Fast Error and Erasure Decoding Algorithm for Reed-Solomon Codes

Nianqi Tang, Chao Chen, Yunghsiang S. Han, *Fellow, IEEE*

***Abstract*—Reed-Solomon (RS) codes are widely used to correct errors and erasures. This paper proposes a fast error and erasure decoding algorithm for RS codes. It achieves the best-known complexity $O(n\log(n-k)+(n-k)\log^2(n-k))$, where n, k are the code length and dimension, respectively. Furthermore, the proposed method is efficient for practical codes. For decoding RS$(255, 223)$, compared with the existing method, the new algorithm saves 32% field operations.**

***Keywords*—Reed-Solomon codes, decoding algorithm, fast Fourier transform.**

I. Introduction

Reed-Solomon (RS) codes, the most commonly used error-correcting codes, have been adopted in many applications, such as storage devices, digital television and data transmission. Hence, investigating fast decoding algorithms for RS codes is an important research topic. As RS codes can be defined by finite field Fourier transform, designing decoding algorithms that take advantage of fast Fourier transform (FFT) draws much attention. For example, by using composite cyclotomic Fourier transform, [1] derived fast syndrome-based decoders. An efficient syndrome calculation method was proposed in [2] by improving the inverse cyclotomic discrete Fourier transform. In [3], an interpolation-based decoding algorithm was designed, whose complexity is $O(t\log^2(t))$ over FFT-friendly field, where t is the correction capability. Based on the additive FFT, an error-only decoding algorithm of RS codes was proposed in [4], whose complexity is $O(n\log(n-k)+(n-k)\log^2(n-k))$, where n and k are the code length and dimension, respectively, and $n-k$ must be a power of two. This achieves the best complexity to date. By proposing the partial FFT algorithm, the constraint on $n-k$ was removed in [5]. By reconstructing the modular approach, [6] further improved this error-only decoding algorithm to make it competitive for practical codes, e.g., RS(255, 223) used in optical communication.

In many applications, RS codewords are disturbed by errors and erasures, for example, in Compact Disc devices or Quick-Response codes. In [7], an error-and-erasure decoding technique was proposed that uses prime-factor discrete Fourier transform. It should be noted that an error and erasure decoding algorithm based on FFT has been provided in [8]. However, the key equation in [8] is incomplete, such that the proposed decoding algorithm cannot decode up to all decodable error and erasure patterns. In this paper, we define the syndrome for the error-and-erasure decoding algorithm and derive the key equation, which generalizes the algorithm in [6] to an error-and-erasure decoding algorithm. The complexity of the proposed algorithm is $O(n\log(n-k)+(n-k)\log^2(n-k))$, which is the best complexity for error-and-erasure decoding of RS codes to date. The comparison shows that the proposed algorithm is superior to known methods in complexity for practical codes.

This paper is organized as follows. Section II describes FFT algorithms. Then we derive the error-and-erasure decoding algorithm in Section III. Section IV provides a detailed implementation of the proposed algorithm. A comparison is given in Section V, which compares the proposed method with others in the literature.

This work was supported by National Key Research and Development Program of China under Grant 2022YFA1004902.

Nianqi Tang (724973040@qq.com) and Yunghsiang S. Han (yunghsiangh@gmail.com) are with the Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, Shenzhen, China. Chao Chen (Corresponding author, cchen@xidian.edu.cn) is with the State Key Lab of ISN, Xidian University, Xi'an, China.

II. FFT Algorithm

This section reviews the n-point FFT algorithm with complexity $O(n\log n)$ over finite fields presented in [9]. Throughout this paper, unless otherwise stated, we assume that the elements appeared are in $\mathbb{F}_{2^m}$ and the polynomials are in $\mathbb{F}_{2^m}[x]$. Given a basis of $\mathbb{F}_{2^m}$ over $\mathbb{F}_2$, denoted by $\{v_0, v_1, \ldots, v_{m-1}\}$, the elements in $\mathbb{F}_{2^m}$ can be represented by

$$\omega_l = l_0v_0 + l_1v_1 + \cdots + l_{m-1}v_{m-1}, 0 \le l < 2^m, \quad (1)$$

where $(l_0, l_1, \ldots, l_{m-1})$ is the binary representation of l. For $0 \le \tau \le m$, the subspace polynomial is defined by

$$s_\tau(x) = \prod_{l=0}^{2^\tau-1}(x-\omega_l).$$

The set $\bar{\mathbb{X}} = \{\bar{X}_0(x), \bar{X}_1(x), \ldots, \bar{X}_{2^m-1}(x)\}$, where

$$\bar{X}_l(x) = \frac{s_0(x)^{l_0}s_1(x)^{l_1}\cdots s_{m-1}(x)^{l_{m-1}}}{s_0(v_0)^{l_0}s_1(v_1)^{l_1}\cdots s_{m-1}(v_{m-1})^{l_{m-1}}},$$

is a basis of $\mathbb{F}_{2^m}[x]/(x^{2^m}-x)$ over $\mathbb{F}_{2^m}$. We define p_l as

$$p_l = s_0(v_0)^{l_0}s_1(v_1)^{l_1}\cdots s_{m-1}(v_{m-1})^{l_{m-1}}, 0 \le l < 2^m.$$

For a polynomial $f(x) \in \mathbb{F}_{2^m}[x]/(x^{2^m}-x)$ of degree less than 2^τ, given its coordinate vector $\bar{\mathbf{f}} = (\bar{f}_0, \bar{f}_1, \ldots, \bar{f}_{2^\tau-1})$ with respect to $\bar{\mathbb{X}}$ and $\beta \in \mathbb{F}_{2^m}$, an FFT algorithm computes $\mathbf{F} = (f(\omega_0+\beta), f(\omega_1+\beta), \ldots, f(\omega_{2^\tau-1}+\beta))$ within $O(2^\tau\log(2^\tau))$ field operations, which is denoted by

$$\mathbf{F} = \text{FFT}_{\bar{\mathbb{X}}}(\bar{\mathbf{f}}, \tau, \beta).$$

The inverse transform is written as

$$\bar{\mathbf{f}} = \mathrm{IFFT}_{\bar{\mathbb{X}}}(\mathbf{F}, \tau, \beta).$$

Detailed descriptions of $\mathrm{FFT}_{\bar{\mathbb{X}}}$ and $\mathrm{IFFT}_{\bar{\mathbb{X}}}$ are shown in Algorithms 1 and 2, respectively. For more discussions, we refer to [10].

Algorithm 1 $\mathrm{FFT}_{\bar{\mathbb{X}}}$ [9]

Input: $\bar{\mathbf{f}} = (\bar{f}_0, \bar{f}_1, \ldots, \bar{f}_{2^\tau-1})$, τ, β.
Output: $(f(\omega_0+\beta), f(\omega_1+\beta), \ldots, f(\omega_{2^\tau-1}+\beta))$.

if $\tau = 0$ **then**
  return $\bar{f}_0$
end if
for $l = 0, 1, \ldots, 2^{\tau-1}-1$ **do**
  $a_l^{(0)} = \bar{f}_l + \dfrac{s_{\tau-1}(\beta)}{s_{\tau-1}(v_{\tau-1})}\bar{f}_{l+2^{\tau-1}}$
  $a_l^{(1)} = a_l^{(0)} + \bar{f}_{l+2^{\tau-1}}$
end for
$\mathbf{a}^{(0)} = (a_0^{(0)}, \ldots, a_{2^{\tau-1}-1}^{(0)}), \mathbf{a}^{(1)} = (a_0^{(1)}, \ldots, a_{2^{\tau-1}-1}^{(1)})$
Calculate $\mathbf{A}_0 = \mathrm{FFT}_{\bar{\mathbb{X}}}(\mathbf{a}^{(0)}, \tau-1, \beta)$, $\mathbf{A}_1 = \mathrm{FFT}_{\bar{\mathbb{X}}}(\mathbf{a}^{(1)}, \tau-1, v_{\tau-1}+\beta)$
return $(\mathbf{A}_0, \mathbf{A}_1)$

Algorithm 2 $\mathrm{IFFT}_{\bar{\mathbb{X}}}$ [9]

Input: $\mathbf{F} = (f(\omega_0+\beta), f(\omega_1+\beta), \ldots, f(\omega_{2^\tau-1}+\beta)), \tau, \beta$
Output: $\bar{\mathbf{f}}$ such that $\mathbf{F} = \mathrm{FFT}_{\bar{\mathbb{X}}}(\bar{\mathbf{f}}, \tau, \beta)$

if $\tau = 0$ **then**
  return $f(\omega_0+\beta)$
end if
$\mathbf{A}_0 = (f(\omega_0+\beta), \ldots, f(\omega_{2^{\tau-1}-1}+\beta)), \mathbf{A}_1 = (f(\omega_{2^{\tau-1}}+\beta), \ldots, f(\omega_{2^\tau-1}+\beta))$
$\mathbf{a}^{(0)} = \mathrm{IFFT}_{\bar{\mathbb{X}}}(\mathbf{A}_0, \tau-1, \beta)$, $\mathbf{a}^{(1)} = \mathrm{IFFT}_{\bar{\mathbb{X}}}(\mathbf{A}_1, \tau-1, v_{\tau-1}+\beta)$
for $l = 0, 1, \ldots, 2^{\tau-1}-1$ **do**
  $\bar{f}_{l+2^{\tau-1}} = a_l^{(0)} + a_l^{(1)}$
  $\bar{f}_l = a_l^{(0)} + \dfrac{s_{\tau-1}(\beta)}{s_{\tau-1}(v_{\tau-1})}\bar{f}_{l+2^{\tau-1}}$
end for
return $\bar{\mathbf{f}}$

III. The Proposed Algorithm

This section derives the fast error and erasure decoding algorithm.

Let $\epsilon = n-k$. An (n,k) RS code over $\mathbb{F}_{2^m}$ is defined by

$$\{(f(\omega_0), f(\omega_1), \ldots, f(\omega_{n-1})) \mid \deg(f(x)) < 2^m - \epsilon, \\ f(\omega_l) = 0, l = n, n+1, \ldots, 2^m-1\}.$$

Detailed explanation of this definition is referred to [5]. To simplify the discussion, we use the original codeword, that is

$$\mathbf{F} = (f(\omega_0), f(\omega_1), \ldots, f(\omega_{2^m-1})), \tag{2}$$

instead of the shortened codeword. Obviously, since $f(\omega_l) = 0$ for $n \le l < 2^m$, these positions need not be sent, and thus they are not disturbed by the noise.

The received vector may contain both errors and erasures. We say an error occurs if a different symbol is received for a transmitted symbol and an erasure occurs if the transmitted symbol is unreadable or lost at the receiver. The error pattern $\mathbf{E}$ is a vector $(e_0, e_1, \ldots, e_{2^m-1})$ such that a nonzero component implies an error occurs in that position. In the same manner, the erasure pattern $\mathbf{W}$ is a vector $(w_0, w_1, \ldots, w_{2^m-1})$ such that a nonzero component implies an erasure occurs. The received vector is written as

$$\mathbf{R} = \mathbf{F} + \mathbf{E} + \mathbf{W},$$

where $\mathbf{E}$ and $\mathbf{W}$ are the error and erasure patterns, respectively. After assigning 0 to all erased positions, by finite field Fourier transform, there exists $r(x)$ of degree $\deg(r(x)) < 2^m$ satisfying

$$\mathbf{R} = (r(\omega_0), r(\omega_1), \ldots, r(\omega_{2^m-1})). \tag{3}$$

The notation E_w represents the set of erasure locators known in advance. Let E_e be the set of error locators, i.e., $E_e = \{\omega_i \mid f(\omega_i) \neq r(\omega_i), \omega_i \notin E_w\}$. Clearly, we have $f(\omega_i) \neq r(\omega_i)$ if $\omega_i \in E_e$ and $r(\omega_i) = 0$ if $\omega_i \in E_w$. The decoder needs to find the set E_e, the error value $f(\omega_i) - r(\omega_i)$ for $\omega_i \in E_e$ and the erasure value $f(\omega_i)$ for $\omega_i \in E_w$. Without loss of generality, we assume that $|E_e| = g$, $|E_w| = h$ and $2g + h \le n - k$.

The error locator polynomial and erasure locator polynomial are defined as

$$\lambda(x) = \prod_{a \in E_e}(x-a), \gamma(x) = \prod_{a \in E_w}(x-a),$$

respectively. For all $\omega_l \in \mathbb{F}_{2^m}$, we have

$$f(\omega_l)\lambda(\omega_l)\gamma(\omega_l) = r(\omega_l)\lambda(\omega_l)\gamma(\omega_l).$$

This implies that, for all ω_l,

$$f(x)\lambda(x)\gamma(x) \equiv r(x)\lambda(x)\gamma(x) \pmod{(x-\omega_l)}.$$

According to the Chinese remainder theorem, we have

$$f(x)\lambda(x)\gamma(x) \equiv r(x)\lambda(x)\gamma(x) \pmod{s_m(x)}.$$

Hence, there exists $q(x)$ such that

$$f(x)\lambda(x)\gamma(x) = r(x)\lambda(x)\gamma(x) + q(x)s_m(x), \tag{4}$$

where $s_m(x) = x^{2^m} - x$. As $\deg(f(x)\lambda(x)\gamma(x)) < 2^m - \epsilon + g + h$, $\deg(r(x)\lambda(x)\gamma(x)) < 2^m + g + h$ and $\deg(s_m(x)) = 2^m$, we have $\deg(q(x)) < g + h$.

Let μ be the smallest integer such that $2^\mu \ge \epsilon$. By polynomial division, we have the following equations:

$$\begin{aligned} f(x)\lambda(x)\gamma(x) &= z_1(x)p_{2^m-2^\mu}\bar{X}_{2^m-2^\mu}(x) + \eta_f(x), \\ r(x) &= u_1(x)p_{2^m-2^\mu}\bar{X}_{2^m-2^\mu}(x) + \eta_r(x), \\ s_m(x) &= (s_\mu(x) + s_\mu(v_\mu))p_{2^m-2^\mu}\bar{X}_{2^m-2^\mu}(x) + \eta_s(x). \end{aligned}$$

When dividing $p_{2^m-2^\mu}\bar{X}_{2^m-2^\mu}(x)$ on both sides of (4) and keeping the quotients, we obtain

$$z_1(x) = u_1(x)\lambda(x)\gamma(x) + q(x)(s_\mu(x) + s_\mu(v_\mu)) + \theta(x),$$

where $\theta(x)$ corresponds to the quotient of $\lambda(x)\gamma(x)\eta_r(x) + q(x)\eta_s(x)$. Clearly, $\deg(\theta(x)) < g+h$. Let $z_2(x) = z_1(x) - \theta(x) - q(x)s_\mu(v_\mu)$. It follows that

$$z_2(x) = u_1(x)\lambda(x)\gamma(x) + q(x)s_\mu(x). \tag{5}$$

Here, we have $\deg(z_2(x)) < 2^\mu - \epsilon + g + h$. By polynomial division, we have the following equations:

$$\begin{aligned} z_2(x) &= z_3(x)\prod_{i=\epsilon-h}^{2^\mu-1}(x-\omega_i) + \eta_{z_2}(x), \\ u_1(x)\gamma(x) &= u_2(x)\prod_{i=\epsilon-h}^{2^\mu-1}(x-\omega_i) + \eta_{u_1}(x), \\ s_\mu(x) &= \prod_{i=0}^{\epsilon-h-1}(x-\omega_i)\prod_{j=\epsilon-h}^{2^\mu-1}(x-\omega_j). \end{aligned} \tag{6}$$

Hence, dividing both sides of (5) by $\prod_{i=\epsilon-h}^{2^\mu-1}(x-\omega_i)$ and keeping the quotients, we obtain that

$$z_3(x) = u_2(x)\lambda(x) + q(x)\prod_{i=0}^{\epsilon-h-1}(x-\omega_i) + \theta_1(x), \tag{7}$$

where $\theta_1(x)$ corresponds to the quotient of $\lambda(x)\eta_{u_1}(x)$. It is easy to verify that $\deg(\theta_1(x)) < g$. If we let $z_4(x) = z_3(x) - \theta_1(x)$, we then obtain the key equation

$$z_4(x) = u_2(x)\lambda(x) + q(x)\prod_{i=0}^{\epsilon-h-1}(x-\omega_i), \tag{8}$$

where $\deg(z_4(x)) < \deg(\lambda(x)) = g$.

The key equation (8) can be solved by the modular approach proposed in [6] and the coordinate vectors of $\lambda(x)$ and $z_4(x)$ for $\bar{\mathbb{X}}$ can be obtained.

Once we are given the error locator polynomial $\lambda(x)$, the error locators can be computed by

$$\mathrm{FFT}_{\bar{\mathbb{X}}}(\bar{\lambda}, \mu, \omega_i), i = 0, 1, \ldots, \lceil n/2^\mu \rceil - 1, \tag{9}$$

where $\bar{\lambda}$ is the coordinate vector of $\lambda(x)$ with respect to $\bar{\mathbb{X}}$.

It remains to compute the error and erasure values. The ordinary derivative of (4) is

$$\begin{aligned} &f^{'}(x)\lambda(x)\gamma(x) + f(x)\lambda^{'}(x)\gamma(x) + f(x)\lambda(x)\gamma^{'}(x) \\ &= r^{'}(x)\lambda(x)\gamma(x) + r(x)\lambda^{'}(x)\gamma(x) + r(x)\lambda(x)\gamma^{'}(x) + \\ &\quad q^{'}(x)s_m(x) + q(x)s_m^{'}(x). \end{aligned} \tag{10}$$

Recall that $s_m^{'}(x) = 1$. If $a \in E_e$, by substituting a into (10), we have

$$f(a)\lambda^{'}(a)\gamma(a) = r(a)\lambda^{'}(a)\gamma(a) + q(a).$$

Therefore, the error value

$$(f(a) - r(a)) = \frac{q(a)}{\lambda^{'}(a)\gamma(a)}.$$

If $a \in E_w$, by substituting a into (10), one has

$$f(a)\lambda(a)\gamma^{'}(a) = r(a)\lambda(a)\gamma^{'}(a) + q(a).$$

Since $r(a) = 0$ if a is an erasure, thus the erasure value

$$f(a) = \frac{q(a)}{\lambda(a)\gamma^{'}(a)}.$$

To sum up, if we let $\Lambda(x) = \lambda(x)\gamma(x)$, Forney's formula for computing both error values and erasure values is

$$\frac{q(a)}{\Lambda^{'}(a)}. \tag{11}$$

Notice that if $\lambda(x)$ and $z_4(x)$ are known, $q(x)$ can be computed by the key equation (8).

According the above discussion, it is easy to see that the decoding algorithm is capable of correcting g errors and h erasures for all $2g + h \leq n - k$.

IV. Implementation and Comparison

This section presents the details of implementation and analyzes the computational complexity. Then a comparison between the proposed method and other known algorithms is provided.

The whole description of the algorithm is shown in Algorithm 3.

Algorithm 3 Error-and-Erasure Decoding Algorithm

Input: Received vector $\mathbf{R} = \mathbf{F} + \mathbf{E} + \mathbf{W}$.
Output: The codeword $\mathbf{F}$.

1: Given E_w, compute the erasure locator polynomial $\gamma(x)$.
2: Compute the syndrome polynomial $u_1(x)$ according to (12).
3: Evaluate $u_1(x)$ at points $\omega_0, \omega_1, \ldots, \omega_{2^\mu-1}$ by Algorithm 1.
4: Given $\eta_{u_1}(\omega_l) = u_1(\omega_l)\gamma(\omega_l)$ for $l = \epsilon - h, \ldots, 2^\mu - 1$, compute $\eta_{u_1}(x)$ by $\mathrm{IFFT}_{\bar{\mathbb{X}}}$.
5: Evaluate $u_1(x), \gamma(x), \prod_{i=\epsilon-h}^{2^\mu-1}(x-\omega_i)$ and $\eta_{u_1}(x)$ at $\omega_{2^\mu}, \ldots, \omega_{2^{\mu+1}-1}$ and compute $u_2(x)$ according to (6).
6: Given $u_2(x)$, compute the error locator polynomial $\lambda(x)$ and the error evaluator polynomial $z_4(x)$ by the modular approach [6].
7: Find the error locations by (9).
8: Given $z_4(x), u_2(x), \lambda(x)$ and $\prod_{i=0}^{\epsilon-h-1}(x-\omega_i)$, compute $q(x)$.
9: Compute $\Lambda(x) = \lambda(x)\gamma(x)$.
10: Compute the error and erasure pattern by (11).
11: **return** $\mathbf{F} = \mathbf{R} + \mathbf{E} + \mathbf{W}$.

Given E_w, the computation of $\gamma(x)$ takes $O(h\log^2 h)$ field operations by repeatedly using the convolution theorem. As $h \leq n-k$, this step costs at most $O((n-k)\log^2(n-k))$ operations (Step 1 in Algorithm 3). Note that, by using the fast Walsh-Hadamard transform, [11] proposed a method of complexity $O(2^m \log 2^m)$ for computing $\gamma(x)$. However, it is more suitable for low-rate RS codes.

Let $\mathbf{R}_{i,\mu} = (r(\omega_{i\cdot 2^\mu}), r(\omega_{i\cdot 2^\mu+1}), \ldots, r(\omega_{i\cdot 2^\mu+2^\mu-1}))$, which is a sub-vector of $\mathbf{R}$. The coordinate vector of $u_1(x)$ with respect to $\bar{\mathbb{X}}$ is computed by (see [4])

$$\sum_{i=0}^{\lceil n/2^\mu \rceil - 1} \mathrm{IFFT}_{\bar{\mathbb{X}}}(\mathbf{R}_{i,\mu}, \mu, \omega_{i\cdot 2^\mu})/p_{2^m-2^\mu}. \tag{12}$$

This costs $O(n\log(n-k))$ field operations (Step 2 in Algorithm 3).

According to (6), we have $\eta_{u_1}(\omega_l) = u_1(\omega_l)\gamma(\omega_l)$ for $l = \epsilon-h, \epsilon-h+1, \ldots, 2^\mu-1$. As $\deg(\eta_{u_1}(x)) < 2^\mu-\epsilon+h$, thus $\eta_{u_1}(x)$ is determined uniquely. Hence, we first evaluate $u_1(x)$ and $\gamma(x)$ at $\omega_0, \omega_1, \ldots, \omega_{2^\mu-1}$ and then compute $\eta_{u_1}(x)$ by $\mathrm{IFFT}_{\bar{\mathbb{X}}}$. Next, $u_1(x)$, $\gamma(x)$, $\prod_{i=\epsilon-h}^{2^\mu-1}(x-\omega_i)$ and $\eta_{u_1}(x)$ are evaluated at $\omega_{2^\mu}, \omega_{2^\mu+1}, \ldots, \omega_{2^{\mu+1}-1}$. By (6), we can obtain $u_2(\omega_{2^\mu}), \ldots, u_2(\omega_{2^{\mu+1}-1})$. Therefore, $u_2(x)$ can be computed by $\mathrm{IFFT}_{\bar{\mathbb{X}}}$. This step costs $O((n-k)\log(n-k))$ operations (Step 3, 4 and 5 in Algorithm 3).

Given $u_2(x)$, the key equation (8) can be solved by frequency domain modular approach or fast modular approach (Algorithm 4 or 6 in [6]) and $\lambda(x)$ and $z_4(x)$ are obtained. This step costs at most $O((n-k)\log^2(n-k))$ operations (Step 6 in Algorithm 3). More detailed discussions are referred to [6].

The roots of $\lambda(x)$ are computed by (9). Notice that only the elements $\omega_0, \omega_1, \ldots, \omega_{n-1}$ are taken into account. This process needs $O(n\log(n-k))$ operations (Step 7 in Algorithm 3).

Given the evaluations of $z_4(x), u_2(x), \lambda(x)$ and $\prod_{i=0}^{\epsilon-h-1}(x-\omega_i)$ at $\omega_{2^\mu}, \omega_{2^\mu+1}, \ldots, \omega_{2^{\mu+1}-1}$, then by (8), the evaluations of $q(x)$ can be computed. So the polynomial $q(x)$ is obtained by $\mathrm{IFFT}_{\bar{\mathbb{X}}}$. This step takes $O((n-k)\log(n-k))$ operations (Step 8 in Algorithm 3). Recall that we already have the evaluations of $\lambda(x)$ and $\gamma(x)$. Thus, $\Lambda(x)$ can be computed by multiplying them and executing $\mathrm{IFFT}_{\bar{\mathbb{X}}}$. Finally, the error and erasure values are computed by (11). This process needs $O((n-k)\log(n-k))$ operations (Step 9 and 10 in Algorithm 3). Hence, the whole decoding algorithm has a complexity of $O(n\log(n-k)+(n-k)\log^2(n-k))$.

Remarks: For RS codes with various parameters, the decoding procedures may be slightly different. The reason is that if the polynomial to be evaluated has a small degree, a straightforward computation is more efficient than $\mathrm{FFT}_{\bar{\mathbb{X}}}$. This also holds for $\mathrm{IFFT}_{\bar{\mathbb{X}}}$. On the other hand, if the number of points to be transformed is not a power of two, the $\mathrm{FFT}_{\bar{\mathbb{X}}}$ and $\mathrm{IFFT}_{\bar{\mathbb{X}}}$ should be slightly modified. More discussions have been provided in [12], where the $\mathrm{FFT}_{\bar{\mathbb{X}}}$ and $\mathrm{IFFT}_{\bar{\mathbb{X}}}$ are generalized to arbitrary points. In practice, one should carefully design the decoding algorithm for specific RS codes to obtain further improvement.

Table I compares the proposed method and the algorithm in [7] when decoding $\lfloor (n-k)/2 \rfloor$ errors. For decoding $\mathrm{RS}(255, 223)$ codes, the proposed method saves 32% field operations. Furthermore, the proposed method can be used for RS codes with arbitrary parameters, which is more flexible than the technique in [7]. Table II provides the number of field operations needed for $\mathrm{RS}(255, 223)$ when decoding various numbers of errors and erasures. Obviously, more operations are needed if more errors occur since there are more iterations in the key equation solver.

TABLE I
COMPLEXITY COMPARISON BETWEEN ALGORITHM IN [7] AND THE PROPOSED METHOD

Codes	Algorithm in [7]			Proposed method		
	Mul.	Add.	Div.	Mul.	Add.	Div.
RS(255, 223)	9,476	12,892	0	6,458	8,691	148
RS(511, 447)	41,606	54,064	0	18,714	23,451	212
RS(1023, 895)	131,741	164,211	0	78,022	88,657	608

TABLE II
COMPLEXITY OF THE PROPOSED METHOD WHEN DECODING VARIOUS NUMBERS OF ERRORS AND ERASURES FOR RS(255, 223)

g \ h	$h=24$			$h=16$		
	Mul.	Add.	Div.	Mul.	Add.	Div.
$g=0$	4,468	7,371	156	3,750	6,851	148
$g=2$	4,977	8,043	158	4,345	7,491	150
$g=4$	5,362	8,467	160	4,769	7,883	152

V. CONCLUSION

An important research issue is designing an efficient algorithm for the error-and-erasure decoding of Reed-Solomon codes. A fast error and erasure decoding algorithm of complexity $O(n\log(n-k)+(n-k)\log^2(n-k))$ is proposed for RS codes in this work. This algorithm also suits practical codes such as RS(255, 223). The proposed algorithm is very efficient under software implementation; however, implementing it in hardware is one potential future research direction.

REFERENCES

[1] X. Wu, Z. Yan, and J. Lin, "Reduced-complexity decoders of long Reed–Solomon codes based on composite cyclotomic Fourier transforms," vol. 60, no. 7, pp. 3920–3925, 2012.

[2] S. V. Fedorenko, "Efficient syndrome calculation via the inverse cyclotomic discrete Fourier transform," vol. 26, no. 9, pp. 1320–1324, 2019.

[3] W. K. Kadir, H.-Y. Lin, and E. Rosnes, "Efficient interpolation-based decoding of reed-solomon codes," in *2023 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2023, pp. 997–1002.

[4] 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," vol. 62, no. 10, pp. 5343–5358, Oct. 2016.

[5] N. Tang and Y. Lin, "Fast encoding and decoding algorithms for arbitrary (n, k) Reed–Solomon codes over $\mathbb{F}_{2^m}$," vol. 24, no. 4, pp. 716–719, April 2020.

[6] N. Tang and Y. S. Han, "A new decoding method for Reed-Solomon codes based on FFT and modular approach," *IEEE Transactions on Communications*, vol. 70, no. 12, pp. 7790–7801, 2022.

[7] T. Truong, P. Chen, L. Wang, and T. Cheng, "Fast transform for decoding both errors and erasures of Reed-Solomon codes over GF(2^m) for $8 \leq m \leq 10$," *IEEE Transactions on Communications*, vol. 54, no. 2, pp. 181–186, 2006.

[8] Y. S. Han, C. Chen, S.-J. Lin, and B. Bai, "On fast Fourier transform-based decoding of Reed-Solomon codes," *Int. J. Ad Hoc and Ubiquitous Computing*, vol. 36, no. 3, pp. 180–187, 2021.

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

[10] 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," vol. 62, no. 11, pp. 6284–6299, Nov. 2016.

[11] G. Robinson, "Logical convolution and discrete Walsh and Fourier power spectra," *IEEE Transactions on Audio and Electroacoustics*, vol. 20, no. 4, pp. 271–280, 1972.

[12] N. Tang and Y. S. Han, "New decoding of Reed-Solomon codes based on FFT and modular approach," 2022, https://arxiv.org/abs/2207.11079.