fig. 6. Coding throughputs of different rateless codes when the packet loss probability $\epsilon = 0.2$ and individual packet size $B = 256$ bits (where k is the number of source packets and only RLRS codes can guarantee zero reception overhead).

Raptor code and uses the binary extension field $\mathbb{F}_{2^8}$ to improve performance. In the Cauchy-based RLRS codes, the preservative field extension is implemented by the field tower presented in this paper, as TABLE II shows that it is superior in both $Mul(2M)$ and ξ. All simulations were performed on the virtual machine equipped with Ubuntu 22.04 LTS operating system, 12-th Gen Intel Core i7-12700H (2.30GHz), and 4 GB 4800 MHz DDR5 SDRAM. All RLRS codes set the initial field to $\mathbb{F}_{2^8}$. All rateless codes were implemented in C/C++ language, with RaptorQ obtained from an open-source code on Github [27]. In particular, the implementation of the RaptorQ provides two versions with and without pre-calculation. To obtain stable results, all simulation results under each setting are taken as the average value after multiple repetitions. Specifically, when $k \leq 256$, the number of repetitions is 1000; when $k > 256$, the number of repetitions is set to 20 due to the increase in consumed time. Note that k is the number of source packets.

To begin with, Fig. 5 shows the difference in operation numbers between the proposed RLRS codes and Cauchy-based RLRS codes. We reduce all operations to packet-by-packet additions over $\mathbb{F}_{2^8}$. Each packet-by-symbol multiplication over $\mathbb{F}_{2^8}$ can be converted to the packet-by-packet addition via the factor η in (28). According to experience, addition over $\mathbb{F}_{2^8}$ is about four times more efficient than multiplication, that is, $\eta = 1/4$. Fig. 5 shows a comparison when η takes more possible values. Moreover, any operation in $\mathbb{F}_{2^{16}}$ can be converted to $\mathbb{F}_{2^8}$ for execution, as shown in TABLE II. The vertical axis represents the result of subtracting the operation number required by the Cauchy-based RLRS codes from the proposed RLRS codes. Thus, the smaller the vertical axis, the higher the operational efficiency of the proposed RLRS codes. From Fig. 5, it can be seen that the proposed RLRS codes perform better than the Cauchy-based RLRS codes, especially as the packet loss probability ϵ increases. The two RLRS codes have similar operational efficiency when $k(\leq 64)$ is small. In that case, the theoretical complexity advantage is not obvious,

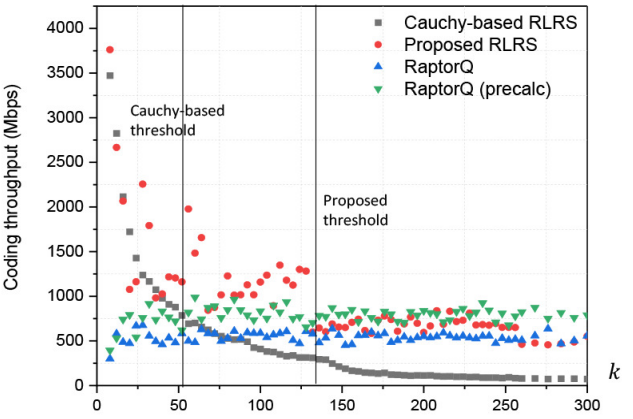

Fig. 7. Coding throughputs of different rateless codes when the packet loss probability $\epsilon = 0.4$ and individual packet size $B = 256$ bits (where k is the number of source packets and only RLRS codes can guarantee zero reception overhead).

and the Cauchy-based RLRS codes can reduce decoding complexity by utilizing successfully received systematic encoded packets. From Fig. 5, one can also see that the difference in operation number does not change smoothly, i.e., there is fluctuation when k is slightly greater than a power of two. This is due to the fact that the proposed RLRS codes need to align the total number of source packets to a power of two by filling empty packets. When k is far away from the smallest power of two not less than it, the more operations caused by empty packets drag down the performance. This is especially obvious when k is slightly larger than 128, due to the presence of a large number of operations over the large field $\mathbb{F}_{2^{16}}$ in the proposed RLRS codes. At this time, the Cauchy-based RLRS codes still only need to be performed in $\mathbb{F}_{2^8}$. We will see from the simulations later that the actual performance difference between the two RLRS codes is consistent with the comparison in operation numbers.

Fig. 6-8 show the performances of different rateless codes

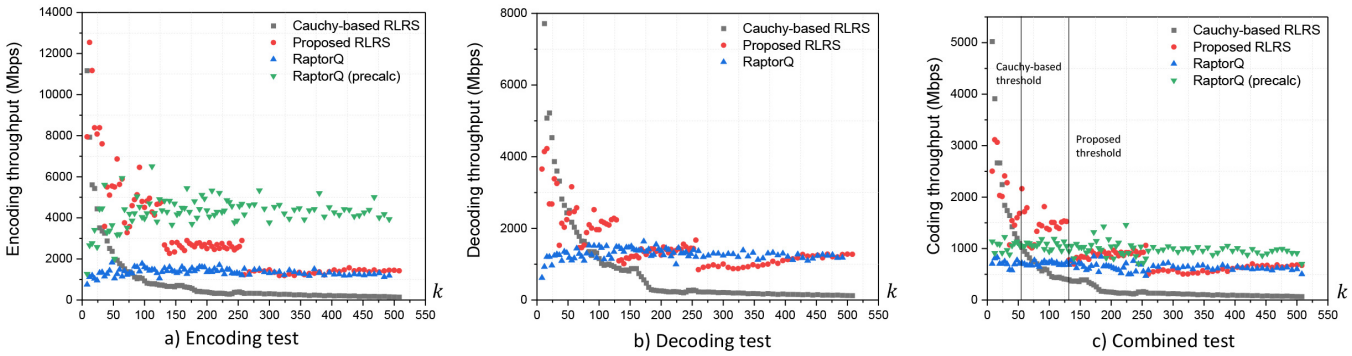

Fig. 8. Encoding and decoding throughputs of different rateless codes when the packet loss probability $\epsilon = 0.3$ and individual packet size $B = 256$ (where k is the number of source packets and only RLRS codes can guarantee zero reception overhead).

under different parameters. The coding throughput is defined as the ratio of the total size of k source packets to the sum of the time spent on encoding and decoding. Due to the fact that RLRS codes always maintain zero reception overhead, the overhead in the implementation of RaptorQ is set to zero by default so it is not always decoded successfully (the probability of successful decoding is about 99% [26]). From Fig. 6 with the packet loss probability $\epsilon = 0.2$ and individual packet size $B = 256$, one can see that RLRS codes always perform better than RaptorQ codes if k is less than a certain threshold. Obviously, the threshold of the proposed RLRS codes is larger than that of the Cauchy-based RLRS codes, and the difference between the two thresholds increases with ϵ increasing, as shown in Fig. 7 with ϵ being 0.4. When k is large, the proposed RLRS codes perform better than the Cauchy-based RLRS codes due to their lower computational complexity. In addition, both Fig. 6 and Fig. 7 show that the threshold between the proposed RLRS codes and RaptorQ codes is $k = 128$. This offers the possibility for the proposed RLRS codes to replace RaptorQ in short codes. Note that in addition to having a performance advantage in short codes, the proposed RLRS codes can always guarantee zero reception overhead, while the RaptorQ codes cannot.

From Fig. 6 and 7, we remark that the proposed RLRS codes have significant performance fluctuations. This is consistent with the previous discussion on the number of operations. Briefly, the proposed RLRS codes need to align the total number of source packets to a power of two by filling empty packets. When k is not a power of two, the encoding/decoding process is the same as when the total number of source packets is the smallest power of two greater than k, thus increasing the computational cost. Note that, like the principle of shortened RS codes [21], filling empty packets does not increase the proposed RLRS code's reception overhead. To show the performance differences in more detail, Fig. 8 gives the encoding/decoding performance of different rateless codes at $\epsilon = 0.3$ and $B = 256$. The encoding (resp. decoding) throughput is defined as the ratio of the total size of k source packets to the time spent on encoding (resp. decoding). It can be observed that the performance advantage of the RaptorQ in large k comes from the pre-calculation of encoding.

In simulations, we also tested the performances of different rateless codes with the individual packet size $B = 512$ and 1024. The results show that the performance advantage of the proposed RLRS codes compared to other codes decreases as B increases. From the above, we conclude that the proposed RLRS codes are more suitable for the scenario with high packet loss probability ϵ and small k, B.

VII. Conclusions

In this paper, a new construction of the RLRS codes is proposed based on Vandermonde generator matrices. Compared with the Cauchy-based RLRS codes in literature, the proposed RLRS codes have the following advantages. The first is that the proposed preservative field extension can be easily implemented without searching for quadratic irreducible polynomials with specific forms. The second is that LCH/ILCH transforms can be employed in the proposed RLRS codes to obtain the encoding and decoding complexities of both $\mathcal{O}(\log_2 k)$ per source packet, less than those of both $\mathcal{O}(k)$ in the Cauchy-based RLRS codes. Finally, the proposed RLRS codes have a lower speed of computational complexity growth caused by field extension. Note that in practice field extension occurs at most once, so it does not affect the order of the above computational complexity. This paper also proposes a scheduling scheme to generate encoded packets on demand to avoid unexpected computation costs of the LCH transform that always produces burst outputs. Simulation results show that the performance of the proposed RLRS codes is competitive compared to that of other rateless codes.

References

[1] P. Elias, "Coding for two noisy channels," in *Information Theory, 3rd London Symposium, London, England, Sept. 1955*, 1955.
[2] V. Paxson, "End-to-end internet packet dynamics," *IEEE/ACM transactions on networking*, vol. 7, no. 3, pp. 277–292, 1999.
[3] C. M. Kozierok, *The TCP/IP guide: a comprehensive, illustrated Internet protocols reference*. No Starch Press, 2005.
[4] J. C. Pasquale, G. C. Polyzos, E. W. Anderson, and V. P. Kompella, "The multimedia multicast channel," in *International Workshop on Network and Operating System Support for Digital Audio and Video*. Springer, 1992, pp. 197–208.
[5] L. Rizzo and L. Vicisano, "A reliable multicast data distribution protocol based on software FEC techniques," in *In Proc. 4th IEEE Workshop Highperform. Commun. Syst. (HPCS '97)*, 1997, pp. 116–125.
[6] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Efficient erasure correcting codes," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 569–584, 2001.

[7] M. Luby, "LT codes," in *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.* IEEE, 2002, pp. 271–280.

[8] J. W. Byers, M. Luby, and M. Mitzenmacher, "A digital fountain approach to asynchronous reliable multicast," *IEEE journal on selected areas in communications*, vol. 20, no. 8, pp. 1528–1540, 2002.

[9] A. Shokrollahi, "Raptor codes," *IEEE transactions on information theory*, vol. 52, no. 6, pp. 2551–2567, 2006.

[10] R. Abbas, M. Shirvanimoghaddam, T. Huang, Y. Li, and B. Vucetic, "Novel design for short analog fountain codes," *IEEE Communications Letters*, vol. 23, no. 8, pp. 1306–1309, 2019.

[11] J. Huang, Z. Fei, C. Cao, M. Xiao, and X. Xie, "Reliable broadcast based on online fountain codes," *IEEE Communications Letters*, vol. 25, no. 2, pp. 369–373, 2020.

[12] V. Bioglio, "MRB decoding of LT codes over AWGN channels," *IEEE Wireless Communications Letters*, vol. 8, no. 2, pp. 548–551, 2018.

[13] L. Yuan, J. Pan, and K. Deng, "A modified design of Raptor codes for small message length," *Wireless Networks*, vol. 25, pp. 2437–2447, 2019.

[14] D. Liben-Nowell, H. Balakrishnan, and D. Karger, "Analysis of the evolution of peer-to-peer systems," in *Proceedings of the twenty-first annual symposium on Principles of distributed computing*, 2002, pp. 233–242.

[15] R. R. Borujeny and M. Ardakani, "A new class of rateless codes based on Reed–Solomon codes," *IEEE Transactions on Communications*, vol. 64, no. 1, pp. 49–58, 2015.

[16] 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.

[17] S. B. Wicker and V. K. Bhargava, *Reed-Solomon codes and their applications*. John Wiley & Sons, 1999.

[18] L. Rizzo and L. Vicisano, "A reliable multicast data distribution protocol based on software FEC techniques," in *The Fourth IEEE Workshop on High-Performance Communication Systems*. IEEE, 1997, pp. 116–125.

[19] L. Vicisano, J. Crowcroft, and L. Rizzo, "TCP-like congestion control for layered multicast data transfer," in *Proceedings. IEEE INFOCOM'98, the Conference on Computer Communications. Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies. Gateway to the 21st Century (Cat. No. 98*, vol. 3. IEEE, 1998, pp. 996–1003.

[20] U. Aho, J. D. Ullman, and E. John, "Hopcroft,"data structures and algorithms"," 1983.

[21] F. J. MacWilliams and N. J. A. Sloane, *The theory of error-correcting codes*. Elsevier, 1977, vol. 16.

[22] F. Didier, "Efficient erasure decoding of Reed-Solomon codes," *arXiv preprint arXiv:0901.1886*, 2009.

[23] S.-J. Lin, W.-H. Chung, and Y. S. Han, "Novel polynomial basis and its application to Reed-Solomon erasure codes," in *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, 2014, pp. 316–325.

[24] S.-J. Lin, T. Y. Al-Naffouri, and Y. S. Han, "FFT algorithm for binary extension finite fields and its application to Reed–Solomon codes," *IEEE Transactions on Information Theory*, vol. 62, no. 10, pp. 5343–5358, 2016.

[25] S.-J. Lin, T. Y. Al-Naffouri, Y. S. Han, and W.-H. Chung, "Novel polynomial basis with fast Fourier transform and its application to Reed–Solomon erasure codes," *IEEE Transactions on Information Theory*, vol. 62, no. 11, pp. 6284–6299, 2016.

[26] L. Minder, A. Shokrollahi, M. Watson, M. Luby, and T. Stockhammer, "RaptorQ Forward Error Correction Scheme for Object Delivery," RFC 6330, Aug. 2011. [Online]. Available: https://www.rfc-editor.org/info/rfc6330

[27] Project webpage, https://github.com/sleepybishop/nanorq.

[28] W. Ryan and S. Lin, *Channel codes: classical and modern*. Cambridge university press, 2009.

[29] D. G. Cantor, "On arithmetical algorithms over finite fields," *Journal of Combinatorial Theory, Series A*, vol. 50, no. 2, pp. 285–300, 1989.

[30] S.-L. Yang, "On the LU factorization of the Vandermonde matrix," *Discrete applied mathematics*, vol. 146, no. 1, pp. 102–105, 2005.

[31] J. Von Zur Gathen and J. Gerhard, *Modern computer algebra*. Cambridge university press, 2013.

[32] J. Lee and S. Park, "Optimum UDP packet sizes in ad hoc networks," *IEICE transactions on communications*, vol. 88, no. 2, pp. 815–820, 2005.

[33] A. Weimerskirch and C. Paar, "Generalizations of the Karatsuba algorithm for polynomial multiplication," *Submitted to Design, Codes and Cryptography*, 2002.

Leilei Yu received the B.Eng. degree in electronic information engineering from the Tianjin University of Technology, Tianjin, China, in 2015, and the Ph.D. degree in cyberspace security from the University of Science and Technology of China, Hefei, China, in 2021. From 2021 to 2022, he was a cybersecurity researcher with the Purple Mountain Laboratories, Nanjing, China. He is currently a postdoctoral research fellow with the Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China. His research focuses on coding theory and high-performance computing.

Sian-Jheng Lin (M'16) received the B.Sc., M.Sc., and Ph.D. degrees in computer science from National Chiao Tung University, Hsinchu, Taiwan, in 2004, 2006, and 2010, respectively. From 2010 to 2014, he was a postdoc with the Research Center for Information Technology Innovation, Academia Sinica. From 2014 to 2016, he was a postdoc with the Electrial Engineering Department at King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia. From 2016 to 2021, he was a researcher with the School of Information Science and Technology at University of Science and Technology of China (USTC), Hefei, China. He is currently a senior scientist with the Theory Lab, Cenrtal Research Institute, 2012 Labs, Huawei Technology Co. Ltd. In recent years, his research focuses on the codes for storage systems, channel coding for memories and lossless data compressions.

Yunghsiang S. Han (S'90-M'93-SM'08-F'11) was born in Taipei, Taiwan, 1962. He received B.Sc. and M.Sc. degrees in electrical engineering from the National Tsing Hua University, Hsinchu, Taiwan, in 1984 and 1986, respectively, and a Ph.D. degree from the School of Computer and Information Science, Syracuse University, Syracuse, NY, in 1993. He was from 1986 to 1988 a lecturer at Ming-Hsin Engineering College, Hsinchu, Taiwan. He was a teaching assistant from 1989 to 1992, and a research associate in the School of Computer and Information Science, Syracuse University from 1992 to 1993. He was, from 1993 to 1997, an Associate Professor in the Department of Electronic Engineering at Hua Fan College of Humanities and Technology, Taipei Hsien, Taiwan. He was with the Department of Computer Science and Information Engineering at National Chi Nan University, Nantou, Taiwan from 1997 to 2004. He was promoted to Professor in 1998. He was a visiting scholar in the Department of Electrical Engineering at University of Hawaii at Manoa, HI from June to October 2001, the SUPRIA visiting research scholar in the Department of Electrical Engineering and Computer Science and CASE center at Syracuse University, NY from September 2002 to January 2004 and July 2012 to June 2013, and the visiting scholar in the Department of Electrical and Computer Engineering at University of Texas at Austin, TX from August 2008 to June 2009. He was with the Graduate Institute of Communication Engineering at National Taipei University, Taipei, Taiwan from August 2004 to July 2010. From August 2010 to January 2017, he was with the Department of Electrical Engineering at National Taiwan University of Science and Technology as Chair Professor. From February 2017 to February 2021, he was with School of Electrical Engineering & Intelligentization at Dongguan University of Technology, China. Now, he is with the Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China and as a consultant of Huawei Technology company. His research interests are in error-control coding, wireless networks, and security.

Dr. Han was a winner of the 1994 Syracuse University Doctoral Prize and a Fellow of IEEE. One of his papers won the prestigious 2013 ACM CCS Test-of-Time Award in cybersecurity.