Reed-Solomon Decoding Algorithms and Their Complexities at the DVB-H Link-Layer

Tero Jokela, Eero Lehtonen

Turku Centre for Computer Science, Department of Information Technology, University of Turku
Joukahaisenkatu 3-5, FI-20520 Turku, Finland
tetajo@utu.fi
elleht@utu.fi

***Abstract*— DVB-H (Digital Video Broadcasting for Handheld terminals) is a data broadcasting standard based on DVB-T (Digital Video Broadcasting - Terrestrial) that enables delivery of various Internet Protocol (IP) based services to mobile receivers. To combat the effects of mobility, a Forward Error Correction (FEC) scheme called MPE-FEC is introduced at the link layer. Different mechanisms for decoding the link layer MPE-FEC based on Reed-Solomon (RS) code have been suggested, but their effect on the decoding complexity as compared to the standard solution has not been analyzed so far. Yet, the complexity is still a crucial issue in battery powered portable devices. In this paper, complexities of different RS decoding algorithms together with their application in DVB-H are analyzed.**

I. Introduction

DVB-H (Digital Video Broadcasting for Handheld terminals) is a relatively new data broadcasting standard [1] that enables delivery of various Internet Protocol (IP) based services to mobile receivers. The DVB-H standard, which is based on and is compatible with DVB-T (Digital Video Broadcasting - Terrestrial), introduces solutions to the problems caused by the mobility of the handheld terminals receiving digital broadcast. These solutions were required to lower power consumption of the terminal, to add flexibility in the network planning, to provide good enough performance in mobile channels and to add compatibility with IP networks. Enhancements to conventional DVB-T systems include the addition of time-slicing and one more stage of error correction called the MPE-FEC (Multi-Protocol Encapsulation - Forward Error Correction) at the link layer. Time-slicing means that the transmission is time division multiplexed, i.e. one service is sent in bursts separated in time. The power-saving is achieved due to the fact that the receiver can switch off radio components between the bursts. The MPE-FEC includes a utilization of Reed-Solomon (RS) code combined with time interleaving to combat channel fading.

In this paper the complexities of different implementations of RS decoder (time and frequency domain) presented in [2] are be investigated based on number of Galois Field (GF) operations. Further, the effect of the decoding scheme presented in [3] called hierarchical decoding on the complexity of the MPE-FEC decoding is investigated using the frequency domain RS decoder.

The paper is organized as follows. First, different decoding algorithms are discussed in section II and their complexities analyzed in section III. Then the decoding schemes in DVB-H MPE-FEC and their complexities are discussed in section IV. Finally, concluding remarks are presented in section V.

II. Decoding algorithms

In this section, we present two known decoding algorithms for Reed-Solomon codes. We begin with frequency domain decoder, which is the main object of study in this paper. Later, an algorithm in the time domain will also be investigated. Both decoding algorithms are based on the Berlekamp-Massey algorithm.

A. Frequency domain decoding

Broadly speaking, decoding in frequency domain means Fourier transforming the received vector and performing the decoding tasks on the transformed vector. In the end of decoding the inverse Fourier transform is taken and some codeword is obtained. This way of decoding is natural for Reed-Solomon codes due to their definition.

As usual, by erasure we mean an error, whose location is known. The number or erasures in a received vector v is denoted by ρ and the number of undetected errors in the vector is denoted by ν. As for bounded distance decoding of any error-correcting code, a received vector is uniquely decodable if

$$\delta \geq 2\nu + \rho + 1,$$

where δ is code minimum distance.

We need one more definition before going to the decoding algorithm. Assume, that vectors e and f correspond to the error- and erasure-vector of a received vector v, that is, for some codeword c

$$c = v - (e + f).$$

Then we say, that the *error-erasure-locator polynomial* Λ is the polynomial, which has minimal degree and satisfies the (cyclic) convolutional equation

$$\Lambda * (E + F) = 0.$$

Here we identify the vector

$$\Lambda = (\Lambda_0, \ldots, \Lambda_{n-1})$$

whose components are coefficients of polynomial

$$\Lambda(x) = \Lambda_0 + \ldots + \Lambda_{n-1}x^{n-1}.$$

The next decoding algorithm is represented as in [2].

Algorithm 2.1 (Frequency domain decoder):

1) Assume, that v is the received vector having $\rho < \delta$ erasures in the locations $i_1, \ldots, i_\rho$, where δ is the distance of the code. First, fill these erasures with zeros. Next, determine the erasure-locator polynomial $\Psi(x)$ from the formula

$$\Psi(x) = \prod_{l=1}^{\rho}(1 - x\omega^{i_l}),$$

where ω is a primitive element of the Galois field $GF(q)$. Also Fourier transform of v is taken, thus obtaining the vector V.
2) The Berlekamp-Massey algorithm explained below is executed on the vector V with Ψ to obtain the error-erasure-locator polynomial $\Lambda(x)$.
3) Next, extend recursively the known $\delta-1$ values of $E+F$ to n values by using the convolutional equation

$$\Lambda * (E + F) = 0$$

and the known values of Λ.
4) The decoding is completed by taking the inverse Fourier transform of the vector

$$C = V - (E + F).$$

Next we study more carefully the step 2 of the algorithm. The Berlekamp-Massey algorithm may be written in the following form:

Algorithm 2.2 (Berlekamp-Massey in frequency domain [2]):

1) Initialization: $\Lambda^{(\rho)}(x) = B^{(\rho)}(x) = \Psi(x), L_\rho = 0, r = \rho$, where $\Psi(x)$ is the erasure-locator polynomial.
2) For $r = \rho + 1, \ldots, \delta - 1$:

$$\Delta_r = \sum_{j=0}^{L_{r-1}+\rho} \{\Lambda^{(r-1)}(x)\}_j V_{r-1-j},$$

$$L_r = t_r(r - \rho - L_{r-1}) + (1 - t_r)L_{r-1},$$

$$\begin{pmatrix} \Lambda^{(r)}(x) \\ B^{(r)}(x) \end{pmatrix} = \begin{pmatrix} 1 & -\Delta_r x \\ t_r\Delta_r^{-1} & (1-t_r)x \end{pmatrix} \cdot \begin{pmatrix} \Lambda^{(r-1)}(x) \\ B^{(r-1)}(x) \end{pmatrix},$$

where $t_r = 1$ if both $\Delta_r \neq 0$ and $2L_{r-1} \leq r - \rho - 1$, otherwise $t_r = 0$.
3) Then $\Lambda^{(\delta-1)}(x)$ is the locator polynomial for both errors and erasures for the sequence $V_0, \ldots, V_{\delta-2}$.

In the step 3 of the algorithm 2.1 we need to solve the convolutional equation

$$\Lambda * (E + F) = 0. \tag{1}$$

This is done iteratively by extending the known values of $E+F$ with the help of the vector Λ from the previous step. The equation (1) can be written in the following form:

$$(E + F)_{((k))} = \sum_{j=1}^{\nu+\rho} \bar{\Lambda}_j (E + F)_{((k-j))}, \quad k = \delta - 1, \ldots, n - 1 \tag{2}$$

Now the right-hand side uses only known values of the vector $E+F$, and hence all of its components can be iteratively obtained.

B. Time-domain decoding

In time-domain decoding, we operate with the received vector itself rather than taking the Fourier transform of it in the very beginning of the algorithm. However, with Reed-Solomon codes the natural approach is to use the Fourier transform at least implicitly and as we shall see, the algorithm in time domain is closely related to the one we already discussed. The most evident way of decoding in time-domain is to take the inverse Fourier transform of each step of the algorithm 2.1. The algorithm presented here is derived in this manner in [2].

Algorithm 2.3 (Time-domain decoder):

1) First, determine the erasure-locator polynomial $\Psi(x)$ as in step 1) in algorithm 2.1. Next, take the inverse Fourier transform of the vector Ψ to obtain a time-domain vector ψ.
2) Execute the Berlekamp-Massey algorithm in time-domain explained below on the vectors v and ψ. As a result, a time-domain erasure-error locator λ is obtained.
3) Let us denote $u^{(\delta-2)} = v$. Now, for $r = \delta-1, \ldots, n-1$, let

$$\Delta_r = \sum_{i=0}^{n-1} \omega^{ir} \lambda_i u_i^{(r-1)}$$

$$u_i^{(r)} = u_i^{(r-1)} - \Delta_r \omega^{-ir} \quad \text{for all } i.$$

4) The codevector c can now be calculated as $c = v - u^{(n-1)}$.

Again, it is time to concentrate on the Berlekamp-Massey algorithm, now in time-domain.

Algorithm 2.4 (Berlekamp-Massey in time-domain [2]):

1) Initialization: $\lambda^{(\rho)} = b^{(\rho)} = \psi, L_\rho = 0, r = \rho$, where ψ is as above.
2) For $r = \rho + 1, \ldots, \delta - 1$,

$$\Delta_r = \sum_{j=0}^{n-1} \omega^{j(r-1)} \lambda_j^{(r-1)} v_j$$

$$L_r = t_r(r - \rho - L_{r-1}) + (1 - t_r)L_{r-1}$$

$$\begin{pmatrix} \lambda_i^{(r)} \\ b_i^{(r)} \end{pmatrix} = \begin{pmatrix} 1 & -\Delta_r\omega^{-i} \\ t_r\Delta_r^{-1} & (1-t_r)\omega^{-i} \end{pmatrix} \begin{pmatrix} \lambda_i^{(r-1)} \\ b_i^{(r-1)} \end{pmatrix}$$

for all *i*, where $t_r = 1$ if both $\Delta_r \neq 0$ and $2L_{r-1} \leq r - \rho - 1$, otherwise $t_r = 0$.
3) Then $\lambda^{(\delta-1)}$ is the error-erasure locator.

III. Complexity analysis of the decoding algorithms

In this section, we will study the complexities of the given decoding algorithms. The complexities will mainly be calculated with respect to the number of multiplications needed in the Galois field. Since the frequency domain decoder is based on Fourier transform, a few words concerning the Fourier transform's complexity are in order.

Fast Fourier transform (FFT) is a well-known, time-wise efficient recursive algorithm. Its complexity is of the order $O(n \log n)$, although this complexity does depend on how the exponentation of the primitive element is organized. In the following, we will assume that some look-up table (i.e. a table of logarithms and anti-logarithms with respect to the primitive element) is constructed, so that exponentation of the primitive element ω can be seen as a constant-time operation.

Let us next study the complexity of some polynomial operations. Assume, that we have two polynomials in Galois field,

$$\begin{aligned} a(x) &= a_0 + a_1 x + \ldots + a_m x^m \quad \text{and} \\ b(x) &= b_0 + b_1 x + \ldots + b_n x^n, \end{aligned} \quad .$$

To obtain their product (in monomial form) $c(x) = a(x)b(x)$, we clearly need $(m+1)(n+1)$ multiplications in the Galois field. This number of multiplications may be decreased if more sophisticated methods are used. However, here we use the *school algorithm* for polynomial multiplications. If $a_0 = b_0 = 1$, the number of needed multiplications reduces to mn, since multiplying by 1 is a trivial operation. Let now α be some element of the Galois field. To calculate the substitution $a(\alpha)$, we need m multiplications, which can be seen by writing the polynomial in different form (Horner scheme),

$$a(\alpha) = (\ldots((a_m\alpha + a_{m-1})\alpha + a_{m-2})\alpha \cdots)\alpha + a_0.$$

A. Frequency domain decoder

In the first step of the algorithm 2.1 the erasure-locator polynomial

$$\Psi(x) = \prod_{l=1}^{\rho}(1 - x\omega^{i_l})$$

is constructed. With the previous considerations we can calculate, that to obtain $\Psi(x)$ in monomial form, we need $(1/2)(\rho-1)\rho$ multiplications in the Galois field. In this step, also Fourier transform is taken hence adding the term $n \log n$ to the overall complexity.

The second step requires calculating the complexity of the Berlekamp-Massey algorithm. The first observation is, that in algorithm 2.2 always $L_r \leq r - \rho$. For every value of r the computation of Δ_r takes at most

$$L_{r-1} + \rho + 1 \leq (r-1) - \rho + \rho + 1 = r$$

multiplications. No multiplications in the Galois field are needed for computing L_r, but in the matrix equation we need at most $2(r+1)$ multiplications. Altogether, for every value of r, we need at most

$$r + 2(r+1) = 3r + 2$$

multiplications in the Galois field. Counting over all values of r, we get that the number of multiplications is at most

$$\begin{aligned} \sum_{r=\rho+1}^{\delta-1} 3r + 2 &= 2(\delta - \rho - 1) + 3\big(\sum_{r=1}^{\delta-1} r - \sum_{r=1}^{\rho} r\big) \\ &= 2\big(\delta - \rho - 1\big) + \big(3/2\big)\big((\delta-1)\delta - \rho(\rho+1)\big) \\ &= 2(\delta - \rho - 1) + (3/2)(\delta + \rho)(\delta - \rho - 1) \\ &= \big(\delta - \rho - 1\big)\big((3/2)(\delta + \rho) + 2\big). \end{aligned}$$

The complexity of the third step in the algorithm 2.1 can be derived from the equation (2). The number of needed iterations of this equation is

$$(n-1) - (\delta - 1) + 1 = n - \delta + 1.$$

For every iteration element, we need at most $\nu + \rho$ multiplications in the Galois field. Thus, the complexity of the third step is at most $(\nu + \rho)(n - \delta + 1)$.

The fourth step of the algorithm introduces another Fourier transform and hence is of complexity $n \log n$.

The overall complexity of the decoding algorithm 2.1 with respect to the number of multiplications in the Galois field $GF(q)$ is at most

$$\begin{aligned} \mathcal{C}(\nu, \rho) = \tfrac{(\rho-1)\rho}{2} + 2n\log n + \\ \big(\delta - \rho - 1\big)\big((3/2)(\delta + \rho) + 2\big) + (\nu + \rho)(n - \delta + 1). \end{aligned} \quad (3)$$

Note that one easily verifies that the number of additions needed in the algorithm 2.1 is of the same magnitude as the number of multiplications. Hence the overall complexity of the algorithm is of the order mentioned in the previous equation.

B. Time-domain decoder

Let us then study the complexity of the algorithm 2.3. As we see, it begins with nearly the same operations as in the previous case, the overall complexity of the first step being $(1/2)(\rho-1)\rho + n \log n$.

The second step introduces the Berlekamp-Massey algorithm in time-domain. For each step of the iteration, calculating Δ_r takes at most $2n$ multiplications. The matrix multiplications take additional $4n$ multiplications, so the overall complexity for each iteration step is $6n$. Since there are $\delta - \rho - 1$ of these steps, the complexity of the Berlekamp-Massey algorithm is at most $6n(\delta - \rho - 1)$.

In the third step of the algorithm 2.3 we need in each iterations at most $3n$ multiplications. There are $n - \delta - 1$ iterations in total, and thus the complexity of this third step is at most $3n(n - \delta - 1)$.

Thus the complexity of the algorithm 2.3 is at most

$$\mathcal{C}(\nu, \rho) = \frac{(\rho-1)\rho}{2} + n\log n + 6n(\delta - \rho - 1) + 3n(n - \delta - 1). \quad (4)$$

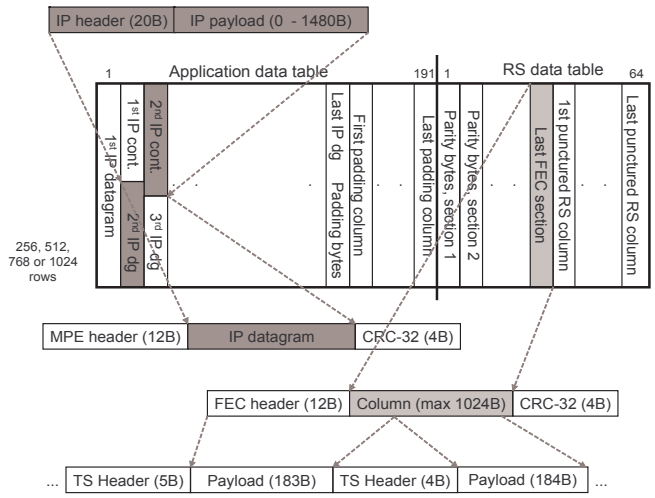

Fig. 1. Link layer operations [5]

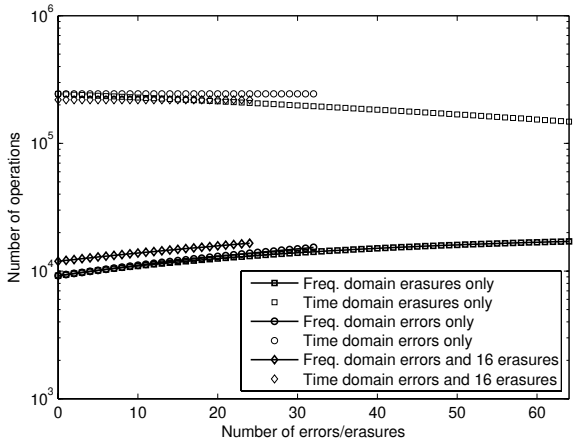

Fig. 2. Dependence of complexity on number of errors and erasures for RS(255,191), $\delta = 65$ code

This complexity has no dependence on the number of errors. This is mainly due to two reasons: First of all, the complexity analysis was rather coarce since the number of multiplications may be diminished by not counting the trivial multiplications. Secondly, this algorithm was constructed in such manner, that most of the calculations are done *for all* i, that is, for all components of a vector of size n.

IV. Decoding schemes in DVB-H

In DVB-H MPE-FEC, different options exist to obtain the erasure information for the RS decoder. It is suggested in [4] that the erasure information could be obtained from the CRC error detection mechanism embedded in MPE(-FEC) sections in the encapsulation process. Another option is to use the error information contained in the TS (Transport Stream) packet headers. In [3] two decoding methods for MPE-FEC based on correcting both errors and erasures were proposed. In the proposed methods also possibly erroneous data is inserted into the MPE-FEC frame, in contradiction to the method suggested in the standard, where all unreliable data is erased and possibly many correct bytes are lost. It is also possible to ignore available erasure information, and use pure error RS decoding. Regardless of the source of erasure information (CRC or TS packet header) the RS erasure decoding procedure may be performed basing on the widely known algebraic algorithms like the ones described in section II.

MPE-FEC frame can be thought as a matrix, where data from the application layer is inserted columnwise. RS encoding is done in the rowwise manner introducing time interleaving for the codewords, since the data is again transmitted columnwise. In the receiver the MPE-FEC frame is reassembled and erasure information created. Then all the rows of the MPE-FEC frame are decoded with RS decoder that can utilize the erasure information. The link layer operations and the structure of the MPE-FEC frame is show in Fig. 1. The MPE-FEC frame and its transmission is more thoroughly presented in [4]. From the complexity point of view the thing that makes the change when the code and decoding algorithm are fixed, is the numbers of undetected errors and erasures hitting RS codewords. The dependence of the complexity on numbers of errors and erasures for frequency and time domain decoders for the MPE-FEC RS(255,191) code is given in Fig. 2. As examples of possible scenarios, errors only, erasures only and combination of both are presented. The complexities are calculated from the expressions (3) and (4) presented in section III.

A. Complexities of the MPE-FEC decoding schemes

From the complexity analysis of the RS decoding algorithms and from Fig. 2 we deduce that the more efficient algorithm is the one operating in frequency domain. Its complexity is given in equation (3). Let us analyse the expression. First of all, we notice that the rightmost component of the sum,

$$(\nu + \rho)(n - \delta + 1), \tag{5}$$

is linear with respect to the sum of ν and ρ. Let us now denote the actual number of erroneus symbols in the received codevector by κ. Then always

$$\kappa \leq \nu + \rho,$$

equality holding when no correct symbols are erased. Thus to minimize the linear term, we would like to have no erasures corresponding to correct symbols in the codevector.

Let us next study the term

$$\frac{(\rho-1)\rho}{2} + \big(\delta - \rho - 1\big)\big((3/2)(\delta+\rho)+2\big). \tag{6}$$

Let us denote it by $g(\rho)$ and then minimize it. Calculating the first derivative of this function, we get

$$\begin{aligned} g'(\rho) &= \frac{2\rho-1}{2} - \frac{3(\delta+\rho)+4}{2} + \frac{3(\delta-\rho-1)}{2} \\ &= \frac{-4\rho-8}{2} = 0, \end{aligned}$$

when $\rho = -2$. From this, and the form of expression (6) we see, that $g(\rho)$ is a decreasing function for $\rho \geq 0$. Thus to minimize the term (6), we would like to have as many erased symbols as possible. This reduces the overall complexity of the frequency domain decoder, since it reduces the complexity of Berlekamp-Massey algorithm in frequency domain. In fact, if we know that we can decode using only erasures, we do not have to execute the Berlekamp-Massey algorithm at all, since we already know the locations of the errors.

Next we need to find the minimizing scheme for the whole algorithm 2.1. For this we need to compare the expressions (5) and (6). Comparing the first derivatives of these terms with respect to the variable ρ we find that

$$\begin{aligned}(n-\delta+1)+g'(\rho) &= n-\delta+1-2\rho-4 \\ &\geq n-\delta+1-2(\delta-1)-4 \\ &= n-3\delta-1 \geq 0\end{aligned}$$

when $3\delta \leq n-1$. This is the case for $n = 255$ and $\delta = 65$, which are the parameters in DVB-H Link Layer Reed-Solomon code. From this we deduce that for this code, taking extra erasures will only increase the complexity of the frequency domain decoding if those erasures correspond to correct symbols. Hence, to minimize the complexity of the decoding using frequency domain RS decoding, the most beneficial way is to have as reliable erasure information as possible.

This is the main idea of the decoding scheme (hierarchical decoding) suggested in [3]. The analysis in [5] shows that it is the best of the discussed decoding schemes with respect to the probability of successful decoding. Here we saw, that it also minimizes the complexity of the frequency domain decoder.

V. Conclusions

In this paper the complexities of frequency and time domain Reed-Solomon decoding algorithms together with their application at the DVB-H link-layer were analyzed. From the compared Reed-Solomon decoding algorithms, the one operating in frequency domain is better from the complexity point of view. The significant result is that the MPE-FEC decoding methods suggested in [3] that have better error correction capabilities also minimize the amount of operations necessary in the Reed-Solomon decoding.

References

[1] ETSI, "Digital Video Broadcasting (DVB); Transmission System for Handheld Terminals (DVB-H)", European Telecommunication Standard, EN 302-304, November 2004.

[2] Richard E. Blahut. *Algebraic Codes for Data Trasmission*, Cambridge University Press, 2003.

[3] H. Joki and J. Paavola, "A Novel Algorithm for Decapsulation and Decoding of DVB-H Link Layer Forward Error Correction", Proc. ICC 2006 (IEEE International Conference on Communications), Vol. 11, pp.5283-5288, June 2006.

[4] ETSI, "Digital Video Broadcasting (DVB); DVB-H Implementation Guidelines", Technical report, TR 102-377 (V1.1.1), January 2005

[5] J. Paavola, H. Himmanen, T. Jokela, J. Poikonen and V. Ipatov. "The Performance Analysis of MPE-FEC Decoding Methods at the DVB-H Link Layer for Efficient IP Packet Retrieval", IEEE Transactions on Broadcasting, Vol. 53, No. 1 (part 1), pp. 263-275 , March 2007.