Implementation of Reed Solomon Code Over GF(9) and 9-Ary Symmetric Channel

Elyes Balti, *Student Member, IEEE*

***Abstract*—Reed-Solomon and related codes have recently become very important for erasure correction in large disk arrays used in data centers. In this paper, we will implement a 3-error correcting Reed-Solomon encoder and decoder over the field GF(9) generated by the primitive polynomial $D^2 + D + 2$ over GF(3) and the decoding is carried out by the Berlekamp. We simulate the encoder and decoder using Monte-Carlo simulations over the 9-ary symmetric channel that outputs the correct symbol with probability $(1 - p)$, and outputs one of the other 8 possible incorrect symbols with probability $p/8$. Then, we compare the simulated probability of symbol error $P(E)$ of out code with the union upper bound.**

***Index Terms*—Reed-Solomon Code, Berlekamp, Galois Field, Generator Polynomial, M-ary symmetric channel, union upper bound.**

I. Introduction

CHANNEL coding is basically from the class of signal transformations designed to improve the communication performance by enabling the transmitted signal to better resist the effects of various channel impairments such as noise fading and jamming [1]. The goal of channel coding is to improve the bit error rate (BER) performance of power limited and/or bandlimited channels by adding redundancy to the transmitted data [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18].

The mechanism by which Reed-Solomon codes correct errors is that the encoder adds redundant bits to the digital input data block. The decoder attempts to correct and restore the original data by removing the errors that are introduced in transmission, due to many reasons that might be the noise in the channel or the scratches on a CD. There are different families of Reed-Solomon codes and each has its own abilities to correct the number and type of errors [19], [20], [21], [22], [23], [24], [25], [26].

These codes are linear and a subsection of BCH codes and can be denoted as RS(n, k) with s-bit (Where s are symbols represented as a bit) symbols.

An n-symbol codeword is made by the k data symbols of s bits each and the encoder adds parity bits. Errors in a codeword can be corrected by the decoder up to t symbols where, $2t = n-k$.

II. Design Specifications

In this project, the Reed-Solomon (RS) code has the following specifications defined by Table 1.1.

Elyes Balti is with the Wireless Networking and Communications Group, Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX 78712 USA e-mail: ebalti@utexas.edu.

TABLE I: Code Specifications

Parameter	Specified value
Galois Field	GF(9)
p	3
m	2
n	8
k	2
t	3
Primitive polynomial	$2 + D + D^2$

III. Elements of GF(9)

A. Representation

The elements of GF(9) are represented in terms of powers, polynomials, and tuples representations as follows

TABLE II: Representation of elements of GF(9)

Power of α	Polynomial	Tuple
0	0	0 0
α^0	1	0 1
α^1	α	1 0
α^2	2α+1	2 1
α^3	2α+2	2 2
α^4	2	0 2
α^5	2α	2 0
α^6	α+2	1 2
α^7	α+1	1 1

B. Addition Table

TABLE III: Addition Table

+	0	1	α	α^2	α^3	α^4	α^5	α^6	α^7
0	0	1	α	α^2	α^3	α^4	α^5	α^6	α^7
1	1	α^4	α^7	α^3	α^5	0	α^2	α	α^6
α	α	α^7	α^5	1	α^4	α^6	0	α^3	α^2
α^2	α^2	α^5	1	α^6	α	α^5	α^7	0	α^4
α^3	α^3	α^5	α^4	α	α^7	α^2	α^6	1	0
α^4	α^4	0	α^6	α^5	α^2	1	α^3	α^7	α
α^5	α^5	α^2	0	α^7	α^6	α^3	α	α^4	1
α^6	α^6	α	α^3	0	1	α^7	α^4	α^2	α^5
α^7	α^7	α^6	α^2	α^4	0	α	1	α^5	α^3

C. Multiplication Table

TABLE IV: Multiplication Table

x	0	1	α	α^2	α^3	α^4	α^5	α^6	α^7
0	0	0	0	0	0	0	0	0	0
1	0	1	α	α^2	α^3	α^4	α^5	α^6	α^7
α	0	α	α^2	α^3	α^4	α^5	α^6	α^7	1
α^2	0	α^2	α^3	α^4	α^5	α^6	α^7	1	α
α^3	0	α^3	α^4	α^5	α^6	α^7	1	α	α^2
α^4	0	α^4	α^5	α^6	α^7	1	α	α^2	α^3
α^5	0	α^5	α^6	α^7	1	α	α^2	α^3	α^4
α^6	0	α^6	α^7	1	α	α^2	α^3	α^4	α^5
α^7	0	α^7	1	α	α^2	α^3	α^4	α^5	α^6

IV. MATLAB Representation

During the MATLAB simulation, I adopt the power representation of the elements of GF(9). For the sake of simplicity, I also used the following representation.

TABLE V: MATLAB Representation

Powers	MATLAB Representation
0	-1
α^0 = 1	0
α^1	1
α^2	2
α^3	3
α^4	4
α^5	5
α^6	6
α^7	7

V. Generator Polynomial

The generator polynomial $g(D)$ of this RS-code has a degree of 2*t = 6, and 2*t+1 = 7 terms. The roots of $g(D)$ are $\alpha, \alpha^2, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6$. The generator polynomial can be factorized as follows:

$$g(D) = (D-\alpha)(D-\alpha^2)(D-\alpha^3)(D-\alpha^4)(D-\alpha^5)(D-\alpha^6), \quad (1)$$

After expanding $g(D)$, it can be written as follows

$$g(D) = \alpha^5+\alpha^4 D+\alpha D^2+\alpha^7 D^3+\alpha^2 D^4+\alpha^6 D^5+D^6, \quad (2)$$

Note that $g(D)$ is a monic polynomial. In MATLAB, a given polynomial is represented by a vector having the coefficients of the polynomial. I adopt representation each polynomial starting by the coefficient relative to the lowest degree. For example, the generator polynomial is represented as follows

$$g = [5 \;\; 4 \;\; 1 \;\; 7 \;\; 2 \;\; 6 \;\; 0], \quad (3)$$

VI. Encoding

In this project, I adopt the systematic encoding of the codeword. Assume that we have a polynomial message of two elements. To perform the systematic encoding, we need to go through the following steps

1) T(D) = D^{2t} * message(D).
2) R(D) = T(D) mod g(D).
3) Tx(D) = T(D) - R(D).

VII. 9-Ary Symmetric Channel

This channel is designed so that various errors can be introduced into the transmitted codeword. Let is consider a random variable X having the same length of the codeword (n = 8) that follows the Bernoulli distribution with probability prob if an error occurs and (1 - prob) if there is no error. Randomly, we generate the vector X taking "1" if an error occurs and "0", otherwise. An example of the random variable X generated with prob = 0.4 is

$$X = 0 \;\; 1 \;\; 1 \;\; 1 \;\; 0 \;\; 1 \;\; 1 \;\; 0 \quad (4)$$

This is equivalent to there are 5 errors are introduced to the codeword at the positions 2, 3, 4, 6, 7.
The output of the 9-ary symmetric channel is the received codeword Rx(D).

VIII. Decoding

A. Syndrome Computation

The syndrome polynomial has (2t = 6) terms and each coefficient S_i is given by

$$S_i = Rx(\alpha^i), \;\; 1 \leq i \leq 6, \quad (5)$$

If S_i are all zeros for $1 \leq i \leq 6$, then there is no error in the received codeword $Rx(D)$. We can also check whether the codeword is altered by error or no by computing the summation of all the coefficients of the syndromes as follows

$$h = \sum_{i=1}^{6} S_i, \quad (6)$$

if h is zero, the codeword is received without an error, otherwise, the codeword is corrupted by the errors introduced by the 9-ary symmetric channel.

B. Error Locator Polynomial: Berlekamp Algorithm

To perform the decoding, we need to compute the error locator polynomial $\sigma(D)$ via the Berlekamp algorithm. The following figure illustrates the iterative way to compute $\sigma(D)$.

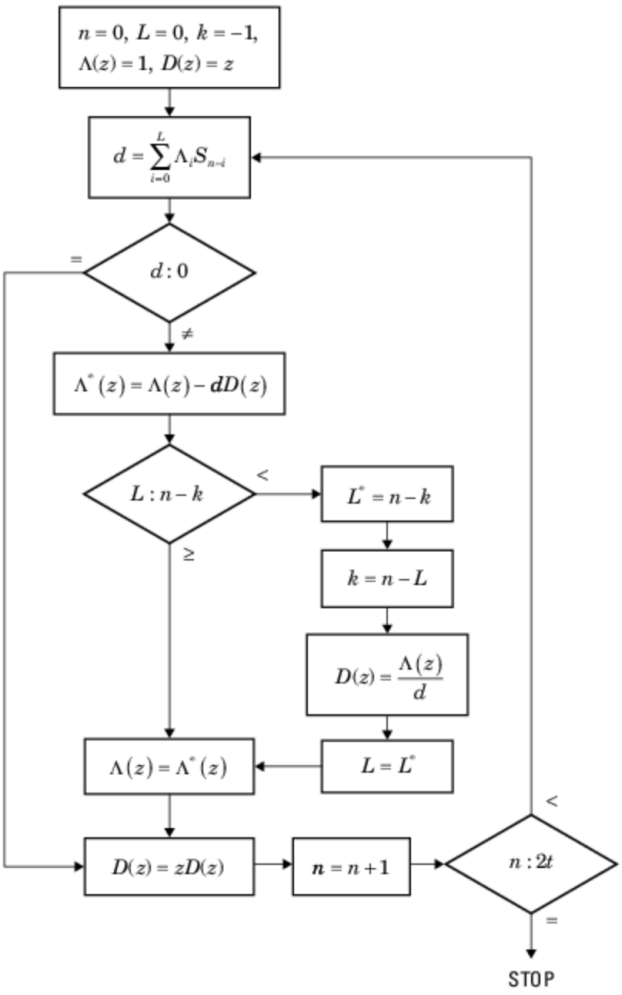

Fig. 1: Berlekamp Algorithm [27].

C. Error Locations and Roots: Chien Search Algorithm

The problem is to find the roots of the error locator polynomial $\sigma(D)$ over the finite field GF(9).
Chien substitutes the elements in the generator's order α^1, α^2, α^3,...etc, into the $\sigma(D)$ of degree c.
The Chien search is based on two observations

- Each non-zero β may be expressed as α^{i_β} for some i_β, where α is a primitive element of GF(9), i_β is the power number of primitive element α. Thus the powers α^i for $0 \leq i < 8$ cover the entire field (excluding the zero element).
- The following relationship exists:

$$\begin{aligned}\sigma(\alpha^i) &= \sigma_0 + \sigma_1(\alpha^i) + \sigma_2(\alpha^i)^2 + ... + \sigma_c(\alpha^i)^c \\ &= \gamma_{0,i} + \gamma_{1,i} + \gamma_{2,i} + ... + \gamma_{c,i},\end{aligned} \tag{7}$$

We define each $\Lambda(\alpha^i)$ as the sum of a set of terms $\{\gamma_{j,i} | 0 \leq j \leq c\}$, from which the next set of coefficients may be derived by

$$\gamma_{j,i+1} = \gamma_{j,i}\alpha^j, \tag{8}$$

In this way, we may start at $\gamma_{j,0} = \sigma_j$, and iterate through each value of i up to 8. If at any stage the resultant summation is zero, i.e. $\sum_{j=0}^{t} \gamma_{j,i} = 0$, then $\sigma(\alpha^i) = 0$ also, so α^i is a root. In this way, we check every element in the field.
Once the roots are found via the Chien Search algorithm, we determine the error locations $error_loc$ as the inverses of the roots of $\sigma(D)$. To illustrate the case, let's take the following example

1) $\sigma(D) = \alpha^5 + D + \alpha^3 D^2 + \alpha^2 D^3 + \alpha D^4$.
2) Apply Chien Search algorithm to find the roots which are $\alpha, \alpha^2, \alpha^4, \alpha^5$.
3) The error locations are the inverses of the roots which are $\alpha^7, \alpha^6, \alpha^4, \alpha^3$. In terms of polynomial, the error locations polynomial is $L(D) = D^3 + D^4 + D^6 + D^7$.

D. Error Evaluator Polynomial

The error evaluator polynomial $\Omega(D)$ is given by

$$\sigma(D)S(D) = \Omega(D) \text{mod} D^{2t}, \tag{9}$$

$\Omega(D)$ can be also interpreted as the convolution of $\sigma(D)$ and $S(D)$.

E. Error Magnitude

The error magnitude e_{i_v} is the error magnitude in the i_v position in the codeword, v is a value less than the error-correcting capability t of the code. e_{i_v} can be expressed as follows

$$e_{i_v} = \frac{\Omega(\alpha^{-i_v})}{\sigma'(\alpha^{-i_v})}, \tag{10}$$

$\sigma'(D)$ is the formal derivative of the error locator polynomial, $\sigma(D)$, and α is the primitive element of the Galois field GF(9). In our example, $i_v = 3, 4, 6, 7$.

F. Error Pattern Polynomial

Once we determine the error magnitudes at the error locations, the error pattern polynomial $e(D)$ is given by

$$e(D) = \sum_v e_{i_v} D^{i_v}, \tag{11}$$

G. Error Correction Checking

This code can only correct 3 or fewer errors. To check whether the received codeword is correctly decoded or not, we need to compute the difference between the received codeword and the error pattern. If the difference is equal to the transmitted codeword, then the decoding is successful, otherwise, the code is unable to correct the errors.

IX. Theoretical Symbol Error Probability Upper Bound

For a t-error-correcting block code is used strictly for error correction on a 9-ary symmetric channel with transition probability p, the probability that the decoder commits an erroneous decoding is upper bounded by [28, Eq. (3.26)]

$$P(E) \leq \sum_{i=t+1}^{n} \binom{n}{i} p^i (1-p)^{n-i}, \tag{12}$$

X. MATLAB Simulation

A. Scenario Illustration

We first generate a message polynomial as follows

$$Message = 5 \quad 6 \tag{13}$$

Then, perform the systematic encoding using the MATLAB function **Systematic_encoding** as follows

$$Tx = 7 \quad 0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \tag{14}$$

Using the function **Nine_Ary_symmetric_channel**, we introduce random errors with probability 0.4 to the transmitted codeword. AT this simulation, X and the received codeword Rx are given by

$$X = 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \tag{15}$$

$$Rx = -1 \quad 0 \quad 1 \quad 2 \quad 3 \quad 3 \quad 5 \quad 4 \tag{16}$$

The next step is to compute the syndrome S and Check_sum using the **Syndrome_computation** function. The result shows the following

$$S = 6 \quad 3 \quad 6 \quad 6 \quad 4 \quad 3 \tag{17}$$

$$Check\_sum = 1 \tag{18}$$

Check_sum is not zero (-1 over GF(9)), which means that the received codeword is corrupted by the error.
After calling the function **rs_berlekamp**, we get the error location polynomial $\sigma(D)$ and the error evaluator polynomial $\Omega(D)$. The results show the following

$$\sigma(D) = 6 \quad 4 \quad 6 \quad -1 \tag{19}$$

$$\Omega(D) = 4 \quad 7 \quad 1 \tag{20}$$

To get the roots and the error locations, we call the function **Chien_search_roots**. The simulation results show that

$$Roots = 1 \quad 7 \tag{21}$$

$$error\_loc = 7 \quad 1 \tag{22}$$

Equivalently, the relative error locations polynomial is $D+D^7$. We call the function **error_value** to compute the error magnitude. The results shows that

$$error = 0 \quad 1 \tag{23}$$

The next step is to compute the error pattern polynomial by calling the function **error_pattern**. The result shows that

$$error\_pattern\_poly = 3 \quad -1 \quad -1 \quad -1 \quad -1 \quad 5 \quad -1 \quad 5 \tag{24}$$

To get the decoded codeword, we compute the difference between the received codeword and the error_pattern_poly, by calling the function **subtraction**. The results shows that

$$Decoded = 7 \quad 0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \tag{25}$$

We observe that the decoded codeword is the same as the transmitted codeword. Hence, the decoding is successful. In addition, we can predict this result since the number of error is 3 with is equal to the error-correction capability t.
What happens if more than t errors corrupt the transmitted codeword. Let's illustrate this case by a simulation.

For the same systematic transmitted codeword Tx used in the first simulation, we generate the random variable X that introducing 6 errors (more than t = 3). The vector X is given by

$$X = 1 \quad 1 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1 \quad 1 \tag{26}$$

The received codeword Rx is

$$Rx = 6 \quad 1 \quad 1 \quad 7 \quad 4 \quad 4 \quad 0 \quad 3 \tag{27}$$

The error pattern polynomial

$$error\_pattern\_poly = -1 \quad -1 \quad -1 \quad -1 \quad -1 \quad 5 \quad -1 \quad -1 \tag{28}$$

The decoded codeword in this case if given by

$$decoded = 1 \quad 4 \quad 6 \quad 5 \quad 3 \quad 4 \quad 5 \quad 3 \tag{29}$$

Definitely, the decoded codeword is different from the transmitted codeword Tx. Hence the decoder failed to correctly decode the received codeword, since more than t errors (6 errors) corrupt the transmitted codeword.

XI. Codeword Error Probability

To compute the symbol error probability, we create a vector of probability values of the 9-ary symmetric channel as follows $prob = 0.1 : 0.1 : 1$. For each value of a given probability, we generate 1000 messages/codewords, and compute the average total number of errors that are not corrected.
Repeat this scenario for all the values of the probability of the 9-ary symmetric channel, so that to get the symbol error probability.

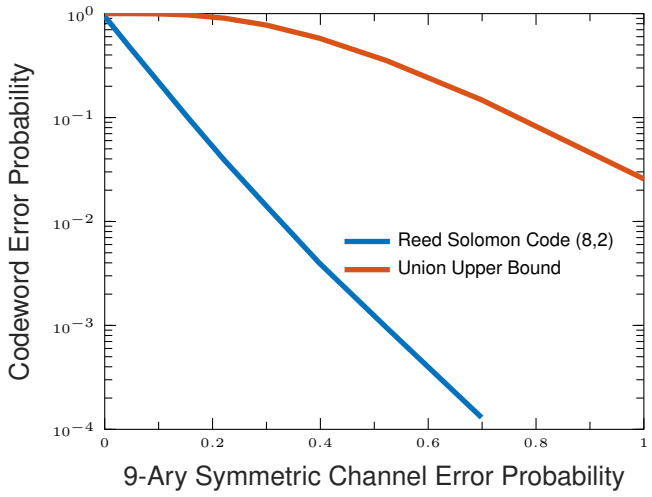

Fig. 2: Codeword error probability.

Figure 2 shows the variations of the symbol error probability with respect to the probability of the transition of 9-ary symmetric channel. Clearly, the decoder achieves better error performance compared to the upper bound of uncorrected symbol error probability of the linear block code. In addition, the upper bound is not the tightest to the error performance since there is a noticeable gap between the two curves. Clearly, the error performance achieved by the RS decoder with 3 error correcting is much better than for the linear block code.

XII. Conclusion

Reed-Solomon codes in comparison to linear block codes are difficult to implement but their performance is much better and consistent than linear block codes since they can handle long bursts of errors. These codes showed good performance compared to the upper bound of the symbol error probability of linear block codes.

Hence with the system that requires good performance but it can't handle complex architecture then RS codes might be more suitable for the practical purpose.

References

[1] S. L. William E. Ryan, *Channel Codes: Classical and Modern*, 1st ed., 2009.

[2] E. Balti and M. Guizani, "Mixed rf/fso cooperative relaying systems with co-channel interference," *IEEE Transactions on Communications*, vol. 66, no. 9, pp. 4014–4027, 2018.

[3] E. Balti and B. K. Johnson, "Tractable approach to mmwaves cellular analysis with fso backhauling under feedback delay and hardware limitations," *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 410–422, 2020.

[4] E. Balti, M. Guizani, B. Hamdaoui, and B. Khalfi, "Mixed rf/fso relaying systems with hardware impairments," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, 2017, pp. 1–6.

[5] E. Balti, "Analysis of hybrid free space optics and radio frequency cooperative relaying systems," Master's thesis, 2018.

[6] E. Balti, M. Guizani, B. Hamdaoui, and B. Khalfi, "Aggregate hardware impairments over mixed rf/fso relaying systems with outdated csi," *IEEE Transactions on Communications*, vol. 66, no. 3, pp. 1110–1123, 2018.

[7] E. Balti and B. K. Johnson, "On the joint effects of hpa nonlinearities and iq imbalance on mixed rf/fso cooperative systems," 2020.

[8] E. Balti, M. Guizani, and B. Hamdaoui, "Hybrid rayleigh and double-weibull over impaired rf/fso system with outdated csi," in *2017 IEEE International Conference on Communications (ICC)*, 2017, pp. 1–6.

[9] E. Balti and B. K. Johnson, "Stochastic geometry analysis of uplink cellular networks with fso backhauling: Cooperative relaying vs. reflecting surfaces," 2020.

[10] E. Balti, "Adaptive gradient search beamforming for full-duplex mmwave MIMO systems," 2020.

[11] E. Balti, N. Mensi, and D. B. Rawat, "Temporal csi correlation in mixed rf/fso cooperative relaying systems under joint effects of hpa nonlinearities and iq imbalance," 2021.

[12] E. Balti and B. L. Evans, "Adaptive self-interference cancellation for full-duplex wireless communication systems," 2021.

[13] N. Mensi, D. B. Rawat, and E. Balti, "Securing v2i communications in 5g and beyond wireless system using gradient ascent approach," 2021.

[14] Y. Maalej and E. Balti, "Cuda-accelerated application scheduling in vehicular clouds under advanced multichannel operations in wave," 2020.

[15] N. Mensi, D. B. Rawat, and E. Balti, "Pls for v2i communications using friendly jammer and double kappa-mu shadowed fading," 2020.

[16] Y. Maalej, A. Abderrahim, M. Guizani, B. Hamdaoui, and E. Balti, "Advanced activity-aware multi-channel operations1609.4 in vanets for vehicular clouds," in *2016 IEEE Global Communications Conference (GLOBECOM)*, 2016, pp. 1–6.

[17] N. Mensi, D. B. Rawat, and E. Balti, "Physical layer security for v2i communications: Reflecting surfaces vs. relaying," 2020.

[18] E. Balti and B. K. Johnson, "Rate and power adaptation for multihop regenerative relaying systems," 2021.

[19] E. Balti, "Hybrid precoding for mmwave v2x doubly-selective multiuser mimo systems," 2021.

[20] E. Balti, M. Guizani, B. Hamdaoui, and Y. Maalej, "Partial relay selection for hybrid rf/fso systems with hardware impairments," in *2016 IEEE Global Communications Conference (GLOBECOM)*, 2016, pp. 1–6.

[21] E. Balti and B. K. Johnson, "Asymmetric rf/fso relaying with hpa non-linearities and feedback delay constraints," 2019.

[22] E. Balti and N. Mensi, "Zero-forcing max-power beamforming for hybrid mmwave full-duplex mimo systems," in *Proc. Int. Conf. on Adv. Systems and Emergent Tech.*, 2020, pp. 344–349.

[23] E. Balti and B. K. Johnson, "Sub-6 ghz microstrip antenna: Design and radiation modeling," 2019.

[24] E. Balti and M. Guizani, "Impact of non-linear high-power amplifiers on cooperative relaying systems," *IEEE Transactions on Communications*, vol. 65, no. 10, pp. 4163–4175, 2017.

[25] E. Balti and B. K. Johnson, "Mmwaves cellular v2x for cooperative diversity relay fast fading channels," 2020.

[26] E. Balti and B. L. Evans, "Hybrid beamforming design for wideband mmwave full-duplex systems," 2021.

[27] "The mathworks site." [Online]. Available: https://www.mathworks.com/help/comm/ug/error-detection-and-correction.html

[28] S. Costello, Daniel J.; Lin, *Error control coding : fundamentals and applications*, 2nd ed. Pearson Education, 2004.