Novel Burst Error Correction Algorithms for Reed-Solomon Codes

Yingquan Wu, *Senior Member, IEEE*

***Abstract*—In this paper, we present three novel burst error correcting algorithms for an (n, k) Reed-Solomon code. The algorithmic complexities are of the same order as error-and-erasure decoding, $O(rn)$, where $r = n - k$. In particular, their hardware implementation shares elements of Blahut error-and-erasure decoding. In contrast, all existing single-burst error correcting algorithms, which are equivalent to the proposed first algorithm, have complexity $O(r^2n)$. The first algorithm corrects the shortest single-burst with length f up to $r - 1$. The algorithm follows the key characterization that the ending locations of all candidate bursts can be purely determined by the roots of a polynomial which is a linear function of syndromes, and moreover, the shortest burst is associated with the longest sequence of consecutive roots. The algorithmic miscorrection probability is bounded by $q^{-(r-1-f)}$, where q denotes the field size. The second algorithm extends the first one to correct the shortest burst with length $f \leq r - 3$ and additionally a random symbol error. The algorithmic miscorrection probability is bounded by $q^{-(r-3-f)}$. The third algorithm probabilistically corrects the shortest burst with length $f \leq r - 1 - 2\delta$ and additionally δ (a small constant) random symbol errors. The algorithmic miscorrection and failure probabilities are both bounded by $q^{-(r-1-2\delta-f)}$. Our simulation results for (60, 40) and (30, 16) shortened Reed-Solomon codes verify that the miscorrection probability for three algorithms and the failure probability for the third algorithm all decay exponentially (at the rate of q^{-1}) with respect to the length of burst.**

***Index Terms*—Burst error correction, Chien search, consecutive roots, exponential decay, miscorrection probability, Reed-Solomon codes.**

I. Introduction

MANY channels such as wireless, magnetic recording, and recently, flash channels tend to suffer from degradation of data in which their reliability deteriorates significantly over time to a degree that compromises data integrity. Often, due to physical/mechanical failures, errors are clustered into bursts. In such cases, error-burst correction codes are employed to protect application data. Reed-Solomon codes are often employed due to their "maximum distance separable" property and moreover to the existence of efficient algorithms to correct up to half the minimum distance of random errors. In this paper, we will devise efficient algorithms for Reed-Solomon codes that correct a long burst with length close to minimum distance and a small number of random errors.

A Reed-Solomon code (n, k) defined over GF(q) is capable of correcting up to $t \triangleq \lfloor \frac{r}{2} \rfloor$ random symbol errors, where $r \triangleq n-k$. However, a single burst of length up to $r-1$ can be identified, with certain miscorrection probability [2], [3], [5], [12]. In [2], the single burst is trapped by a re-encoder shift register. In [3], Chen and Owsley first utilized syndromes to determine all single-bursts. The method is theoretically straightforward but computationally costly, due to having to solve a possibly singular linear equation system. In [5], Dawson and Khodkar observed that the complicated equation system in [3] can be obtained alternatively from the well-known key equation, and presented a simplified method to determine all single-bursts. Nevertheless, the method still performs trial-and-error for all possible burst lengths. In [12], Yin, *et al.* presented a cyclic shift method, as in [2]. Forney's erasure-only decoding is used in [12], whereas re-encoding is employed in [2]. Overall, all above algorithms require computational complexity $O(r^2n)$.

Section II introduces basics of Reed-Solomon codes. Section III presents a novel single-burst correction algorithm which has complexity $O(rn)$. The implementation of the algorithm shares elements of Blahut unified error-and-erasure decoding [7]. Its core is based on new insights that all candidate single-bursts with length up to $r - 1$, are governed by the roots of a special polynomial which is a linear function of syndromes, and that the desired shortest single-burst is determined by the longest sequence of the consecutive roots. The new method effectively circumvents the need for trial-and-error on the burst length. The algorithmic miscorrection probability is bounded by $q^{-(r-1-f)}$, where q denotes the field size and f the length of the true burst. It is worth noting that Reed-Solomon codes along with the proposed algorithm compare favorably to the existing burst-correcting codes [8], [10]. Finally, the proposed (frequency-domain) approach can be viewed as the dual of the (time-domain) zero-span approach presented in [11], where circulant parity-check matrices are utilized to correct burst errors for general cyclic codes.

Section IV extends the algorithm presented in Section III to correct a long burst with length f up to $r - 3$ and additionally a random symbol error, with minor algorithmic modifications while maintaining the same complexity. Its miscorrection probability is bounded by $q^{-(r-3-f)}$. Finally, Section V presents an algorithm to probabilistically correct a long burst with length f up to $r - 1 - 2\delta$ and up to δ random symbol errors (where δ is a small constant). Its miscorrection and failure probabilities are both bounded by $q^{-(r-1-2\delta-f)}$.

Manuscript received January 29, 2009; revised March 01, 2011; accepted July 09, 2011. Date of current version February 08, 2012. The material in this paper was presented in part at the 47th Allerton Conference on Communication, Control, and Computing, Monticello, IL, October 2009. Filed under U.S. Patent No. 7,962,839, June 14, 2011.

The author is with Link_A_Media Devices Corp., Santa Clara, CA 95051 USA (e-mail: yingquan_wu@yahoo.com).

Communicated by Gadiel Seroussi, Associate Editor for Coding Theory.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIT.2011.2173623

II. Reed-Solomon Codes

Let a (possibly shortened) Reed-Solomon code $\mathcal{C}(n,k)$ ($r \triangleq n-k$) over GF(q) ($n<q$) be defined by the generator polynomial

$$G(x) \triangleq \prod_{i=0}^{r-1}(x-\alpha^{b+i})$$

where α is a primitive element of GF(q) and b denotes the starting power (For simplicity, we always choose $b=1$ for examples throughout the paper). A polynomial of degree less than n is a codeword polynomial if and only if it is divisible by the generator polynomial $G(x)$. As can be readily seen, a codeword polynomial $C(x)$ satisfies

$$C(\alpha^{b+i})=0, \quad i=0,1,2,\ldots,n-k-1.$$

In practice, a message polynomial $M(x)$ of degree up to $k-1$ is systematically encoded to a codeword polynomial $C(x)$ via a linear feedback shift register circuit which divides $x^rM(x)$ by $G(x)$ and results in its remainder polynomial $\Psi(x)$ of degree up to $r-1$ [1]

$$C(x) \triangleq x^rM(x)-\Psi(x).$$

(Throughout the paper we do not distinguish between a vector $\mathbf{A}=[A_0,A_1,A_2,\ldots,A_l]$ and its polynomial representation $A(x)=A_0+A_1x+A_2x^2+\cdots+A_lx^l$.) The minimum Hamming distance of the code is $d_{\min}=n-k+1$, a feature known as maximum distance separable [1].

Let $C(x)$ denote the transmitted codeword polynomial and $R(x)$ the received word polynomial after appropriate channel quantization. Next, we briefly introduce the formulation of frequency-domain decoding. Let syndromes be defined as follows

$$S_i=R(\alpha^{i+b}), \quad i=0,1,2,\ldots,r-1.$$

Let e denote the (unknown) number of errors, $X_1=\alpha^{i_1}, X_2=\alpha^{i_2},\ldots,X_e=\alpha^{i_e}$ denote error locators (corresponding to error locations $i_i,i_2,\ldots,i_e$, respectively), and $Y_1,Y_2,\ldots,Y_e$ denote the corresponding error magnitudes. Define the *syndrome* polynomial

$$S(x) \triangleq S_0+S_1x+S_2x^2+\cdots+S_{r-1}x^{r-1} \tag{1}$$

the *error locator* polynomial

$$\Lambda(x) \triangleq \prod_{i=1}^{e}(1-X_ix)=1+\Lambda_1x+\Lambda_2x^2+\cdots+\Lambda_ex^e \tag{2}$$

and the *error evaluator* polynomial

$$\begin{aligned}\Omega(x) &\triangleq \sum_{i=1}^{e}Y_iX_i^b\prod_{j=1,j\neq i}^{e}(1-X_jx)\\ &=\Omega_0+\Omega_1x+\Omega_2x^2+\cdots+\Omega_{e-1}x^{e-1}.\end{aligned} \tag{3}$$

The three polynomials satisfy the key equation [1]

$$\Omega(x)=\Lambda(x)S(x) \pmod{x^r}. \tag{4}$$

The Berlekamp-Massey algorithm can be used to solve (4) and correct up to $\lfloor \frac{r}{2}\rfloor$ errors [1].

When ρ ($\rho \le r$) erasures, $\alpha_0,\alpha_1,\ldots,\alpha_{\rho-1}$, are present, error-and-erasure decoding succeeds if $2e+\rho \le r$, where e denotes the number of erroneous symbols. In other words, error-and-erasure decoding is more powerful than error-only decoding if more than half the number of the erased symbols are truly erroneous. The Blahut Error-and-Erasure Decoding Algorithm treats errors and erasures in a unified fashion which results in great hardware amenity. The pseudo code of the Blahut Error-and-Erasure Decoding Algorithm is given below [1]:

Blahut Error-and-Erasure Decoding Algorithm:

```
• Inputs: $\mathbf{S}=[S_0,S_1,\ldots,S_{r-1}]$, $[\alpha_0,\alpha_1,\ldots,\alpha_{\rho-1}]$.
• Initialization: $\Lambda^{(0)}(x)=B^{(0)}(x)=1$, and $L=0$
• For $j=0,1,2,\ldots,\rho-1$, do:
  — Compute $\Lambda^{(j+1)}(x) \leftarrow (1-\alpha_jx)\Lambda^{(j)}(x)$
  — Set $B^{(j+1)}(x) \leftarrow \Lambda^{(j+1)}(x)$, and $L \leftarrow L+1$
• For $j=\rho,\rho+1,\rho+2,\ldots,r-1$, do:
  • Compute $\Delta^{(j)} \leftarrow \sum_{i=0}^{L}\Lambda_i^{(j)} \cdot S_{j-i}$
  • Compute $\Lambda^{(j+1)}(x) \leftarrow \Lambda^{(j)}(x)-\Delta^{(j)} \cdot xB^{(j)}(x)$
  • If $\Delta^{(j)} \neq 0$ and $2L \le j+\rho$, then
        * Set $B^{(j+1)}(x) \leftarrow (\Delta^{(j)})^{-1}\Lambda^{(j)}(x)$ and $L \leftarrow j+\rho+1-L$
  • Else
        * Set $B^{(j+1)}(x) \leftarrow xB^{(j)}(x)$
  • Compute $\Omega^{(r)}(x) \leftarrow \Lambda^{(r)}(x)S(x) \pmod{x^r}$
  • Outputs: $\Lambda^{(r)}(x)$, $\Omega^{(r)}(x)$, $L$
```

Note that in the above description, we used the superscript "$^{(j)}$" to stand for the jth iteration and the subscript "$_i$" to stand for the i-th coefficient.

Finally, we note that each symbol location of a codeword corresponds to a different locator. Specifically, let a codeword be $\mathbf{c}=[c_0,c_1,c_2,\ldots,c_{n-1}]$, then the i-th symbol c_i corresponds to the locator α^i, i.e., an error occurring on c_i is associated with the root α^{-i} of the error locator polynomial $\Lambda(x)$ (note that an error locator is the inverse of the corresponding root of $\Lambda(x)$). The Chien search is an efficient circuit implementation method to perform root search in the sequential order of $1,\alpha^{-1},\alpha^{-2},\ldots,\alpha^{-(n-1)}$ [4].

III. Correction Algorithm for a Long Burst

We denote by $\mathsf{B}(s,f,\mathbf{u})$ a burst of errors $\mathbf{u} \triangleq [u_0,u_1,\ldots,u_{f-1}]$ that starts from s and has length f, such that, (i) more than half the elements of the error vector $\mathbf{u}$ are nonzeros, (ii) the boundary locations s and $s+f-1$ are erroneous, i.e., $u_0 \neq 0$, $u_{f-1} \neq 0$, and (iii) it is maximal in the sense that no super interval $[s',s'+f'-1] \supset [s,s+f-1]$ satisfies (i) and (ii). Note when less than $\lceil \frac{f}{2}\rceil$ elements of $\mathbf{u}$ are nonzeros, the errors are more convenient to be treated as random errors. On the other hand, a burst with length $f \ge r$ is uncorrectable, as there always exists a solution for any r consecutive locations (via erasure-only decoding). Therefore, the limit on the burst correction capability of an (n,k) Reed-Solomon code is $r-1$. In this section, we devise an

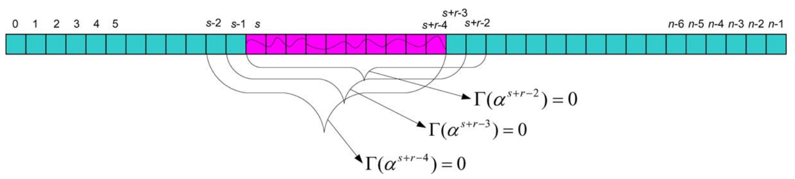
0 1 2 3 4 5
s-2 s-1 s
s+r-3
s+r-4 s+r-2
n-6 n-5 n-4 n-3 n-2 n-1
$\Gamma(\alpha^{s+r-2})=0$
$\Gamma(\alpha^{s+r-3})=0$
$\Gamma(\alpha^{s+r-4})=0$

Fig. 1. An example of three consecutive roots associated with a single burst of length $f = r - 3$.

efficient algorithm to correct a long burst with length up to the limit $r - 1$.

The following lemma is implied by the key equation (4).

Lemma 1: If a single burst $\mathsf{B}\ (s, f, \mathbf{u})$ has occurred such that the burst interval

$$[s, s + f - 1] \subseteq [s', s' + r - 2], \tag{5}$$

then, regardless of the actual burst length f

$$S_{r-1}\Lambda_0 + S_{r-2}\Lambda_1 + \cdots + S_0\Lambda_{r-1} = 0 \tag{6}$$

where $\Lambda(x) = (1 - \alpha^{s'}x)(1 - \alpha^{s'+1}x)\ldots(1 - \alpha^{s'+r-2}x)$.

Proof: By (4), (6) holds for a burst of length $r-1$. When the length f is less than $r - 1$, it can always be enlarged (trivially) to a burst of length $r - 1$ by including additional $r - 1 - f$ error-free symbols at the beginning (of actual burst), the end, or both.

We observe that the burst interval $[s, s + f - 1]$ is the largest common subset of all burst intervals $[s', s' + r - 2]$ satisfying (5), which immediately leads to a new sight of single-burst, as stated by the following lemma. Its proof follows straightforwardly from Lemma 1.

Lemma 2: A single burst $\mathsf{B}\ (s, f, \mathbf{u})$ has occurred if and only if

$$\begin{cases} S_{r-1}\Lambda_0^{(0)} + S_{r-2}\Lambda_1^{(0)} + \cdots + S_0\Lambda_{r-1}^{(0)} = 0 \\ S_{r-1}\Lambda_0^{(1)} + S_{r-2}\Lambda_1^{(1)} + \cdots + S_0\Lambda_{r-1}^{(1)} = 0 \\ \vdots \\ S_{r-1}\Lambda_0^{(r-1-f)} + S_{r-2}\Lambda_1^{(r-1-f)} + \cdots + S_0\Lambda_{r-1}^{(r-1-f)} = 0 \end{cases} \tag{7}$$

where $\Lambda^{(i)}(x) = (1 - \alpha^{s-i}x)(1 - \alpha^{s-i+1}x)\ldots(1 - \alpha^{s-i+r-2}x)$ for $i = 0, 1, 2, \ldots, r - 2$.

Define

$$\bar{\Lambda}^{(p)}(x) \triangleq \prod_{i=-(p-1)}^{0} (1 - \alpha^i x) \tag{8}$$

where p is related to the maximum allowable length of burst (it is $r - 1$ in the formulation of a single burst, while varying in different scenarios).

Define

$$\Gamma(x) \triangleq S_{r-1}\bar{\Lambda}_0^{(r-1)} + S_{r-2}\bar{\Lambda}_1^{(r-1)} \cdot x + \cdots + S_0\bar{\Lambda}_{r-1}^{(r-1)} \cdot x^{r-1} \tag{9}$$

which yields

$$\begin{aligned} &S_{r-1}\Lambda_0^{(i)} + S_{r-2}\Lambda_1^{(i)} + S_{r-3}\Lambda_2^{(i)} + \cdots + S_0\Lambda_{r-1}^{(i)} \\ &= S_{r-1}\bar{\Lambda}_0^{(r-1)} + S_{r-2}\bar{\Lambda}_1^{(r-1)}\alpha^{s-i+r-2} \\ &\quad + \cdots + S_0\bar{\Lambda}_{r-1}^{(r-1)}\alpha^{(r-1)(s-i+r-2)} \\ &= \Gamma(\alpha^{s-i+r-2}). \end{aligned} \tag{10}$$

According to (10), Lemma 2 is translated to

Theorem 1: A single burst $\mathsf{B}\ (s, f, \mathbf{u})$ has occurred if and only if $\Gamma(x)$ as defined in (9) has $r - f$ consecutive roots $\alpha^{s+f-1}, \alpha^{s+f}, \ldots, \alpha^{s+r-2}$.

A pictorial illustration of Theorem 1 is shown in Fig. 1.

We observe that the first index of the consecutive root sequence corresponds to the end of the burst. For this reason, we shall keep track of the end of a burst, instead of its start. Also, note that $\Gamma(x)$ has degree up to $r - 1$ and thus may have up to $r - 1$ valid roots, i.e., there are $r - 1$ candidate single-bursts in the worst case. Our objective, naturally, is to determine the shortest one. Theorem 1 also indicates that the root search starts from $\alpha^{\lceil r/2 \rceil}$ while ending at α^{n+r-3}. Note when n is close to field size q, it is possible that

$$n + r - 3 - (q - 1) \geq \lceil \frac{r}{2} \rceil.$$

In this case, the code is treated as a cyclic code by padding some zeros, where a burst happening at the beginning and a burst at the end are regarded as a single (merged) burst, and the entire field $\mathrm{GF}(q)\backslash\{0\}$ is searched.

One way to compute error magnitudes is re-encoding through cyclic shifting of a received word. It is simple but incurs a long delay for high-rate codes. An alternative way, which avoids long delay, is through Forney's formula [6]

$$Y_i = -\frac{\Omega(X_i^{-1})}{X_i^{-(b-1)}\Lambda'(X_i^{-1})}$$

where X_i^{-1} denotes a root of $\Lambda(x)$, and $\Lambda'(x)$ the formal derivative of $\Lambda(x)$. Given a burst $\mathsf{B}\ (s, f, \mathbf{u})$, the error locator polynomial $\Lambda(x)$ can be computed by

$$\Lambda(x) = \bar{\Lambda}^{(f)}(\alpha^{s+f-1}x)$$

and the error evaluator polynomial $\Omega(x)$ is then computed through (4).

We are now ready to present the algorithmic procedure as follows.

Algorithm 1: Correcting a Long Burst:

1) Precompute the coefficients of $\bar{\Lambda}^{(r-1)}(x)$.
2) Compute syndromes $S_0, S_1, S_2, \ldots, S_{r-1}$.
3) Apply the Berlekamp-Massey algorithm. If successful, then stop.
4) Compute the coefficients of $\Gamma(x)$ as defined in (9).

5) Apply the Chien search to $\Gamma(x)$ to obtain the shortest single-burst by tracking the longest consecutive root sequence. Record the end position of the desired shortest burst.
6) Compute burst error magnitudes using Forney's formula or re-encoding.

Algorithm 1 has computational complexity $O(rn)$, which is in the same order as error-and-erasure decoding. Moreover, it shares the computation units of the Blahut error-and-erasure decoding [7]. This remains true for the subsequent two derived algorithms. Therefore, the proposed algorithms exhibit great hardware amenity. We next present an example to illustrate the procedure of Algorithm 1.

Example 1: Consider the (24, 16) shortened Reed-Solomon code over $\mathrm{GF}(2^5)$ (in this and subsequent examples, the generator polynomials $G(x)$ are always chosen with starting power $b = 1$, for simplicity). Let the transmitted codeword be

$$\mathbf{c} = [\alpha^8, \alpha^3, \alpha^{21}, \alpha^8, \alpha^{21}, \alpha^0, \alpha^3, \alpha^{18}, \alpha^1, \alpha^{27}, \alpha^1, \alpha^{25}, \alpha^{12}, \alpha^{26}, \alpha^{10}, \alpha^{10}, \alpha^{17}, \alpha^{28}, \alpha^2, \alpha^{29}, \alpha^{18}, \alpha^3, \alpha^8, \alpha^{22}]$$

and the received word

$$\mathbf{r} = \left[\alpha^8, \underline{\alpha^{26}, \alpha^1, \alpha^2, \alpha^{17}, \alpha^{14}, \alpha^{29}}, \alpha^{18}, \alpha^1, \alpha^{27}, \alpha^1, \alpha^{25}, \alpha^{12}, \alpha^{26}, \alpha^{10}, \alpha^{10}, \alpha^{17}, \alpha^{28}, \alpha^2, \alpha^{29}, \alpha^{18}, \alpha^3, \alpha^8, \alpha^{22}\right].$$

Its syndrome values are computed as:

$$\mathbf{S} = [\alpha^{28}, \alpha^{12}, \alpha^4, \alpha^4, \alpha^{27}, \alpha^{13}, \alpha^{28}, \alpha^{17}]$$

which yield

$$\Gamma(x) = \alpha^7 x^7 + \alpha^{26} x^6 + \alpha^{10} x^5 + \alpha^{16} x^4 + \alpha^{11} x^3 + \alpha^{28} x^2 + \alpha^{26} x^1 + \alpha^{17}.$$

The Chien search identifies five valid roots, α^6, α^7, α^9, α^{16}, α^{22}, respectively. By Theorem 1, the first two consecutive roots result in a burst of length 6, such that

$$\mathbf{c}_1 = [\alpha^8, \underline{\alpha^3, \alpha^{21}, \alpha^8, \alpha^{21}, \alpha^0, \alpha^3}, \alpha^{18}, \alpha^1, \alpha^{27}, \alpha^1, \alpha^{25}, \alpha^{12}, \alpha^{26}, \alpha^{10}, \alpha^{10}, \alpha^{17}, \alpha^{28}, \alpha^2, \alpha^{29}, \alpha^{18}, \alpha^3, \alpha^8, \alpha^{22}],$$

which is the transmitted codeword. The root α^9 leads to a candidate codeword

$$\mathbf{c}_2 = [\alpha^8, \alpha^{26}, \alpha^1, \underline{\alpha^{27}, \alpha^{15}, \alpha^3, \alpha^{20}, \alpha^0, \alpha^{24}, \alpha^9}, \alpha^1, \alpha^{25}, \alpha^{12}, \alpha^{26}, \alpha^{10}, \alpha^{10}, \alpha^{17}, \alpha^{28}, \alpha^2, \alpha^{29}, \alpha^{18}, \alpha^3, \alpha^8, \alpha^{22}].$$

where the underlined symbols are corrected. The root α^{16} leads to a candidate codeword

$$\mathbf{c}_3 = \left[\alpha^8, \alpha^{26}, \alpha^1, \alpha^2, \alpha^{17}, \alpha^{14}, \alpha^{29}, \alpha^{18}, \alpha^1, \alpha^{27}, \underline{\alpha^{15}, \alpha^9}, \underline{0, \alpha^5, \alpha^3, \alpha^9, \alpha^{30}}, \alpha^{28}, \alpha^2, \alpha^{29}, \alpha^{18}, \alpha^3, \alpha^8, \alpha^{22}\right].$$

The last root α^{22} results in a candidate codeword

$$\mathbf{c}_4 = \left[\alpha^8, \alpha^{26}, \alpha^1, \alpha^2, \alpha^{17}, \alpha^{14}, \alpha^{29}, \alpha^{18}, \alpha^1, \alpha^{27}, \alpha^1, \alpha^{25}, \alpha^{12}, \alpha^{26}, \alpha^{10}, \alpha^{10}, \underline{\alpha^2, \alpha^{23}, \alpha^{21}, \alpha^{17}, \alpha^{15}, \alpha^{28}, \alpha^{15}}, \alpha^{22}\right].$$

The algorithm picks the shortest burst, which corresponds to the longest consecutive root sequence, thereby successfully retrieves the transmitted codeword.

Given a genuine burst $\mathsf{B}\ (s, f, \mathbf{u})$ that results in syndrome polynomial $S(x)$, we call another burst $\mathsf{B}\ (s', f', \mathbf{u}')$ a *spurious* burst if it yields the same syndrome polynomial $S(x)$. Noting that Algorithm 1 picks the first shortest burst out of all candidate bursts, a spurious burst $\mathsf{B}\ (s', f', \mathbf{u}')$ causes Algorithm 1 to miscorrect if $f' < f$ or $f' = f$ & $s' < s$. For analytical simplicity, we conservatively claim that a miscorrection has happened if a spurious burst $\mathsf{B}\ (s', f', \mathbf{u}')$ satisfies $f' \le f$.

We denote by $P_m(\mathsf{B}\ (s, f, \mathbf{u}) | f)$ $(f > \frac{r}{2})$ the miscorrection probability of Algorithm 1 with respect to a random burst $\mathsf{B}\ (s, f, \mathbf{u})$ with given length f, where s is uniformly distributed within $[0, n - f]$ and $\mathbf{u}$ is uniformly distributed over $\{\mathbf{u} \in \mathrm{GF}(q)^f \mid u_0 \neq 0, u_{f-1} \neq 0, w(\mathbf{u}) > f/2\}$, where $w(\cdot)$ denotes Hamming weight. Formally, $P_m(\mathsf{B}\ (s, f, \mathbf{u}) | f)$ is defined as the probability that there exists a spurious burst $\mathsf{B}\ (s', f', \mathbf{u}')$ which satisfies $f' \le f$ while yielding the same syndromes as $\mathsf{B}\ (s, f, \mathbf{u})$.

We proceed to present a upper bound on $P_m(\mathsf{B}\ (s, f, \mathbf{u})\, | f)$. Let $\mathbf{c}$ be the transmitted codeword, and $\mathbf{y}$ be the received word corrupted by a single burst $\mathsf{B}\ (s, f, \mathbf{u})$ such that

$$\begin{aligned} &y_i = c_i, i \notin [s, s + f - 1] \\ &\mathbf{u} = [c_s, c_{s+1}, \ldots, c_{s+f-1}] \oplus [y_s, y_{s+1}, \ldots, y_{s+f-1}] \\ &y_s \neq c_s, y_{s+f-1} \neq c_{s+f-1}. \end{aligned}$$

By definition, more than $\lceil \frac{f}{2} \rceil$ symbols are erroneous (otherwise the errors are considered as random), the overall number of possibilities is upper bounded by $(q-1)^{\lceil \frac{f}{2} \rceil} q^{\lfloor \frac{f}{2} \rfloor} \approx q^f$. Now consider the subset that contains a spurious burst $\mathsf{B}\ (s', f', \mathbf{u}')$ $(f' \le f)$, and for analytical simplicity, at same time satisfies $f' + f \ge r + 2$ (otherwise the Berlekamp-Massey algorithm succeeds). Note that (s', f') must obey

$$s + r - 1 < s' + f' - 1 \text{ or } s' + r - 1 < s + f - 1$$

which is equivalent to

$$(s' - s) + f' > r \text{ or } (s - s') + f > r,$$

otherwise, the two resulting consecutive root sequences are jointly consecutive and thus Theorem 1 is violated. We first investigate the case that the two bursts $\mathsf{B}\ (s, f, \mathbf{u})$ and $\mathsf{B}\ (s', f', \mathbf{u}')$ are non-overlapping. Without loss of generality, we assume that $s' > s$. The pictorial analysis is illustrated in Fig. 2. Define $\mathbf{y}'$ such that

$$\begin{array}{ll} y'_i = y_i = c_i, & i \notin [s, s + f - 1] \cup [s', s' + f' - 1] \\ y'_i = y_i, & i \in [s + f' - r,\ s + f - 1] \\ y'_i = 0, & i \in [s,\ s + f' - r - 1] \cup [s', s' + f' - 1]. \end{array}$$

Applying erasure-only decoding on $\mathbf{y}'$ results in a codeword, say $\mathbf{c}'$. Define a received word $\mathbf{y}''$

$$\begin{array}{ll} y''_i = y_i = c_i, & i \notin [s,\ s + f - 1] \\ y''_i = y_i, & i \in [s + f' - r,\ s + f - 1] \\ y''_i = c'_i, & i \in [s,\ s + f' - r - 1]. \end{array}$$

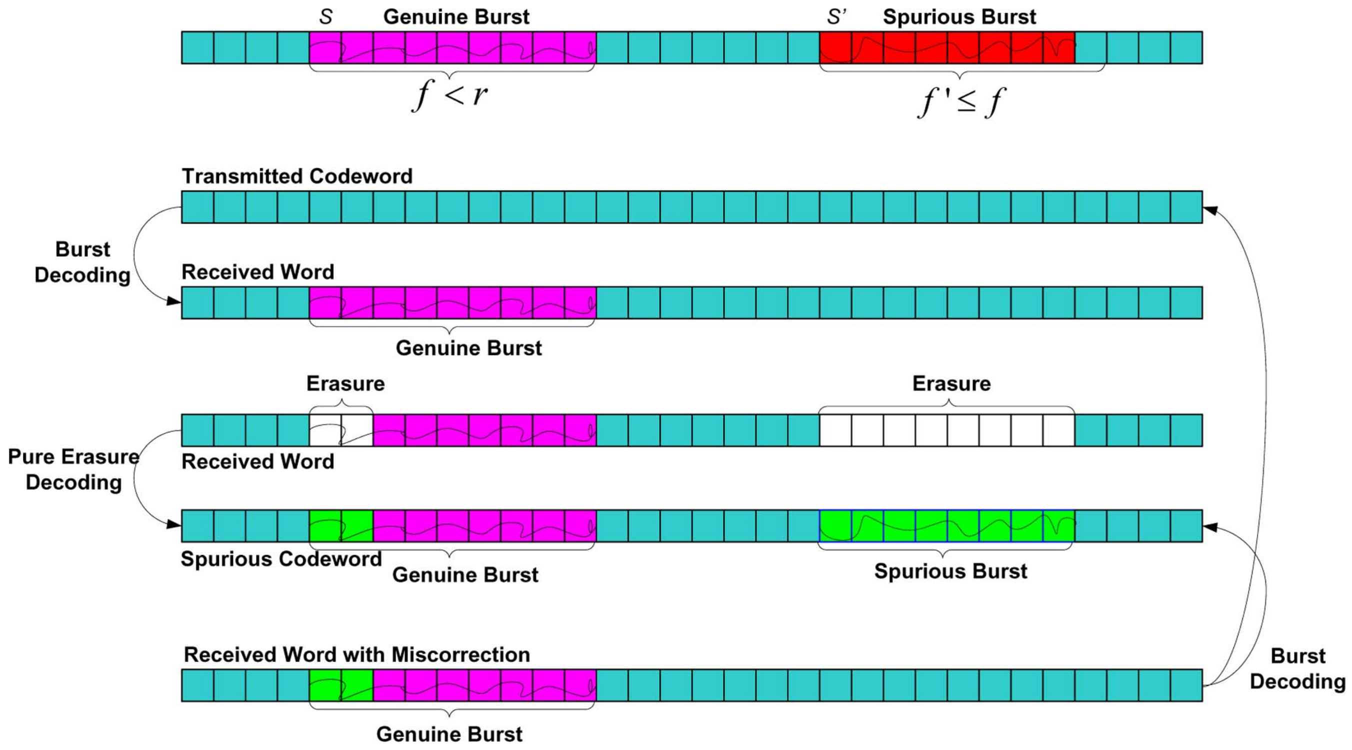

Fig. 2. An explicit construction of non-overlapping case where a burst-corrupted word leads to miscorrection.

Clearly, $\mathbf{y}''$ is a special case of $\mathbf{y}$ suffering a burst at $[s, s+f-1]$. Moreover, $\mathbf{y}''$ exhibits a spurious burst at $[s', s'+f'-1]$ (leading to the miscorrection toward $\mathbf{c}'$). We observe that there are at most $q^{f+f'-r}$ different $\mathbf{y}''$ following the freedom of the above second assignment $y''_i = y_i, i \in [s+f'-r, s+f-1]$, wherein the phrase "at most" is due to the constraint that at least half the symbols within the spurious burst interval $[s', s'+f'-1]$ are different from the corresponding received symbols. In addition, there are at most $n-(f+f')$ different choices of s', where "$-$" is due to the condition of nonoverlapping with genuine burst and "at most" is due to the ignorance of word end. Hence, there are at most

$$(n-(f+f'))q^{f+f'-r} \tag{11}$$

possibilities for given burst lengths f and f'.

Next we investigate the alternative case that the two bursts $\mathsf{B}\ (s, f, \mathbf{u})$ and $\mathsf{B}\ (s', f', \mathbf{u})$ overlap. Without loss of generality, we still assume that $s' > s$. The pictorial analysis is illustrated in Fig. 3. Define $\mathbf{y}'$ such that

$$\begin{array}{ll} y'_i = y_i = c_i, & i \notin [s, s+f-1] \cup [s', s'+f'-1] \\ y'_i = y_i, & i \in [s+f'-r, s'-1] \\ y'_i = 0, & i \in [s, s+f'-r-1] \cup [s', s'+f'-1]. \end{array}$$

Applying erasure-only decoding on $\mathbf{y}'$ results in a codeword, say $\mathbf{c}'$. Define a received word $\mathbf{y}''$

$$\begin{array}{ll} y''_i = y_i = c_i, & i \notin [s, s+f-1] \\ y''_i = y_i, & i \in [s+f'-r, s'-1] \\ y''_i = c'_i, & i \in [s, s+f'-r-1] \cup [s', s+f-1]. \end{array}$$

Clearly, $\mathbf{y}''$ is a special case of $\mathbf{y}$ suffering a burst at $[s, s+f-1]$, while exhibiting an overlapping spurious burst at $[s', s'+f'-1]$ (leading to the miscorrection toward $\mathbf{c}'$). We observe that there are at most $q^{s'-s+f'-r}$ different $\mathbf{y}''$ following the freedom of the assignment $y''_i = y_i, i \in [s+f'-r, s'-1]$, where s' ranges from $s-f'+r+1$ to $s+f-1$. Alternatively, when $s > s'$, there at most $q^{s-s'+f-r}$ different $\mathbf{y}''$ with s ranging from $s'-f+r+1$ to $s'+f'-1$. Hence, there are

$$\sum_{s'=s-f'+r-1}^{s+f-1} q^{s'-s+f'-r} + \sum_{s=s'-f+r-1}^{s'+f'-1} q^{s-s'+f-r} \approx 2q^{f+f'-r-1} \tag{12}$$

possibilities for a given f and f'.

The following theorem provides a simple union bound, based upon the above discussions.

Theorem 2: Given a burst $\mathsf{B}\ (s, f, \mathbf{u})$ with length f ($f \leq r-1$) has occurred, the miscorrection probability of the proposed algorithm is upper bounded by

$$P_m(\mathsf{B}\ (s, f, \mathbf{u})|f) \leq q^{r-1-f}. \tag{13}$$

Proof: The number of overall miscorrection possibilities is upper bounded by summing (11) and (12) over all possible values of f'. Therefore, we obtain the following union bound:

$$\begin{aligned} &P_m(\mathsf{B}\ (s, f, \mathbf{u})|f) \leq \\ &\leq q^{-f} \sum_{f'=r+2-f}^{f} \left(2q^{f+f'-r-1} + (n-f-f')q^{f+f'-r}\right) \\ &\leq n \cdot q^{-(r-f)} \leq q^{-(r-f-1)}. \end{aligned}$$

Remarks: It is worth noting that the above analysis is independent of the starting location s. Thus, the bound given in (13) applies universally to each s. It is possible to refine the above

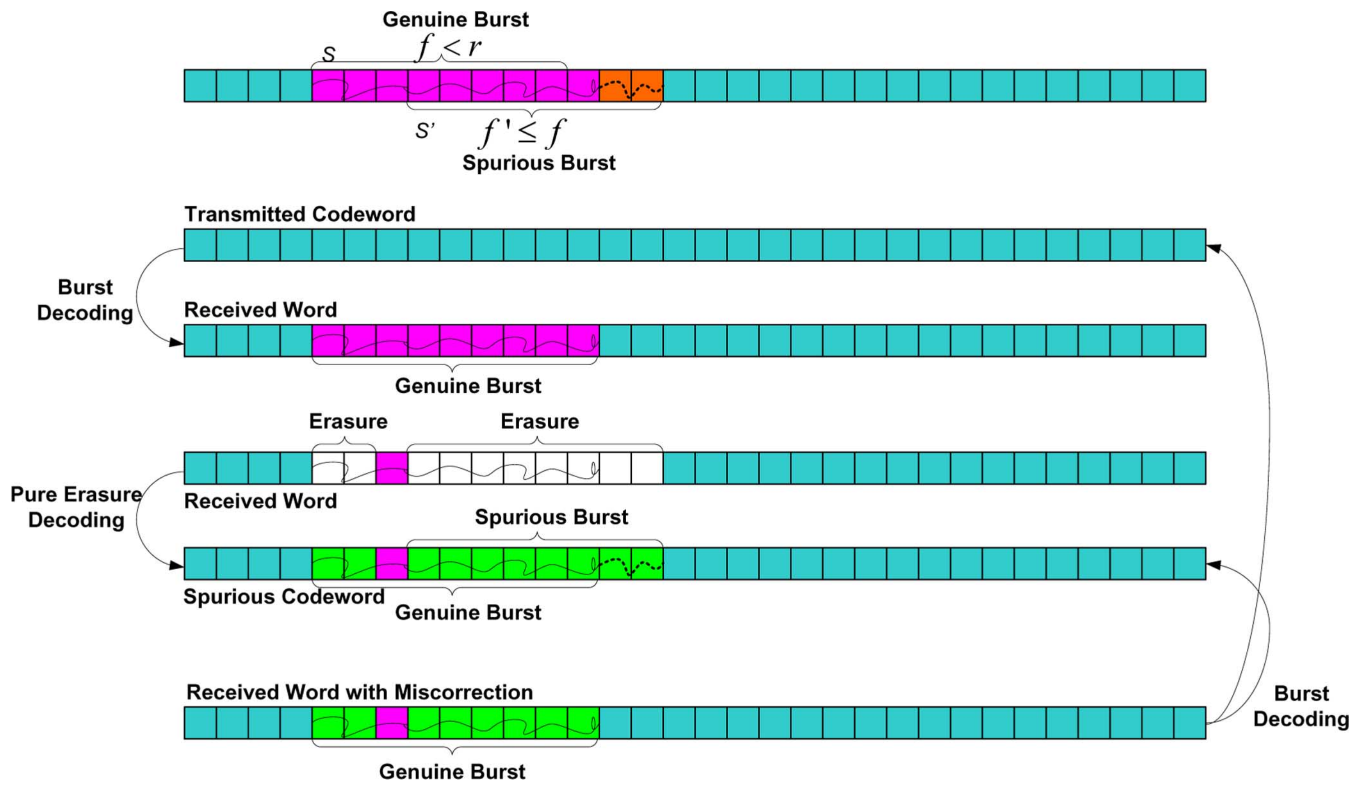

Fig. 3. An explicit construction of overlapping case where a burst-corrupted word leads to miscorrection.

analysis to obtain a tighter first order term, however, the effect is negligible compared to the exponential decay factor. The above analysis still holds if the burst of consecutive symbols is substituted by an equal number of randomly located symbols.

Theorem 2 reveals that the miscorrection probability decays exponentially with respect to the burst length. Since in practical applications the code length n is chosen to be close to field size q for format efficiency, the miscorrection probability of a single burst with length $f = r - 1$ is non-negligible. However, by sacrificing slightly the burst correction capability, the miscorrection probability turns to be negligible. For example, data storage uses the format of 512-byte sector which is protected by a shortened Reed-Solomon code (where typically $r \in [32, 48]$) over $\mathrm{GF}(2^{10})$. The resulting miscorrection probability is less than $q^{-6} \approx 10^{-18}$ for correcting a single burst of length up to $r - 7$.

Theorem 2 can be re-interpreted from an intuitive angle. Note that $\Gamma(x)$ has degree $r - 1$, and divides $\prod_{i=s+f-1}^{s+r-2}(x - \alpha^i)$ by Theorem 1. Accordingly, $\frac{\Gamma(x)}{\prod_{i=s+f-1}^{s+r-2}(x-\alpha^i)}$ has $q^{r-1-(r-f)}$ possibilities. Given that it contains a spurious burst starting at s' and with length f', $\Gamma(x)$ must be divisible by $\prod_{i=s'+f'-1}^{r+s'-2}(x - \alpha^i)$. There are $q^{r-1-(r-f)-(r-f')} = q^{f+f'-r-1}$ possibilities for the polynomial $\frac{\Gamma(x)}{\prod_{i=s+f-1}^{s+r-2}(x-\alpha^i)\prod_{j=s'+f'-1}^{s'+r-2}(x-\alpha^j)}$. Therefore, the probability of miscorrection is governed by

$$\begin{aligned} P_m(\mathbf{B}\,(s,f,\mathbf{u})|f) &\le \frac{\sum_{s',f'} q^{f+f'-r-1}}{q^{r-1-(r-f)}} \\ &= \frac{\sum_{f'=r+2-f}^{f}(n-(r-1-f'))q^{f+f'-r-1}}{q^{r-1-(r-f)}} \\ &\le q^{r-1-f}. \end{aligned}$$

Finally, we draw comments on the f-list decoding of bursts, which is obtained by tracking all sequences of at least $r-f$ consecutive roots of $\Gamma(x)$. Clearly, the list size l is upper bounded by

$$l \le \left\lfloor \frac{r-1}{r-f} \right\rfloor \tag{14}$$

whereas a tight lower bound which attains the Reiger bound is given in [9]

$$l \ge \left\lceil \frac{f}{r-f} \right\rceil . \tag{15}$$

Furthermore, the proposed algorithm can be trivially tailored for (l, f)-burst list decoding that achieves the Reiger bound. The following example shows the upper bound is attainable.

Example 2: Consider the (31, 23) primitive Reed-Solomon code. Let the all-zero codeword be corrupted by a single burst

$$[0,0,0,0,0,0,0,0,0,0,0,0,0,\alpha^{15},\alpha^{18},\alpha^{8},\alpha^{28},\alpha^{15},\alpha^{3},\alpha^{0},\\ 0,0,0,0,0,0,0,0,0,0,0]$$

which has syndromes

$$\mathbf{S} = [\alpha^{16},\alpha^{29},\alpha^{24},\alpha^{6},\alpha^{1},\alpha^{9},\alpha^{6},\alpha^{14}].$$

The resulting $\Gamma(x)$ has 7 valid roots

$$\begin{aligned} \Gamma(x) =& \alpha^{26}x^7 + \alpha^{12}x^6 + \alpha^{30}x^5 + \alpha^{18}x^4 + \alpha^{16}x^3 + \alpha^{24}x^2 \\ &+ \alpha^4x^1 + \alpha^{14} \\ =& (x-\alpha^0)(x-\alpha^4)(x-\alpha^7)(x-\alpha^9)(x-\alpha^{17}) \\ &(x-\alpha^{19})(x-\alpha^{25}), \end{aligned}$$

each one resulting in a different candidate burst, following Theorem 1.

IV. Correction Algorithm for a Long Burst With a Random Error

In this section, we consider the correction of a long burst $\mathsf{B}\ (s, f, \mathbf{u})$ with length $f \leq r-3$ along with a possible random error denoted by (β, Y), where β and Y are locator and magnitude, respectively. We are interested in determining the shortest burst and possibly a random error, when there are multiple candidates.

The following lemma is analogous to Lemma 2 which connects a burst to a sequence of consecutive roots.

Lemma 3: A burst $\mathsf{B}\ (s, f, \mathbf{u})(f \leq r-3)$ along with a random error (β, Y) (when there is no random error, β is set to 0) has occurred, if and only if the following system of equations are satisfied

$$\begin{cases} S_{r-1}\Lambda_0^{(0)} + S_{r-2}\Lambda_1^{(0)} + \cdots + S_0\Lambda_{r-1}^{(0)} = 0 \\ S_{r-1}\Lambda_0^{(1)} + S_{r-2}\Lambda_1^{(1)} + \cdots + S_0\Lambda_{r-1}^{(1)} = 0 \\ \vdots \\ S_{r-1}\Lambda_0^{(r-2-f)} + S_{r-2}\Lambda_1^{(r-2-f)} + \cdots + S_0\Lambda_{r-1}^{(r-2-f)} = 0 \end{cases} \tag{16}$$

where $\Lambda^{(i)}(x) = \prod_{j=s-i}^{s-i+r-3}(1-\alpha^j x)\cdot(1-\beta x)$ for $i = 0, 1, 2, \ldots, r-2-f$.

It is worth noting that the artificial burst length $r-2$ is one more than the maximum allowable burst length, $r-3$, which is different from that of single burst as discussed in Section III. This redundancy is utilized to identify the random error locator β, as will be explained shortly.

Define

$$\Gamma_1(x) \triangleq \sum_{i=0}^{r-2} S_{r-1-i}\bar{\Lambda}_i^{(r-2)}x^i \tag{17}$$

$$\Gamma_2(x) \triangleq \sum_{i=0}^{r-2} S_{r-2-i}\bar{\Lambda}_i^{(r-2)}x^i \tag{18}$$

where the coefficients $\bar{\Lambda}_i^{(r-2)}$, $i = 0, 1, \ldots, r-2$, are defined as in (8). We observe for $j > 0$

$$\Lambda_j^{(i)} = \bar{\Lambda}_j^{(r-2)}\alpha^{j(s-i+r-3)} - \beta\bar{\Lambda}_{j-1}^{(r-2)}\alpha^{(j-1)(s-i+r-3)} \tag{19}$$

which immediately yields

$$\begin{aligned} & S_{r-1}\Lambda_0^{(i)} + S_{r-2}\Lambda_1^{(i)} + S_{r-3}\Lambda_2^{(i)} + \cdots + S_0\Lambda_{r-1}^{(i)} \\ &= S_{r-1} + \sum_{j=1}^{r-1} S_{r-1-j}\Lambda_j^{(i)} \\ &= S_{r-1} + \sum_{j=1}^{r-1} S_{r-1-j}\cdot \\ &\quad \left(\bar{\Lambda}_j^{(r-2)}\alpha^{j(s-i+r-3)} - \beta\bar{\Lambda}_{j-1}^{(r-2)}\alpha^{(j-1)(s-i+r-3)}\right) \\ &= \Gamma_1(\alpha^{s-i+r-3}) - \beta\Gamma_2(\alpha^{s-i+r-3}) \end{aligned} \tag{20}$$

for $i = 0, 1, 2, \ldots, r-2-f$. If $\Gamma_1(\alpha^j) = 0$ (regardless of $\Gamma_2(\alpha^j)$), then $\beta = 0$, which represents the case of a single burst without random error. On the other hand, if $\Gamma_1(\alpha^j) \neq 0$ and $\Gamma_2(\alpha^j) = 0$, then β is non-existent (in next example it is denoted by ∞).

The above discussions translate Lemma 3 into the following concise assertion

Theorem 3: A burst $\mathsf{B}\ (s, f, \mathbf{u})(f \leq r-3)$, along with a random error (β, Y), has occurred, if and only if

$$\frac{\Gamma_1(\alpha^i)}{\Gamma_2(\alpha^i)} = \beta, \quad i = s+f-1, s+f, \ldots, s+r-3. \tag{21}$$

Remarks: A random error locator, β, is a valid candidate only if it applies to at least two consecutive indices i's in (21), which justifies the exploitation of $\bar{\Lambda}^{(r-2)}(x)$, instead of $\bar{\Lambda}^{(r-3)}(x)$. For the special case of a single burst crossing $r-2$ symbols, which can be viewed as a single burst of length $r-3$, combined with a random error from either end of the burst, the above theorem needs to be relaxed to allow $f = r-2$ with respect to $\beta = 0$. For a shortened code, we also need to make sure the resulting β is valid such that $\beta \in \{1, \alpha, \alpha^2, \ldots, \alpha^{n-1}\}$.

We are now ready to present the algorithmic procedure as follows.

Algorithm 2: Correcting a Long Burst and a Random Error:

1) Precompute the coefficients of $\bar{\Lambda}^{(r-2)}(x)$.
2) Compute syndromes $S_0, S_1, S_2, \ldots, S_{r-1}$.
3) Apply the Berlekamp-Massey algorithm. If successful, then stop.
4) Compute the coefficients of $\Gamma_1(x)$ and $\Gamma_2(x)$.
5) Apply the Chien search to $\Gamma_1(\alpha^i)$ and $\Gamma_2(\alpha^i)$ and compute $\beta = \frac{\Gamma_1(\alpha^i)}{\Gamma_2(\alpha^i)}$, for $i = \lceil\frac{r}{2}\rceil, \lceil\frac{r}{2}\rceil+1, \lceil\frac{r}{2}\rceil+2, \ldots, n+r-4$. Tracking the longest consecutive indices i's that result in identical and valid β and record the corresponding burst end location.
6) Compute error magnitudes utilizing Forney's formula.

Note when the searching end overlaps with start such that $n+r-4-(q-1) \geq \lceil\frac{r}{2}\rceil$, then the entire field GF$(q)\backslash\{0\}$ is searched instead.

The following example sheds light on the insight of Algorithm 2.

Example 3: Consider the (24, 16) shortened Reed-Solomon code in GF(2^5).

(i). Let a transmitted codeword be

$$\mathbf{c} = [\alpha^{14}, \alpha^{11}, \alpha^{11}, \alpha^{25}, \alpha^{11}, 0, \alpha^7, \alpha^1, \alpha^9, \alpha^2, \alpha^{11}, \alpha^8, \alpha^{15} \\ \alpha^{29}, \alpha^{25}, \alpha^7, \alpha^{10}, \alpha^{15}, \alpha^{21}, \alpha^{30}, \alpha^3, \alpha^3, \alpha^{10}, \alpha^8]$$

and the received word

$$\mathbf{r} = [\alpha^{14}, \alpha^{11}, \alpha^{11}, \alpha^{25}, \alpha^{11}, 0, \alpha^7, \alpha^1, \alpha^9, \alpha^2, \alpha^{11}, \alpha^8, \alpha^{15}, \\ \underline{\alpha^{28}, \alpha^{16}, \alpha^{21}, \alpha^4}, \alpha^{15}, \alpha^{21}, \alpha^{30}, \alpha^3, \alpha^3, \alpha^{10}, \alpha^8].$$

Syndromes are computed

$$\mathbf{S} = [\alpha^1, \alpha^{13}, \alpha^9, \alpha^{22}, \alpha^{24}, \alpha^{22}, \alpha^{30}, \alpha^4].$$

Subsequently, $\Gamma_1(x)$ and $\Gamma_2(x)$ are obtained

$$\Gamma_1(x) = \alpha^{29}x^6 + \alpha^3x^5 + \alpha^{14}x^4 + \alpha^{30}x^3 + \alpha^{19}x^2$$

$$+ \alpha^3 x^1 + \alpha^4$$
$$\Gamma_2(x) = \alpha^{17}x^6 + \alpha^7 x^5 + \alpha^1 x^4 + \alpha^{28}x^3 + \alpha^{21}x^2 + \alpha^{26}x^1 + \alpha^{30}.$$

Evaluating $\frac{\Gamma_1(x)}{\Gamma_2(x)}$ over the consecutive sequence, $\alpha^4, \alpha^5, \alpha^6, \ldots, \alpha^{28}$, gives the following:

$$[\underline{0}, \alpha^{17}, \underline{0, 0}, \alpha^4, \underline{\alpha^{18}, \alpha^{18}}, \infty, \alpha^{16}, \underline{\alpha^5, \alpha^5}, \alpha^{16}, \underline{0, 0, 0}, \alpha^{13}, \alpha^{30}, \alpha^{11}, \underline{\alpha^4, \alpha^4}, \alpha^8, \alpha^{19}, \alpha^{26}, \alpha^2]$$

where ∞ indicates that $\Gamma_2 = 0$ but $\Gamma_1 \neq 0$. Theorem 3 shows that there are 6 candidates, as shown on top of next page.

Applying Forney's formula yields 6 candidate codewords corresponding to the above vector (see the first equation at the bottom of the page), where the first burst is incomplete due to the shortening of the code (note the length is supposed to be 6). By choosing the candidate corresponding to the shortest burst (which is $\mathbf{c}_5$), the transmitted codeword is successfully retrieved.

(ii). Let a transmitted codeword be

$$\mathbf{c} = [\alpha^{25}, \alpha^{10}, \alpha^7, \alpha^{17}, \alpha^{28}, \alpha^7, \alpha^9, \alpha^{11}, \alpha^9, \alpha^{14}, \alpha^{29}, \alpha^{21} \alpha^5, \alpha^{15}, \alpha^{14}, \alpha^7, \alpha^{27}, \alpha^{13}, \alpha^{28}, \alpha^{22}, \alpha^5, \alpha^{16}, \alpha^{17}, \alpha^8]$$

and the received word

$$\mathbf{r} = \big[\alpha^{25}, \alpha^{10}, \alpha^7, \alpha^{17}, \alpha^{25}, \alpha^7, \alpha^9, \alpha^{11}, \alpha^9, \alpha^{14}, \alpha^{29}, \alpha^{11}, \alpha^{24}, \alpha^{24}, \alpha^{14}, \alpha^7, \alpha^{27}, \alpha^{13}, \alpha^{28}, \alpha^{22}, \alpha^5, \alpha^{16}, \alpha^{17}, \alpha^8\big].$$

Syndromes are computed

$$\mathbf{S} = [a^{28}, \alpha^{19}, \alpha^{28}, \alpha^{11}, \alpha^{28}, \alpha^{10}, \alpha^{18}, \alpha^{12}].$$

Subsequently, $\Gamma_1(x)$ and $\Gamma_2(x)$ are obtained

$$\Gamma_1(x) = \alpha^4 x^6 + \alpha^{22}x^5 + \alpha^3 x^4 + \alpha^3 x^3 + \alpha^7 x^2 + \alpha^{22}x^1 + \alpha^{12}$$
$$\Gamma_2(x) = \alpha^{13}x^6 + \alpha^{13}x^5 + \alpha^{20}x^4 + \alpha^{17}x^3 + \alpha^{25}x^2 + \alpha^{14}x^1 + \alpha^{18}.$$

Evaluating $\frac{\Gamma_1(x)}{\Gamma_2(x)}$ over the consecutive sequence, $\alpha^4, \alpha^5, \alpha^6, \ldots, \alpha^{28}$, gives the following:

$$\big[\alpha^{24}, \alpha^3, \underline{a^{21}, \alpha^{21}}, \alpha^{12}, \alpha^{13}, \alpha^{26}, \alpha^{14}, -1, \underline{a^4, \alpha^4, \alpha^4, \alpha^4}, \alpha^{25}, \alpha^{28}, \alpha^8, \underline{0}, \alpha^{27}, \underline{0}, \alpha^{16}, \alpha^{15}, \alpha^{29}, \alpha^{30}, \alpha^{22}\big].$$

Theorem 3 shows that there are 4 candidates, as shown on the next page.

Applying Forney's formula yields 4 candidate codewords corresponding to the above vector, where the transmitted word $\mathbf{c}_2$ exhibits the shortest burst along with a random error. (See the second equation at the bottom of the page.)

We denote by $P_m(\mathsf{B}\ (s, f, \mathbf{u}), (\beta, Y)|f)$ the miscorrection probability of Algorithm 2 with respect to a random burst $\mathsf{B}\ (s, f, \mathbf{u})$ with given length f and a random error (β, Y), where s is uniformly distributed within $[0, n - f]$, $\mathbf{u}$ is uniformly distributed over $\{\mathbf{u} \in \mathrm{GF}(q)^f \mid u_0 \neq 0, u_{f-1} \neq 0, w(\mathbf{u}) > f/2\}$, where $w(\cdot)$ denotes Hamming weight, $\log(\beta)$ is uniformly distributed over $[0, n-1] \backslash [s, s+f-1]$, and Y is uniformly distributed within $\mathrm{GF}(q) \backslash \{0\}$. Formally, $P_m(\mathsf{B}\ (s, f, \mathbf{u}), (\beta, Y)|f)$ is defined as the probability that there exists a spurious burst $\mathsf{B}\ (s', f', \mathbf{u}')$ with $f' \leq f$ and possibly a random error (β', Y') that generates the same syndromes as $\mathsf{B}\ (s, f, \mathbf{u})$ and (β, Y).

Note that the analysis on the miscorrection of Algorithm 1 actually has nothing to do with bursty condition but rather the number of errors. Therefore, it also applies to this case, more

$$\mathbf{r} = [\alpha^{14}, \alpha^{11}, \alpha^{11}, \alpha^{25}, \alpha^{11}, 0, \alpha^7, \alpha^1, \alpha^9, \alpha^2, \alpha^{11}, \alpha^8, \alpha^{15}, \underline{\alpha^{28}, \alpha^{16}, \alpha^{21}, \alpha^4}, \alpha^{15}, \alpha^{21}, \alpha^{30}, \alpha^3, \alpha^3, \alpha^{10}, \alpha^8]$$
$$\mathbf{c}_1 = [\underline{\alpha^{21}, \alpha^4, \alpha^1, \alpha^{25}, \alpha^{27}}, 0, \alpha^7, \alpha^1, \alpha^9, \alpha^2, \alpha^{11}, \alpha^8, \alpha^{15}, \alpha^{28}, \alpha^{16}, \alpha^{21}, \alpha^4, \alpha^{15}, \alpha^{21}, \alpha^{30}, \alpha^3, \alpha^3, \alpha^{10}, \alpha^8]$$
$$\mathbf{c}_2 = [\alpha^{14}, \alpha^{11}, \underline{\alpha^{22}, \alpha^{15}, \alpha^{13}, \alpha^2, \alpha^4}, \alpha^1, \alpha^9, \alpha^2, \alpha^{11}, \alpha^8, \alpha^{15}, \alpha^{28}, \alpha^{16}, \alpha^{21}, \alpha^4, \alpha^{15}, \alpha^{21}, \alpha^{30}, \alpha^3, \alpha^3, \alpha^{10}, \alpha^8]$$
$$\mathbf{c}_3 = [\alpha^{14}, \alpha^{11}, \alpha^{11}, \alpha^{25}, \alpha^{11}, \underline{\alpha^{19}, \alpha^1, \alpha^6, \alpha^1, \alpha^8}, \alpha^{11}, \alpha^8, \alpha^{15}, \alpha^{28}, \alpha^{16}, \alpha^{21}, \alpha^4, \alpha^{15}, \underline{\alpha^4}, \alpha^{30}, \alpha^3, \alpha^3, \alpha^{10}, \alpha^8]$$
$$\mathbf{c}_4 = [\alpha^{14}, \alpha^{11}, \alpha^{11}, \alpha^{25}, \alpha^{11}, \underline{\alpha^{10}}, \alpha^7, \alpha^1, \alpha^9, \underline{\alpha^{12}, 0, \alpha^{29}, \alpha^{22}, \alpha^{25}}, \alpha^{16}, \alpha^{21}, \alpha^4, \alpha^{15}, \alpha^{21}, \alpha^{30}, \alpha^3, \alpha^3, \alpha^{10}, \alpha^8]$$
$$\mathbf{c}_5 = [\alpha^{14}, \alpha^{11}, \alpha^{11}, \alpha^{25}, \alpha^{11}, 0, \alpha^7, \alpha^1, \alpha^9, \alpha^2, \alpha^{11}, \alpha^8, \alpha^{15}, \underline{\alpha^{29}, \alpha^{25}, \alpha^7, \alpha^{10}}, \alpha^{15}, \alpha^{21}, \alpha^{30}, \alpha^3, \alpha^3, \alpha^{10}, \alpha^8]$$
$$\mathbf{c}_6 = [\alpha^{14}, \alpha^{11}, \alpha^{11}, \alpha^{25}, \underline{\alpha^{13}}, 0, \alpha^7, \alpha^1, \alpha^9, \alpha^2, \alpha^{11}, \alpha^8, \alpha^{15}, \alpha^{28}, \alpha^{16}, \alpha^{21}, \alpha^4, \alpha^{15}, \underline{\alpha^{28}, \alpha^2, \alpha^2, \alpha^{25}, \alpha^2}, \alpha^8]$$

$$\mathbf{r} = [\alpha^{25}, \alpha^{10}, \alpha^7, \alpha^{17}, \alpha^{25}, \alpha^7, \alpha^9, \alpha^{11}, \alpha^9, \alpha^{14}, \alpha^{29}, \alpha^{11}, \alpha^{24}, x\alpha^{24}, \alpha^{14}, \alpha^7, \alpha^{27}, \alpha^{13}, \alpha^{28}, \alpha^{22}, \alpha^5, \alpha^{16}, \alpha^{17}, \alpha^8]$$
$$\mathbf{c}_1 = [a^{25}, \alpha^{10}, \underline{a^{28}, \alpha^6, \alpha^0, \alpha^{24}, \alpha^7}, \alpha^{11}, \alpha^9, \alpha^{14}, \alpha^{29}, \alpha^{11}, \alpha^{24}, \alpha^{24}, \alpha^{14}, \alpha^7, \alpha^{27}, \alpha^{13}, \alpha^{28}, \alpha^{22}, \alpha^5, \underline{a^{25}}, \alpha^{17}, \alpha^8]$$
$$\mathbf{c}_2 = [a^{25}, \alpha^{10}, \alpha^7, \alpha^{17}, \underline{a^{28}}, \alpha^7, \alpha^9, \alpha^{11}, \alpha^9, \alpha^{14}, \alpha^{29}, \underline{a^{21}, \alpha^5, \alpha^{15}}, \alpha^{14}, \alpha^7, \alpha^{27}, \alpha^{13}, \alpha^{28}, \alpha^{22}, \alpha^5, \alpha^{16}, \alpha^{17}, \alpha^8]$$
$$\mathbf{c}_3 = [a^{25}, \alpha^{10}, \alpha^7, \alpha^{17}, \alpha^{25}, \alpha^7, \alpha^9, \alpha^{11}, \alpha^9, \alpha^{14}, \alpha^{29}, \alpha^{11}, \alpha^{24}, \alpha^{24}, \alpha^{14}, \underline{a^{15}, \alpha^3, \alpha^{11}, \alpha^{15}, \alpha^{29}, \alpha^{25}}, \alpha^{16}, \alpha^{17}, \alpha^8]$$
$$\mathbf{c}_4 = [a^{25}, \alpha^{10}, \alpha^7, \alpha^{17}, \alpha^{25}, \alpha^7, \alpha^9, \alpha^{11}, \alpha^9, \alpha^{14}, \alpha^{29}, \alpha^{11}, \alpha^{24}, \alpha^{24}, \alpha^{14}, \alpha^7, \alpha^{27}, \underline{a^{12}, \alpha^3, \alpha^6, \alpha^{27}, \alpha^{18}, \alpha^{15}}, \alpha^8].$$

specifically, given a genuine burst with length f and a random locator β, the probability of the existence of a spurious burst with length $f' \leq f$ possibly along with a spurious random locator β' is upper bounded by $q^{-(r-2-f)}$. In addition, β' can be arbitrarily apart from the spurious burst and thus may have $n - f'$ different options. In summary, we obtain the following union bound.

Theorem 4: Given that a burst $\mathbf{B}\ (s, f, \mathbf{u})$ with length $f (f \leq r-3)$ possibly along with a random error (β, Y) have occurred, the miscorrection probability of Algorithm 2 is governed by

$$P_m(\mathbf{B}\ (s, f, \mathbf{u}), (\beta, Y)|\, f) \leq (n-f) q^{-(r-2-f)} \leq q^{-(r-3-f)}. \tag{22}$$

Next, we provide an alternative intuitive interpretation on the above bound for the miscorrection probability. Let $\mathbf{B}\ (s, f) (f \leq r-3)$ denote the burst and β the random error locator. Theorem 4 indicates that

$$\Gamma_1(\alpha^{s+f-1+j}) = \beta \Gamma_2(\alpha^{s+f-1+j}), \quad j = 0, 1, \ldots, r-2-f, \tag{23}$$

where $S_0, S_1, \ldots, S_{r-1}$ are the unknowns. There are r unknowns subject to $r-1-f$ linear constraints. So the degrees of freedom of this system of equations is $r - (r-1-f) = f+1$.

On the other hand, the existence of a spurious burst with length $f' \leq f$ and a random error locator β' indicates another set of linear constraints

$$\Gamma_1(\alpha^{s'+f'-1+j}) = \beta' \Gamma_2(\alpha^{s'+f'-1+j}), j = 0, 1, \ldots, r-2-f'. \tag{24}$$

The two set of linear constraints must be linearly independent (otherwise Theorem 3 is violated). Therefore, the degrees of freedom associated with miscorrection is at most $f + 1 - (r - f' - 1) = f + f' - r + 2$. Finally, we observe that the negative difference of the two degrees of freedom is exactly the exponent of the miscorrection probability of Algorithm 2.

V. Correction Algorithm for a Long Burst and Up to Constant Number of Random Errors

In this section, we devise an efficient algorithm to probabilistically correct a long burst with length $f \leq r - 1 - 2\delta$ along with up to δ random errors, where δ is a small constant, i.e., fixed when r goes to infinity.

We present the following decoding algorithm while justifying it thereafter.

Algorithm 3: Correcting a Long Burst and up to Constant Number of Random Errors:

```
1) Precompute the coefficients of $\bar{\Lambda}^{(r-2\delta)}(x)$.
2) Compute syndromes $S_0, S_1, S_2, \ldots, S_{r-1}$.
3) Apply the Berlekamp-Massey algorithm, if successful, then stop.
4) For $l = 0, 1, 2, \ldots, n-1$, do:
   • Initialize $\Lambda^{(r-2\delta)}(x) \leftarrow \bar{\Lambda}^{(r-2\delta)}(\alpha^l x)$, $L \leftarrow r - 2\delta$, $B^{(r-2\delta)}(x) \leftarrow \bar{\Lambda}^{(r-2\delta)}(\alpha^l x)$.
   • Initialize $\lambda^{(r-2\delta)}(x) \leftarrow 1$, $b^{(r-2\delta)}(x) \leftarrow 1$.
   • For $j = r-2\delta, r-2\delta+1, \ldots, r-1$, do:
     — Compute $\Delta^{(j)} = \sum_{i=0}^{L} \Lambda_i^{(j)} \cdot S_{j-i}$
     — Compute $\Lambda^{(j+1)}(x) = \Lambda^{(j)}(x) - \Delta^{(j)} \cdot x B^{(j)}(x)$
     — Compute $\lambda^{(j+1)}(x) = \lambda^{(j)}(x) - \Delta^{(j)} \cdot x b^{(j)}(x)$
     — If $\Delta^{(j)} \neq 0$ and $2L \leq j + (r - 2\delta)$,
         * Set $B^{(j+1)}(x) \leftarrow (\Delta^{(j)})^{-1} \Lambda^{(j)}(x)$
         * Set $b^{(j+1)}(x) \leftarrow (\Delta^{(j)})^{-1} \lambda^{(j)}(x)$ and $L \leftarrow j + r + 1 - 2\delta - L$
   • Else
         * Set $B^{(j+1)}(x) \leftarrow x B^{(j)}(x)$
         * Set $b^{(j+1)}(x) \leftarrow x b^{(j)}(x)$
   • Track the longest sequence of consecutive indices $l$'s that result in identical $\lambda^{(r)}(x)$, and record the last element $l^*$ of the consecutive $l$'s.
5) Apply the Chien search to the recorded $\lambda^{(r)}(x)$. If the number of valid roots of $\lambda^{(r)}(x)$ is equal to its degree, then apply Forney's formula over the overall error-locator polynomial $\Lambda^{(r)}(x) = \lambda^{(r)}(x) \cdot \bar{\Lambda}^{(r-2\delta)}(\alpha^{l^*} x)$ to compute error magnitudes and output the desired codeword. Otherwise, declare a decoding failure.
```

The desired output $\lambda^{(r)}(x)$ is the error locator polynomial corresponding to up to δ random errors. For simplicity, it is referred to as the random-error locator polynomial. A candidate $\lambda^{(r)}(x)$ to be tracked has the following property: $\lambda^{(r)}(x)$ is identical for at least two consecutive l's, corresponding to a burst with length $f \leq r - 1 - 2\delta$ and up to δ (possibly spurious) random errors, or $\lambda^{(r)}(x) = 1$, which corresponds to a burst of length $f \leq r - 2\delta$ without random error. Each iteration of Step 3 is essentially stand-alone Blahut error-and-erasure decoding. Furthermore, the following intrinsic connection holds at the lth iteration:

$$\begin{aligned} \Lambda^{(r)}(x) &= \lambda^{(r)}(x) \cdot \bar{\Lambda}^{(r-2\delta)}(\alpha^l x). \\ B^{(r)}(x) &= b^{(r)}(x) \cdot \bar{\Lambda}^{(r-2\delta)}(\alpha^l x). \end{aligned}$$

When a long burst with length $f \leq r - 1 - 2\delta$ is captured by $\bar{\Lambda}^{(r-2\delta)}(\alpha^l x)$ and the number of random errors is no greater than δ, $\lambda^{(r)}(x)$ is the error locator polynomial for the random errors. Moreover, the resulting $\lambda^{(r)}(x)$ is identical for exactly $r - 2\delta - f + 1$ consecutive l's. However, its converse is not true, i.e, if a $\lambda^{(r)}(x)$ is resulted identically for exactly $r - 2\delta - f + 1$ consecutive l's, it is not necessarily a valid error locator polynomial in the sense that its number of distinct roots within $\{1, \alpha^{-1}, \alpha^{-2}, \ldots, \alpha^{-(n-1)}\}$ is not equal to its degree. The algorithm incurs a failure if the random-error locator polynomial $\lambda^{(r)}(x)$ chosen by the algorithm is invalid, even though there may exist valid error locator polynomials.

Remarks: It is worth noting that Algorithm 3 applies for any value of δ. The reason behind for setting δ to be a small constant and independent of n and r is to maintain the computational complexity $O(nr)$. On the other hand, if to choose δ proportional to r, then the resulting complexity becomes $O(nr^2)$.

We denote by $P_m(\mathbf{B}\ (s, f, \mathbf{u}), \{(\alpha_i, Y_i)\}_{i=1}^{\delta}\ |f)$ the miscorrection probability of Algorithm 3 with respect to a random burst $\mathbf{B}\ (s, f, \mathbf{u})$ with given length f and δ random errors, $\{(\alpha_i, Y_i)\}_{i=1}^{\delta}$, where $\lfloor \frac{r}{2} \rfloor - \delta < f < r - 2\delta$, where s is uniformly distributed within $[0, n-f]$, $\mathbf{u}$ is uniformly distributed over $\{\mathbf{u} \in \mathrm{GF}(q)^f | u_0 \neq 0, u_{f-1} \neq 0, w(\mathbf{u}) > f/2\}$, where $w(\cdot)$ denotes Hamming weight, $\log(\alpha_i)$, $i = 1, 2, \ldots, \delta$, are distinct and uniformly distributed over $[0, n-1] \backslash [s, s+f-1]$, and Y_i, $i = 1, 2, \ldots, \delta$, are uniformly distributed within $\mathrm{GF}(q) \backslash \{0\}$. Formally, $P_m(\mathbf{B}\ (s, f, \mathbf{u}), \{(\alpha_i, Y_i)\}_{i=1}^{\delta} | f)$ is probability that there exists a spurious burst $\mathbf{B}\ (s', f', \mathbf{u}')$ with

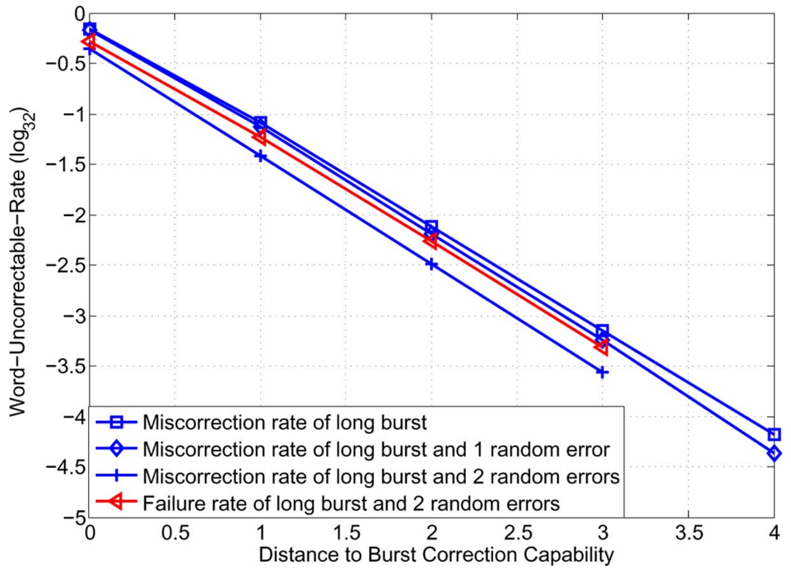

Fig. 4. Simulated miscorrection and failure rates as functions of distance to capability for the (30, 16) shortened Reed-Solomon code defined over $\mathrm{GF}(2^5)$.

$f' \le f$ and up to δ random errors that yields the same syndromes as the burst $\mathsf{B}(s,f,\mathbf{u})$ and δ errors. Likewise, define $P_f(\mathsf{B}(s,f,\mathbf{u}),\{(\alpha_i,Y_i)\}_{i=1}^{\delta}|f)$ as the failure probability that there exists a spurious burst $\mathsf{B}(s',f',\mathbf{u}')$ with $f' \le f$ and an invalid random-error locator polynomial $\lambda(x)$ (its number of distinct and valid roots is not equal to its degree) that generates the same syndromes as the burst $\mathsf{B}(s,f,\mathbf{u})$ and δ errors.

Next theorem characterizes the miscorrection and failure probabilities of Algorithm 3.

Theorem 5: Given that a burst $\mathsf{B}(s,f,\mathbf{u})(f \le r-1-2\delta)$ and δ random errors have occurred, the miscorrection and failure probabilities of Algorithm 3 are both bounded by

$$P_m(\mathsf{B}(s,f,\mathbf{u}),\{(\alpha_i,Y_i)\}_{i=1}^{\delta} \mid f) \le q^{-(r-1-2\delta-f)} \quad (25)$$
$$P_f(\mathsf{B}(s,f,\mathbf{u}),\{(\alpha_i,Y_i)\}_{i=1}^{\delta} \mid f) \le q^{-(r-1-2\delta-f)}. \quad (26)$$

Proof: Let $\lambda(x)$ be a spurious random-error locator polynomial with degree up to δ recorded in Algorithm 3, such that $\lambda(x)$ is the resulting polynomial for $r-2\delta+1-f'(f' \le f)$ consecutive l's in Algorithm 3. Define the modified syndrome polynomial

$$\tilde{S}(x) \triangleq \left[S(x)\lambda^{(r)}(x) \pmod{x^r} - S(x)\lambda(x) \pmod{x^{2\delta}}\right]/x^{2\delta} \quad (27)$$

which is essentially the syndrome polynomial of the (unknown) burst by removing the known erasure locator polynomial $\lambda(x)$ [6]. Following the analysis for Theorem 2, the probability that there exists a spurious burst of length $f'(f' \le f \le r-2\delta-1)$ resulting in the syndrome polynomial $\tilde{S}(x)$ is upper bounded by $q^{-(r-1-\delta-f)}$. On the other hand, by choosing up to δ distinct roots, there are $\binom{n+1-f}{\delta}$ possibilities for a valid error locator polynomial $\lambda(x)$, for the case of miscorrection, where $n+1-f$ is the number of random-error locators, composed of $\{0\}$ and all n error locators excluding f burst locators. Note there are totally q^δ choices for a normalized $\lambda(x)$ by letting $\lambda_0 = 1$. Therefore, for the case of failure, there are $q^\delta - \binom{n+1-f}{\delta}$ possibilities for an invalid polynomial $\lambda(x)$. Overall, both terms are bounded by $q^{-(r-1-2\delta-f)}$.

Remarks: When $\delta = 1$, the random-error locator polynomial $\lambda^{(r)}(x)$ obtained from Algorithm 3 is reduced to a linear polynomial, and thus does not need the Chien search to determine its root. Its only root is the random error locator when it lies in $\{1,\alpha^{-1},\alpha^{-2},\ldots,\alpha^{-(n-1)}\}$. Therefore, Algorithm 2 is essentially a simplified version of Algorithm 3. The proof of Theorem 5 indicates that the miscorrection probability roughly differs from the failure probability by a factor of

$$\frac{P_m(\mathsf{B}(s,f,\mathbf{u}),\{(\alpha_i,Y_i)\}_{i=1}^{\delta} \mid f)}{P_f(\mathsf{B}(s,f,\mathbf{u}),\{(\alpha_i,Y_i)\}_{i=1}^{\delta} \mid f)} \approx \frac{(n-f)^\delta/\delta!}{q^\delta-(n-f)^\delta/\delta!}. \quad (28)$$

Fig. 4 depicts the simulated miscorrection and failure rates as functions of distance to burst correction capability for the (30, 16) shortened Reed-Solomon code defined over $\mathrm{GF}(2^5)$, wherein the distance to burst correction capability stands for the difference of the length of injected burst and the corresponding burst correction capabilities, i.e., $r-1=13$ for the case of single burst, $r-3=11$ for the case of single burst and a random error, and $r-5=9$ for the case of single burst and two random errors. Note there is no miscorrection and failure for a single burst of length 5 and 2 random errors (corresponding to the two missing data points in the plot) since the errors fall into capability of hard-decision decoding. Fig. 5 depicts the simulated miscorrection and failure rates as functions of distance to burst correction capability for the (60, 40) shortened Reed-Solomon code defined over $\mathrm{GF}(2^6)$. The simulated results are consistent with the claims that the miscorrection and failure probabilities decay exponentially (at the rate of q^{-1}) with respect to the length of burst. In addition, the gaps between the miscorrection curve and the failure curve of Algorithm 3 in both figures are coincident with the assertion (28).

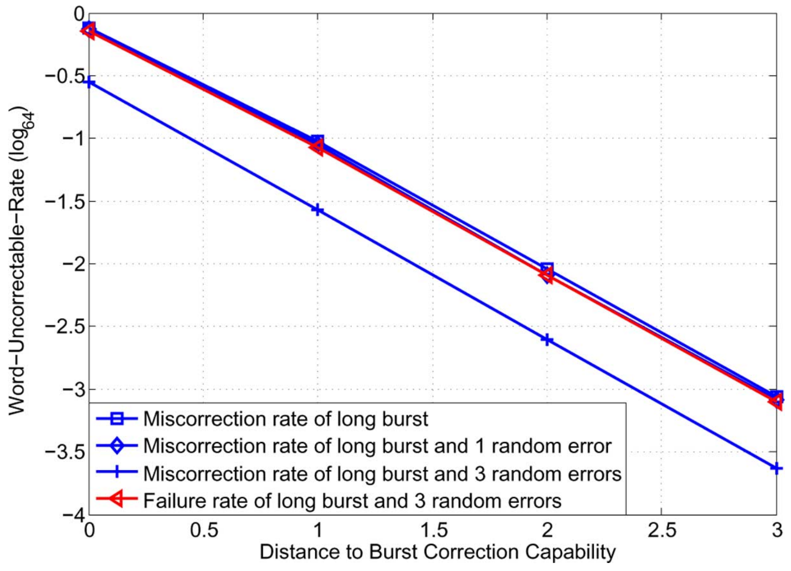

Fig. 5. Simulated miscorrection and failure rates as functions of distance to capability for the (60, 40) shortened Reed-Solomon code defined over $\mathrm{GF}(2^6)$.

VI. Conclusion

We presented three novel burst error correcting algorithms for an (n, k) Reed-Solomon code. The first one corrects a long burst with length up to $r-1$, where $r = n-k$, the second one corrects a long burst with length up to $r-3$ and a random error, and the last one corrects a long burst with length up to $r-1-2\delta$ and up to δ (a small constant) random errors. The algorithmic complexities are of the same order as error-and-erasure decoding, $O(rn)$, moreover, their hardware implementation shares the elements of Blahut error-and-erasure decoding. In contrast, all existing single-burst error correcting algorithms, which are equivalent to the proposed first algorithm, have complexity $O(r^2n)$. The miscorrection probability for all three algorithms and the failure probability for the third algorithm all decay at the rate of q^{-1} with respect to the length of burst, where q is the field size. Overall, our developments demonstrate that Reed-Solomon codes exhibit near-optimal burst correction capability and compare favorably to existing burst correcting codes in literature.

Acknowledgment

The author wishes to thank Dr. G. Ungerboeck for clarifying the section on single burst correction, Dr. M. Fossorier for pointing out the dual relation to the approach in [11], and the anonymous referee as well as the editor G. Seroussi for highly detailed and constructive comments.

References

[1] R. E. Blahut, *Algebraic Codes for Data Transmission*. Cambridge, U.K.: Cambridge Univ. Press, 2003.
[2] E. R. Berlekamp, "Elongated Burst Trapping," U.S. 4 916 702, 1990.
[3] J. Chen and P. Owsley, "A burst-error-correcting algorithm for Reed-Solomon codes," *IEEE Trans. Inf. Theory*, vol. 38, pp. 1807–1812, Nov. 1992.
[4] R. T. Chien, "Cyclic decoding procedures for Bose-Chauduri-hocquenghem codes," *IEEE Trans. Inf. Theory*, vol. 10, pp. 357–363, 1964.
[5] E. Dawson and A. Khodkar, "Burst error-correcting algorithm for Reed-Solomon codes," *Electron. Lett.* , vol. 31, pp. 848–849, 1995.
[6] G. D. Forney Jr., "On decoding BCH codes," *IEEE Trans. Inf. Theory*, vol. 11, pp. 549–557, 1965.
[7] J.-H. Jeng and T.-K. Truong, "On decoding of both errors and erasures of a Reed–Solomon code using an inverse-free Berlekamp–Massey algorithm," *IEEE Trans. Commun.*, vol. 47, pp. 1488–1494, Oct. 1999.
[8] D. Raphaeli, "The burst error correcting capabilities of a simple array code," *IEEE Trans. Inf. Theory*, vol. 51, pp. 722–728, Feb. 2005.
[9] R. M. Roth and P. O. Vontobel, "List decoding of burst errors," *IEEE Trans. Inf. Theory*, vol. 55, pp. 4179–4190, Sep. 2009.
[10] R. M. Roth and G. Seroussi, "Reduced-redundancy product codes for burst error correction," *IEEE Trans. Inf. Theory*, vol. 44, pp. 1395–1406, Jul. 1998.
[11] S. Song, S. Lin, K. Abdel-Ghaffar, A. Ding, W. H. Fong, and M. P. C. Fossorier, "Burst decoding of cyclic codes based on circulant parity-check matrices," *IEEE Trans. Inf. Theory*, vol. 56, pp. 1038–1047, Mar. 2010.
[12] L. Yin, J. Lu, K. B. Letaief, and Y. Wu, "Burst-error-correcting algorithm for Reed-Solomon codes," *Electron. Lett.*, vol. 37, pp. 695–697, 2001.

Yingquan Wu (M'07–SM'08) was born in the People's Republic of China. He received both the B.S. and M.S. degrees in mathematics from Harbin Institute of Technology, P.R. China, in July 1996 and July 1997, respectively. He received the M.S. degree in electrical engineering from the State University of New York at Buffalo in August 2000. He received the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign in October 2004.

He currently works at Link_A_Media Devices Corp., Santa Clara, CA, where he has been developing algorithms and architectures for error control coding and digital signal processing for data storage. His research interests lie in the areas of algebraic coding theory, soft-decision decoding techniques, and the efficient VLSI design of general communication algorithms.