UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

ERROR-CORRECTION CODING FOR HIGH-DENSITY MAGNETIC RECORDING CHANNELS

A Dissertation

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

degree of

Doctor of Philosophy

By

HAITAO XIA
Norman, Oklahoma
2004

ERROR-CORRECTION CODING FOR HIGH-DENSITY MAGNETIC RECORDING CHANNELS

A Dissertation APPROVED FOR THE
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

BY

Dr. Joao R. Cruz, Committee Chair

Dr. John Cheung

Dr. Joseph Havlicek

Dr. Tian-You Yu

Dr. Tomasz Przebinda

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude to my advisor, Dr. J. R. Cruz, for his mentoring, guidance, support and encouragement throughout my Ph.D. studies. It is his technical expertise and wisdom that make this work possible. His valuable teaching and advice, both academic and non-academic, will benefit me forever.

Secondly, I would also like to thank Dr. John Cheung, Dr. Tianyou Yu and Dr. Joseph Havelick, and Dr. Tomasz Przebinda for their support and their time serving on my committee.

I also thank my fellow students in the Communication Signal Processing Laboratory (CSPLab) for the helpful discussions and the friendship.

Finally, I would like to thank Hitachi, Ltd. for providing a graduate fellowship, which made this work possible.

To my parents and my wife Xi

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Future high density magnetic recoding channels (MRCs) are subject to more noise contamination and intersymbol interference, which make the error-correction codes (ECCs) become more important. Recent research of replacement of current Reed-Solomon (RS)-coded ECC systems with low-density parity-check (LDPC)-coded ECC systems obtains a lot of research attention due to the large decoding gain for LDPC-coded systems with random noise. In this dissertation, systems aim to maintain the RS-coded system using recent proposed soft-decision RS decoding techniques are investigated and the improved performance is presented.

The soft-decision RS decoding algorithms and their performance on magnetic recording channels have been researched, and the algorithm implementation and hardware architecture issues have been discussed. Several novel variations of KV algorithm such as soft Chase algorithm, re-encoded Chase algorithm and forward recursive algorithm have been proposed. And the performance of nested codes using RS and LDPC codes as component codes have been investigated for bursty noise magnetic recording channels.

Finally, a promsing algorithm which combines RS decoding algorithm with LDPC decoding algorithm together is investigated, and a reduced-complexity modification has been proposed, which not only improves the decoding performance largely, but also gurantees a good performance in high signal-to-noise ratio (SNR), in which area an error floor is experienced by LDPC codes.

Chapter 1

Introduction to Magnetic Recording Systems

1.1 Introduction

A magnetic recording system such as a hard disk drive normally includes the disk, the read and write head, the read/write electronics and the controller. The disk is coated with the recording media on which the user data is stored. The disk consists of concentric tracks, each track is further divided into sectors where the user data is stored in bit cells, typically each sector contains 512 bytes of user data plus some overhead. Using write and read heads mounted at the end of the arm, we can write and read user data from the hard disk. The controller interfaced with the host computer controls the arm motion, and performs error-correcting encoding and decoding. The write process is generally a non-linear process which uses a square wave to magnetize the recording media. If the amplitude of the write current is large enough, the media is magnetized to saturation, and the magnetization is a spatial copy of the write current. When the system reads the data from the recording media, the read head such as an inductive head or a magnetoresistive (MR) head is used to capture the changes in magnetization on the disk. The readback signal is amplified and further processed by the read channel detector, which translates the analog waveform into data $\hat{x}_k$ which will be sent to the error-correction code (ECC) decoder. Fig. 1.1 shows a diagram of a longitudinal magnetic recording system.

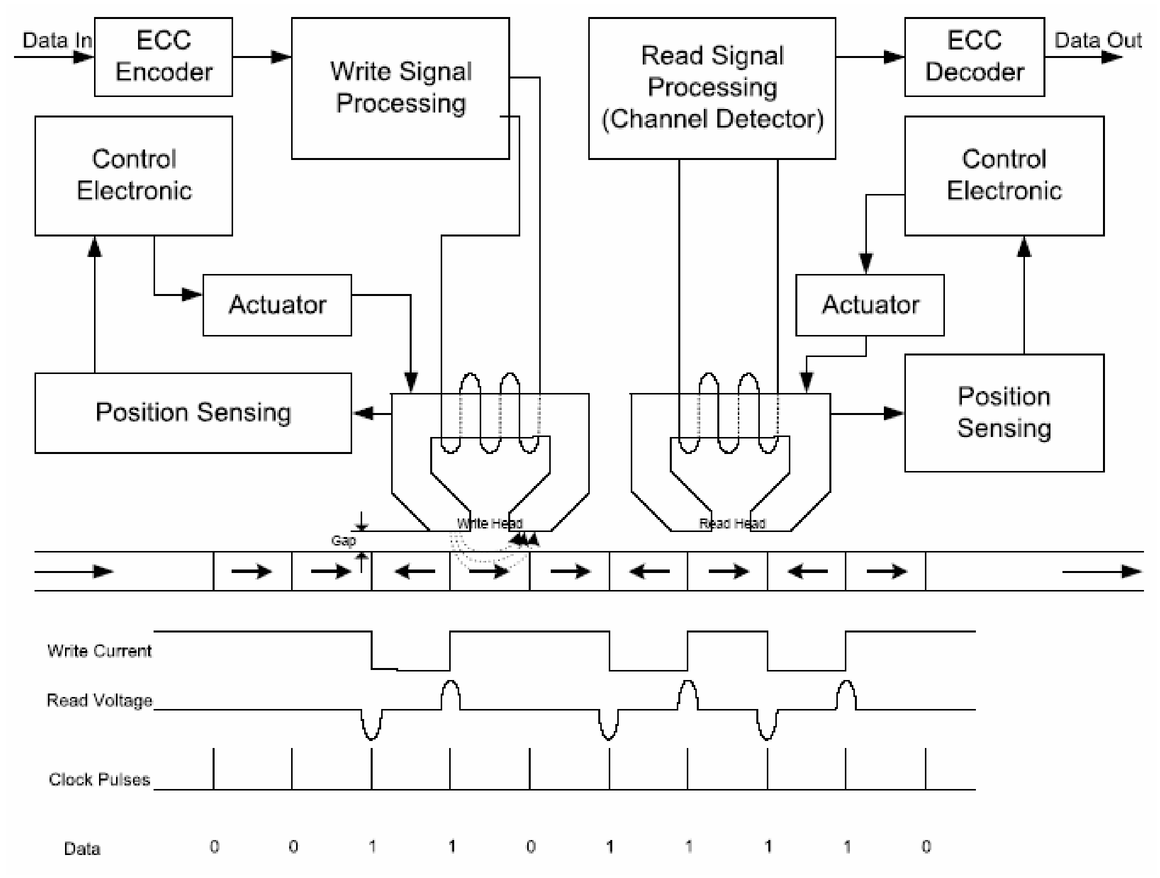


Fig. 1.1. A schematic diagram of a magnetic recording system using longitudinal recording techniques.

1.2 Partial Response Channel and Detection

A magnetic head senses the transitions in the direction of magnetization, which corresponds to a step signal in the write current. The magnetic recording channel response is thus characterized by the step response $s(t)$. Given a sequence of transitions recorded on the media, the readback signal is equal to the sum of signals from each transition. So the channel with discrete-time transfer function can be generally represented as $G(D)=\sum_{i=0}^{N} g_i D^i$ where $g_0=1$, D is the unit-delay operator corresponding to one modulation interval T of the channel, are termed partial response (PR) channels. In real magnetic recording systems, the readback isolated pulse $s(t)$ cannot be ideal, which means that some signal will leak to the adjacent transition, and the isolated pulses from each transition start to overlap, which is called inter-symbol interference (ISI). The closer each transition is, the larger the ISI will be, as well as the recording density.

Although most of the density increase is due to the improvement of the heads and recording media, digital signal processing and coding do have played an important role recently, especially since the implementation of the first commercial partial response maximum-likelihood (PRML) channel detector. The read head senses the transitions in the direction of magnetization, which corresponds to a step signal in the write current. The magnetic recording channel response is thus characterized by the step response $s(t)$. A commonly used model of $s(t)$ for longitudinal recording is the Lorentzian function [1], which is:

$$s(t) = \frac{V_0}{1 + \left(\frac{2t}{PW_{50}} \right)^2}, \tag{1.1}$$

where V_0 is the peak amplitude of $s(t)$, PW_{50} is a parameter specifying the pulse width at half the peak amplitude and is determined by the transition width in the recording media and the head-to-media distance. Besides longitudinal recording techniques, perpendicular recording has become a promising technique for the next generation of magnetic recording systems. As the channel density becomes larger and larger, the super-paramagnetic effect becomes the major obstacle to further increasing the recording areal density. Perpendicular magnetic recording enables high-density magnetic recording. In a perpendicular recording channel model, the isolated waveform reproduced from a recording transition can be approximated by a hyperbolic-tangent function [2]:

$$s(t) = V_0 \tanh\left(\frac{\ln 3}{T_{50}} t\right). \tag{1.2}$$

A common measure of the recording density for longitudinal recording is $S_c = PW_{50} / T$ with T is the symbol period, and for perpendicular recording is $K = T_{50} / T$ where T_{50} is time width required for $s(t)$to rise from $-V_0 / 2$ to $+V_0 / 2$. The read signal can be considered as the superposition of the channel step responses, each corresponding to a change (-1 to +1 or +1 to –1) in the channel input x_k: $y(t) = \sum_k (x_k - x_{k-1}) s(t - kT) + n(t)$ or simplified as $y(t) = \sum_k x_k p(t - kT) + n(t)$, with $n(t)$ as the noise and $p(t) = s(t) - s(t - T)$ as the channel pulse response which depends on the symbol period T of the channel. Besides the additive noise $n(t)$ introduced by channel, another noise called transition jitter noise is also a dominant noise for high density channels. The jitter noise Δt_k is modeled as a random shift in the transition position whose probability distribution function is truncated Gaussian with zero mean and variance σ_j^2 with σ_j being a percentage of T. So the readback signal if we consider transition jitter noise can be written as [3], [4]

$$y(t) = \sum_k x_k p(t - kT + \Delta t_k) + n(t). \tag{1.3}$$

And $p(t + \Delta t)$ can be approximated as $p(t + \Delta t) \approx p(t) + \Delta t \frac{d}{dt} p(t)$. A system model can be represented as in Fig. 1.2, where the readback signal $y(t)$ is filtered by a low-pass filter and sampled at the symbol rate. The received sequence, s_k, is then equalized to the sequence c_k, and afterwards a MLSE detector such as the Viterbi detector (VD) [5] will be used to determine the most likely input sequence $\hat{x}_k$.

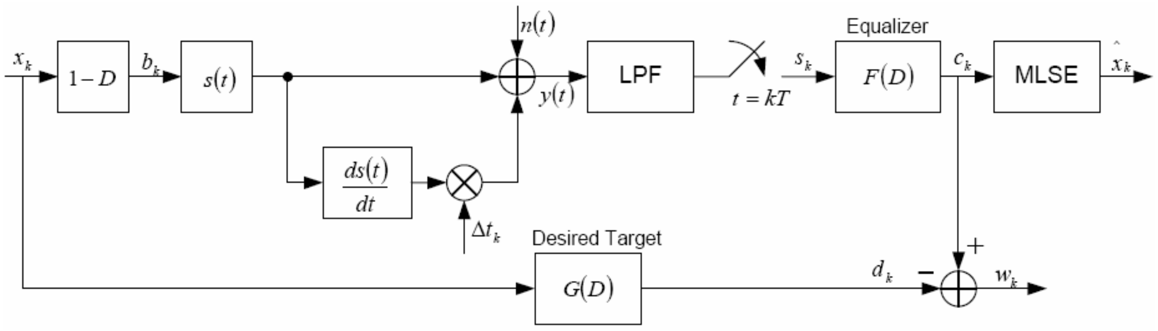


Fig. 1.2. System model for a longitudinal magnetic recording system.

It is shown in Fig. 1.2 that, in order to achieve the best channel detection performance, one needs to make the difference between the equalizer output c_k and the desired output d_k as small as possible. A straight forward way of designing the equalizer is setting $G(D) = 1$, which corresponding to a zero-forcing equalization. However, the equalizer designed by such method will lead to a noise enhancement [1]. Another commonly used method is the minimum mean-squared error (MMSE) approach [1], [4]. Let w_k be the difference between the desired output d_k and the equalizer output c_k, according to Fig. 1.2, it is shown that [4], [6]

$$E\{w_k^2\} = E\{[(s_k * f_k) - (x_k * g_k)]^2\} \tag{1.4}$$

where $E\{\bullet\}$ represents the expectation operation and $*$ represents the convolution operation. So given the channel input and a PR target $\mathbf{G} = [\mathrm{g}_0, g_1, \cdots, g_{L-1}]^{\mathrm{T}}$, we want to design the coefficients $\mathbf{F} = [f_{-M}, \cdots, f_0, \cdots f_M]^{\mathrm{T}}$ to minimize (1.4) subject to certain constraints, which will give [6]

$$\lambda = \frac{1}{\mathbf{I^T}\left(\mathbf{X} - \mathbf{C_{s,x}^T R^{-1} C_{s,x}}\right)^{-1}\mathbf{I}}, \tag{1.5}$$

$$\mathbf{G} = \lambda\left(\mathbf{X} - \mathbf{C_{s,x}^T R^{-1} C_{s,x}}\right)^{-1}\mathbf{I}, \tag{1.6}$$

$$\mathbf{F} = \mathbf{R}^{-1}\mathbf{C_{s,x}G}, \tag{1.7}$$

where λ is the Lagrange multiplier, and the L-element column vector $\mathbf{I}$ has a first element 1 and all other elements are zero. $\mathbf{X}$ is an $L \times L$ autocorrelation matrix of the input sequence x_k, $\mathbf{C_{s,x}}$ is an $N \times L$ cross-correlation matrix of sequences s_k and x_k, and $\mathbf{R}$ is an $N \times N$ autocorrelation matrix of a sequence s_k.

Of particular interest in longitudinal magnetic recording are partial response polynomials of the form $G(D) = (1-D)(1+D)^n$, where $n = 0,\ 1,\ 2, \cdots$. For $n = 0$, the channel is the $1-D$ or “dicode” partial response channel. The channel for $n=1$ is termed as the class-IV partial response (PR4) channel and the channel for $n=2$ is termed as the extended class-IV partial response (EPR4) channel [7]. Higher values of n result in more ISI and permit higher recording densities. In practical systems, a generalized partial response (GPR) target [8], [9] with arbitrary coefficients other than an integer-coefficient PR target is used which leads to a considerably better equalization performance. In perpendicular magnetic recording, the response to the magnetization pattern in the perpendicular media corresponds to a low-pass-filtered waveform of the write current. Unlike a longitudinal recording channel, this channel passes the DC component and thus presents challenges for read-channel design. In [2], several classes of $(L-1)$ -th order GPR targets in terms of $G(D)=(1-\alpha D)(p_0+p_1 D+\cdots p_{L-2}D^{L-2})$ are proposed for both additive white Gaussian noise (AWGN) and media noise channels. In this polynomial, the parameter $\alpha(0\le\alpha\le 1)$ for the $(1-\alpha D)$ operator is selected to adjust the DC component of the overall target response. For $\alpha=0$ or 1, this polynomial denotes the PR1-based DC full response or PR4-based DC-free response, respectively.

In Fig. 1.2, after the equalizer output, a maximum likelihood sequence estimator (MLSE) such as hard-decision Viterbi decoder is used to estimate the data sequence. In the next generation systems, a soft-output Viterbi algorithm (SOVA) [10] or a Bahl-Cocke-Jelinek-Raviv (BCJR) [11] algorithm could be used to supply soft information to the ECC decoder to improve the decoding performance.

1.3 Modulation Codes and Precoding

Magnetic recording systems use modulation codes to reduce ISI, as well as provide timing information, error monitoring information, etc. [1]. Since in magnetic recording systems, the readback signal only responds to transitions, a precoding process needs to be done for the input data, i.e. convert the data sequence to a sequence that each bit in the modulation code is a transition marker, where "0" represents an non-transition and "1" represents a transition. This process can be done by a precoder, which can be taken as a rate-1 encoder, such as the one $1/\left(1 \oplus D^2\right)$ precoder used in EPR4 systems. Since the EPR4 polynomial is $1+D-D^2-D^3$, this choice of precoder does not increase the decoder complexity. Also, the modulation codes must have a constraint on the maximum run of non-transitions. The commonly used modulation codes are run-length-limited (RLL) codes and maximum-transition-run (MTR) codes [1], [12], [13]. For example, for a (d,k) RLL code, the maximum run-length of "0"s between two "1"s is d and the maximal run length of "0"s is k. The effect of placing d zeros between successive "1"s is to spread the transitions apart enough to reduce the overlapping of the channel response due to successive transitions, which will reduce the ISI. Setting a upper limit of continuous "0"s with k ensures that transitions occur frequently enough for timing recovery. Therefore, when modulation codes are used, and errors happen, by looking at the violation of the code constraint, we will be able to tell if an error occured.

In the dissertation, we assume that the magnetic recording system exhibits perfect synchronization. And since we are mainly focus on the algorithms evaluation and development of ECC decoding algorithms, no modulation codes have been used in our simulation.

1.4 Signal-to-Noise Ratio (SNR)

Noise in the read signal comes from two major sources: the electronics noise arises from the read head and the pre-amplifier; the media noise exists because of the media defects and the imperfect magnetic domain alignment in the media. Electronics noise is white Gaussian noise and can be modeled as an additive component at the output of the recording channel. Media noise at low to middle recording densities can also be modeled as white Gaussian noise, but it undergoes the same read process as the user data; at high recording densities, media noise can be modeled as transition position jitter, the pulse amplitude and width jitter and partial-erasure effects.

Assume the step response of the magnetic recording channel is given as $s(t)$, the SNR is defined as

$$SNR = \frac{V_0^2 / T}{N_{AWGN} + N_{jitter}} \tag{1.8}$$

where V_0 is the peak amplitude of $s(t)$, and if we assume that the single-sided power spectrum density is N_0, then the in band additive noise power $N_{AWGN} = \frac{N_0}{2T}$ and the jitter noise power N_{jitter} is given as $N_{jitter} = \frac{1}{T}\sigma_j^2 \int_{-\infty}^{+\infty} \left| \frac{d}{dt} s(t) \right|^2 dt$ where σ_j^2 is the variance of the jitter. And

$$SNR = \frac{V_0^2 / T}{N_{AWGN} + N_{jitter}} = \frac{V_0^2 / T}{N_0 / 2T + (1/T)\sigma_j^2 \int_{-\infty}^{+\infty} \left| \frac{d}{dt} s(t) \right|^2 dt} = \frac{V_0^2}{N_0 / 2 + \sigma_j^2 \int_{-\infty}^{+\infty} \left| \frac{d}{dt} s(t) \right|^2 dt} \tag{1.9}$$

Since (1.9) gives an SNR in the form of energy per bit over total noise power, the definition of such SNR is invariant to channel density changes, which is different when coding is applied. For a Lorentzian response, the noise power of jitter noise is $N_{jitter} = \frac{1}{T}\sigma_j^2 \int_{-\infty}^{+\infty} \left| \frac{d}{dt} s(t) \right|^2 dt = \frac{\pi V_0^2 \sigma_j^2}{2T \cdot PW_{50}}$. For the noise power of jitter noise for other channels, the interested reader can refer to [14].

1.5 Error-Correcting Codes

Better equalization targets achieve better performance; however, the performance of the best equalization with ML detection cannot exceed the Matched Filter Bound (MFB) of the channel. In order to further improve the performance and maintain reliable information retrieval, a channel coding strategy needs to be used. Error-correction codes ensure higher noise tolerance at the receiver by adding redundancy into the user data to achieve a better separation of the data sequence. The code rate R is given as the ratio of the length information K over the length of the code transmitted given a certain encoding scheme. In magnetic recording channels, the code rate can also be represented as $R = \frac{S_u}{S_c}$ where S_u is the user density and S_c is the channel density. In [15], it is shown that on Lorentzian channels with only AWGN, the effective SNR is given as

$$SNR_{eff} \approx \frac{2\int_{-\infty}^{+\infty} |s(t)|^2 dt}{N_0 / 2} \frac{R^2}{S_u^2} \tag{1.10}$$

for high user density. From (1.10), it can be seen that the effective SNR is proportional to the square of the code rate R, which is different for most communication channels such as the AWGN channel which is proportional to the code rate. In other words, the coding gain of an encoding scheme should be larger to compensate for the code rate loss in magnetic recording systems.

In current magnetic recording systems, Reed-Solomon (RS) codes [16] have been used, in a configuration which includes: an RS encoder, a modulation encoder such as RLL encoder, a PR channel, a Viterbi detector, a modulation decoder and an RS decoder (See Fig. 1.3). Recently, with the invention of turbo codes [17] and turbo equalization, and the re-discovery of low-density parity-check codes, other coding systems have been the focus of attention of the magnetic recording industry. Turbo codes, introduced by Berrou *et al.* in 1993, provide a large coding gain for the memoryless AWGN channel and bring the system performance to within 1 dB of the Shannon capacity limit. In addition, another category of capacity-achieving codes, the low-density parity-check (LDPC) codes [18], [19] using message passing decoding algorithms [20] exhibit a performance within 0.13 dB of the AWGN channel capacity limit, which is closer than any other code discovered to date. The original turbo and LDPC codes were designed for memoryless AWGN channels, but researchers have applied these codes to magnetic recording channels [21]-[23].

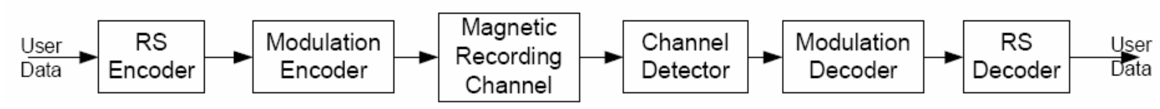


Fig. 1.3. Current magnetic recording system.

1.6 Problem Statement

Since large coding gains have been reported by replacing RS codes with LDPC codes for magnetic recording systems, LDPC codes have been considered as possible ECC codes for the next generation of magnetic recording systems.

However, the uncertainty of the performance of LDPC codes at high SNR is still a problem for magnetic recording systems where burst noise is the dominant noise. Compared to LDPC codes, RS codes have the ability to correct error bursts and the performance at high SNR is determined [24]. Also, the replacement of current RS-coded magnetic recording systems with an LDPC-coded system means a radical change of the system circuitry, which makes manufacturers hesitant to replace RS-coded systems with LDPC-coded systems. Moreover, in 1999, Guruswami and Sudan (GS) [25] proposed a new algorithm to improve the decoding capability of RS codes beyond the traditional error-correction capability, the half minimum distance. Later, Koetter and Vardy (KV) [26] further improved the GS algorithm by utilizing the soft information from the channel output, which gives a more than 1-dB gain for low rate RS code on an AWGN channel.

All the above motivates us to research soft-decision RS decoding algorithms over magnetic recording channels, which includes performance evaluation, performance improvement of the algorithms and reduced-complexity implementations.

1.7 Overview of the Dissertation

In this dissertation, the soft-decision RS decoding algorithms for magnetic recording channels have been researched. The algorithm implementation and hardware architecture issues and performance evaluations have been discussed for both longitudinal and perpendicular magnetic recording channels.

Several variations of the KV algorithm such as the soft Chase algorithm [27], the re-encoded Chase algorithm [28] and the forward recursive algorithm [29] have been proposed. The performance of nested codes with RS and LDPC codes as component codes have been investigated for bursty noise magnetic recording channels. Also, a recently proposed iterative decoding algorithm [30] for RS codes has been investigated and a reduced-complexity modified algorithm [31] has been presented.

Chapter 2 gives an overview of the decoding algorithms for RS codes which includes traditional hard-decision algorithms as well as newly proposed interpolation-based soft-decision RS decoding algorithms such as GS, KV and re-encoding algorithms.

Chapter 3 investigates the application of soft-decision RS decoding algorithms on magnetic recording systems, in terms of performance, reduced-complexity implementation and burst noise protection capability.

Chapter 4 gives a hardware implementation and an architecture discussion of the major step in the interpolation-based soft-decision RS decoding algorithms.

Chapter 5 proposes a new reliability-based soft-decision RS decoding algorithm called forward recursive algorithm, and evaluates its performance and implementation.

Chapter 6 discusses the performance of nested code schemes using RS codewords or LDPC codewords as component codes, and investigates their performance on bursty noise channels.

Chapter 7 investigates the performance of iterative RS decoding algorithms for magnetic recording channels.

Chapter 8 gives the conclusions and discussion for future research.

Bibliography

[1] J. W. M. Bergmans, *Digital Baseband Transmission and Recording*. Boston, MA: Kluwer Academic Publishers, 1996.

[2] H. Sawaguchi, Y. Nishida, H. Takano and H. Aoi, “Performance analysis of modified PRML channels for perpendicular recording systems,” *J. Magn. Magn. Mater.*, vol. 235, pp. 265-272, 2001.

[3] T. R. Oenning and J. Moon, “The effect of jitter noise on binary input intersymbol interference channel capacity,” in *Proc. IEEE Intl. Commun. Conf.*, pp. 2416-2420, 2001.

[4] P. Kovintavewat, I. Ozgunes, E. Kurtas, J. R. Barry and S. W. McLaughlin, “Generalized partial-response targets for perpendicular recording with jitter noise,” *IEEE Trans. Magn.*, vol. 38, pp. 2340-2342, Sept. 2002.

[5] G.D. Forney, Jr., “Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference,” *IEEE Trans. Inform. Theory*, vol. 18, pp. 363-378, May 1972.

[6] J. Moon and W. Zeng, “Equalization for maximum likelihood detector,” *IEEE Trans. Magn.*, vol. 31, pp. 1083-1088, Mar. 1995.

[7] P. Kabal and S. Pasupathy, “Partial-response signaling,” *IEEE Trans. Commun.*, vol. 23, pp. 921-934, Sept. 1975.

[8] R. D. Cideciyan, E. Eleftheriou, B.H. Marcus and D. S. Modha, “Maximum transition run codes for generalized parital response channels,” *IEEE J. Selected Areas Commun.*, vol. 19, pp. 619-634, Apr. 2001.

[9] N. M. Zayed and L. R. Carley, “Generalized partial response signaling and efficient MLSD using linear Viterbi branch metrics,” *in Proc. Global Telecommun. Conf.*, pp. 949-954, 1999.

[10] J. Hagenauer and P. Hoeher, “A Viterbi algorithm with soft-decision outputs and its applications,” in *Proc. IEEE Global Telecommun. Conf.*, pp. 1680-1686, 1989.

[11] L. R. Bahl, J. Cocke, F. Jelinek and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. 20, pp. 284-287, Mar. 1974.

[12] K. A. Schouhamer Immink, P. H. Siegel and J. K. Wolf, "Codes for digital recorders," *IEEE Trans. Inform. Theory*, vol. 44, pp. 2260-2299, Oct. 1998.

[13] K. A. Schouhamer Immink, *Coding Techniques for Digital Recorders*. Hemel Hempstead: Prentice Hall, 1991.

[14] J. Moon, "Signal-to-noise ratio definition for magnetic recording channels with transition noise," *IEEE Trans. Magn.*, vol. 36, pp. 3881-3883, Sept. 2000.

[15] W. E. Ryan, "Optimal code rates for concatenated codes on a PR4-equalized magnetic recording channel," *IEEE Trans. Magn.*, vol. 36, pp. 4044-4049, Nov. 2000.

[16] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *SIAM J. Applied Mathematics*, vol. 8, pp. 300-304, 1960.

[17] C. Berrou, A. Glavieux and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes," in *Proc. IEEE Intl. Conf. Commun.*, pp. 1064-70, 1993.

[18] D.J.C. MacKay and R.M. Neal, "Near Shannon limit performance of low density parity check codes," *Electronics Letters*, vol. 32, pp. 1645, Aug. 1996.

[19] R.G. Gallager, "Low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 8, pp. 21-28, Jan. 1962.

[20] D.J.C. MacKay, “Good error-correcting codes based on very sparse matrices,” *IEEE Trans. Inform. Theory*, vol. 45, pp. 399-431, Mar. 1999.

[21] W. Ryan, “Performance of high rate turbo codes on a PR4 equalized magnetic recording channel,” in *Proc. IEEE Intl. Conf. Commun.*, pp. 947-951, Atlanta, 1998.

[22] Z.-N. Wu, *Coding and Iterative Detection for Magnetic Recording Channels*. New York, NY: Kluwer Academic Publishers, 2000.

[23] H. Song, R.M. Todd and J.R. Cruz, “Application of low-density parity-check codes to magnetic recording channels,” *IEEE J. Select. Areas Commun.*, vol. 19, pp. 918-923, May 2001.

[24] S. B. Wicker, *Error Control Systems for Digital Communication and Storage*. Upper Saddle River, NJ: Prentice Hall, 1995.

[25] V. Guruswami and M. Sudan. “Improved decoding of Reed-Solomon and algebraic-geometric codes,” *IEEE Trans. Inform. Theory*, vol. 45, pp.1757-1767, Sept. 1999.

[26] R. Koetter and A. Vardy, “Algebraic soft-decision decoding of Reed-Solomon codes”, *IEEE Trans. Inform. Theory*, vol. 49, pp. 2809-2825, Nov. 2003.

[27] H. Xia, C. Zhong and J. R. Cruz, “A Chase-type algorithm for soft-decision Reed-Solomon codes on Rayleigh fading channels,” in *Proc. IEEE Global Commun. Conf.*, pp. 1751-1755, 2003.

[28] H. Xia and J. R. Cruz, “Application of soft-decision Reed-Solomon decoding on magnetic recording channels,” *IEEE Trans. Magn.*, vol. 40, pp. 3419-3430, Sept. 2003.

[29] H. Xia and J. R. Cruz, “Reliability-based forward recursive algorithms for algebraic soft-decision decoding of Reed-Solomon codes,” submitted to *IEEE Trans. Commun.*, 2004.

[30] J. Jiang and K. R. Narayanan, “Iterative soft decision decoding of Reed-Solomon codes based on adaptive parity check matrices,” to appear in *Proc. IEEE Intl. Symp. Inform. Theory*, 2004.

[31] H. Xia and J. R. Cruz, “Performance of reliability-based iterative soft-decision Reed-Solomon decoding over magnetic recording channels,” submitted to *IEEE Trans. Magn.*, 2004.

Chapter 2

Reed-Solomon Codes and Their Decoding Algorithms

2.1 Introduction

Reed-Solomon (RS) codes were proposed by Reed and Solomon in [1], and have found a lot of applications such as in magnetic recording systems, deep space communication systems, etc., because of their property of preventing burst errors. The original method of generating an RS code can be described as follows:

Let denote the information sequence be $\mathbf{f} = (f_0, f_1, \cdots, f_{k-1})$, where each entry $\mathbf{f}$ is an element of a finite field $GF(q)$ with $q = n + 1$ (The detailed finite field operations such as addition, multiplication, etc., can be found in [2, p. 93]). The RS (n,k) codes with codelength n and information length k can be generated by evaluating the information polynomial $f(x) = f_0 + f_1 x^1 + \cdots + f_{k-1} x^{k-1}$ over the nonzero distinct elements of $GF(q)$, that is:

$$RS : \mathrm{c} = \{f(x_0), f(x_1), \cdots, f(x_{n-1}) | x_0, x_1, \cdots, x_{n-1} \in GF(q)\}, \tag{2.1}$$

where the distinct elements are $x_i = \alpha^i, i = 0,1,\cdots,n-1$ with α as the primitive element in $GF(q)$. The generation of an RS codeword can also be represented in a matrix form

$$\mathbf{c}^{\mathbf{T}} = \begin{bmatrix} 1 & 1 & 1 & \cdots & \cdots & 1 \\ 1 & \alpha & \alpha^{2} & \cdots & \cdots & \alpha^{k-1} \\ \vdots & & & & & \vdots \\ 1 & \alpha^{i} & \alpha^{2i} & & & \alpha^{(k-1)i} \\ \vdots & & & & & \vdots \\ 1 & \alpha^{(k-1)} & \alpha^{2(k-1)} & \cdots & \cdots & \alpha^{(k-1)(n-1)} \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_i \\ \vdots \\ f_{k-1} \end{bmatrix} = \mathbf{Gf}^{T} \tag{2.2}$$

Also, RS codes can be taken as the subset of Bose-Chaudhuri-Hocquenghem (BCH) codes, which leads to an efficient systematic encoding method for RS codes. Given an information polynomial $f(x) = f_0 + f_1 x + \cdots + f_{k-1}x^{k-1}$ with degree $\deg(f(x)) < k$, and a generator polynomial $g(x) = (x-\alpha)(x-\alpha^2)\cdots(x-\alpha^{2t})$, the RS codeword can be encoded as the coefficients of polynomial $c^{'}(x) = f(x)g(x)$. For implementation purposes, we would like to send out the information symbols directly, and attach the generated parity check information at the end of information sequence. Let $b(x) = x^{2t} f(x) \bmod g(x)$, the RS codeword consists of $\mathbf{c}' = (b_0, b_1, \cdots, b_{2t-1,} f_0, f_1, \cdots, f_{k-1})$, where the parameter t satisfies $2t = n-k$. The systematic encoding method for RS codes can also be interpreted as the generator matrix $\mathbf{G}^{'}$ multiplying with the information vector $\mathbf{f}^{T}$, i.e. $\mathbf{c'}^{\mathbf{T}} = \mathbf{G'f}^{T}$. By carefully selecting the generator matrices $\mathbf{G}$ and $\mathbf{G'}$ such that $\mathbf{GG'}^{\mathbf{-1}}$ is non-singular, we can obtain the transformation between generalized RS codewords and RS codewords as $\mathbf{c}^{\mathbf{T}} = \mathbf{GG'}^{\mathbf{-1}}\,\mathbf{c'}^{T}$ (Detailed discussion can be found in [3]). It has been proved that RS codes are maximum-distance separable (MDS) codes, and the minimum distance of an $RS(n,k)$ is $d_{\min} = n-k+1$ [4, p. 188].

2.2 Hard-Decision Decoding of RS Codes

Suppose an RS codeword generated systematically by

$$RS : \mathbf{c} = \{c_0, c_1, \cdots, c_{n-1} \,|\, c_i \in coefficient\ \ of\ \ f(x)g(x), i = 0,1,\cdots,n-1\} \tag{2.3}$$

is sent out, and because of the noise introduced by the channel, we will get a noise contaminated hard-decision vector $\mathbf{y} = \mathbf{c} + \mathbf{e}$ for the input of RS decoder, where $\mathbf{e}$ represents an error vector. The error polynomial is given as:

$$e(x) = e_0 + e_1 x + \cdots + e_{n-1} x^{n-1}, \tag{2.4}$$

where we assume v errors occur, that is v coefficients of $e(x)$ are nonzero, and the locations of these v errors are $l_1, l_2, \cdots, l_v$. Then the error polynomial given above can be re-written as:

$$e(x) = e_{l_1} x^{l_1} + e_{l_2} x^{l_2} + \cdots + e_{l_v} x^{l_v}, \tag{2.5}$$

with $e_{l_1}, e_{l_2}, \cdots, e_{l_v}$ as the error magnitudes. If we evaluate the RS decoder input $\mathbf{y}$ or its polynomial represented as $y(x) = c(x) + e(x) = f(x)g(x) + e(x)$ with $2t$ distinct elements $\alpha, \alpha^2, \cdots, \alpha^{2t}$, we will obtain syndromes

$$S_j = y(\alpha^j) = e(\alpha^j) = e_{l_1} \alpha^{jl_1} + e_{l_2} \alpha^{jl_2} + \cdots + e_{l_v} \alpha^{jl_v}, \tag{2.6}$$

for $j = 1,2,\cdots,2t$ because $g(x) = (x-\alpha)(x-\alpha^2)\cdots(x-\alpha^{2t})$ becomes zero at those distinct elements. So now the decoding problem becomes solving $2t$ equations to find out the v error locations $l_1, l_2, \cdots, l_v$ and their magnitudes $e_{l_1}, \cdots, e_{l_v}$, that is $2v$ constraints. It is obvious that we will have a correct answer as long as $v \le t$.

The set of equations are not easy to solve directly. Here we would like to define some intermediate variables that can be computed from syndromes and from which we can find the error locations and its magnitudes. Let $X_i = \alpha^{l_i}, i = 1, \cdots, v$, and define a so-called error location polynomial $\Lambda(x)$ as:

$$\Lambda(x) = 1 + \Lambda_1 x + \cdots + \Lambda_v x^v = (1 - xX_1)(1 - xX_2)\cdots(1 - xX_v), \tag{2.7}$$

so if we know the coefficients of $\Lambda(x)$ and find the zeros of $\Lambda(x)$, we will obtain the error locations. What is the connection between this error location polynomial and the syndromes we compute above? Let us multiply both sides of the above equation by $e_{l_i}(X_i)^{j+v}$ and let $x = X_i^{-1}$ for $i = 1, 2, \cdots, v$, then we will get

$$e_{l_i}(X_i)^{j+v}\left(1 + \Lambda_1 X_i^{-1} + \cdots + \Lambda_v X_i^{-v}\right) = 0$$

or

$$e_{l_i}\left((X_i)^{j+v} + \Lambda_1 (X_i)^{j+v-1} + \cdots + \Lambda_v (X_i)^j\right) = 0 \tag{2.8}$$

Furthermore, let us sum up the equations from (2.8) for $i = 1, \cdots, v$, we get

$$\sum_{i=0}^{v} e_{l_i}\left((X_i)^{j+v} + \Lambda_1 (X_i)^{j+v-1} + \cdots + \Lambda_v (X_i)^j\right) = 0$$

or

$$\sum_{i=0}^{v} e_{l_i}\left(X_i\right)^{j+v} + \Lambda_1 \sum_{h=0}^{v} e_{l_i}\left(X_i\right)^{j+v-1} + \cdots + \sum_{i=0}^{v} \Lambda_v e_{l_i}\left(X_i\right)^{j} = 0 \qquad (2.9)$$

From (2.9), we know that

$$S_{j+v} + \Lambda_1 S_{j+v-1} + \Lambda_2 S_{j+v-2} + \cdots + \Lambda_v S_j = 0 \qquad (2.10)$$

More generally, we have

$$\Lambda_1 S_{j+v-1} + \Lambda_2 S_{j+v-2} + \cdots + \Lambda_v S_j = -S_{j+v} \qquad j = 1,2,\cdots,v\,, \qquad (2.11)$$

and writing the above equations in a matrix format,

$$\begin{bmatrix} S_1 & S_2 & \cdots & S_{v-1} & S_v \\ S_2 & S_3 & \cdots & S_v & S_{v+1} \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ S_{v-1} & S_v & \cdots & S_{2v-3} & S_{2v-2} \\ S_v & S_{v+1} & \cdots & S_{2v-2} & S_{2v-1} \end{bmatrix} \begin{bmatrix} \Lambda_v \\ \Lambda_{v-1} \\ \vdots \\ \Lambda_2 \\ \Lambda_1 \end{bmatrix} = \begin{bmatrix} -S_{v+1} \\ -S_{v+2} \\ \vdots \\ -S_{2v-1} \\ -S_{2v} \end{bmatrix}. \qquad (2.12)$$

The solution of this matrix can be obtained as long as the matrix is nonsingular, and in [5, Theorem 7.2.2], it is proved that when there are fewer than t errors, the syndrome matrix is nonsingular. So by computing the coefficients of $\Lambda(x)$ using

$$\begin{bmatrix} \Lambda_v \\ \Lambda_{v-1} \\ \vdots \\ \Lambda_2 \\ \Lambda_1 \end{bmatrix} = \begin{bmatrix} S_1 & S_2 & \cdots & S_{v-1} & S_v \\ S_2 & S_3 & \cdots & S_v & S_{v+1} \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ S_{v-1} & S_v & \cdots & S_{2v-3} & S_{2v-2} \\ S_v & S_{v+1} & \cdots & S_{2v-2} & S_{2v-1} \end{bmatrix}^{-1} \begin{bmatrix} -S_{v+1} \\ -S_{v+2} \\ \vdots \\ -S_{2v-1} \\ -S_{2v} \end{bmatrix}, \qquad (2.13)$$

and finding the zeros of $\Lambda(x)$ using Chien search [6], the location of the errors are obtained.

Now we have the error location polynomial $\Lambda(x) = \prod_{i=1}^{v}(1 - X_i x)$ for the case of v errors. If we express the syndromes in a polynomial representation, we have

$$\begin{aligned}
1 + S(x) &= 1 + \sum_{j=1}^{\infty} S_j x^j \\
&= 1 + \sum_{j=1}^{\infty}\left(\sum_{i=1}^{v} e_{l_i} X_i^j\right)x^j \\
&= 1 + \sum_{i=1}^{v} e_{l_i} \sum_{j=1}^{\infty}(X_i x)^j \\
&= 1 + \sum_{i=1}^{v} e_{l_i}\left(\frac{X_i x}{1 - X_i x}\right)
\end{aligned} \tag{2.14}$$

If we let

$$\begin{aligned}
\Omega(x) &= \Lambda(x)(1 + S(x)) \\
&= \left(\prod_{i=1}^{v}(1 - X_i x)\right)\left(1 + \sum_{i=1}^{v} e_{l_i}\left(\frac{X_i x}{1 - X_i x}\right)\right) \\
&= \Lambda(x) + \sum_{i=1}^{v} e_{l_i} X_i x \prod_{j \neq i}(1 - X_j x)
\end{aligned} \tag{2.15}$$

be the error evaluation polynomial, we can have the error magnitudes computed as $e_{l_i} = \dfrac{-X_i \Omega(X_i^{-1})}{\Lambda'(X_i^{-1})}$ by Forney's algorithm [7], where $\Lambda'(x)$ is the formal derivative of $\Lambda(x)$ as shown in [4, p. 221].

2.2.1 Berlekamp-Massey Algorithm

As shown above, the most computationally complex step in the decoding of an RS code comes from the matrix inversion, which is proportional to v^3. Obviously, for a moderate v, the complexity is still reasonable. However, if we want to correct a large number of errors, an efficient decoding algorithm is needed. In [8], Berlekamp proposed an efficient way of decoding RS codes, and in [9], Massey used a shift-register-based interpretation of Berlekamp's algorithm. From (2.10) we know that, the syndromes $S_j, j = j - v$ to j and the coefficients of the error locator polynomial $\Lambda(x)$ satisfy

$$S_{j+v} + \Lambda_1 S_{j+v-1} + \Lambda_2 S_{j+v-2} + \cdots + \Lambda_v S_j = 0$$

that is

$$S_{j+v} = -\Lambda_1 S_{j+v-1} - \Lambda_2 S_{j+v-2} - \cdots - \Lambda_v S_j \tag{2.16}$$

so each syndrome s_j can be calculated by the coefficient of $\Lambda(x)$ and previous syndromes $S_{j-1}, \cdots, S_{j-v}$. Equation (2.16) can be represented as a linear feedback shift register (LFSR) as shown in Fig. 2.1.

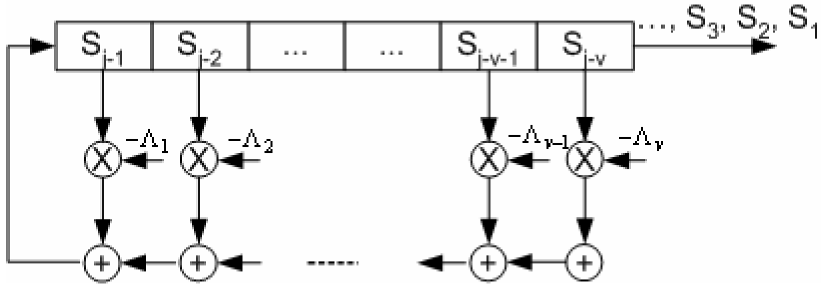


Fig. 2.1. Linear feedback shift register for generating syndromes.

Now the decoding problem becomes how to find an LFSR whose first $2t$ outputs are $S_1, \cdots, S_{2t}$. If we can successfully find such LFSR, the coefficient Λ_i for this LFSR provides the desired error location polynomial $\Lambda(x)$ which can be used to correct up to t errors in a received vector. For details on the Berlekamp-Massey (BM) algorithm and its proof, please refer to [4], [8], [9] and references therein.

2.2.2 Welch-Berlekamp Algorithm

Besides the algorithms described above, still there are many other hard-decision RS decoding algorithms such as the Euclid's algorithm [4, p. 224]. In this section, we would like to revisit an algebraic algorithm proposed by Welch and Berlekamp in [10], which will lead to the breakthrough algorithm proposed by Gurusawmi and Sudan in [11].

Suppose we received a channel output hard-decision vector $\mathbf{r} = \mathbf{c} + \mathbf{e}$ (the notation of the channel output vector $\mathbf{r}$ is slightly different to the decoder input $\mathbf{y}$ here) given an RS codeword $\mathbf{c}$ is sent out, the Welch-Berlekamp (WB) algorithm is based on the fact that if one can find nonzero polynomials $D(x)$ and $N(x)$ such that

$$D(x_i)\, r_i = N(x_i) \tag{2.17}$$

for $i = 0, 1, \cdots, n-1$ with minimal degree $deg(D(x))$, and $deg(N(x)) \leq deg(D(x))$, then $f(x) = N(x)/D(x)$ if $N(x)$ can be divided by $D(x)$, otherwise the codeword is not decodable. Furthermore, taking the last k points $(r_{n-k}, r_{n-k+1}, \cdots, r_{n-1})$, and using the generator polynomial to systematically re-encode them to generate another vector $\mathbf{r'}$, then by subtracting $\mathbf{r'}$ from $\mathbf{r}$, we get a vector $\mathbf{y} = (y_0, y_1, \cdots, y_{n-k-1}, 0, \cdots, 0)$ which is corrupted by the same error pattern $\mathbf{e}$. By finding two nonzero polynomials $D(x)$ and $N(x)$ satisfying (2.17) with r_i replaced by y_i, we can correctly decode it, as long as the number of errors in the codeword is less than $\lfloor (n-k+1)/2 \rfloor$. Note that the problem can be solved by a minimal interpolation at the distinct points $x_i = \alpha^i, i = 0, 1, \cdots, q-2$, and that we need to find a bivariate polynomial $Q(x, y) = D(x)y + N(x)$ that passes through all the point pairs $\{x_{i,}\ y_i\}$, given that subtraction is the same as addition in $GF(q)$. Since the bivariate polynomial needs to pass through k zeroes, we can further simplify the interpolation as

$$\begin{aligned} Q(x,y) &= D(x)y + N(x) \\ &= D(x)y + N'(x)V(x) \\ &= V(x)\left(D(x)\frac{y}{V(x)} + N'(x) \right) \\ &= V(x)Q(x, \overline{y}) \end{aligned} \tag{2.18}$$

where $V(x) = (x - \alpha^{n-k}) \cdots (x - \alpha^{n-1})$, $\overline{y} = y / V(x)$ and $Q(x, \overline{y}) = D(x)\overline{y} + N'(x)$. The interpolation of n points has been reduced to $n-k$ points $\{x_{i,}\ \overline{y}_i\}$ with

$$\overline{y}_i = y_i / V(x_i), \ \ i = 0, 1, \cdots, n-k-1.$$

***Example 2.1*:** Consider an example similar to the one in [12], using a (7, 3) RS code over $GF(8)$. Suppose that the generator polynomial is given by

$g(x)=(x-\alpha)(x-\alpha^2)(x-\alpha^3)(x-\alpha^4)$ and the information sequence is $(1, 0, 0)$, then the codeword $\mathbf{c}=(\alpha^3, \alpha, 1, \alpha^3, 1, 0, 0)$ is sent, but $\mathbf{r}=(\alpha, \alpha, 1, \alpha^3, 1, 0, \alpha)$ is received. Re-encoding the last three entries in $\mathbf{r}$, and subtracting it from $\mathbf{r}$, we get

$$\begin{aligned}\mathbf{y}=\mathbf{r}-\mathbf{r'}=\mathbf{r}-&(\alpha^4, \alpha^6, \alpha^3, \alpha^2, 1, 0, \alpha)\\ =&(\alpha^2, \alpha^5, \alpha, \alpha^5, 0, 0, 0)\end{aligned}$$

with $V(x)=(x-\alpha^4)(x-\alpha^5)(x-\alpha^6)$, and the modified interpolation points are given by $x_0=1, x_1=\alpha, x_2=\alpha^2,\ x_3=\alpha^3$ and

$$\overline{y}_0=\alpha^2/V(1)=\alpha^5, \overline{y}_1=\alpha^5/V(\alpha)=\alpha^6,$$

$$\overline{y}_2=\alpha/V(\alpha^2)=\alpha^4, \overline{y}_3=\alpha^5/V(\alpha^3)=1.$$

By interpolation, we find a bivariate polynomial

$Q(x,\overline{y})=\alpha x^2\overline{y}+\alpha x\overline{y}+\overline{y}+\alpha^5 x+\alpha^5$, and using (2.18) we can re-write it as

$$\begin{aligned}Q(x,y)&=(\alpha x^2+\alpha^3 x+1)y+(\alpha^5 x+\alpha^5)(x-\alpha^4)(x-\alpha^5)(x-\alpha^6)\\ &=\alpha(x-\alpha^6)(x+1)y+\alpha^5(x+1)(x-\alpha^4)(x-\alpha^5)(x-\alpha^6)\\ &=\alpha(x-\alpha^6)(x+1)(y-\alpha^4(x-\alpha^4)(x-\alpha)^5).\end{aligned}$$

After factorization, we get $y-\alpha^4(x-\alpha^4)(x-\alpha^5)$, which has nonzero values at $e_1=\alpha^2-\alpha^6=1$, and $e_6=0-\alpha=\alpha$. Subtracting the errors from the received vector gives the correct output $\mathbf{r}-\mathbf{e}=(\alpha^3, \alpha, 1, \alpha^3, 1, 0, 0)$.

■

2.3 Soft-Decision RS Decoding Algorithms

An decoding algorithm is said to be a soft-decision algorithm if utilize channel reliability information to help the decoding. Among them, generalized minimum distance (GMD) algorithm [13] and the Chase algorithm [14] are two most important soft-decision RS decoding algorithms.

(1) GMD algorithm: The basic idea behind Forney's GMD algorithm is that for hard-decision RS decoding algorithms, if we flag some unreliable symbols as erasures, we can further improve the decoding performance, given that the error-erasure correction capability is $2e+f<n-k+1$, where e is the number of errors, and f is the number of erasures in a received channel output sequence.

(2) Chase algorithm: There are three types of the Chase algorithm, but the basic idea of such algorithms is: Given a received channel output, search for the p least reliable bits in a received sequence, since each bit has two possibilities, either "0" or "1", so we can generate 2^p test patterns in terms of $(0,0,\cdots,0),(0,0,\cdots,1),\cdots,(1,1,\cdots,1)$ and for each test pattern, we will replace the p least reliable bits in a received sequence with a test pattern, then a hard-decision RS decoding is executed to see if a correct answer can be found.

Both of these two algorithms need to find the least reliable bits/symbols, then execute a hard-decision RS decoding algorithm. Besides these two soft-decision RS decoding algorithms and their variations [15], there are some other type of soft-decision algorithms which will be described in the following subsections.

2.3.1 Guruswami-Sudan Algorithm

Strictly, the Guruswami-Sudan (GS) algorithm [11] is not a soft-decision decoding algorithm, however, this algorithm leads to a more powerful implementation proposed by Koetter and Vardy in [16], which uses the soft-decision channel output information. Therefore here in this paper, we would like to discuss the GS algorithm in soft-decision RS decoding section. GS algorithm is an interpolation-based algorithm which can be taken as an extension of WB algorithm described above.

Let the input of the decoder be $\mathbf{y} = \mathbf{c} + \mathbf{e} = (y_0, y_1, \cdots, y_{n-1})$, where the RS codeword $\mathbf{c}$ is generated using (2.1). With the corresponding distinct elements over $GF(q)$ as $(x_0, x_1, \cdots, x_{n-1})$, we get a set of pairs $\{x_i, y_i\}_{i=0}^{n-1}$ (See Fig. 2.2). The WB algorithm described above implies that if we can find a bivariate polynomial $Q(x, y) = D(x)y + N(x)$ passing through all the points $\{x_i, y_i\}_{i=0}^{n-1}$ once, we can correctly decode a received vector given the error vector $\mathbf{e}$ has no more than $\lfloor (n-k+1)/2 \rfloor$ nonzero entries.

$$
\begin{array}{cccccc}
x_i = & x_0 & x_1 & x_2 & \cdots & x_{n-1} \\
f(x_i) = & f(x_0) & f(x_1) & f(x_2) & \cdots & f(x_{n-1}) \\
 & & & \downarrow & \cdots & \\
c_i = & c_0 & c_1 & c_2 & \cdots & c_{n-1} \\
 & & & \Downarrow & & \\
y_i = x_i + e_i = & y_0 & y_1 & y_2 & \cdots & y_{n-1} \\
 & & & \Updownarrow & & \\
(x_i, y_i) = & (x_0, y_0) & (x_1, y_1) & (x_2, y_2) & \cdots & (x_{n-1}, y_{n-1})
\end{array}
$$

Fig. 2.2. Conversion of channel output vector $\mathbf{y}$ into point pairs $\{x_i, y_i\}_{i=0}^{n-1}$.

This raises a question: What if we let the bivariate polynomial $Q(x, y)$ pass through those points more than once, say m times? The answer to this question is provided by Guruswami and Sudan [11], [17]. We can find such bivariate polynomial, which will make the performance of the interpolation-based algorithm far better than the traditional hard-decision RS decoding algorithms.

Before we go into the details, we would like to establish some notations, similar notation can be found in [11], [16], [17].

Notation 1. Let $B(x, y) = \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} b_{ij} x^i y^j$ be a bivariate polynomial over $GF(q)$, b_{ij} is the coefficient of $B(x, y)$. And let w_x, w_y be some integer numbers. The (w_x, w_y)-weighted degree of $B(x, y)$ is denoted as

$$\deg_{w_x, w_y}(B(x, y)) = \max\{iw_x + jw_y \mid b_{ij} x^i y^j \textit{ is a monomial in } B(x, y), b_{ij} \neq 0\}, \tag{2.19}$$

where it is said $\deg_{w_x,w_y}\left(x^{i'}y^{j'}\right) < \deg_{w_x,w_y}\left(x^{i}y^{j}\right)$ given $i'w_x + j'w_y < iw_x + jw_y$ or $i'w_x + j'w_y = iw_x + jw_y \ and \ j' < j$ or $i'w_x + j'w_y = iw_x + jw_y \ and \ j' = j \ and \ i' < i$. It worth noting that the $(1,1)$-weighted degree of $B(x,y)$ is simply the degree of $B(x,y)$. Also, the number of monomials of (w_x, w_y)-weighted degree no larger than δ is given as

$$N_{w_x,w_y}(\delta) \overset{\Delta}{=} \left|x^i y^j : i, j \geq 0 \ and \ iw_x + jw_y \leq \delta\right| \tag{2.20}$$

where $|\bullet|$ represents the cardinality. In [11], [16], it was proved that $N_{w_x,w_y}(\delta)$ satisfies

$$N_{1,k-1}(\delta) = \frac{(\delta+1)^2}{2(k-1)} + \frac{k-1}{2}\left(\left\lceil\frac{\delta+1}{k-1}\right\rceil - \left(\left\lceil\frac{\delta+1}{k-1}\right\rceil - \frac{\delta+1}{k-1}\right)^2\right) > \frac{\delta^2}{2(k-1)} \tag{2.21}$$

for $w_x = 1, w_y = k-1$, or more general lower bound $N_{w_x,w_y}(\delta) > \delta^2 / 2w_x w_y$.

Notation 2: A bivariate polynomial $B(x,y)$ is said to pass through a given point (α, β) with multiplicity m if the shift bivariate polynomial $B(x+\alpha, y+\beta)$ contains only monomials of degree larger than m, but no monomial has degree less than m, i.e.,

$$\sum_{i=i'}^{\infty}\sum_{j=j'}^{\infty}\binom{i}{i'}\binom{j}{j'}\alpha^{i-i'}\beta^{j-j'}b_{ij} = 0 \qquad \forall\, i', j' \geq 0 \ such\ that\ i'+j' < m \tag{2.22}$$

For a detailed proof, please see [11]. From (2.22), we can see that if we want to let a bivariate polynomial $B(x,y)$ pass through a given point with multiplicity m, the coefficients b_{ij} must satisfy $m(m+1)/2$ constraints. So if we have n points, then the total number of constraints $nm(m+1)/2$ needs to be satisfied. Let $\delta = \deg_{1,k-1}(B(x,y))$, if the total number of monomials in $B(x,y)$ is larger than the number of constraints, i.e.

$$N_{1,k-1}(\delta) > \frac{nm(m+1)}{2}, \tag{2.23}$$

we will have more unknowns in a set of linear equations than constraints, which definitely leads to a solution. The detailed GS algorithm is given below (the reader can also refer to [11], [16], [17] for a detailed proof).

Guruswami-Sudan Algorithm:

Inputs: 1. Receive hard-decision channel output vector $\mathbf{y} = \mathbf{c} + \mathbf{e}$

2. Multiplicity m

Step 1: Generate a sequence $\{(x_i, y_i)\}_{i=0}^{n-1}$ as shown in Fig. 2.2.

Step 2: (Interpolation Step) Find a nonzero bivariate polynomial $Q_m(x,y)$ which passes through point pairs $\{(x_i, y_i)\}_{i=0}^{n-1}$ with multiplicity m.

Step 3: (Factorization Step) Find all the polynomials $f(x)$ such that: $y - f(x)$ is a factor of $Q_m(x,y)$, with $deg(f(x)) < k$.

Step 4: (List-Decoding Step) Regenerate codeword $\mathbf{c}$ from all found $f(x)$, and find the one with least Euclidian distance to $\mathbf{y}$ as the output.

End.

So in the GS algorithm, by finding a bivariate polynomial $Q_m(x, y)$ that passes through $\{x_i, y_i\}_{i=0}^{n-1}$ with a constant multiplicity m, and identifying all its factors of the form $y - f(x)$ with $deg(f(x)) < k$, the coefficients of the polynomial $f(x)$ will be output as the decoding answer. The GS algorithm includes two major steps, which are *interpolation*, *factorization steps*. Several algorithms for implementing the interpolation and factorization steps can be found in [17], [18]. It is clear that if we set $m = 1$, the GS algorithm becomes the WB algorithm. And as multiplicity $m \to \infty$, the error-correction capability of the GS algorithm goes up to $t \leq n - \sqrt{n(k-1)}$ errors, far better than the traditional hard-decision RS decoding capability $\lfloor (n-k+1)/2 \rfloor$ particularly for samll code rates k/n.

2.3.2 Koetter-Vardy Algorithm

Although the GS algorithm improves the error-correction capability, there are still some ussatisfied question: How to determine the multiplicities value m, since we know that the higher the value of m, the better the performance is, but the decoding complexity increases since more constraints need to be satisfied which means more linear equation need to be solved. Recently, Koetter and Vardy [16] suggested a smart way to determine the multiplicity m of the GS algorithm by referring to the channel reliability information, and allowing m not be a constant.

Now instead of using the hard-decision vector $\mathbf{y}$, we can use the "soft" channel information, i.e., the probability of each output bit to be a '1' or a '0'. This probability can be converted into a reliability matrix $\Pi[i,j]$, where each entry represents the probability of the channel output $y_j = i$, $j = 0,\cdots,n-1; i = 0,\cdots,q-1$. Using this reliability matrix and a given parameter, the *total multiplicity* s, we can use *Algorithm A* in [1] to generate a multiplicity matrix $M[i,j]$. Each nonzero entry in M can be taken as a point $(x_t = i, y_t = j)$, whose multiplicity value is $m_t = m_{ij}$. Therefore, the channel output information has been translated into a sequence of points $\{(x_t, y_t), m_t\}_{t=0}^{N-1}$, where N is not necessary to be equal to n. The sum of multiplicities m_i of each point pair $\{x_i, y_i\}_{i=0}^{n-1}$ will be referred to as the *total multiplicity* s, which is a parameter determining decoding performance. The GS algorithm [11] can be taken as a special case of the Koetter-Vardy (KV) algorithm.

The multiplicity computation algorithm is given as follows:

Multiplicity Computation Algorithm (Algorithm A in [16]):

Inputs: 1. Channel reliability matrix $\Pi[i,j] = \lfloor \pi_{ij} \rfloor$

2. Total multiplicity s

Output: $M[i,j]$

Initialize: Let $\Pi^*[i,j] = \Pi[i,j]$ and $M[i,j] = [\mathbf{0}]$ as all-zero matrix, i.e. each entry $m_{ij} = 0$ in $M[i,j]$

Computation Step:

```
While $s > 0$ {
    Find the largest entry $\pi^*_{ij}$ in $\Pi^*[i,j]$,
    Set
        $\pi^*_{ij} = \frac{\pi_{ij}}{m_{ij}+2}$
        $m_{ij} = m_{ij} + 1$
        $s = s - 1$
}
```

So now the GS algorithm is updated to the KV algorithm with variable multiplicities as:

Koetter-Vardy Algorithm:

Inputs: 1. Receive channel reliability information and create $q \times n$ reliability matrix $\Pi[i,j]$, where each entry $\pi_{i,j}$ of this matrix represents the probability of the j th element to be i in $GF(q)$, i.e., $c_j = i$.

2. Total multiplicity s

Step 1(Multiplicity Computation Step): Compute multiplicity matrix $M[i,j]$ according to $\Pi[i,j]$ and total multiplicity s.

Step 2: Generate a one-dimension sequence $\{(x_t, y_t), m_t\}_{t=0}^{N-1}$ according to the nonzero entries in $M[i, j]$, where N is not necessary equal to the codelength n.

Step 3 (Soft Interpolation Step): Find a nonzero bivariate polynomial $Q_M(x, y)$ which passes through N points $(x_t, y_t)_{t=0}^{N-1}$ with multiplicity m_t respectively.

Step 4 (Factorization Step): Find all the polynomials $f(x)$ such that: $y - f(x)$ is a factor of $Q_M(x, y)$, with $deg(f(x)) < k$.

Step 5: Regenerate codeword **c** from all found $f(x)$, and by referring to the reliability information matrix, find the most-likely codeword as the output.

Compared to the GS algorithm, the KV algorithm has one more step, the *multiplicity computation step*, which generates variable multiplicities instead of constant multiplicities using channel reliability information. Also the *interpolation step* is modified to accommodate the variable multiplicities. The GS algorithm can be viewed as a special case of the KV algorithm. In the following, we will discuss the correctness of such interpolation-based algorithm and the detailed algorithms for the *(soft) interpolation* and *factorization steps* are given.

A. Correctness of Interpolation-based Algorithms

Definition 1 [16]: Let $C = C(M) = \sum_{i=0}^{q-1} \sum_{j=0}^{n-1} m_{ij}(m_{ij} + 1)/2$ be the cost of a given multiplicity matrix. Let δ denote as the degree of a given bivariate polynomial

$Q_M(x,y)$ generated according to the multiplicity matrix M. Denote the score $S_M(\mathbf{c}) = \langle M, [\mathbf{c}] \rangle$ of a multiplicity matrix M given an RS codeword $\mathbf{c}$ is sent out, where $\langle \bullet \rangle$ represents the inner product of two matrices, and $[\mathbf{c}]$ represents an $n \times n$ matrix with each column containing only a "1" at c_i rows, and c_i is the entry of codeword $\mathbf{c}$ for $i = 0, \cdots, n$.

Theorem 2.1 [16]:

Given a multiplicity matrix M, the polynomial $Q_M(x,y)$ generated by passing through those non-zero entries in M has a factor $y - f(x)$ where $f(x)$ can be used to generate codeword $\mathbf{c}$ using (2.1) if

$$S_M(\mathbf{c}) > \delta = \deg_{1,k-1}(Q_M(x,y)) \tag{2.24}$$

Proof:

Let $\mathbf{c} = (c_0, c_1, \cdots, c_{n-1})$ be an RS codeword generated by information polynomial $f(x)$ with the degree $\deg(f(x)) < k$, then $f(x_i) = c_i$. Let us define a univariate polynomial $F(x)$ as $F(x) = Q_M(x, f(x))$, so the degree of this univariate polynomial is

$$\deg(F(x)) = \deg(Q_M(x, f(x))) = \deg_{1,k-1}(Q_M(x,y)) = \delta.$$

Also, we know that

$$S_M(\mathbf{c}) = \langle M, [\mathbf{c}] \rangle = m_1 + m_2 + \cdots + m_n,$$

the summation of all those entries in M corresponding to correct symbols of codeword $\mathbf{c}$. It is proved in [11] that, if a bivariate polynomial $Q_M(x,y)$ passes through a given point (α,β) with multiplicity m, then the univariate polynomial $F(x)=Q_M(x,f(x))$ is divisible by $(x-\alpha)^m$. Since we have $S_M(\mathbf{c})=\langle M,[\mathbf{c}]\rangle=m_1+m_2+\cdots+m_n$, which means at least we need to pass through all those points with the overall multiplicity $m_1+m_2+\cdots+m_n$, so the bivariate polynomial $Q_M(x,y)$ satisfying all the constraints in M should satisfy $F(x)=Q_M(x,f(x))$ is divisible by $(x-x_0)^{m_1}(x-x_1)^{m_1}\cdots(x-x_{n-1})^{m_{n-1}}$, so the degree of $F(x)=Q_M(x,f(x))$ should be larger than $S_M(\mathbf{c})$ or $F(x)=Q_M(x,f(x))=0$. Therefore, if we have a multiplicity matrix M, by which a bivariate polynomial $Q_M(x,y)$ generated satisfies $S_M(\mathbf{c})>\delta=\deg_{1,k-1}(Q_M(x,y))$, then we will have $F(x)=Q_M(x,f(x))=0$, which means the factorization of $Q_M(x,y)$ will lead to a correct factor $y-f(x)$, and $f(x)$ can be used to recover the RS codeword $\mathbf{c}$. ■

Also according to [16], we have $\delta<\sqrt{2(k-1)C}$. Here we can determine that the parameters a,b of the highest monomial $x^a y^b$ in $Q_M(x,y)$ should satisfy

$$a<\delta<\sqrt{2(k-1)C}=\sqrt{(k-1)\sum_{i=0}^{q-1}\sum_{j=0}^{n-1}m_{ij}(m_{ij}+1)} \tag{2.25}$$

and

$$b<\frac{\delta}{k-1}<\frac{\sqrt{2(k-1)C}}{k-1}=\sqrt{\left(\sum_{i=0}^{q-1}\sum_{j=0}^{n-1}m_{ij}(m_{ij}+1)\right)/(k-1)}. \tag{2.26}$$

Soft Interpolation Algorithm

Given a multiplicity matrix M and its one-dimension representation $\{m_t\}_{t=0}^{N-1}$, the interpolation algorithm originally proposed in [17] has been slightly modified as follows:

Soft Interpolation Algorithm:

Inputs: $n, k, \{(x_t, y_t), m_t\}_{t=0}^{N-1}$

Output: $Q_M(x, y) = \Delta(x, y) \in G^{N-1, m_{N-1}}$

Initialize: Compute parameter a, b such that:

1. $a = \sqrt{(k-1)\left(\sum_{t=0}^{N-1} m_t(m_t+1)\right)}$

2. $b = \sqrt{\left(\sum_{t=0}^{N-1} m_t(m_t+1)\right)/(k-1)}$

3. $$\begin{aligned} G^{0,0}(x, y) &= \{G_0, G_1, \cdots G_l, \cdots, G_b\} \\ &= \{1, y^1, \cdots, y^l, \cdots, y^b\} \end{aligned}$$

Soft Interpolation:

```
For all $(x_t, y_t), t = 0, \cdots, N-1$ {
  For $j = 0, 1, \cdots, m_t$ {
```

```
Find the smallest $(1, k-1)$ weighted degree polynomial $\Delta(x, y) \in G^{t,j}$ whose shift polynomial $\Delta(x + x_t, y + y_t) \in G^{t,j}$ has nonzero monomial component of power $j$
    if $\Delta(x, y)$ exists {
        $\begin{pmatrix} G^{t,j}(x, y) \setminus \Delta(x, y) \\ \Delta(x, y) \end{pmatrix} \leftarrow \begin{pmatrix} G^{t,j}(x, y) - \frac{G^{t,j}(x_t, y_t)}{\Delta(x_t, y_t)} \Delta(x, y) \\ (x - x_t)\Delta(x, y) \end{pmatrix}$
    }
  }
}
```

B. Factorization Algorithm

The factorization algorithm can be implemented as follows [18]:

Factorization Algorithm:

Inputs: 1. $Q(x, y), k, i$

Global Array: $\gamma[i], i = 0, \cdots, k-1$

Output: $\gamma[i], i = 0, \cdots, k-1$

Initialize: $\gamma[i] = 0, i = 0, \cdots, k-1,$

$i = 0$.

Factorization:

```
Find the largest integer $v$ which make $Q(x, y)/x^v$ is still a bivariate polynomial
$R(x, y) = Q(x, y)/x^v$
Find all the roots of $R(0, y)$ in $GF(q)$
For each distinct root $\lambda$ of $R(0, y)$
{
$\gamma[i] = \lambda$;
if $i = k-1$, then output $\gamma[i]$ for $i = 0, \cdots, k-1$;
else
    {
        $R^*(x, y) = R(x, xy+\lambda)$;
        Recursively do Factorization Algorithm with parameters $R^*(x, y), k, i+1$.
    }
}
```

***Example 2.2*:** Let us consider an example using the similar parameters as in [16], a RS (5, 2) codeword is generated as $\mathbf{c} = (f(0), f(1), f(2), f(3), f(4)) = (1, 2, 3, 4, 0)$ over $GF(5)$, with information polynomial $f(x) = 1 + x$. Assume the codeword is sent, and the received channel reliability information gives a reliability matrix

$$\Pi = \begin{bmatrix} 0.01 & 0.0025 & 0.05 & 0.14 & 0.20 \\ 0.06 & 0.0025 & 0.09 & 0.14 & 0.05 \\ 0.02 & 0.9900 & 0.15 & 0.07 & 0.20 \\ 0.01 & 0.0012 & 0.61 & 0.44 & 0.40 \\ 0.90 & 0.0038 & 0.10 & 0.21 & 0.15 \end{bmatrix}.$$

The hard-decision of the received vector will be $\mathbf{y} = (4,2,3,3,3)$ which has three symbol errors, beyond the error-correction capability of both traditional hard-decision RS decoding algorithms and the GS algorithm. Here by using the KV algorithm with total multiplicity $s = 4$, we can generate a multiplicity matrix as

$$M_{s=4} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Applying the soft interpolation algorithm, we get

$$\begin{aligned} Q_M(x,y) &= 4x^2 + x - 2y^2 - 2xy - 3 \\ &= (y - x - 1)(-2y - 4x + 3) \bmod 5, \end{aligned}$$

and $\delta = \deg_{1,k-1}(Q_M(x,y)) = 2$, $S_M(\mathbf{c}) = 3 > \delta$, which means that correct decoding can be achieved. The factorization will give us

$$f_1(x) = 1 + x \quad \Rightarrow \quad \mathbf{c} = (1, 2, 3, 4, 0) \quad \Rightarrow \quad \Pr\{(1, 2, 3, 4, 0)\} = 1.521 \times 10^{-3},$$

$$f_2(x) = 4 + 3x \quad \Rightarrow \quad \mathbf{c} = (4, 2, 0, 3, 1) \quad \Rightarrow \quad \Pr\{(4, 2, 0, 3, 1)\} = 9.801 \times 10^{-4}.$$

Obviously $\mathbf{c} = (1, 2, 3, 4, 0)$ will be selected as the output. ■

The multiplicity computation and the choice of total multiplicity s play a very important role in the KV algorithm. It is believed that, as s goes to infinity, the performance of KV algorithm will achieve its best [16], however, the decoding complexity will be very high. Also, when we increase the total multiplicity, the decoding performance does not improve monotonically. Using the same numerical parameters as in Example 2.2, Fig. 2.3 shows the change in the error-correction capability while the total multiplicity increases. The point of correct decoding is $s = 4$, and the second one is $s = 9$, which is the original parameter used in the example in [16]. This relationship of the total multiplicity and correct decoding capability leads to the discussion in the following section as well as to a novel algorithm which will be discussed in Chapter 5.

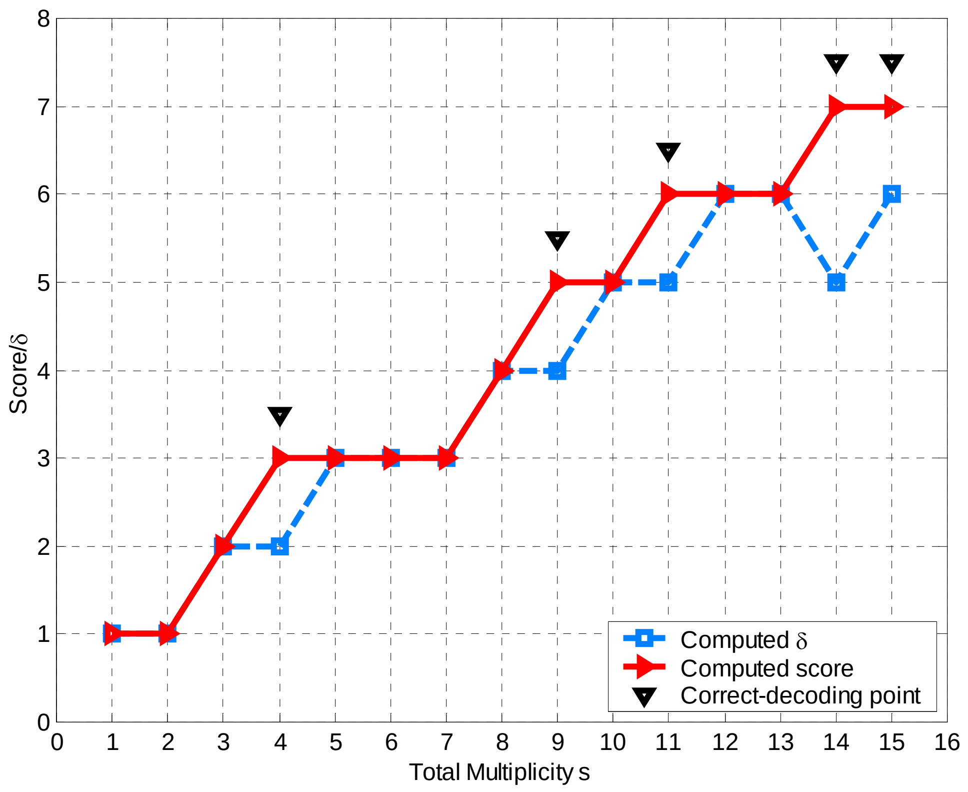


Fig. 2.3. Relationship between total multiplicity and correctability for soft-decision RS decoding given certain reliability matrix Π.

2.3.3 Suboptimal Multiplicity Computation Algorithm

There are still several other algorithms for computing the multiplicity matrix such as in [19], [20]. Here a novel suboptimal multiplicity computing algorithm is also given, which will be very useful for the following *re-encoding algorithm* proposed in [21]. The suboptimal algorithm leads to constant or partially constant entries in the multiplicity matrix. Since the original *Algorithm A* in [16] is optimal, the issue of quantifying the decoding losses when multiplicities are forced to be constant or partially constant arises.

Our original motivation for this suboptimal algorithm was that in the multiplicity matrix M there are only n correct points, and the other $nq-n$ points are incorrect. When we increase the total multiplicity s beyond a certain value, most of the time we will introduce incorrect points. Fig. 2.3 illustrates the point that increasing the total multiplicity does not guarantee correct decoding. The idea behind our suboptimal algorithm is to increase the score $S_M(\mathbf{c})$ without increasing δ, or if that is not possible, to increase $S_M(\mathbf{c})$ more than δ, to guarantee successful decoding. This suboptimal algorithm consists of two parts: 1) First we use a fairly small value for s (but large enough to insure that M contains at least k correct points; this value can be determined by simulation at a given signal-to-noise ratio (SNR)) to compute the multiplicity matrix using algorithms in [16] or [19]; 2) Method 1: Set a threshold θ, and for those entries whose values are larger than this threshold, change them to a constant value τ; or Method 2: Assign a constant value τ to all nonzero entries in the multiplicity matrix. This way, if successful, we increase the weight of correct points, rather than increasing the number of incorrect points, which in turn increases $S_M(\mathbf{c})$ more than δ. The detailed steps of this algorithm, which we call *Algorithm A^+* are given below.

In Fig. 2.4, a comparison of different multiplicity computation algorithms is given for a particular code. It can be seen that the performance of the soft-decision decoding algorithm using Method 1 is the same as the original KV algorithm, while Method 2 is slightly better at higher SNR on an AWGN channel. Gross's algorithm [19] is also used as a reference. The better performance of Method 2 reinforces our expectation that at higher SNR, by forcing the multiplicities to have a constant value, most of the time we increase the score more than δ, which leads to better performance. Note, however, that the decoding complexity is slightly higher in terms of the average total multiplicity as shown in Table 2.1, and therefore these results do not violate the fact that *Algorithm A* is optimal for a given multiplicity. Method 1 achieves the same performance as *Algorithm A*, but with a slightly lower complexity, as measured by the average total multiplicity, which tells us that at higher SNR, even if we reduce the number of interpolation iterations for the correct points, by assigning them a constant multiplicity τ, it does not change the decoding performance appreciably. Furthermore, the additional advantage of using this suboptimal algorithm is that since the multiplicity is constant (Method 2) or partly constant (Method 1), an efficient divide-and-conquer algorithm [22] can be used for the interpolation. Also, when using *Algorithm A*, the interpolation step can only be started after all the multiplicity values have been computed, while for the suboptimal algorithm proposed here, since the final multiplicity value τ for part or all of the interpolation points is known in advance, we can carry out the interpolation step in parallel while computing the multiplicity matrix. Finally, by supplying constant or partially constant multiplicities, the suboptimal algorithm can be used with the re-encoding algorithm described in the next Section.

Algorithm A$^+$: Calculating M^+ from the reliability matrix $\prod$.

Input: Total multiplicity *s*, reliability matrix $\prod$, threshold θ, parameter τ

Speed up step:

```
for $i = 1$ to $n$ do
 1) For each column of the reliability matrix received, find the highest probability entry $j$.
 2) Let $m_{i,j} = \tau$ in $M^+$, output $\{(x_i, y_i), \tau\}$ directly to interpolation step.
end for
```

Compensation step:

(Method 1) Using *Algorithm A* in [15], and total multiplicity *s* compute nonzero entries in M^+ other than those points which have already been computed, output to interpolation step.

(Method 2) Using *Algorithm A* in [15], and total multiplicity *s* compute nonzero entries in M^+ other than those points which have already been computed, also set those nonzero values $m_{i,j} = \tau$, output to interpolation step.

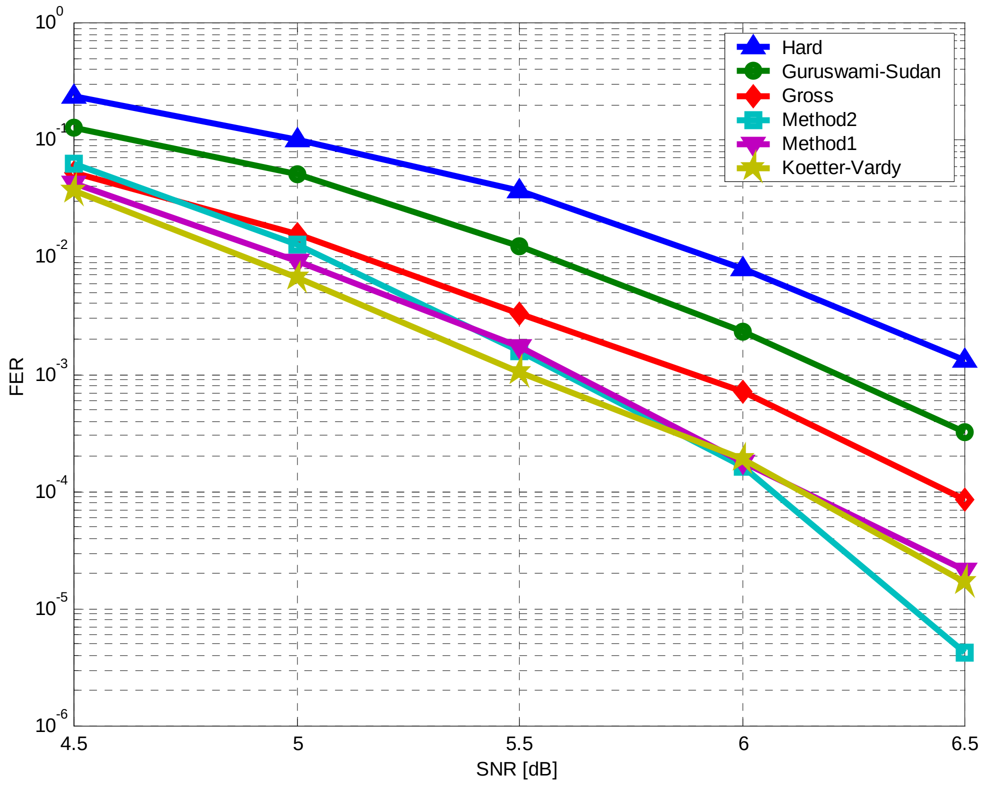


Fig. 2.4. Performance comparison of RS code (31,15) on the AWGN channel with different multiplicity computation algorithms.

TABLE 2.1
PARAMETERS FOR FIG. 2.4 AT SNR=6.5 dB ON AN AWGN CHANNEL

	KV	GROSS	METHOD 1	METHOD 2
PARAMETERS FOR MULTIPLICITY COMPUTATON	$s = 6 \times n$	$\lambda = 8$	$s = 6 \times n$ $\tau = 5$ $\theta = 3$	$s = 6 \times n$ $\tau = 5$
AVERAGE TOTAL MULTIPLCITY	186	215	165	202

2.3.4 Re-Encoding Algorithm

Although the performance improvement of interpolation-based algorithms, such as the GS and KV algorithms, is very large compared to traditional hard-decision RS decoding algorithms, the decoding complexity has been increased simultaneously. Noticing that another interpolation-based RS decoding algorithm, the WB algorithm, has two steps, namely a re-encoding step, and a polynomial interpolation step, Gross *et. al* proposed a re-encoding algorithm in [21], [23] which significantly reduces the decoding complexity of the KV algorithm. The decoding complexity reduction comes from the fact that when a re-encoding step is complete, it will generate at least k point pairs $\{x_{i,}\ y_i\}$ with $y_i = 0$, then the polynomial interpolation of these k points can be implemented with lower complexity.

By choosing the k points with the largest multiplicity from the multiplicity matrix (or equivalently the k most reliable entries in the received vector $\mathbf{r}$), then systematically re-encoding these k points and subtracting the resulting codeword $\mathbf{r'}$ from the received vector $\mathbf{r}$, we can generate a modified vector $\mathbf{y}$, which has at least k zero entries for polynomial interpolation. Suppose we initialize the set of polynomials with $G(x,y) = \{G_0(x,y),\cdots,G_l(x,y),\cdots,G_{b-1}(x,y)\} = \{1,\cdots,y^j,\cdots,y^{b-1}\}$, the set of polynomials passing through those k points $\{x_i,0\}$ becomes:

$$
\begin{aligned}
G'(x,y)=\{&(x-x_i)^{m_i}(x-x_{i+1})^{m_{i+1}}\cdots(x-x_{i+k-1})^{m_{i+k-1}},\\
&(x-x_i)^{m_i-1}(x-x_{i+1})^{m_{i+1}-1}\cdots(x-x_{i+k-1})^{m_{i+k-1}-1}y,\\
&\cdots,\\
&(x-x_i)^{m_i-b+1}(x-x_{i+1})^{m_{i+1}-b+1}\cdots(x-x_{i+k-1})^{m_{i+k-1}-b+1}y^{b-1}\}.
\end{aligned}
$$

The k points will not necessarily be contiguous, but here for notation convenience, we assume them contiguous. Let $v(x)=(x-x_i)(x-x_{i+1})\cdots(x-x_{i+k-1})$ and $V(x)=(x-x_i)^{m_i}(x-x_{i+1})^{m_{i+1}}\cdots(x-x_{i+k-1})^{m_{i+k-1}}$, the bivariate polynomial which passes through N points with different multiplicities can be expressed as:

$$
\begin{aligned}
Q_M(x,y)&=\sum_{j=0}^{b-1}w_j(x)(x-x_i)^{m_i-j}(x-x_{i+1})^{m_{i+1}-j}\cdots(x-x_{i+k-1})^{m_{i+K-1}-j}y^j\\
&=V(x)\sum_{j=0}^{b-1}w_j(x)(y/v(x))^j,
\end{aligned}
\tag{2.27}
$$

where $w_j(x)$ is a univariate polynomial generated in the interpolation step. The polynomial $V(x)$ can be factored out, calculated in advance, and the interpolation step in the GS and KV algorithms is simplified to finding a bivariate polynomial

$$
Q_M(x,\overline{y})=\sum_{j=0}^{b-1}w_j(x)\overline{y}^j \tag{2.28}
$$

with $\overline{y}=y/v(x)$. Correspondingly, the modified vector $\mathbf{y}$ needs to be transformed into $\overline{y}_i=y_i/v(x)$, and the number of interpolation points is reduced to $N-k$ points. Notice that $deg^{(1,k-1)}(\overline{y})=deg^{(1,k-1)}(y)-deg^{(1,k-1)}(v(x))=k-1-k=-1$. It needs to be mentioned here that the value of b in (2.27) must be less or equal to m', where m' is the k-th largest value among the multiplicities. Otherwise, we will get a negative power for the univariate polynomial $x - x_i$. In practice, b can be chosen to be larger than m', but the complexity will be slightly higher, since some points among those k points $\{x_i, 0\}$ whose multiplicity is less than b cannot be the factors in $V(x)$, and regular interpolation for them is required. This will not be a problem if the k points in $V(x)$ have equal multiplicity, which can be achieved by carefully choosing the threshold θ in Method 1.

Furthermore, in [23], a reduced-complexity factorization scheme is proposed. The goal of a factorization algorithm is to find the factors in terms of $y - f(x)$. Let us assume that given a $Q_M(x, y)$ as in (2.27), then this bivariate polynomial should have a factor such as $y - f(x)$ as long as it can be correctly decoded, that is,

$$\begin{aligned} Q_M(x,y) &= V(x)\sum_{j=0}^{b-1} w_j(x)\left(\frac{y}{v(x)}\right)^j \\ &= (y - f(x))D(x,y) \\ &= \frac{(y - f(x))}{v(x)} D(x,y)v(x) \\ &= \left(\frac{y}{v(x)} - \frac{f(x)}{v(x)}\right) D(x,y)v(x). \end{aligned} \tag{2.29}$$

So if we replace $\overline{y} = y / v(x)$ and (2.28) into (2.29), we have

$$\begin{aligned} Q_M\left(x,\overline{y}\right) &= \sum_{j=0}^{b-1} w_j(x)\overline{y}^j \\ &= \left(\overline{y} - \frac{f(x)}{v(x)}\right)\frac{D(x,y)v(x)}{V(x)}. \end{aligned} \tag{2.30}$$

According to (2.30), if we can find the factor $\bar{y} - f(x)/v(x)$ for a bivariate polynomial which passes through only $N-k$ points from a given multiplicity matrix, then we can finally have $f(x)$ to recover the re-encoded codeword $\mathbf{c}-\mathbf{r'}$. Furthermore, since $f(x)$ will zero out all the positions that are not in error for codeword $\mathbf{c}-\mathbf{r'}$, therefore

$$f(x) = \Omega(x) \prod_{i \in R \setminus E} (x - x_i) \tag{2.31}$$

where R represents the set of indices that x_i corresponds to k positions in $\mathbf{r}$ which have been selected to generate the re-encoding codeword $\mathbf{r'}$, and E represents the set of indices of the errors in those k positions in $\mathbf{r}$. Also we can write $v(x)$ as

$$v(x) = \prod_{i \in E} (x - x_i) \prod_{i \in R \setminus E} (x - x_i) \tag{2.32}$$

So now we can have

$$\frac{f(x)}{v(x)} = \frac{\Omega(x) \prod_{i \in R \setminus E} (x - x_i)}{\prod_{i \in E} (x - x_i) \prod_{i \in R \setminus E} (x - x_i)} = \frac{\Omega(x)}{\prod_{i \in E} (x - x_i)} \tag{2.33}$$

where $\Omega(x)$ and $\Lambda(x) = \prod_{i \in E} (x - x_i)$ can be taken as the error evaluation polynomial and error location polynomial in traditional hard-decision RS decoding algorithm such as BM algorithm, respectively. Therefore, if we can compute $\Omega(x)$, $\Lambda(x)$ and know the error locations, then we can use $f(x_i) + e_i = 0$ for those $i \in E$ to recover codeword $\mathbf{c}-\mathbf{r'}$, and codeword $\mathbf{c}$ finally.

2.4 Summary

In Chapter 2, we discussed the hard-decision RS decoding algorithms and soft-decision algorithms such as KV algorithm. The KV algorithm includes three major steps, that is, multiplicity computation step, soft interpolation step and factorization step. Also, a re-encoding algorithm aimed to reduce the complexity of soft interpolation step was discussed. Moreover, we proposed a suboptimal multiplicity computation algorithm which leads to an efficient implementation of re-encoding algorithm.

Bibliography

[1] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *J. Soc. Indust. Applied Mathematics*, vol. 8, pp. 300-304, 1960.

[2] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes.* Amsterdam, The Netherlands: North-Holland, 1977.

[3] W. J. Gross, F. R. Kschischang, R. Koetter and P. G. Gulak, "Towards a VLSI architecture for interpolation-based soft-decision Reed-Solomon decoders," submitted to *J. VLSI Signal Processing*, 2003.

[4] S. B. Wicker, *Error Control Systems for Digital Communication and Storage*. Upper Saddle River, NJ: Prentice Hall, 1995.

[5] R. E. Blahut, *Theory and Practice of Error Control Codes*. Reading, MA: Addison-Wesley, 1983.

[6] R. T. Chien, “Cyclic decoding procedures for Bose-Chaudhuri-Hocquenghem codes,” IEEE Trans. Inform. Theory, vol. 10, pp. 357-363, Oct. 1964.

[7] G. D. Forney, “On decoding BCH codes,” *IEEE Trans. Inform. Theory*, vol. 11, pp. 549-557, Oct. 1965.

[8] E. R. Berlekamp, *Algebraic Coding Theory*. Laguna Hills, CA: Aegean Park, 1984.

[9] J. L. Massey, “Shift-register synthesis and BCH decoding,” *IEEE Trans. Inform. Theory*, vol. 15, pp. 122-127, Jan. 1969.

[10] E. Berlekamp, “Bounded distance+1 soft-decision Reed-Solomon decoding,” *IEEE Trans. Inform. Theory*, vol. 42, pp. 704-720, May 1996.

[11] V. Guruswami and M. Sudan. “Improved decoding of Reed-Solomon and algebraic-geometric codes,” *IEEE Trans. Inform. Theory*, vol. 45, pp.1757-1767, Sept. 1999.

[12] M. Kuijper, “Algorithms for decoding and interpolation,” in *Codes, Systems, and Graphical Models,* B. Marcus and J. Rosenthal, Eds., Berlin, Germany: Springer-Verlag, 2001, pp. 265-282.

[13] G. D. Forney, Jr., “Generalized minimum distance decoding,” *IEEE Trans. Inform. Theory*, vol. 12, pp. 125-131, 1966.

[14] D. Chase, “A class of algorithms for decoding block codes with channel measurement information,” *IEEE Trans. Inform. Theory*, vol. 18, pp.170-182, Jan. 1972.

[15] T.-H. Hu and S. Lin, “An efficient hybrid decoding algorithm for Reed-Solomon codes based on bit reliability,” *IEEE Trans. Commun*. vol. 51, pp. 1073-1081, July, 2003.

[16] R. Koetter and A. Vardy, “Algebraic soft-decision decoding of Reed-Solomon codes”, *IEEE Trans. Inform. Theory*, vol. 49, pp. 2809-2825, Nov. 2003.

[17] R. R. Nielsen and T. Hoholdt, “Decoding Reed-Solomon codes beyond half the minimum distance”, in *Coding Theory, Cryptography and Related Areas*, J. Buchmann, T. Hoholdt, T. Stichtenoth, and H. Tapia-Recillas, Eds., Berlin, Germany: Springer-Verlag, 2000, pp. 221-236.

[18] R. M. Roth and G. Ruckenstein, “Efficient decoding of Reed-Solomon codes beyond half the minimum distance”, *IEEE Trans. Inform. Theory*, vol. 46, pp. 246-258, Jan. 2000.

[19] W. J. Gross, F. R. Kschischang, R. Koetter and P. G. Gulak, “Simulation results for algebraic soft-decision decoding of Reed-Solomon codes,” in *Proc. 21st Biennial Symp. Commun.*, pp. 356-360, 2002.

[20] F. Parvaresh, and A. Vardy, “Mulitplicity assignments for algebraic soft-decoding of Reed-Solomon codes,” in *Proc. IEEE Inter. Symp. Inform. Theory*, pp. 205, 2003.

[21] W. J. Gross, F. R. Kschischang, R. Koetter and P. G. Gulak, “A VLSI architecture for interpolation in soft-decision list decoding of Reed-Solomon codes,” in *Proc. IEEE Workshop Signal Processing Syst.*, pp. 39-44, 2002.

[22] G.-L. Feng and X. Giraud, “Fast algorithms in Sudan decoding procedure for Reed-Solomon codes,” submitted to *IEEE Trans. Inform. Theory*, 2001.

[23] A. Ahmed, R. Koetter and N. R. Shanbhag, “VLSI architecture for soft-decision decoding of Reed-Solomon codes,” submitted to *IEEE Trans. VLSI Systems,* Feb. 2003.

Chapter 3

Applications of Soft-Decision Reed-Solomon Decoding Algorithms to Magnetic Recording Systems

3.1 Introduction

A good ECC for magnetic recording systems must have high code rate, the decoding algorithm must have a reasonable decoding computational complexity, and the decoder must be realizable. In current magnetic recording systems, a hard-decision Viterbi decoding algorithm is used for channel detection, and a hard-decision RS decoder is used as the ECC decoder to correct the errors after the Viterbi decoder. RS codes are maximum distance separable (MDS) codes, and have a low error probability at high signal-to-noise ratios (SNRs). However, the ECC decoder using a hard-decision RS decoding algorithm cannot utilize the "soft" information of the channel detector output, and cannot share information with the channel detector which limits performance. Recently, the invention of turbo codes [1] and the re-discovery of low-density parity-check (LDPC) codes [2] has sparked a great deal of interest in the magnetic recording industry in replacing the RS code with a soft-decision decodable code such as a Turbo code, or an LDPC code for improved performance. Theoretical research has shown that LDPC codes are capable of approaching the Shannon limit. However, the error floor observed for turbo codes and the uncertainty of LDPC code performance at high SNR, coupled with large performance degradation of LDPC codes in the presence of error bursts indicate that there is still a lot of research that needs to be done before turbo codes or LDPC codes can replace RS codes in magnetic recording systems. Also, the high decoding complexity of soft-decision decoding is another obstacle. In [3], Koetter and Vardy proposed a soft-decision RS decoding algorithm, which provides another alternative for future magnetic recording systems.

3.2 Soft Chase-Type Algorithms

The KV algorithm shows a large coding gain for low-rate RS codes over AWGN channels [3]. However, the coding gain decreases as code rate goes up. In magnetic recording systems, high-rate RS codes must be used, therefore, how to improve the decoding gain becomes a major open problem for soft-decision RS decoding of high rate RS codes. Inspired by the Chase algorithm [4], we combine the KV algorithm with the traditional Chase algorithm and propose a soft version of the Chase algorithm (this algorithm has been presented in part in our work [5], [6]).

Given that for a received channel output $\mathbf{c}$, normally only a few bits in a symbol are at risk, due to noise contamination, we search for the p least reliable bits in a received sequence, generate a set of test patterns, and map the zeros to some large positive value for the log-likelihood ratio (LLR), and the ones to some negative value. Then several multiplicity matrices can be generated and soft interpolation and factorization will provide additional candidates for list decoding, which in turn leads to improved decoding performance. Notice that, in those multiplicity matrices, only a few columns are different (especially when we use Method 2 (Section 2.3.3) to compute the multiplicity matrix), and some complexity reduction can be realized.

We divided the sequence of points $\{(x_t, r_t), m_t\}_{t=0}^{N-1}$ for polynomial interpolation into two groups: Group 1 includes $N-k'$ points $\{(x_t, r_t), m_t\}$, whose symbols do not contain any bit involved in bit-flipping; Group 2 includes k' points $\{(x_t, r_t), m_t\}$, whose symbols contain bits involved in bit-flipping. Then an intermediate bivariate polynomial $Q'_M(x,r)$ is generated, which passes through those $N-k'$ points in Group 1, and is stored for future use. For each matrix we let $Q'_M(x,r)$ pass through the appropriate points in Group 2 corresponding to the different test patterns to get $Q_M(x,r)$. Since the value of k' is usually very small, the complexity of the generation of the bivariate polynomial $Q_M(x,r)$ does not increase very much, while the performance improvement can be very significant. So it is interesting that the original idea of combining the KV algorithm with the Chase algorithm is to improve the decoding performance, but the decoding complexity does not increase substantially due to the nice interpolation property of the KV algorithm.

3.2.1 Soft Chase Algorithm

The detailed soft Chase algorithm is described as follows: Suppose an RS codeword $\mathbf{c}$ is generated and sent through the channel, with

$$\mathrm{c} = \left(c_0^{(0)}, c_0^{(1)}, \cdots, c_0^{(u-1)}, \cdots, c_{n-1}^{(0)}, \cdots, c_{n-1}^{(u-1)}\right),\ u = \log_2 q .$$

Step 1: Obtain the channel output reliability for each bit:

$\mathbf{p} = (\mathrm{p}_0, \mathrm{p}_1, \cdots, \mathrm{p}_{\mathrm{n}-1}) = \left(p_0^{(0)}, p_0^{(1)}, \cdots, p_0^{(u-1)}, \cdots, p_{n-1}^{(u-1)}\right)$, which can be converted to a hard-decision vector $\mathbf{r}$, where $p_i^{(j)} = LLR\left(c_i^{(j)}\right) = \log \dfrac{\Pr\left(c_i^{(j)} = 1 \mid \mathrm{r}\right)}{\Pr\left(c_i^{(j)} = -1 \mid \mathrm{r}\right)}$;

Step 2: Find the p least reliable bits by searching the output sequence $\mathbf{p}$, and generate the test patterns corresponding to these p bits;

Step 3: For each test pattern, compute the reliability and multiplicity matrices, generate a sequence of points $\{(x_t, r_t), m_t\}_{t=0}^{N-1}$, and divide them into two groups according to whether the reliability matrix entries are involved in bit-flipping or not;

Step 4: Generate the intermediate polynomial $Q'_M(x,r)$, which passes through points in the first group, and store $Q'_M(x,r)$;

Step 5: Finish the polynomial interpolation step by making $Q'_M(x,r)$ pass through those points in the second group; complete the soft-decision decoding. If it fails, go to Step 5 else go to End;

Step 6: For the next test pattern, re-compute those points and multiplicities in the second group, read $Q'_M(x,r)$ from memory, then go to Step 5;

End;

The stopping criterion can be implemented using (2.24) to check the decoding output codeword, since its score and the weighted degree of $Q_M(x,y)$ can be obtained after decoding. Other stopping criteria, such as distance computation or CRC, can also be used as a supplement.

3.2.2 Re-Encoded Chase Algorithm

Although the soft Chase algorithm described above provides a large decoding gain without substantially increasing the decoding complexity compared to the original KV algorithm, the decoding complexity of the original KV algorithm is still prohibitively high. In order to reduce the complexity while maintaining the decoding gain, we further combined our soft Chase algorithm with the re-encoding algorithm [7] to produce a new algorithm, the re-encoded Chase algorithm [8], which not only improves the decoding performance but also reduces the decoding complexity compared to the KV algorithm. This makes the practical implementation of soft-decision RS decoding for future magnetic recording more attractive.

The concept stems from the following observations:

1) In the soft Chase algorithm, we divide the sequence of polynomial interpolation points $\{(x_t, r_t), m_t\}_{t=0}^{N-1}$ into two groups: one includes $N-k'$ points $\{(x_t, r_t), m_t\}$, whose symbols do not contain any bit involved in bit-flipping; another includes k' points $\{(x_t, r_t), m_t\}$, whose symbols contain bits involved in bit-flipping (k' is related to p).

2) In the re-encoding algorithm, we also divide the sequence of points $\{(x_t, r_t), m_t\}_{t=0}^{N-1}$ into two groups: one includes k points $\{(x_t, y_t), m_t\}$, with the largest multiplicity m_t, which will be used to generate the modified set $\{(x_t, y'_t = 0), m_t\}$; another includes the remaining $N-k$ points $\{(x_t, y_t), m_t\}$.

It is interesting that the points in the first group of the re-encoding algorithm can be made to belong to the first group of interpolation points for the soft Chase algorithm, if the parameter k' is properly chosen. So for the soft Chase algorithm, we first generate a set of multiplicity matrices in accordance to the parameters s and k', then the multiplicities are divided into two groups: one consisting of those multiplicities that are common to all matrices, another consisting of the multiplicities that are generated by each particular test pattern. The first group can be further subdivided into two groups: the k entries with the largest value will be labeled as Group 1, the rest as Group 2. The pattern dependent multiplicities will be labeled as Group 3, which consists of 2^p subsets. With this grouping of the interpolation points, it is easy to see that we can calculate the bivariate polynomial passing through all the points in Groups 1 and 2 only once, which significantly reduces the soft Chase algorithm complexity without loss of performance. Since k' is usually very small, the complexity of finishing the interpolation step, by passing through each subset of points in Group 3, 2^p times, is not much larger than implementing the soft interpolation step just once in the KV algorithm. When combined with the re-encoding algorithm, the decoding complexity can be further reduced. The additional cost incurred by the soft Chase algorithm is that we need to perform 2^p factorizations. A factorization algorithm given in [9, p. 32] which utilizes the conventional hard-decision RS decoder to help perform the factorizations can be used here to reduce the factorization complexity. The implementation of 2^p partial interpolation and factorization steps can be realized in parallel, so the decoding can be achieved with a small delay (Fig. 3. 1(a)). We can also trade off complexity for decoding delay by using only a single hardware/software core which implements the partial interpolation and factorization steps sequentially (Fig. 3.1(b)). The significant performance improvement makes this combined algorithm attractive. A summary of the combined algorithm is as follows:

Initialize: Channel output probabilities for each bit of received codeword.

Step 1: Find the p least reliable bits by searching the output probability sequence, and generate a set of reliability and multiplicity matrices corresponding to the test patterns;

Step 2: For each multiplicity matrix, generate a sequence of points $\{(x_t, y_t), m_t\}_{t=0}^{N-1}$, and assign them to their respective groups;

Step 3: Generate the intermediate polynomial $Q^{(1)}{}_M(x, y)$, which passes through points in Group 2, and store $Q^{(1)}{}_M(x, y)$;

Step 4: Finish the polynomial interpolation step by making $Q^{(1)}{}_M(x, y)$ pass through points in Group 3 to get $Q^{(2)}{}_M(x, y)$; finish soft-decision decoding using the re-encoding algorithm. If it fails, go to Step 5; else go to End;

Step 5: For the next test pattern, read $Q^{(1)}{}_M(x, y)$ from memory, then go to Step 4;

End.

It should be mention here, that soft-decision decoding failure can be determined by checking if the decoded codeword satisfies (2.24).

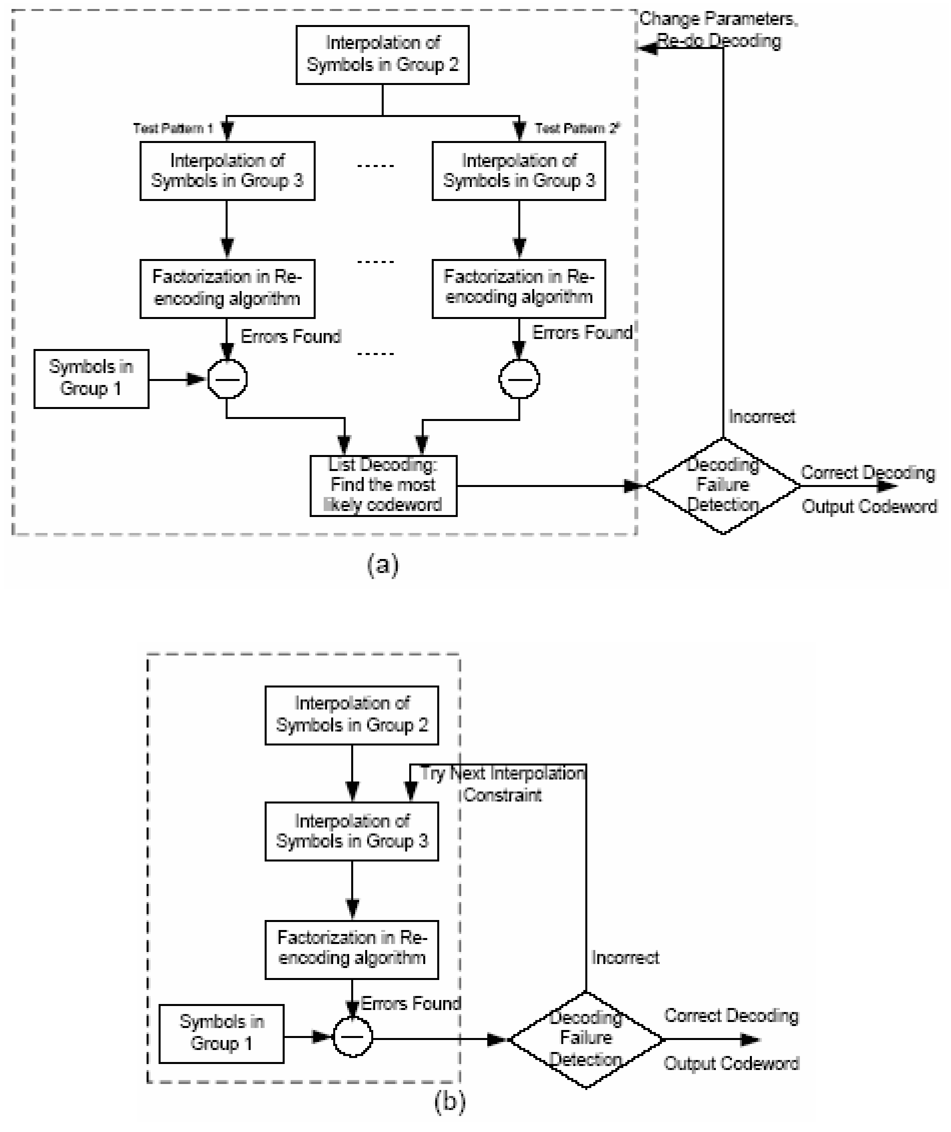

Interpolation of Symbols in Group 2
Test Pattern 1
Test Pattern 2
Interpolation of Symbols in Group 3
Interpolation of Symbols in Group 3
Factorization in Re-encoding algorithm
Factorization in Re-encoding algorithm
Errors Found
Errors Found
Symbols in Group 1
List Decoding: Find the most likely codeword
Change Parameters, Re-do Decoding
Incorrect
Decoding Failure Detection
Correct Decoding
Output Codeword
(a)
Interpolation of Symbols in Group 2
Try Next Interpolation Constraint
Interpolation of Symbols in Group 3
Factorization in Re-encoding algorithm
Errors Found
Symbols in Group 1
Incorrect
Decoding Failure Detection
Correct Decoding
Output Codeword
(b)

Fig. 3.1 Block diagram of two different types of combined soft Chase decoders. (a) Parallel; (b) Recursive.

3.3 Retry Mode Scheme

Current RS decoding systems use a Viterbi algorithm (VA) as the channel output decoder and a hard-decision RS decoding algorithm. The KV algorithm provides an error-correction capability larger than $n-\sqrt{n(k-1)}$, and the decoding gain can significantly exceed this bound when the code rate is low. Although the KV algorithm and its variations bring a performance improvement as well as complexity reduction, the complexity is still larger than hard-decision decoding, especially when the code length n is large. Fig. 3.2 shows that most of the errors that happen in one frame are less than half the minimum distance, so it is not necessary to use the more complicated soft-decision algorithm every time. A hardware or software implemented retry mode system shown in Fig. 3.3 should be much more attractive than completely changing the current decoding system. In such a system, the soft RS decoder is invoked only if the hard RS decoder fails (flagged by a particular checking scheme, such as CRC check in a magnetic recording system), and the gain of the soft decoder can be adjusted by changing the total multiplicity s. In a real implementation of this retry mode system, a hard-decision decoder and retry mode soft-decision decoder can be executed at the same time, if the output of a hard-decision decoder is correct, the operation of the soft-decision decoder can be halted, otherwise continue to finish the soft-decision decoding. Also the interpolation and factorization steps of the soft-decision algorithm can be executed in parallel, for each iteration in the *interpolation step*, and an intermediate bivariate polynomial will be output for factorization. If the correct codeword is found, the decoding process is stopped, which helps to reduce the decoding delay. This system has better performance and lower computational complexity than an on-the-fly soft-decision decoder, which makes it well suited for the high-throughput requirements of current magnetic recording systems.

3.3.1 Retry Mode Implementation

In order to apply the soft-decision decoding algorithm to partial response channels, commonly found in magnetic recording systems, without greatly increasing system complexity, the following retry scheme is proposed:

Input: Maximal multiplicity

Step 1: Receive channel reliability information and convert it to a non-binary sequence:

$$\mathbf{y} = (y_1, y_2, \cdots, y_n), \quad y_i \in GF(q)$$

Store the reliability information in a $q \times n$ matrix $\Pi[i, j]$

Step 2:Hard Decision Step. Input the non-binary sequence $\mathbf{y}$ into the hard-decision decoding algorithm.

Step 3: If the hard decision is wrong (indicated by a CRC check), then switch to the soft decoding algorithm; else End.

Step 4: Soft Decision Step. Using the reliability information matrix $\Pi[i, j]$, and the soft-decision algorithm described in the last section, do soft-decision decoding.

Step 5: Erasure Decision Step (optional). Erase unreasonable symbols in sequence $\mathbf{y}$, go to Step 4.

Step 6: If the soft decision is wrong, increase total multiplicity, then go to Step 4, else End.

Step 7: Continue until the total multiplicity reaches the maximal multiplicity, End.

3.3.2 Discussion on Decoding Complexity

The decoding complexity is not easy to calculate, however we can shed some light on how it compares to hard-decision RS decoding. The complexity of RS hard-decision decoding is very nearly $O(n \log n)$ [10, p. 336] with the error-correction capability no more than $(n-k)/2$. The complexity of the soft-decision decoding algorithm is much higher. Given in [11], the complexity of interpolation-based RS decoding algorithm is $O(\delta^6 / k^3)$, and since

$$\min(\delta) < \sqrt{2(k-1)C_M} = \sqrt{(k-1)\sum_{i=0}^{q-1}\sum_{j=0}^{n-1} m_{ij}\left(m_{ij}+1\right)} \quad , \tag{3.1}$$

the complexity of KV algorithm is approximately

$$Complexity(KV) = O\left(\frac{\delta^6}{k^3}\right) < O\left(\left(\sqrt{(k-1)\sum_{i=0}^{q-1}\sum_{j=0}^{n-1} m_{ij}\left(m_{ij}+1\right)/k^3}\right)^6\right)$$
$$\approx O\left(\left(\sum_{i=0}^{q-1}\sum_{j=0}^{n-1} m_{ij}\left(m_{ij}+1\right)\right)^3\right), \tag{3.2}$$

where m_{ij} is a nonzero entry in the multiplicity matrix. According to (3.2) we can see that the complexity of the soft-decision RS decoding algorithm is proportional to $O(n^3)$. Considering the re-encoding algorithm, (3.2) can be simplified to:

$$Complexity\,(RE) \approx O\left(\left(\sum_{i=0}^{q-1}\sum_{j=0}^{n-1} m_{ij}\left(m_{ij}+1\right) - \sum_{t\in L} m_t\left(m_t+1\right)\right)^3\right) \quad , \tag{3.3}$$

where L presents the set of the k largest entries in the multiplicity matrix M , so the complexity of the re-encoding algorithm is approximately $O\left((n-k)^3\right)$. Since as illustrated by Fig. 3.2, most of the errors occurring in one block will be less than half the minimum distance, which can be correctly decoded by the hard-decision decoding algorithm. So the complexity of the proposed retry mode scheme is only slightly larger than $O(n\,log\,n)$.

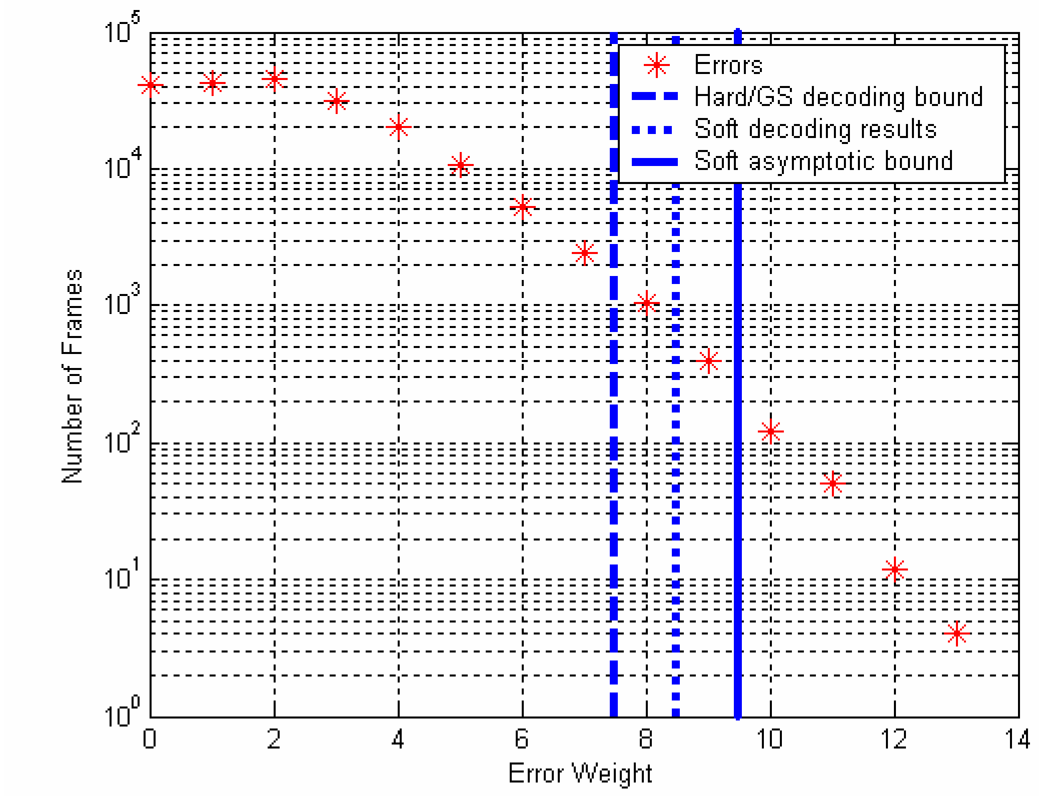


Fig. 3. 2. Error distribution of RS (143,129) on MEEPR4 at SNR=14.5dB.

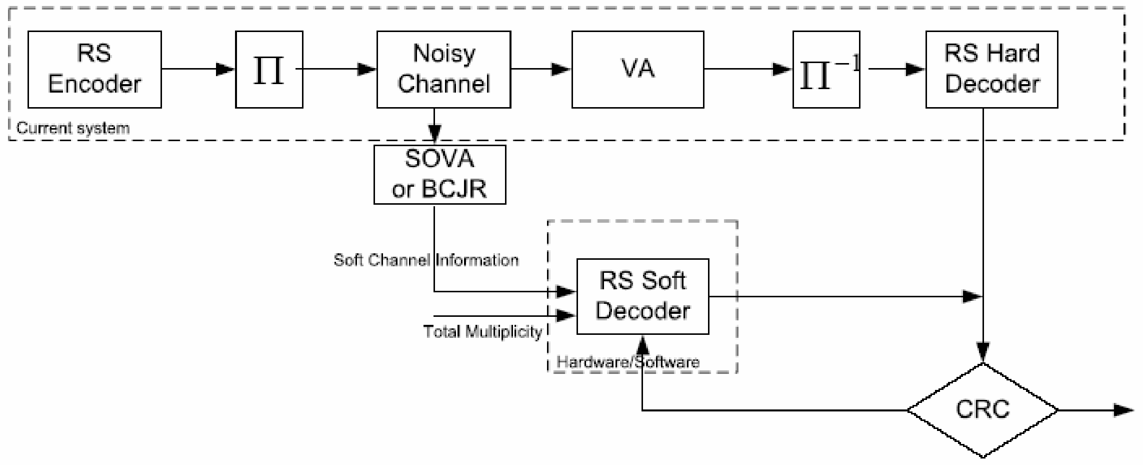


Fig. 3. 3. Proposed system with soft decoding retry scheme.

3.4 Soft-Decision RS Decoding over PR Channels

Here we investigate the performance of the soft-decision RS decoding algorithm on magnetic recording channels equalized to EPR4 [12], and MEEPR4 channels [13]. The BCJR algorithm [14] is used for channel detection to supply soft information to the ECC decoders.

Firstly, the soft-decision RS decoding algorithm has been tested for different code rates over PR channels, which shows a decrease in coding gain when the code rate increases. According to Fig. 3.4, the soft-decision decoding algorithm works better for low-rate RS codes. For RS (255, 144) code with rate R=0.56, a 0.4-dB decoding gain over hard-decision decoding can be obtained with total multiplicity s=635, however, when the code rate increases to 0.9, the decoding gain is only 0.1-dB left for an RS (160, 143) code over EPR4 channel with channel density S_c=2.887.

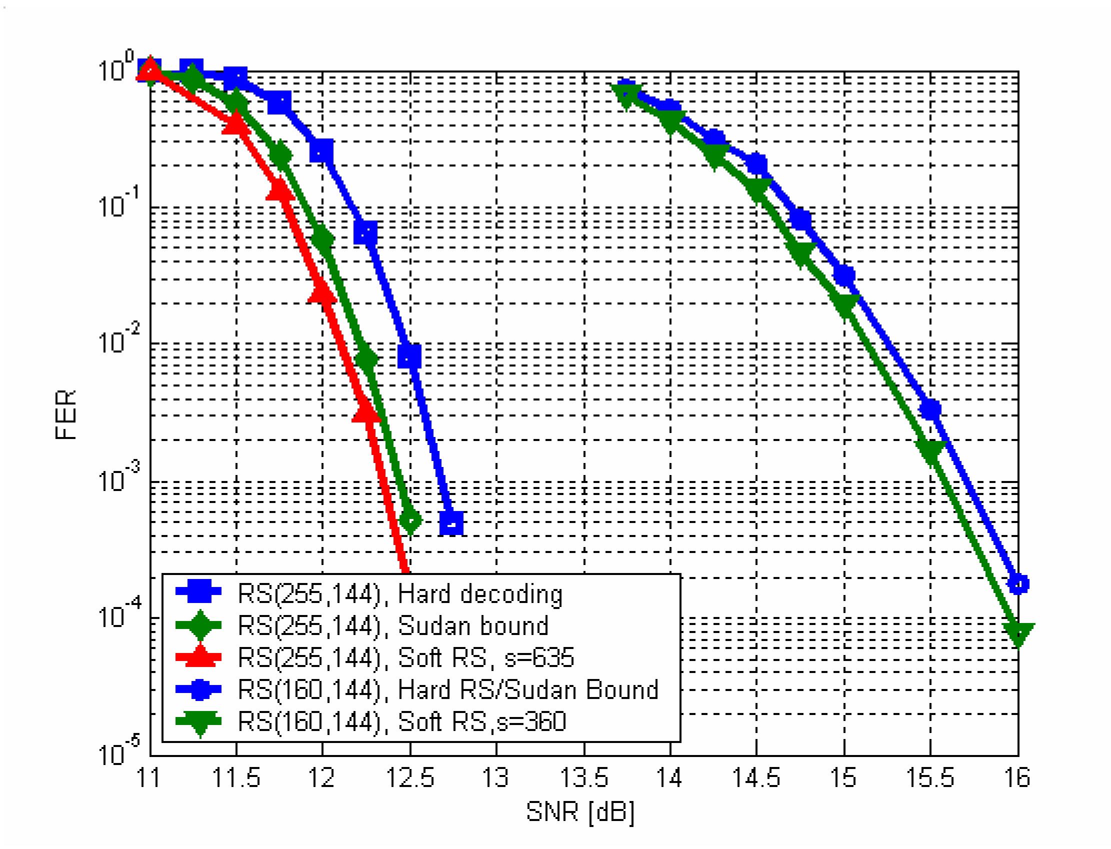


Fig. 3.4. Performance of two RS codes with different code rates using soft-decision RS decoding algorithm, EPR4 channel, no interleaving, channel density 2.887.

Secondly, the performance is evaluated for different values of the total multiplicity s. It is shown in Fig. 3.5 that by increasing the value of the total multiplicity s, we improve performance. Asymptotically, the soft-decision RS decoding algorithm is about 0.4-dB better than the conventional hard-decision decoding algorithm for the MEEPR4 channel with RS (143, 129) code at a sector error rate SER=10^{-4}.

Furthermore, a comparison of different multiplicity computation algorithms (See Section 2.3.3 for the details on the multiplicity computation algorithms) on an equalized magnetic recording channel is given in Fig. 3.6 for an RS (181, 169) code. The performance of Method 2 is 0.1dB worse than the KV algorithm but more than 0.1dB better than the GS algorithm, while the algorithm of Gross *et al.* [15] and our Method 1 are almost as good as the KV algorithm with comparable total multiplicity. If we increase the total multiplicity, the difference in performance among the various algorithms will become more noticeable, but since it will also increase the decoding complexity, which increases decoding latency, we only investigate their performance with a small total multiplicity, which is the case of interest in magnetic recording. Note that these results cannot be directly compared with the AWGN channel results shown in Fig. 2.4, because of the widely different values of the multiplicity and code rate.

In addition, the performance of the soft Chase algorithm using an RS (143, 129) code on an MEEPR4 channel was tested and the results are shown in Fig. 3.7. The total number of bits flipped is set at $p = 6$, which means $2^p = 64$ test patterns. By flipping the least reliable bits in the received sequence, and generating different multiplicity patterns, we expanded the decoding span of a given received codeword. Then from the expanded list of decoding candidates we declared the most likely one as the output. A 0.4-dB gain is observed compared to soft-decision RS decoding without bit-flipping. Higher gains can be expected by increasing the value of p, at the expense of a small increase in decoding complexity compared to original KV algorithm.

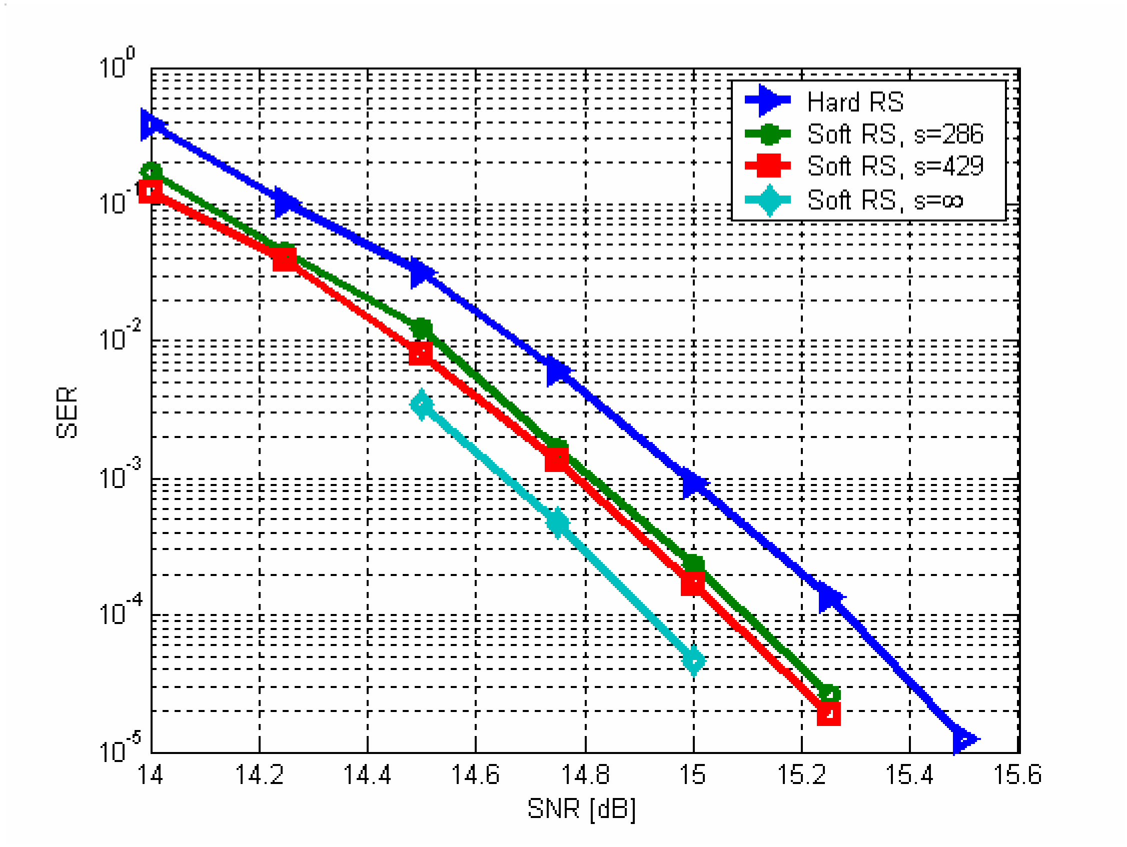


Fig. 3.5. Multiplicity effect on soft-decision algorithm for RS (143, 129) code, four-way interleaved, equalized MEEPR4 channel, channel density is 2.967.

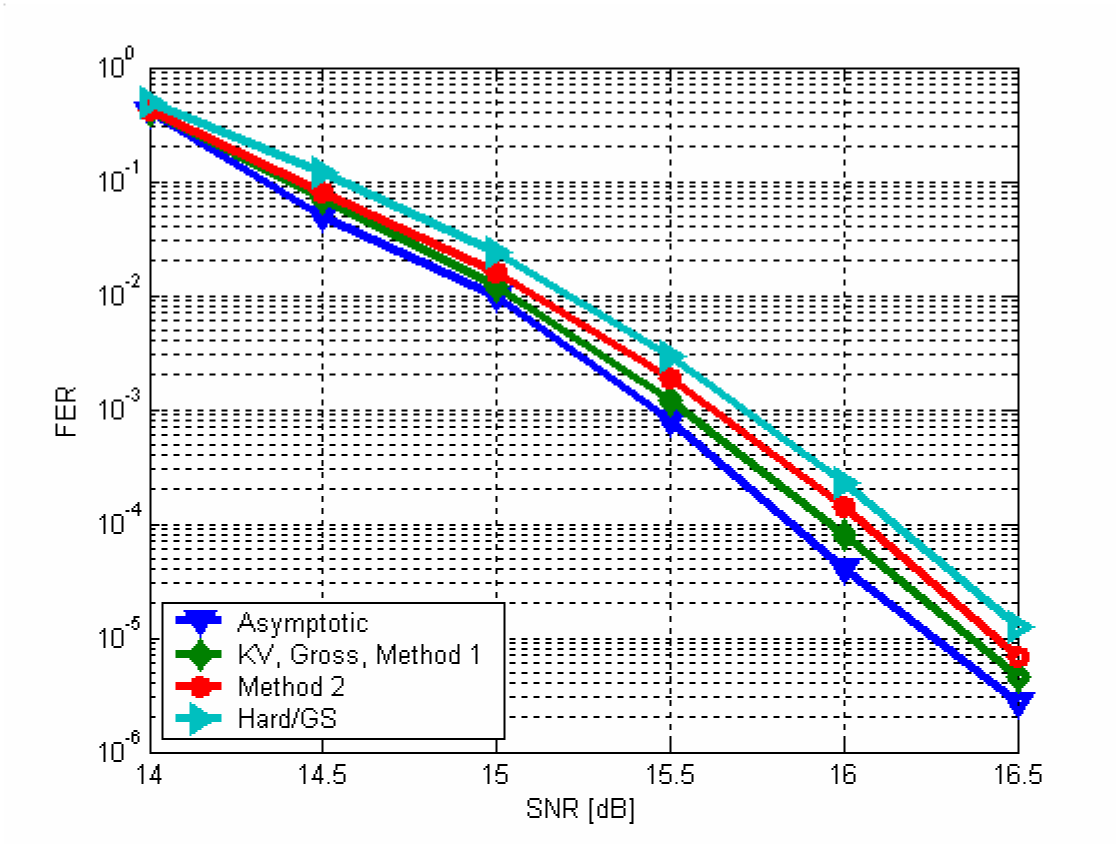


Fig. 3.6. Performance of soft-decision decoding of RS (181,169) with different multiplicity computation algorithms on an equalized MEEPR4 channel, S_c=2.967, s = 543.

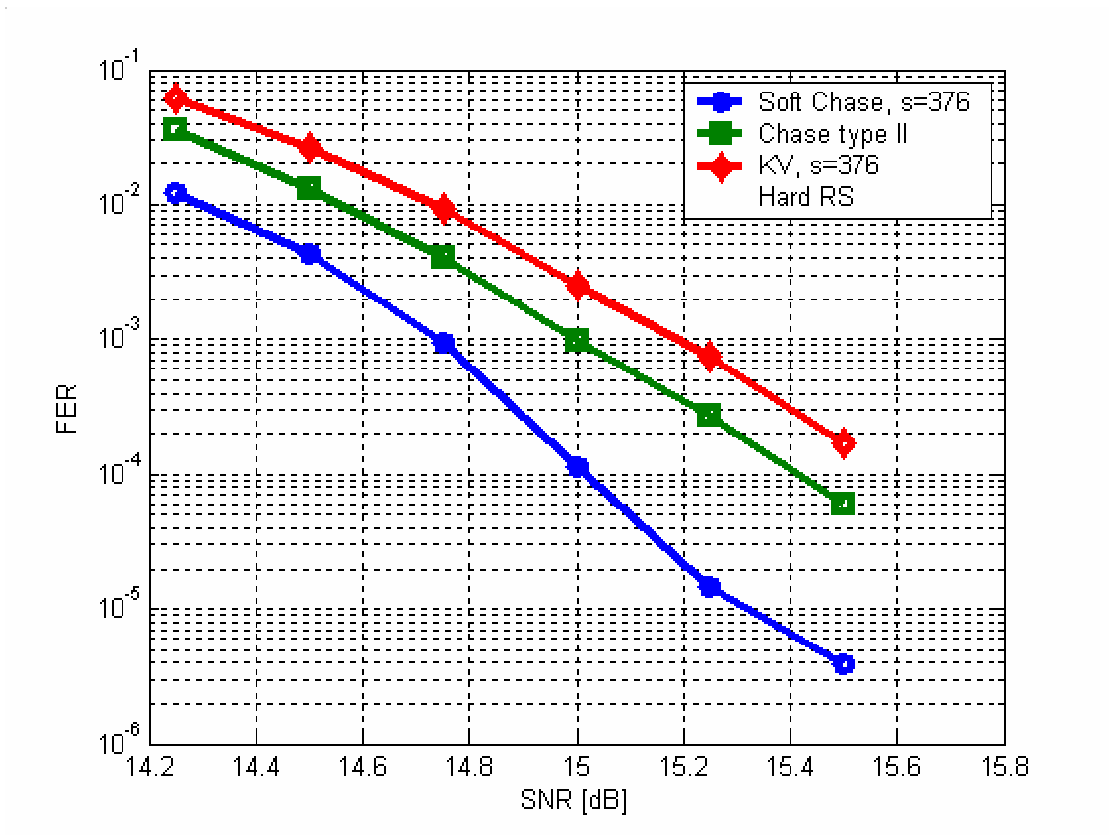


Fig. 3.7. Performance of RS code (143, 129) using the soft Chase algorithm with p = 6 on an equalized MEEPR4 channel, S_c=2.967.

3.5 Concatenation with Inner Codes

In the KV algorithm, a reliability matrix is generated according to the channel output information, which supplies the probability of each symbol of the RS code to be a certain value in $GF(q=2^m)$. Intuitively, one would think that making each symbol of the hard-decision channel output information more reliable would lead to improved reliability matrices.

One way of doing this is symbol-based concatenation. Let us assume a linear code C over an m-ary code in $GF(q = 2^m)$ with codelength n, information length k. For $c_1, c_2 \in C$, the Hamming distance between two codewords c_1 and c_2 is the number of positions where c_1 and c_2 differ. The minimum distance, or simply distance of a code C, is defined to be the minimum Hamming distance $d_{\min}$ between a pair of distinct codewords of C. Given an m-ary (n,k) linear code C_1 with minimum distance $d_{1\min}$ and a binary (n',m) linear code C_2 with $d_{2\min}$, their concatenation $\overline{C} = C_1 \bullet C_2$ is a code which first encodes the message according to C_1 and then encodes each of the symbols of the codeword of C_1 further using C_2 (since each symbol of C_1 has m bits and C_2 is a m-ary code , this encoding is well-defined). The concatenated code $\overline{C}$ has a minimum distance at least the product of the outer and inner codes' minimum distances, that is $d_{1\min} \times d_{2\min}$. Furthermore, when an RS code is used as the outer code, and soft-decision algorithms such as the KV algorithm are used, Guruswami and Sudan have shown in [16] that the error-correction capability goes up to $(1 - 1/m)$. Using the channel output information, each symbol of an RS codeword will be assigned a probability in the reliability matrix. Concatenation with some binary code can provide more reliable information, particularly if a *maximum a posterior* (MAP) decoder is used for the inner code. In turn, a more reliable symbol probability for the RS codeword can be used for the computation of the reliability and multiplicity matrices, which improves the decoding performance.

3.5.1 Concatenation with a Single Parity-Check Code

We have done a simple experiment to evaluate the improvement that concatenation can bring to soft-decision decoding. In an attempt to provide improved channel information to the soft decoder, we considered the concatenation of an RS (143, 126) code with an single parity-check (SPC) (9, 8) code, and compared it with a single RS (143, 112) code with the same overall code rate. The results are shown in Fig. 3.8 for s = 356, from where we can see that the concatenation provides a 0.75-dB gain at a frame error rate FER = 10^{-4} over an AWGN channel and more coding gain can be expected if we increase the total multiplicity. Another example is shown in Fig. 3.9 using the same parameters as the simulation in Fig. 3.8, but with equalized MEEPR4 channel. It is shown that by concatenation, an RS (143, 126) code with an SPC (9, 8) code, a 0.5-dB gain over the RS (143, 126) code can be realized. However the decoding performance of the concatenated code is worse than a single RS (143, 112) code with the same code rate on an equalized MEEPR4 channel. For PR channels, there are certain error events [17] that a simple SPC code cannot correct. However, if we specially design the inner codes which can be used to correct the special error events we might have, on certain PR channels, a better performance since the concatenation will not only improve the reliability information for soft-decision decoding but also help to detect/remove the error events. Since in this dissertation we are mainly focus on the soft-decision decoding algorithms instead of the design of concatenation coding schemes to prevent the error events over PR channels, we will not go further into the various concatenation schemes. Some discussion about the error events and parity check codes designed to remove those error events can be found in [17], [18].

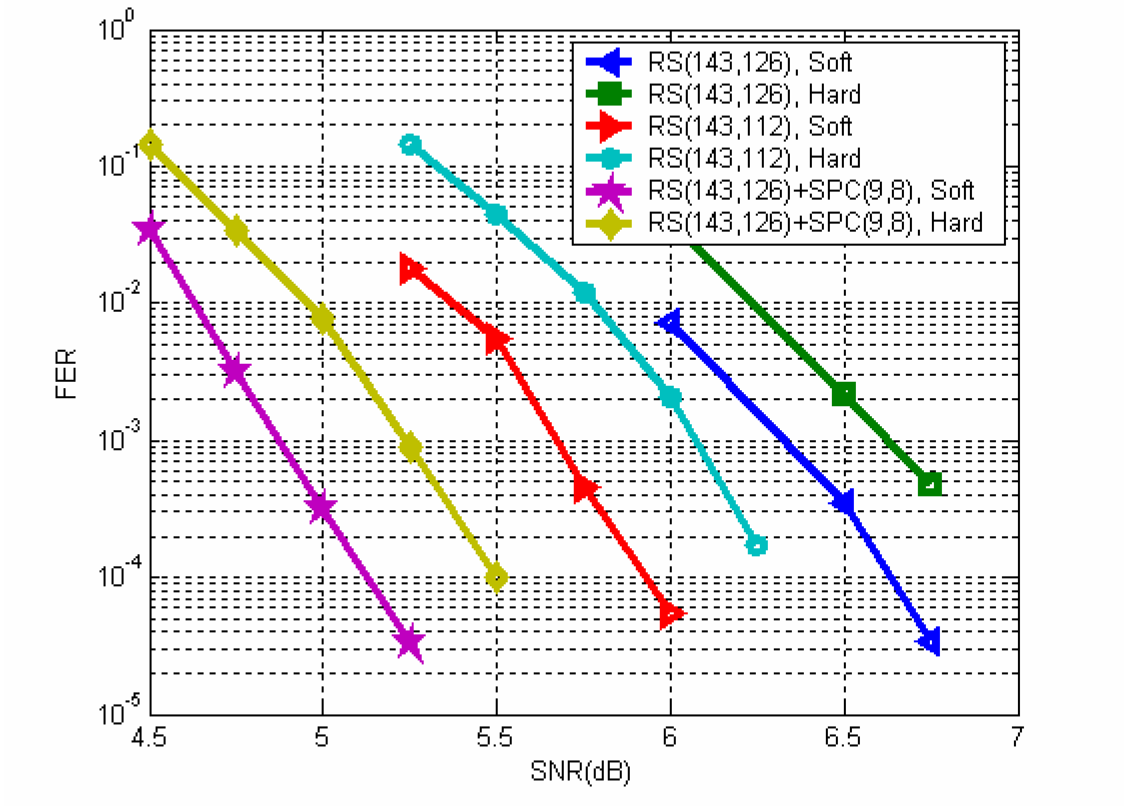


Fig. 3. 8. Effect of concatenation on soft-decision RS decoding over AWGN channels.

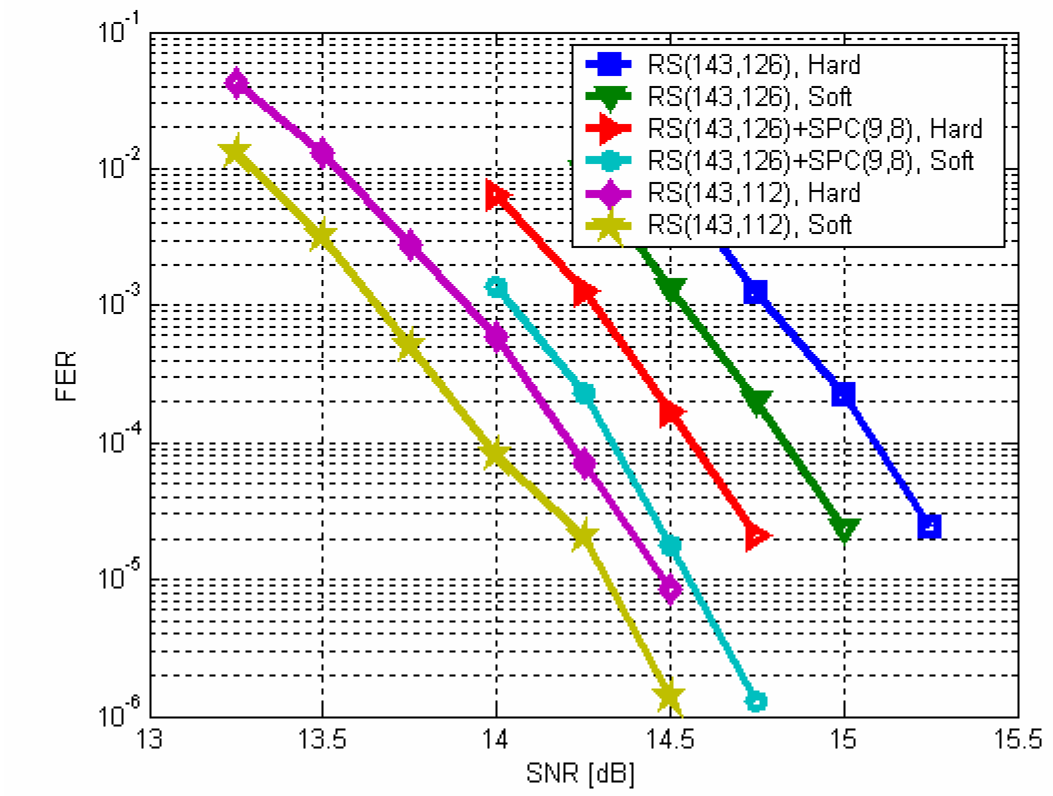


Fig. 3. 9. Effect of concatenation on soft-decision RS decoding over equalized MEEPR4 channels, S_c=2.967.

3.5.2 Concatenation with LDPC Code

The concatenation described above shows that the codelength of the inner code should be small, otherwise there will be a tremendous decoding complexity increase if we use concatenation. Still there is another type concatenation, such as a given (n,k) linear code C_1 and a inner (n',n) linear code C_2, their concatenation $\tilde{C} = C_1 \circ C_2$ is a code which first encodes the message according to C_1 and then encodes the whole codeword of C_1 further using C_2. The inner code will be used to handle the certain error events, then the outer code will take care of the residual errors. As the research on LDPC codes has shown better performance of LDPC codes in magnetic recording systems [19], [20] with random noise, the idea of using serial concatenation of LDPC codes with outer RS codes has been comtemplated. There are two reasons for using such concatenation, one is that LDPC codes have good performance over PR channels; another reason is that since the error probability performance of LDPC codes is not clearly established at high SNRs, a concatenation with an RS code, whose error probability performance is known, will guarantee a good performance in the high SNRs region.

Simulation results for soft-decision RS decoding on the AWGN channel are given in Fig. 3.10 and compared with an LDPC code [21]. Soft-decision RS decoding shows a 0.2-dB gain over hard-decision decoding, but is worse than the LDPC code with similar parameters by more than 2 dB at a sector-error rate SER=10^{-4}. The performance of soft-decision RS decoding improves by 0.5 dB by concatenating it with a single parity check (SPC) (9, 8) code, while the LDPC code improves by 1.2 dB. In both cases the performance of the RS code is worse than an LDPC code with the same overall code rate on the AWGN channel.

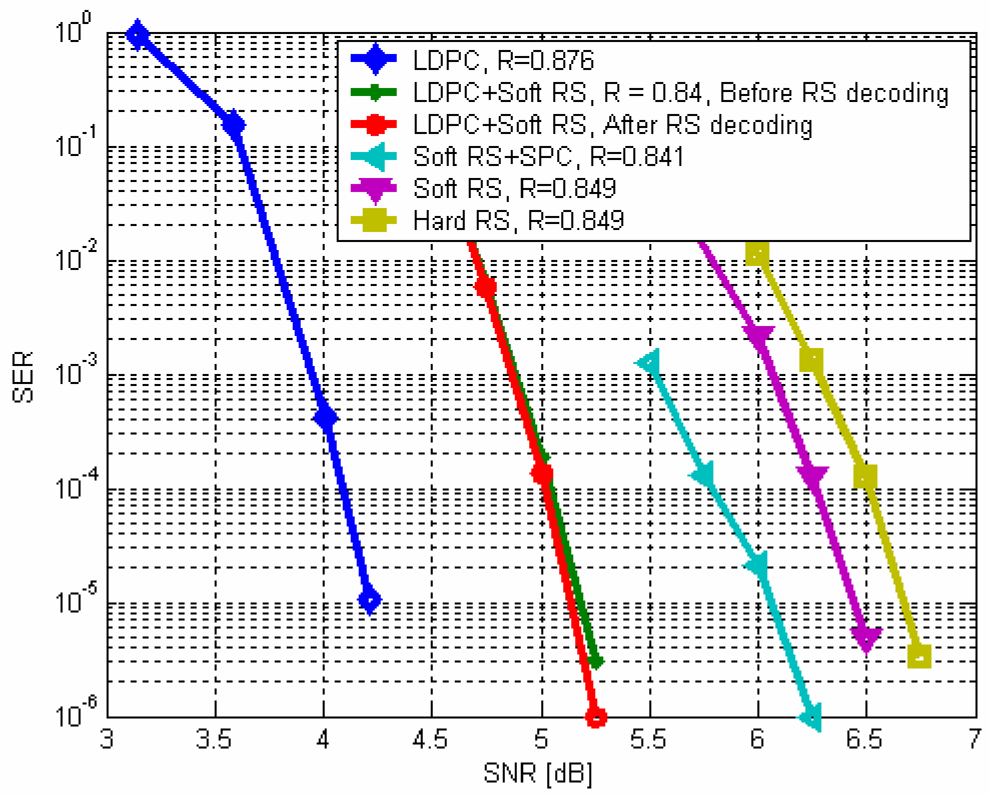


Fig. 3. 10. Performance of LDPC code concatenated with an RS code on AWGN channels.

However, in magnetic recording channels, the dominant impairment may be burst noise caused by media defects (MDs) and thermal asperity (TA). We also tested the performance of the RS code and compared it with an LDPC code on PR channels with and without burst noise (assuming the location of the bursty errors is known). For PR channels, the decoding gap between RS and LDPC codes is not as large as in AWGN channels; there is only a 0.5-dB difference at SER=10^{-4}, and this gap will be even smaller in the presence of erasures. The performance of the four-way interleaved RS (127, 115) code (RS II) concatenated with a (4376, 4094) LDPC code (LDPC II) was evaluated and the results are shown in Fig. 3.11. Compared to single codes with similar code rate, e.g., a four-way interleaved RS (137, 117) code (RS I) and a single (3584, 3140) LDPC code (LDPC I), the concatenation does not provide any coding gain on this PR channel, regardless of whether erasures are present or not. The reason might be that the LDPC decoder generates an approximately uniformly distributed error pattern if it cannot correctly decode the received vector, and this error pattern cannot be decoded by the RS decoder either. This suggests that an RS and LDPC concatenation scheme for future magnetic recording system might not be useful. It is worth noting that code concatenation does bring extra coding gain in PR channels, but if we compensate for the code rate loss, there is no net gain.

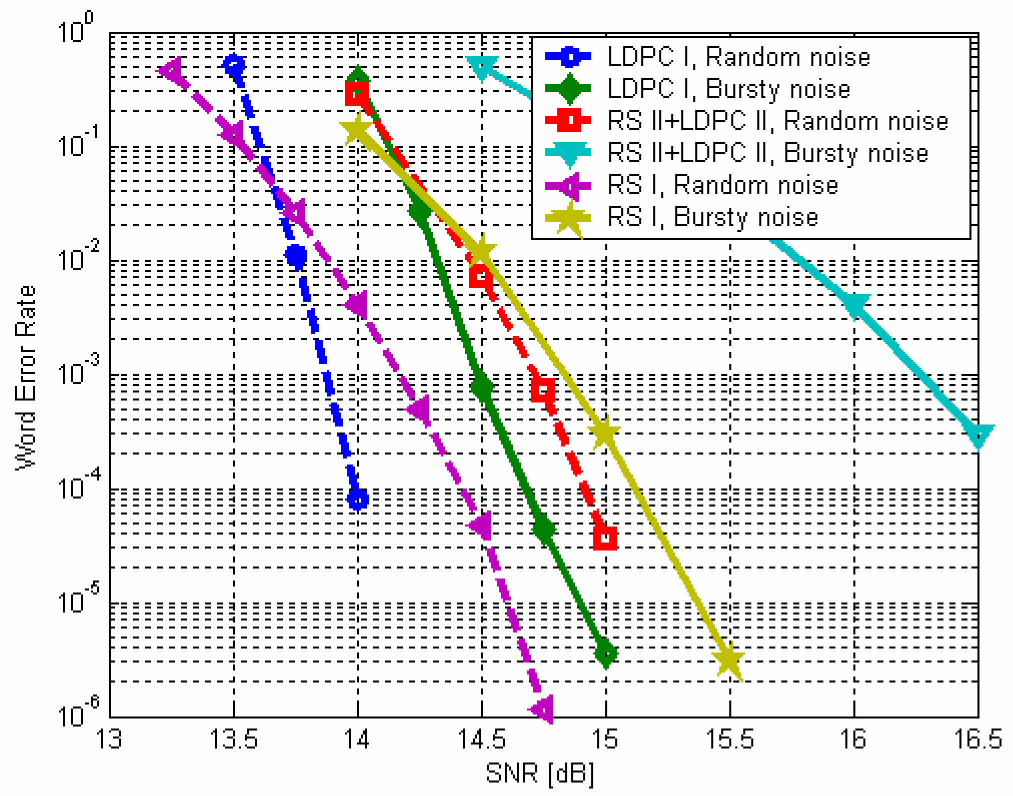


Fig. 3.11. Performance comparison of RS codes concatenated with LDPC codes with random noise and bursty noise over equalized MEEPR4 channels.

3.6 Interaction between Channel Detector and ECC Decoder

Turbo equalization [22] between channel detector and ECC decoder such as a low-density parity-check (LDPC) code has been proposed as a way of obtaining additional coding gain. However, the same concept cannot be applied to RS decoders due to the "hard" output. However, magnetic recording systems normally use several interleaved RS codewords in one sector and the random distribution of errors makes some RS codewords decodable, and some not. Due to this fact, we send those correctly decoded interleaves (RS codewords, checked by CRC) back to the channel detector such as a SOVA or BCJR algorithms for further decoding instead of marking the whole sector in error, and more reliable information from the channel detector can be expected. Fig. 3.12 sheds some light on such a system and Fig. 3.13 shows the performance improvement when we feedback the correctly decoded codeword with some soft version of the information back to the BCJR decoder to both hard-decision and soft-decision decoding algorithms. Three-way interleaved RS (186, 172) codes have been used, and 0.1-dB gain can be observed with three iterations of turbo equalization.

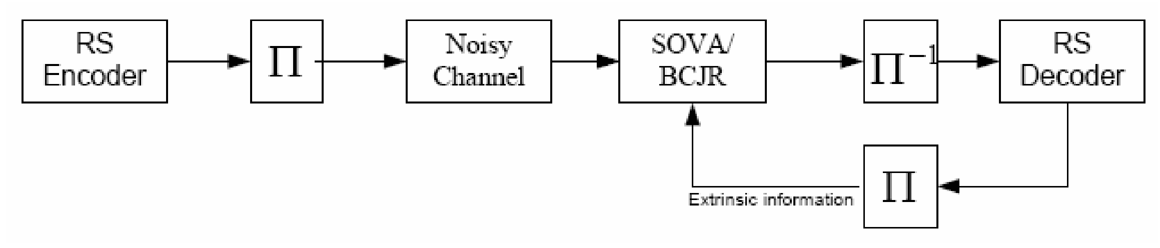


Fig. 3.12. Illustration of the interaction between channel detector and RS decoder.

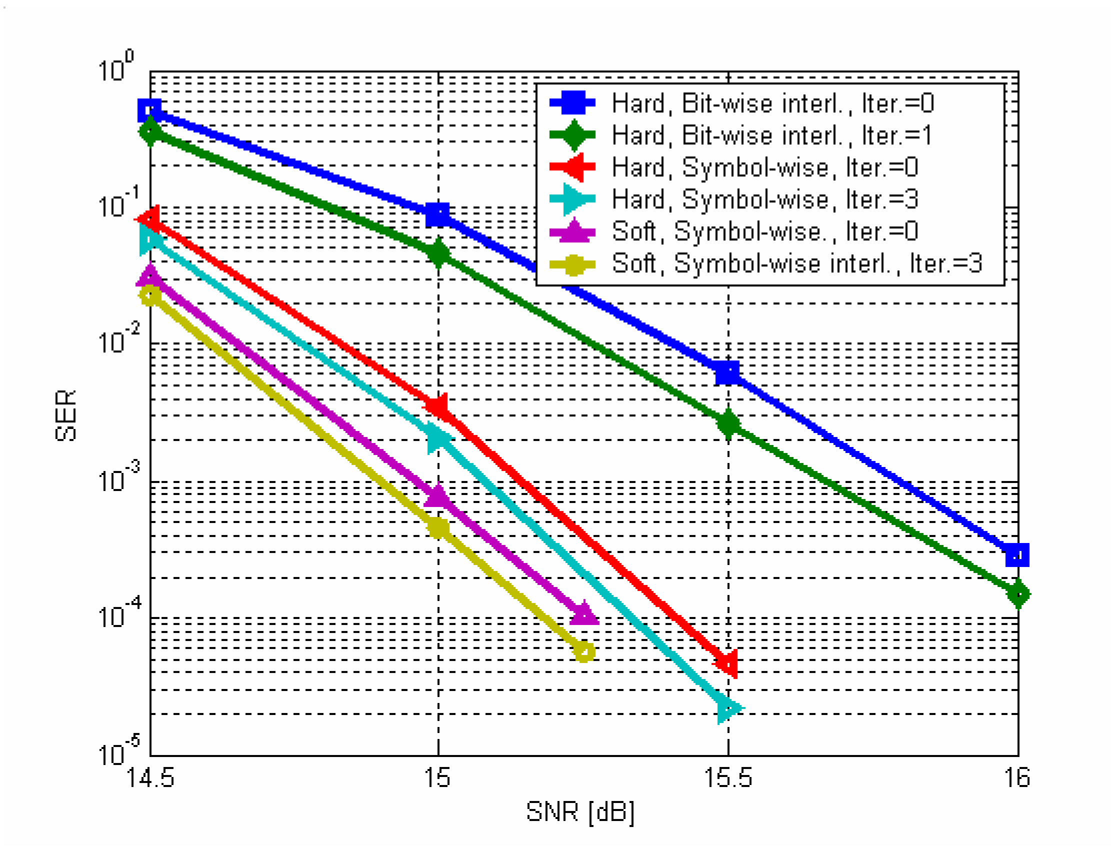


Fig. 3.13. Performance of 3-way interleaved RS (186, 172) code with feedback “hard” extrinsic information from RS decoder to channel detector over equalized MEEPR4 channel.

3.7 Magnetic Recording Channel with Erasures

Besides the random noise introduced by circuits, TA and MDs are the most common problems faced by magnetic recording systems which lead to error bursts. When TA happens, the system can detect it and send a detection flag, channel state information (CSI), to the channel decoder, and the flagged segment of data will be treated as erasure in the following decoding procedure. When MDs happen, their effect depends on the fading depth. In any case, the information data has been contaminated by noise bursts which leads to a contiguous segment of data being partially or fully erased. Here by abuse of notation, we denote such phenomena as "erasure", although it might not be a total information loss. In this work, we are mainly focused on the occurrence of MDs, including both partial and full erasures. The model of MD noise is presented in Fig. 3.14. Normally, the envelope of the signal is a constant value, but when the MD happens, the envelope of the signal decays gradually to zero, then goes up to constant value at the end of MD area. We use two parameters to measure the erasure: parameter η is used to represent the depth of erasure fading, where $\eta = 0 \sim 1$ represents full information loss (full erasure) to no information loss (no erasure), and L is used to represent the length of erasures in bits. Simulation shows that when the fading depth η increase, the decoding performance decreases, which means that the partial decayed channel output still can supply some useful information to the soft decoder. An MD in a real system is composed by different lengths of erasure with different fading depth as shown in Fig. 3. 14, here in order to simplify the process, we assume the MD has the same fading depth for the whole erasure. We use two easily computed bounds to evaluate the real system performance P_e:

$$P(e \mid \eta = 1) < P_e < P(e \mid \eta = 0)$$

where $P(e \mid \eta = 1)$ represents the error probability of full erasure case and $P(e \mid \eta = 0)$ represents error probability of the non erasure case (see also in [23]).

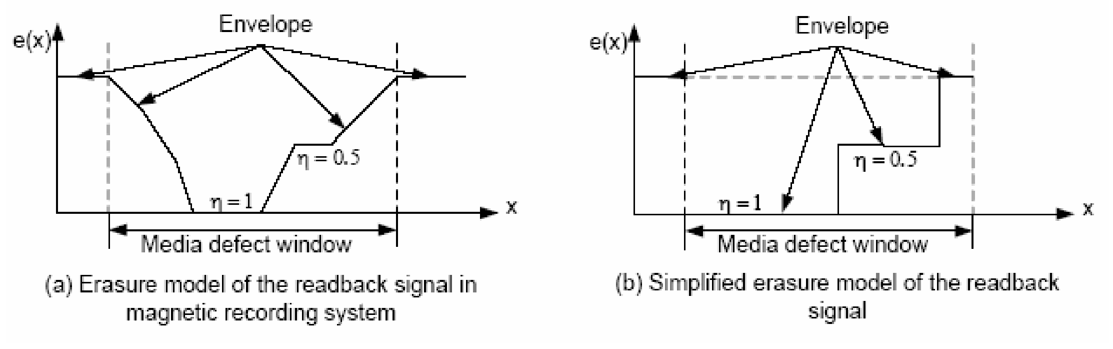

e(x)
Envelope
$\eta = 0.5$
$\eta = 1$
x
Media defect window
(a) Erasure model of the readback signal in magnetic recording system
e(x)
Envelope
$\eta = 0.5$
$\eta = 1$
x
Media defect window
(b) Simplified erasure model of the readback signal

Fig. 3.14. Partial or full erasure caused by MDs in magnetic recording systems.

We investigated the performance of the soft-decision RS decoding algorithm in the presence of erasures with different fading depths without CSI. The results are shown in Fig. 3.15 for 128-bit erasures. A four-way interleaved RS (112, 96) code was used. The fading depth affects the decoding performance significantly. For instance at $\eta = 0.2$, the coding loss caused by the erasure is small, which indicates that the partially erased channel output can still provide useful information to the soft-decision decoder. However, when the fading depth increases to 0.5, the decoding performance becomes almost the same as for a full erasure ($\eta = 1$).

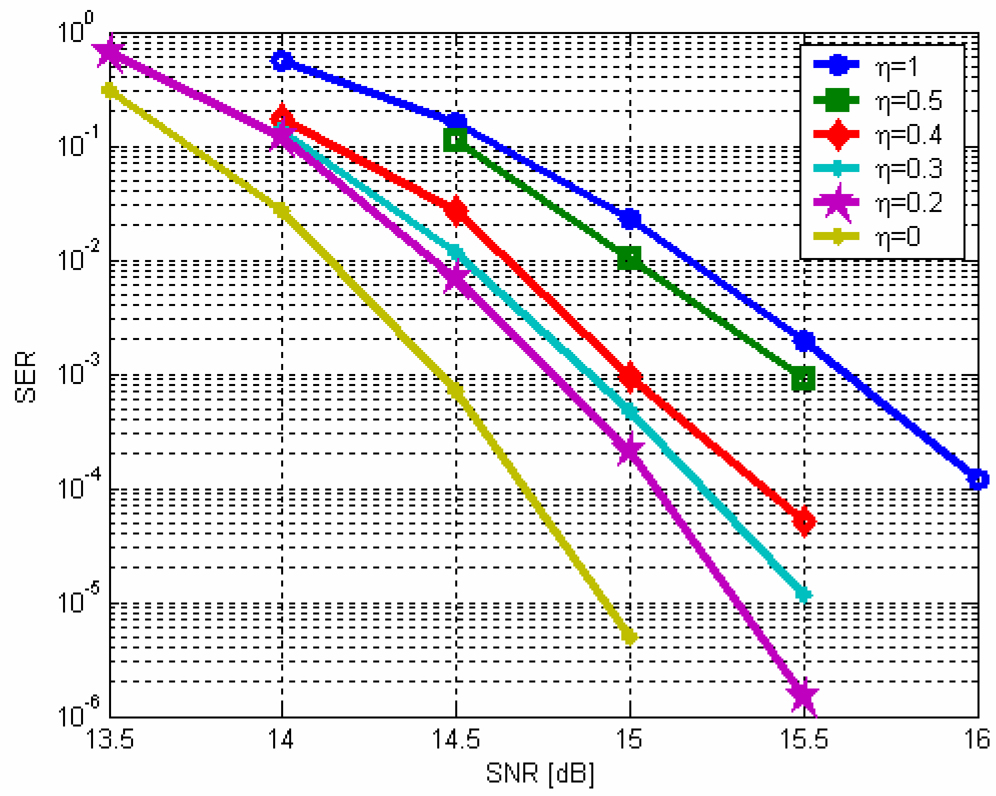


Fig. 3.15. Performance of soft-decision decoding of RS (112, 96) in the presence of erasures, with fading depth as a parameter on an equalized MEEPR4 channel without CSI, S_c=2.967, L=128 bits. The RS code is four-way interleaved, s=336.

The effect of erasure length was also investigated and the results are shown in Fig 3.16. For a four-way interleaved RS (112, 96) code, the maximum erasure correction capability with hard-decision decoding is (112-96)x8x4=512 bits. We tested erasure lengths from 128 to 576 bits. As fading depth increases, the erasure correction capability becomes worse. For a fading depth of 0.5, the SER curve begins to show a floor for $L = 224$ bits, but for a fading depth of 0.2, no error floor is observed for erasure lengths up to $L = 576$ bits.

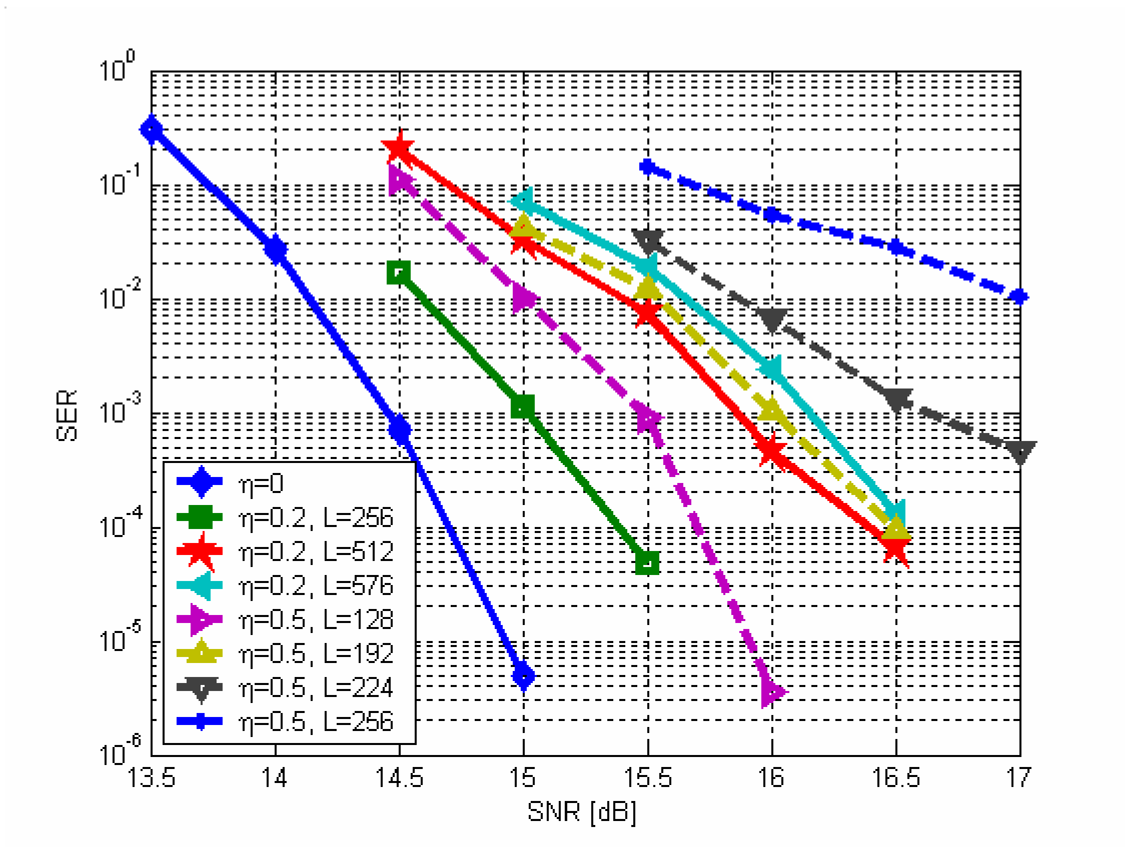


Fig. 3.16. Performance of soft-decision decoding of RS (112, 96) in the presence of erasures, with erasure length and fading depth as parameters, on an equalized MEEPR4 channel without CSI, S_c=2.967. The RS code is four-way interleaved.

Furthermore the effect of knowing CSI is discussed. If CSI can be detected by a certain technique [24], the LDPC decoder can set the loglikelihood ratios (LLRs) of the corresponding bits to zero, and the soft RS decoder can ignore those symbols involved during decoding, which in turn provides a substantial performance gain. A difference between the soft-decision RS and LDPC erasure decoding is that the soft-decision RS decoder completely ignores the errors caused by the erasures, while the

LDPC decoder still needs those erased bits for parity check calculation. Although we mark those erased bits to be unknown, and use parity check information to recover them, an error floor is to be expected for LDPC codes in the presence of erasures, since if more than two bits in a row of the parity check matrix of the LDPC code are involved in an erasure, the LDPC decoder will most likely fail to correctly decode, especially when the erasure length is beyond the LDPC erasure correction capability $\lambda_{\max} \approx 2n / w_r$ [23], where n is the codeword length and w_r is the row weight. As shown in Fig. 3.17, in both the no erasure case and the full erasure and known CSI case, LDPC code performance is much better than RS codes. But somehow if we do not know CSI, which might be the case in a real system, LDPC codes perform very poorly. For erasure length $L = 128$ bits, the LDPC code considered cannot correct any sector without known CSI.

The performance of a 10-bit/symbol RS (547, 487) code on an equalized MEEPR4 channel is given in Fig. 3.18, and compared with a three-way interleaved RS (182, 162) code with the same code rate. The RS (517, 487) code is 0.2-dB better than the RS (182, 162) code with AWGN, but if we consider long erasures, the RS (517, 487) code with 300-bit erasures performs 0.5-dB better than the RS (182, 162) code with 240-bit erasures when the CSI is known.

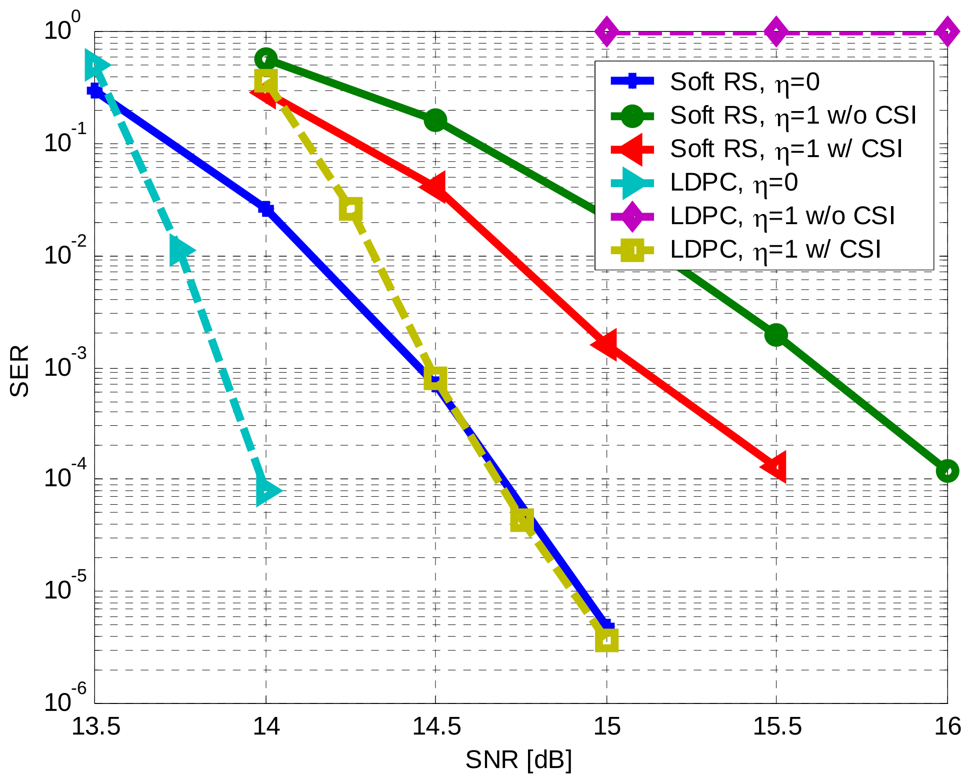


Fig. 3.17. Comparison of LDPC(3584, 3140) code with four-way interleaved RS (112, 96) code on an equalized MEEPR4 channel, S_c=2.967 with different erasure depth, with or without known CSI, L=128 bits.

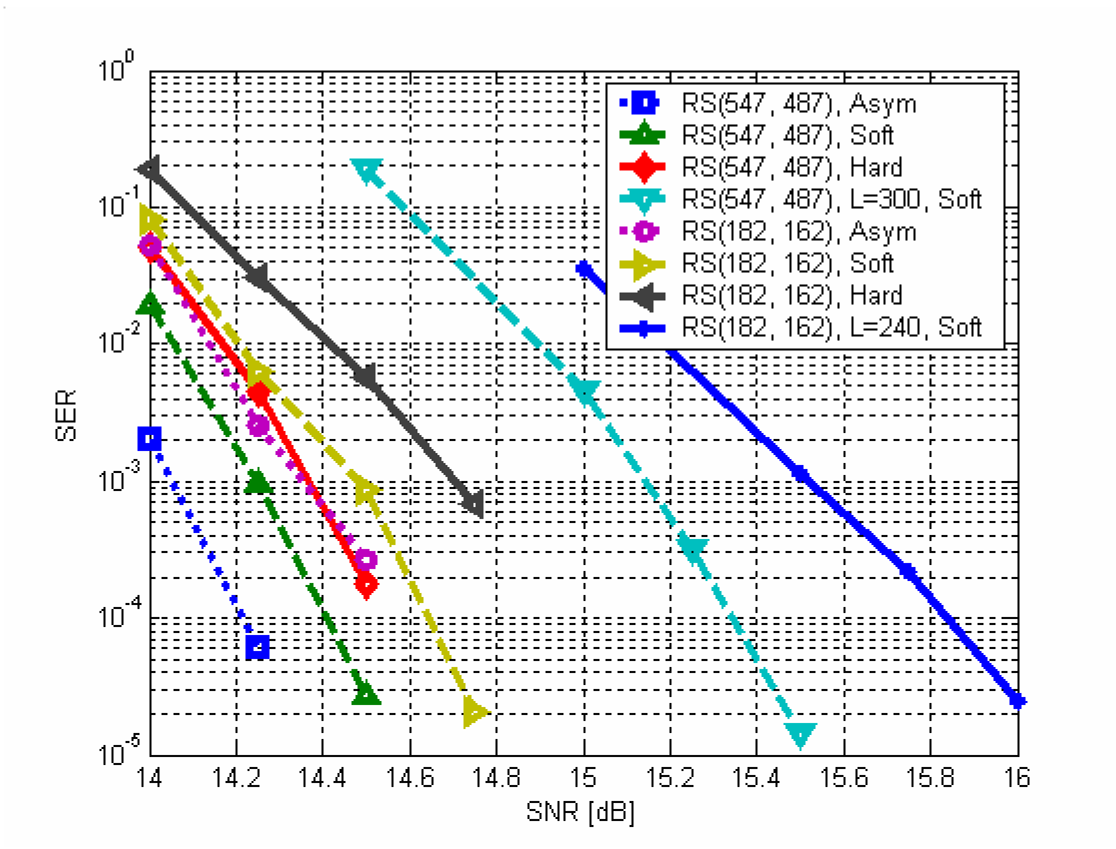


Fig. 3.18. Comparison of 10-bit symbol RS (547, 487) code with three-way interleaved RS (182, 162) code with and without erasures on an equalized MEEPR4 channel, S_c=2.967, with known CSI.

3.7.1 Noise Variance Overestimation

A lot of research has been done on the erasure effect on ECC decoders [23]- [27]. In [25], Song first found out that by overestimating the noise variance for a magnetic recording channel in the presence of erasures, one can have a better decoding performance for the LDPC decoder, and he called this phenomenon "noise overestimation". In [27], Tan *et. al* further investigated this phenomenon, and named it “SNR mismatch”. The basic idea behind this “overestimation” phenomenon is: for soft-decision decoders such as BCJR or LDPC decoders, a noise variance estimation of the channel needs to be supplied to the decoder. Normally the noise variance estimation comes from the practical estimation of a given real system. However, in some extreme cases, the average noise variance estimated by the practical system is not exactly correct. For example, in the case of a segment of contiguous erasures, the average noise variance can be used to decode the part which has no erasures, but the segment of erasures, which means large noise contamination, tells us that a large noise variance is experienced. So if we still use the average noise variance for decoding the erasure segment, definitely we will have worse performance. A larger noise variance is needed for decoding the erasure segment, which is reflected to as noise overestimation. The parameter α is defined as the ratio of the average noise variance used for decoding over the exact noise variance of the channel without erasures. The noise overestimation has only a small effect on the error rate performance of the BCJR decoder, which can be observed in Fig. 3.19.

However, the effect on the probability value on each bit leads to a large performance improvement of both the LDPC decoder and the soft-decision RS decoder. In Fig. 3.20, two RS codes, a three-way interleaved RS (181, 170) code (RS I) and a four-way interleaved RS (136, 128) code (RS II), are compared with a (4376, 4094) LDPC code [21] in the presence of erasures of various lengths. By selecting a noise overestimation factor $\alpha = 3$, the performance of the LDPC code with erasures is much improved. An error floor can still be observed, especially for $L > \lambda_{\max} = 139$

bits. The performance advantage of the LDPC code over the RS I code without erasures is more than 1.5 dB at SER=10^{-4}, but only 1 dB with 128-bit erasures, and becomes even smaller at higher SNRs. The RS code outperforms the LDPC code with 160-bit erasures. Furthermore, by using a smaller interleaving depth, the performance of the RS code improves about 0.5 dB with 128-bit erasures, and more than 1-dB gain is observed with $L = 160$ bits.

We also simulated the system with different precoders. In Fig. 3.21, the LDPC code and the RS I code are used. By changing the precoder from $1/(1 \oplus D)$ to $1/(1 \oplus D^2)$, the performance changes very little for the RS code. However for the LDPC code without erasures, the decoding performance using precoder $1/(1 \oplus D)$ is 0.5dB better at SER=10^{-4}; with $L = 128$ bit erasures, the performance of the system using precoder $1/(1 \oplus D^2)$ is substantially better.

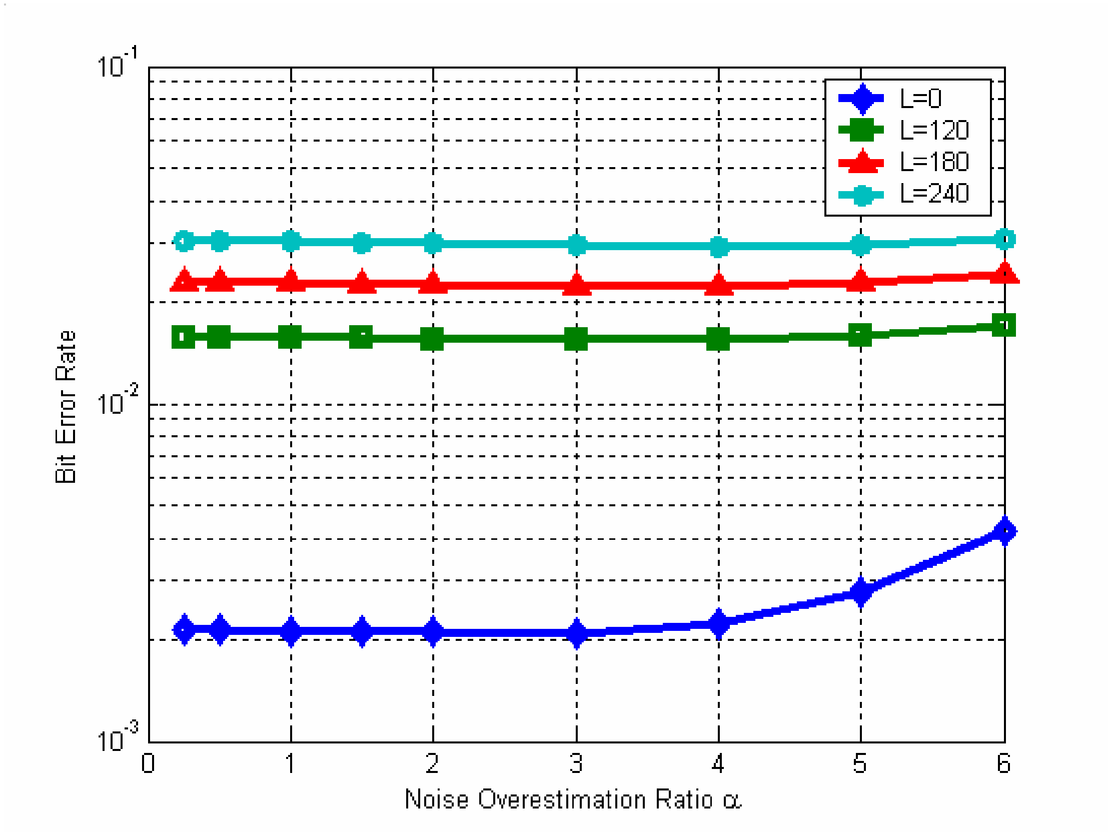


Fig. 3.19. Bit error rate of BCJR output with different noise overestimation ratio at different SNRs.

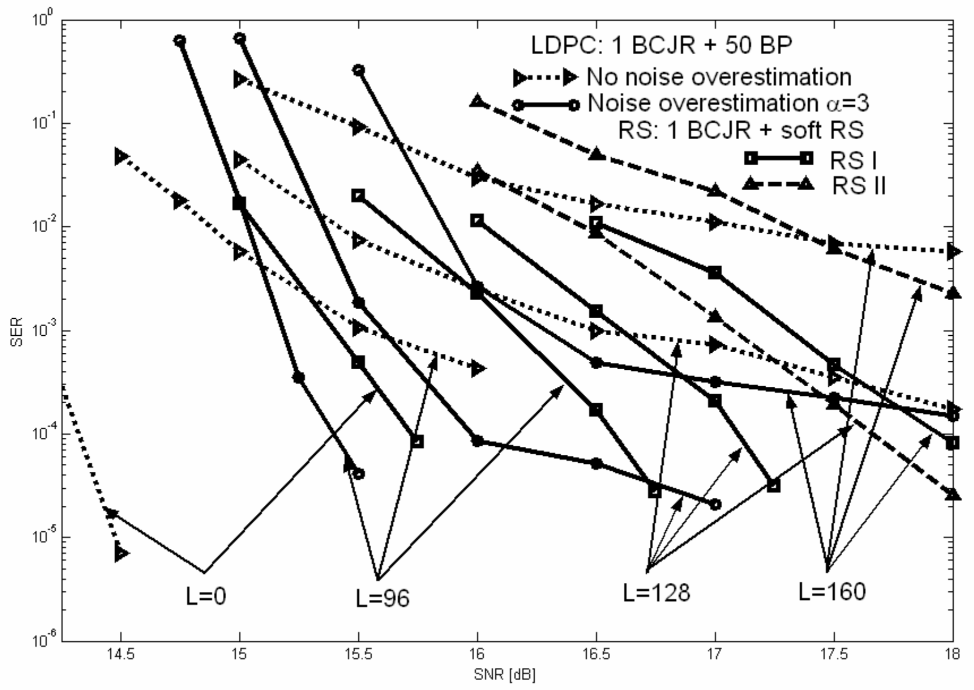


Fig. 3.20 Performance comparison of soft-decision RS and LDPC decoding with erasures on an equalized MEEPR4 channel, S_c=2.967, precoder 1/ (1+D).

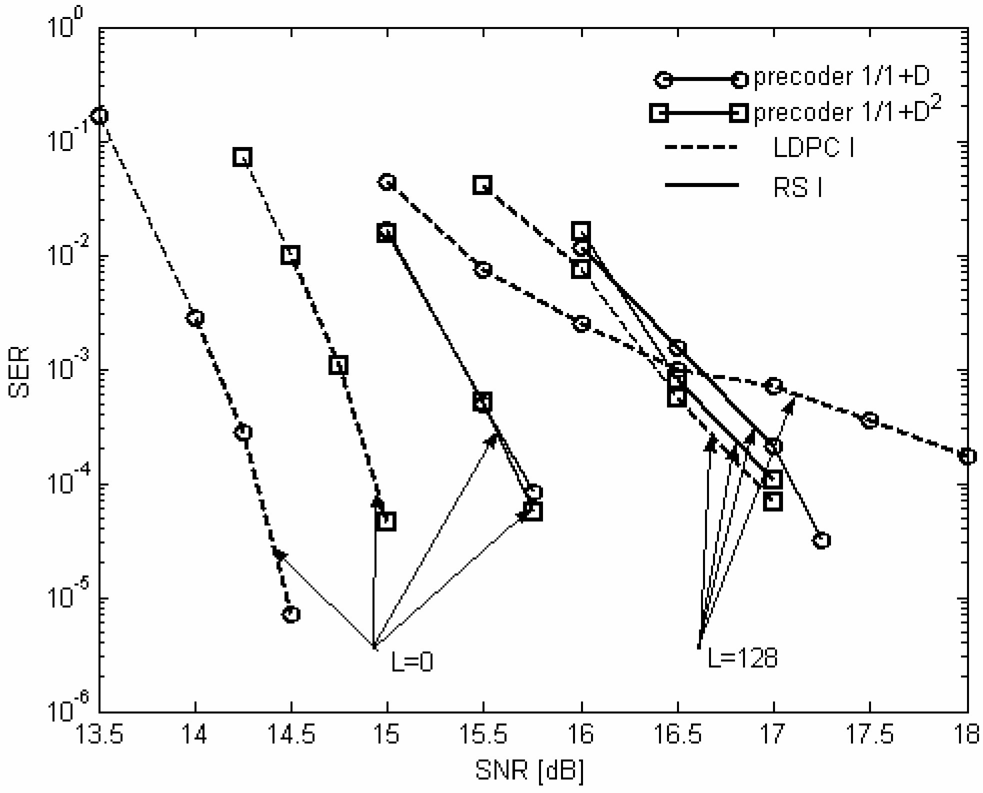


Fig. 3.21 Precoder effect on soft-decision RS and LDPC decoding with erasures on an equalized MEEPR4 channel, S_c=2.967.

For RS codes, shown in Fig. 3.22, when erasures happen with length L=120 bits, and the noise variance overestimation used is $\alpha = 3$, no big difference is observed for hard-decision RS decoding algorithms, however, the performance of soft-decision RS decoding algorithm shows an almost 0.5-dB difference. Moreover, when no erasure happens, using noise variance overestimation leads to a slight degradation of hard-decision RS decoding, and a worse performance of soft-decision algorithm at lower SNRs, however, when the SNR becomes larger, things change: the soft-decision RS decoding algorithm with noise variance overestimation shows a trend of better performance than the one without noise variance overestimation. This can be explained as an optimization of the soft-decision algorithm at high SNRs leads to a better overall system performance, although each component in the system might only be suboptimal.

From simulations, we observed a performance improvement using noise overestimation on both RS and LDPC decoding. A system architecture for decoding using noise overestimation is presented in Fig. 3.23. The channel detector generates an erasure flag when an erasure occurs, and a counter provides the erasure length, which is used in a pre-calculated lookup-table (LUT) that stores the noise overestimation factor for the soft-decision decoder.

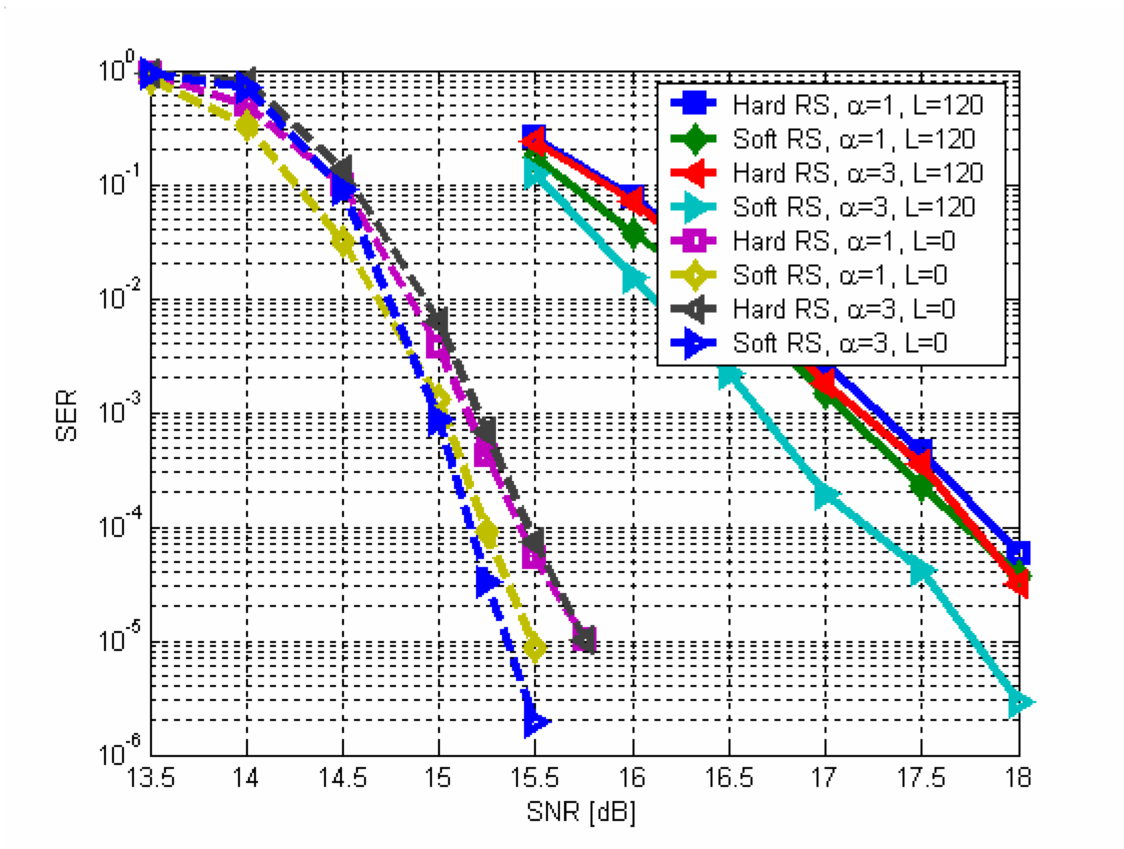


Fig. 3.23 Performance comparison of 3-way interleaved RS (186, 172) code at different noise overestimation ratio with different length of erasures L, $\eta = 0$.

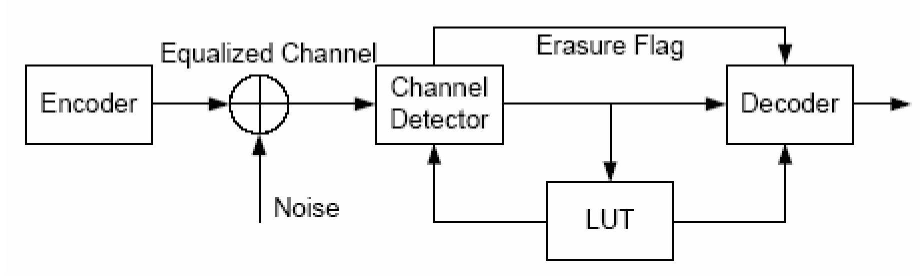


Fig. 3.24 System architecture utilizing noise overestimation effect.

3.8 Summary

In this charpter, we investigated the performance of soft-decision RS decoding algorithms on magnetic recording channels with different parameters. The performance gain of the soft-decision RS decoding algorithm is not as large as for AWGN channels. In order to improve the decoding gain as well as reduce the decoding complexity of soft-decision RS decoding, we proposed the soft Chase algorithm and the re-encoded Chase algorithm which are good for practical implementation in magnetic recording systems. Also the noise overestimation effect when the channel is dominated by burst noise was investiaged.

Bibliography

[1] C. Berrou, A. Glavieux and P. Thitimajshima, “Near Shannon limit error-correcting coding and decoding: Turbo-codes,” in *Proc. IEEE Intl. Conf. Commun.*, pp. 1064-70, 1993.

[2] R.G. Gallager, “Low-density parity-check codes,” *IEEE Trans. Inform. Theory*, vol. 8, pp. 21-28, Jan. 1962.

[3] R. Koetter and A. Vardy, “Algebraic soft-decision decoding of Reed-Solomon codes”, *IEEE Trans. Inform. Theory*, vol. 49, pp. 2809-2825, Nov. 2003.

[4] D. Chase, “A class of algorithms for decoding block codes with channel measurement information,” *IEEE Trans. Inform. Theory*, vol. 18, pp.170-182, Jan. 1972.

[5] H. Xia, C. Zhong and J. R. Cruz, "A Chase-type algorithm for soft-decision Reed-Solomon codes on Rayleigh fading channels," in *Proc. IEEE Global Commun. Conf.*, pp. 1751-1755, 2003.

[6] H. Xia and J.R. Cruz, "Application of soft-decision Reed-Solomon decoding on magnetic recording channels," *IEEE Trans. Magn.*, vol. 40, pp. 3419-3430, Sept. 2004.

[7] W. J. Gross, F. R. Kschischang, R. Koetter and P. G. Gulak, "A VLSI architecture for interpolation in soft-decision list decoding of Reed-Solomon codes," in *Proc. IEEE Workshop Signal Processing Syst.*, pp. 39-44, Oct. 2002.

[8] H. Xia and J. R. Cruz, "Reduced-complexity implementation of soft-decision decoding of Reed-Solomon codes," in *Proc. Intl. Conf. Acoustics, Speech, and Signal Processing,* pp. v33-v36, 2004.

[9] A. Ahmed, R. Koetter and N. R. Shanbhag, "VLSI architecture for soft-decision decoding of Reed-Solomon codes," submitted to *IEEE Trans. VLSI Systems,* Feb. 2003.

[10] R. E. Blahut, *Theory and Practice of Error Control Codes*. Reading, MA: Addison-Wesley, 1983.

[11] V. Guruswami and M. Sudan. "Improved decoding of Reed-Solomon and algebraic-geometric codes," *IEEE Trans. Inform. Theory*, vol. 45, pp.1757-1767, Sept. 1999.

[12] P. Kabal and S. Pasupathy, "Partial-response signaling," *IEEE Trans. Commun.*, vol. 23, pp. 921-934, Sept. 1975.

[13] T. Nishiya, K. Tsukano, T. Hirai, T. Nara, and S. Mita, "Turbo-EEPRML: An

EEPR4 channel with an error-correcting post-processor designed for 16/17 rate quasi-MTR code," in *Proc. IEEE Global Telecommun. Conf.*, pp. 2868-2873, 1998.

[14] L. R. Bahl, J. Cocke, F. Jelinek and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. 20, pp. 284-287, Mar. 1974.

[15] W. J. Gross, F. R. Kschischang, R. Koetter and P. G. Gulak, "Simulation results for algebraic soft-decision decoding of Reed-Solomon codes," in *Proc. 21st Biennial Symp. Commun.*, pp. 356-360, 2002.

[16] V. Guruswami and M. Sudan, "Decoding concatenated codes using soft information", in *Proc. 17th IEEE Annual Conf. Computational Complexity,* pp. 122-131, 2002.

[17] R. Cideciyan, J. Coker, E. Eleftheriou and R. Galbraith, "Noise predictive maximum likelihood detection combined with parity-based post-processing," *IEEE Trans. Magn.*, vol. 37, pp. 714-720, Mar. 2001.

[18] Z. A. Keirn, V. Y. Krachkovsky, E. F. Haratsch, H. Burger, "Use of redundant bits for magnetic recording: single-parity codes and Reed-Solomon error-correcting code," *IEEE Trans. Magn.*, vol. 40, pp. 225-230, Jan. 2004.

[19] Z.-N. Wu, *Coding and Iterative Detection for Magnetic Recording Channels*. New York, NY: Kluwer Academic Publishers, 2000.

[20] H. Song, R. M. Todd and J. R. Cruz, "Application of low-density parity-check codes to magnetic recording channels," *IEEE J. Select. Areas Commun.*,

vol. 19, pp. 918-923, May 2001.

[21] D. J. C. Mackay, available at: http://www.inference.phy.cam.ac.uk/mackay/codes/data.html.

[22] R. Koetter, A. C. Singer and M. Tuechler, "Turbo Equalization," *IEEE Signal Processing Magazine*, vol. 21, pp. 67-80, Jan. 2003.

[23] M. Yang and W. E. Ryan, "Performance of (quasi-)cyclic LDPC codes in noise bursts on the EPR4 channel," in *Proc. IEEE Global Telecom. Conf.*, 2001, vol. 5, pp. 2961-2965.

[24] H. Xia, W. Tan and J. R. Cruz, "Nested codes for perpendicular recording channels," *IEEE Trans. Magn.*, vol. 40, pp. 3111-3113, July 2004.

[25] H. Song, *Applications of Iterative Decoding to Magnetic Recording Channels.* Ph.D. dissertation, The University of Oklahoma, 2002.

[26] H. Xia and J. R. Cruz, "Soft-decision decoding of Reed-Solomon codes on magnetic recording channels with erasures," in *Proc. IEEE Int. Commun. Conf.*, pp. 2909 -2913, 2003.

[27] W. Tan, and J. R. Cruz, "Signal-to-Noise Ratio Mismatch for Low-Density Parity-Check Coded Magnetic Recording Channels," *IEEE Trans. Magn*., vol. 40, pp. 498-506, Mar. 2004.

Chapter 4

Implementation of Soft-Decision Reed-Solomon Decoding Algorithms

4.1 Introduction

After the proposal of the KV algorithm [1], the efficient implementation of the decoding algorithm has been the focus of a lot of research attention. The soft-decision RS decoder using the KV algorithm can be divided into four major steps: reliability/multiplicity matrix computation, soft (polynomial) interpolation, factorization and list decoding. Among these four steps, soft (polynomial) interpolation is the most computationally complex. Here we address the issue of hardware implementation of soft interpolation.

The implementation of the soft interpolation step, the generation of the bivariate polynomial $Q_M(x,y)$ starts with a set of polynomials

$$\begin{aligned}\mathbf{G} &= \{G_0, G_1, \cdots G_l, \cdots, G_b\} \\ &= \{1, y^1, \cdots, y^l, \cdots, y^{b-1}\}\end{aligned}. \tag{4.1}$$

From this set of monomials, we let each polynomial satisfy certain constraints, that is, passing through point (x_i, y_i) with multiplicity m_i given by the multiplicity matrix M . When all the constraints in the multiplicity matrix have been satisfied, the smallest polynomial in the updated polynomial set $\mathbf{G}'$ is the output $Q_M(x,y)$ [2], [3].

It is shown in (2.22) that in order to let a polynomial $G(x,y)$ pass through a point (x_i, y_i) with multiplicity m_i, we need to make the Hasse derivative (HD) [4]

$$G^{(\gamma,\beta)}(x_i, y_i) = \sum_{\substack{\gamma \le j \\ \beta \le l}} \binom{j}{\gamma}\binom{l}{\beta} c_{jh} x_i^{j-r} y_i^{l-\beta} \tag{4.2}$$

of $G(x, y)$ to be zero for every γ and β satisfying $\gamma + \beta < m_i$ and $\gamma, \beta \ge 0$. So the soft interpolation step can be further divided into three sub-steps:

Initialize: 1. A set of polynomials $\mathbf{G}^{(0)} = \{1, y^1, \cdots, y^l, \cdots, y^b\}$

2. A sequence of constraints $\{(x_i, y_i), m_i\}_{i=0}^{N-1}$

1) Compute the Hasse derivative for each polynomial $G_l(x, y)$ in $\mathbf{G}$ with given $x = x_i, y = y_i, \gamma,$ and β. From those $G_l(x, y)$ with nonzero HD value, compute their $(1,\ k-1)$-weighted degree and denote the smallest degree $G_l(x, y)$ to be $\Delta(x, y)$;

2) Update those polynomials whose weighted degree are not the smallest one as $G_l^{'}(x, y) = G_l(x, y) - \dfrac{G_l(x_i, y_i)}{\Delta(x_i, y_i)}\Delta(x, y)$;

3) Update smallest degree polynomial $\Delta^{'}(x, y) = (x - x_i)\Delta(x, y)$.

For those polynomials $G_l(x, y)$ whose HD value is zero, no operation will be done to them. After the set of polynomials satisfy all the constraints, the interpolation step is done.

4.2 Memory Requirement

The hardware implementation of the soft-decision decoding algorithm requires setting up a block of memory to store the coefficients of polynomials $G_l(x,y)$ generated during each iteration, as shown in Fig. 4.1. Since the coefficients of polynomials can be represented as a power of the primitive element α, we would like to store this power in memory instead of the real value of the coefficients of the polynomials. Also, we need to build two look-up tables (LUTs) to deal with the operations in $GF(q)$, that is to change α^i to integer j and vice versa. On-the-fly computation of these LUT entries can be implemented to save memory, at the expense of decoding delay. In practical implementation, LUTs might not be a good choice when q is very large, a real time computation circuit can be easily used to replace the LUTs for the following architectures discussed.

The memory can be categorized into three types:

Type-I memory is used for storing $G_l(x,y)$, which is initialized as a group of monomials such as $\{1, y, y^2, \cdots, y^{b-1}\}$.

Type II is used to store the lowest weighted degree bivariate polynomials $\Delta(x,y)$ generated at every iteration. Since the memory is ordered as

$$x^0y^0, x^1y^0, \cdots x^{a-1}y^0; x^0y^1, \cdots, x^{a-1}y; x^0y^{b-1}, \cdots, x^{a-1}y^{b-1},$$

if we want to multiply $\Delta(x, y)$ or $G_l(x, y)$ by some monomial x^i, we simply need to shift the value inside each segment of memory to the right by i. The highest power of x is bounded by a, so no overflow will occur.

Type-III memory can be further divided into two categories: one is used to store the value of the monomials, such as $\binom{j}{\gamma} x_i^{j-\gamma}$, another is used to store the value of $\binom{l}{\beta} y_i^{l-\beta}$ in (4.2).

The choice of parameters a, b is given in (2.25) and (2.26), respectively.

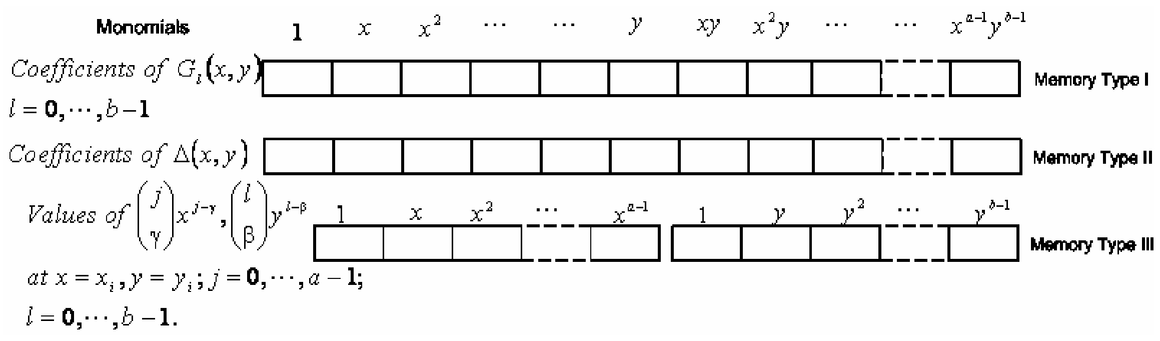

Monomials
1 x x^2 $\cdots$ $\cdots$ y xy x^2y $\cdots$ $\cdots$ $x^{a-1}y^{b-1}$
Coefficients of $G_l(x,y)$
$l = 0,\cdots,b-1$
Memory Type I
Coefficients of $\Delta(x,y)$
Memory Type II
Values of $\binom{j}{\gamma}x^{j-\gamma}, \binom{l}{\beta}y^{l-\beta}$
1 x x^2 $\cdots$ x^{a-1} 1 y y^2 $\cdots$ y^{b-1}
Memory Type III
at $x = x_i, y = y_i; j = 0,\cdots,a-1;$
$l = 0,\cdots,b-1.$

Fig. 4.1 Memory required for hardware implementation of the soft interpolation step.

4.3 Implementation of Soft Interpolation Step

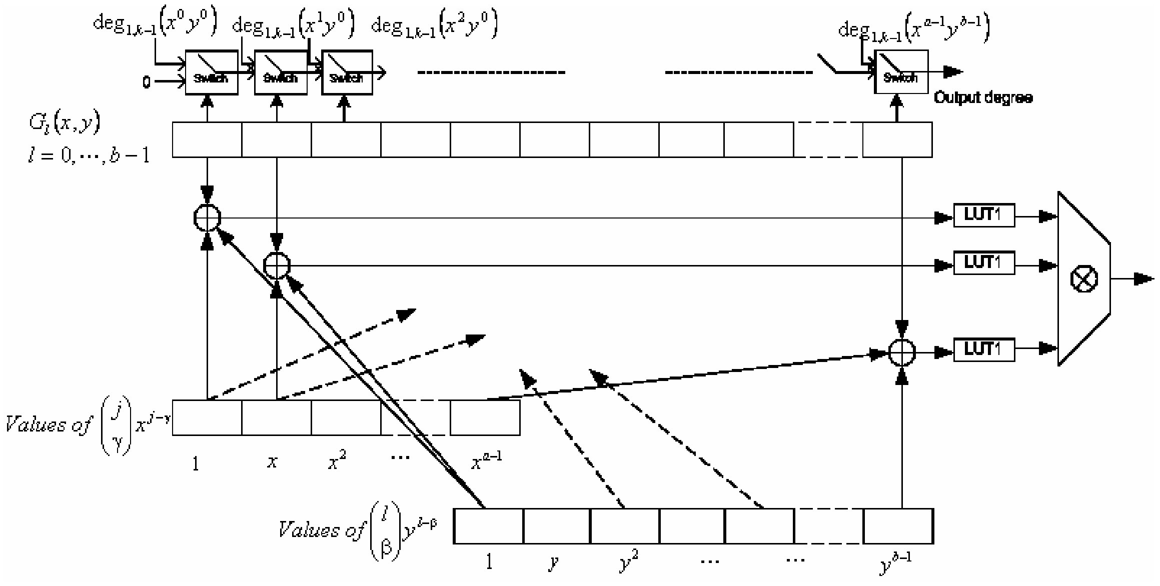
The first step of soft interpolation is to compute the HD value of each $G_l(x, y)$ with given $x = x_i, y = y_i, \gamma$ and β. Fig. 4. 2 shows an implementation diagram of Step 1 with $\oplus$ representing modulo-$(q-1)$ addition and $\otimes$ modulo-2 bitwise addition. As we can see the memory for storing $G_l(x, y)$ is segmented as

$$\left(1, x, \cdots x^{a-1}\right); \left(1, x, \cdots, x^{a-1}\right)y; \cdots ; \left(1, \cdots, x^{a-1}\right)y^{b-1},$$

and the computation of (4.2) can be further divided as:

$$G^{(\gamma,\beta)}(x_i, y_i) = \sum_{\substack{\gamma \le j \\ \beta \le l}} \binom{j}{\gamma}\binom{l}{\beta} c_{jl} x_i^{j-\gamma} y_i^{l-\beta} = \sum_{\substack{l=0 \\ \beta \le l}}^{b-1} \binom{l}{\beta} y_i^{l-\beta} \sum_{\gamma \le j} c_{jl} \binom{j}{\gamma} x_i^{j-\gamma} . \tag{4.3}$$

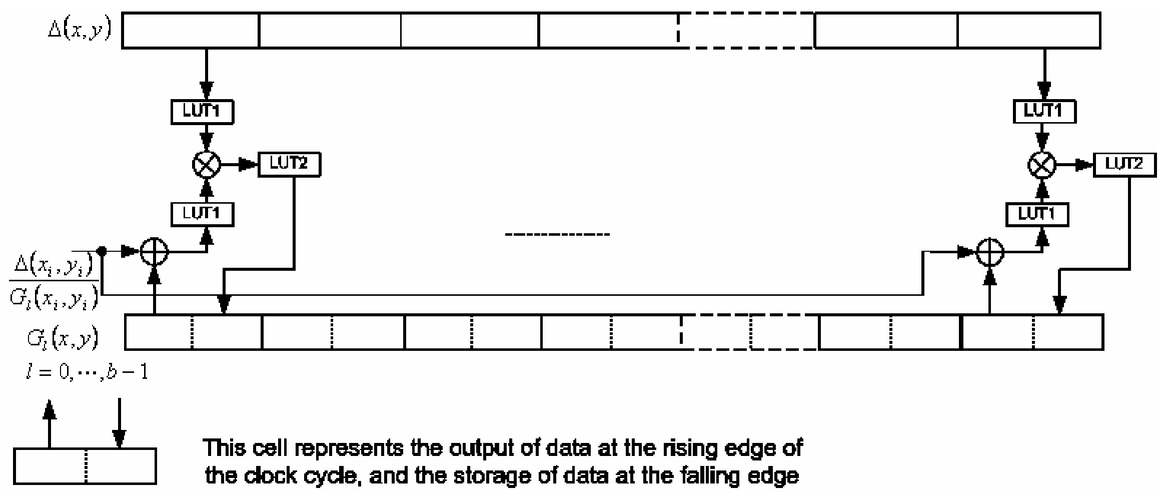
For every iteration, the values of $\binom{j}{\gamma} x_i^{j-\gamma}$ and $\binom{l}{\beta} y_i^{l-\beta}$ are updated and stored in Type-III memory. The computation of the HD value at every iteration needs: $(a + b + a \times b)$ modulo-$(q-1)$ additions and $a \times b$ LUT searches, then $a \times b$ modulo-2 bitwise additions. Notice that in mod-2 arithmetic, $\binom{j}{\gamma}$ or $\binom{l}{\beta}$ will be zero if the value is an even integer, otherwise it is one, so the actual number of computations using (4.3) will be smaller. The computation of the $(1, k-1)$-weighted degree can be realized simply by a series of switch circuits, and the degree of the highest nonzero monomial $x^j y^l$ in $G_l(x, y)$ can be obtained as shown in Fig. 4. 2.



$\deg_{1,k-1}(x^0y^0)$
$\deg_{1,k-1}(x^1y^0)$
$\deg_{1,k-1}(x^2y^0)$
$\deg_{1,k-1}(x^{a-1}y^{b-1})$
0
Switch
Output degree
$G_l(x,y)$
$l=0,\cdots,b-1$
LUT1
Values of $\binom{j}{\gamma}x^{j-\gamma}$
1 x x^2 $\cdots$ x^{a-1}
Values of $\binom{l}{\beta}y^{l-\beta}$
1 y y^2 $\cdots$ $\cdots$ y^{b-1}

Fig. 4.2 Hardware architecture for computing the Hasse derivative and $(1,k-1)$-degree polynomial $G_l(x,y)$.

The implementation of Step 2 is shown in Fig. 4.3, which requires $a \times b$ modulo- $(q-1)$ additions, $3 \times a \times b$ LUT searches and $a \times b$ modulo-2 bitwise additions for every $G_l(x,y)$ update. The computation can be achieved at the rising edge of the system clock, and the result can be restored in memory at the falling edge of the clock. The updating can be realized in parallel for all $G_l(x,y)$. The $\Delta(x,y)$ is initialized as the lowest degree $G_l(x,y)$ whose Hasse derivative is not zero, and the updating in Step 3 with $(x-x_i)\Delta(x,y)$ are just shift and add operations (See Fig. 4.4), which also requires $a \times b$ modulo-$(q-1)$ additions, $3 \times a \times b$ LUT searches and $a \times b$ modulo-2 bitwise additions.



$\Delta(x,y)$
LUT1
LUT2
LUT1
$\dfrac{\Delta(x_i,y_i)}{G_l(x_i,y_i)}$
$G_l(x,y)$
$l = 0,\cdots,b-1$
This cell represents the output of data at the rising edge of the clock cycle, and the storage of data at the falling edge

Fig. 4.3 Hardware architecture for updating polynomial $G_l(x,y)$.

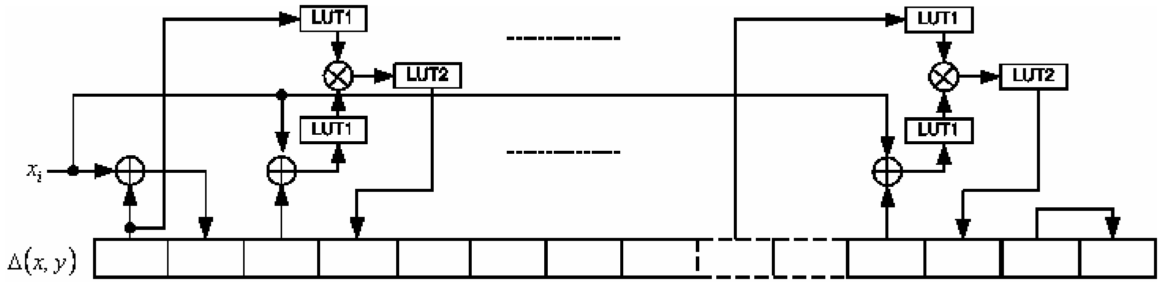


Fig. 4.4 Hardware architecture for updating polynomial $\Delta(x,y)$ with $(x-x_i)\Delta(x,y)$.

The memory needed for the three types of memory described above is $b\times a\times b\times\log q$ bits, $a\times b\times\log q$ bits and $(a+b)\times\log q$ bits, respectively. For the two LUTs, an additional $2\times q\times\log q$ bits are needed. The memory requirement for soft interpolation is approximately $(b+1)\times a\times b\times\log q+(a+b+2q)\times\log q$ bits. The memory requirement and complexity are mainly determined by the values a and b. All the computations described above are additions, and LUT searches, which are amenable to very high speed hardware implementation. This is also the case for the remaining steps in the soft-decision RS decoding algorithm.

4.4 Implementation of Soft Interpolation Step with Re-Encoding

The hardware architectures given above are based on the original GS and KV algorithms. For the reduced complexity algorithm proposed in [4]-[6], the interpolation points can be divided into two groups: one has $n-k$ points, and the interpolation of such group can be realized by the hardware described above; the other group has k points, with $y_i = 0$, and a reduced complexity hardware implementation based on (2.27) can be devised. We now assume, without loss of generality, that the k points are indexed as $i = 0, \cdots, k-1$, and $N-k$ points are indexed as $i = k, \cdots, N-1$, for notation convenience. Multiplying $\sum_{l=0}^{b-1} w_l(x) y^l$ by $(x-x_0)^{m_0-l} \cdots (x-x_{k-1})^{m_{k-1}-l}$ can be accomplished just by shift and add operations. Depending on the value $m' = m_i - l$, the $(x-x_i)^{m'}$ term can be expanded into a polynomial with two terms, if m is a power of two, or more than two terms if it is not. An illustration of the hardware needed to implement the multiplication of $w_l(x)y^l$ by $(x-x_i)^{m'}$ is given in Fig. 4.5. The shift controller stores the information of the computed coefficient and the power of x in $(x-x_i)^{m'}$. The original $w_l(x)y^l$ is multiplied by the coefficient, then shifted by an amount controlled by the power of x, and added to $\phi(x, y)$ which is initialized as zero. After no more than m iterations, we will obtain $\phi(x,y)=(x-x_i)^{m'}w_l(x)y^l$. The computation required for one segment shown in Fig. 4.5 includes $m'\times(m'+1)/2$ shifts, $m'\times 3\times a'$ LUT searches, $m'\times a'$ modulo-$(q-1)$ additions, and $m'\times a'$ modulo-2 bitwise additions. Since the bivariate polynomial in re-encoding algorithm needs to pass through points (x_i, y_i) for $i=k,\cdots,N-1$, similar to the computation in (2.25), the parameter a' can be computed as

$$a'=\sqrt{(k-1)\left(\sum_{i=k}^{N-1}m_i(m_i+1)\right)} \tag{4.4}$$

The whole circuit includes b segments, and for the transformation of

$$Q_M(x,y)=\sum_{l=0}^{b-1}w_l(x)y^l(x-x_0)^{m_0-l}(x-x_1)^{m_1-l}\cdots(x-x_{k-1})^{m_{k-1}-l}, \tag{4.5}$$

the total computation includes $\sum_{i=0}^{k-1}(m_i\times(m_i+1)/2)$ shifts, $\left(b\times\sum_{i=0}^{k-1}m_i\right)\times 3\times a'$ LUT searches, $\left(b\times\sum_{i=0}^{k-1}m_i\right)\times a'$ modulo- $(q-1)$ additions and $\left(b\times\sum_{i=0}^{k-1}m_i\right)\times a'$ modulo-2 bitwise additions, approximately. Since the value a' is smaller than a for the GS and KV algorithms and at least for k points there is no need to compute the HD values (Step 1), the re-encoding algorithm can reduce the memory requirement and computation complexity substantially, especially when the code rate is high, which is the case in magnetic recording systems. Notice that in the shift controller implementation, we would like to set the multiplicity m_i to a constant value since it is easier to implement

$$(x-x_0)^{m_0-l}\cdots(x-x_{k-1})^{m_{k-1}-l}=((x-x_0)\cdots(x-x_{k-1}))^{m_0-l}=\left(\sum_i c_i x^i\right)^{m_0-l} \tag{4.6}$$

with c_i being coefficients. Otherwise, for each segment shown in Fig. 4.5, we need to determine how many $m_i - l$ terms are less than zero, and the computation of $(x-x_0)^{m_0-l}\cdots(x-x_{k-1})^{m_{k-1}-l}$ with different m_i requires additional operations. The comparison of the total number of operations needed for implementing the soft interpolation step for the original GS or KV algorithms and for its re-encoding version are shown in Table 4.1.

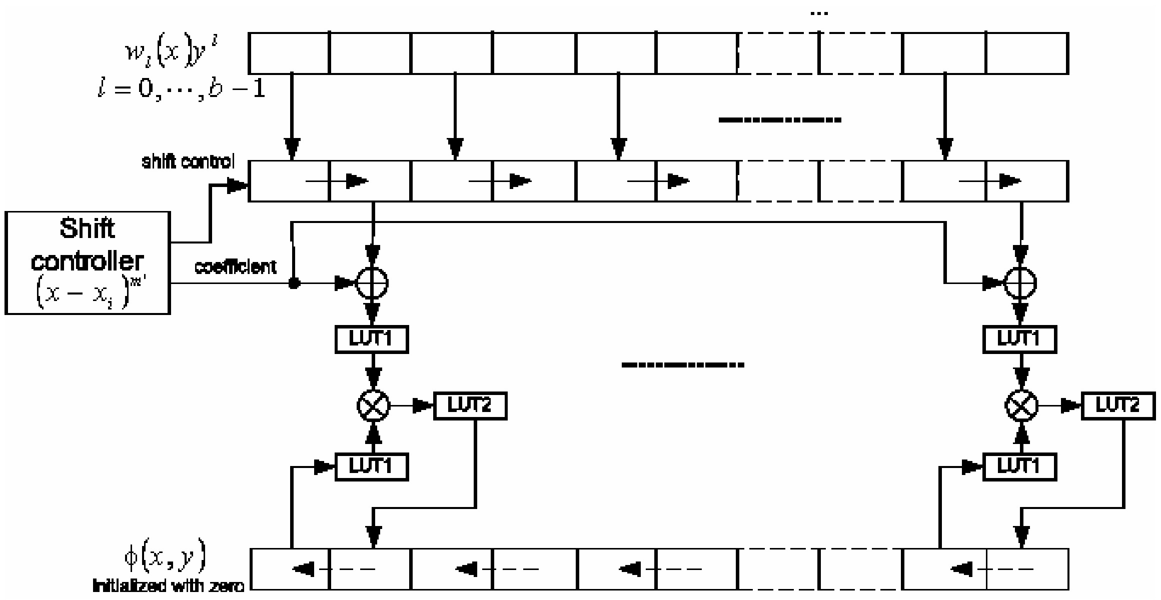


Fig. 4.5 Hardware architecture for updating one segment of memory: $w_l(x)y^l$ with $\phi(x,y)=(x-x_i)^{m'}w_l(x)y^l$.

TABLE 4.1

NUMBER OF OPERATIONS NEEDED FOR IMPLEMENTATION OF THE SOFT INTERPOLATION ALGORITHM

		Modulo-$(q-1)$ additions	Modulo-2 bitwise additions	LUT searches	
GS/KV	Step 1	$(a+b+a\times b)\sum_{i=0}^{N-1} m_i(m_i+1)/2$	$(a\times b)\sum_{i=0}^{N-1} m_i(m_i+1)/2$	$(a\times b)\sum_{i=0}^{N-1} m_i(m_i+1)/2$	
	Step 2	$(a\times b)\sum_{i=0}^{N-1} m_i(m_i+1)/2$	$(a\times b)\sum_{i=0}^{N-1} m_i(m_i+1)/2$	$(3\times a\times b)\sum_{i=0}^{N-1} m_i(m_i+1)/2$	
	Step 3	$(a\times b)\sum_{i=0}^{N-1} m_i(m_i+1)/2$	$(a\times b)\sum_{i=0}^{N-1} m_i(m_i+1)/2$	$(3\times a\times b)\sum_{i=0}^{N-1} m_i(m_i+1)/2$	
		Modulo-$(q-1)$ additions	Modulo-2 bitwise additions	LUT searches	Shifts
Re-encoding	Step 1	$(a'+b+a'\times b)\sum_{i=k}^{N-1} m_i(m_i+1)/2$	$(a'\times b)\sum_{i=k}^{N-1} m_i(m_i+1)/2$	$(a'\times b)\sum_{i=k}^{N-1} m_i(m_i+1)/2$	0
	Step 2	$(a'\times b)\sum_{i=k}^{N-1} m_i(m_i+1)/2$	$(a'\times b)\sum_{i=k}^{N-1} m_i(m_i+1)/2$	$(3\times a'\times b)\sum_{i=k}^{N-1} m_i(m_i+1)/2$	0
	Step 3	$(a'\times b)\sum_{i=k}^{N-1} m_i(m_i+1)/2$	$(a'\times b)\sum_{i=k}^{N-1} m_i(m_i+1)/2$	$(3\times a'\times b)\sum_{i=k}^{N-1} m_i(m_i+1)/2$	0
	Step 4	$\left(b\times\sum_{i=0}^{k-1} m_i\right)a'$	$\left(b\times\sum_{i=0}^{k-1} m_i\right)a'$	$\left(b\times\sum_{i=0}^{k-1} m_i\right)\times 3a'$	$\sum_{i=0}^{k-1}\frac{m_i(m_i+1)}{2}$

$$a' = \sqrt{(k-1)\left(\sum_{i=k}^{N-1} m_i(m_i+1)\right)},\ a = \sqrt{(k-1)\left(\sum_{i=0}^{N-1} m_i(m_i+1)\right)}\ \text{and}\ b = \sqrt{\left(\sum_{i=0}^{N-1} m_i(m_i+1)\right)/(k-1)}$$

4.5 Summary

In this charpter, we discussed the memory requirement and hardware architectures for implementation of the soft interpolation step, which is a major step in the GS and KV algorithms. Also, we investigated the complexity reduction of the interpolation step when the re-encoding algorithm is considered.

Bibliography

[1] R. Koetter and A. Vardy, "Algebraic soft-decision decoding of Reed-Solomon codes", *IEEE Trans. Inform. Theory*, vol. 49, pp. 2809-2825, Nov. 2003.

[2] R. R. Nielsen and T. Hoholdt, “Decoding Reed-Solomon codes beyond half the minimum distance,” in *Coding Theory, Cryptography and Related Areas*, J. Buchmann, T. Hoholdt, T. Stichtenoth, and H. Tapia-Recillas, Eds., Berlin, Germany: Springer-Verlag, 2000, pp. 221-236.

[3] V. Guruswami and M. Sudan. “Improved decoding of Reed-Solomon and algebraic-geometric codes,” *IEEE Trans. Inform. Theory*, vol. 45, pp.1757-1767, Sept. 1999.

[4] W. J. Gross, F. R. Kschischang, R. Koetter and P. G. Gulak, “A VLSI architecture for interpolation in soft-decision list decoding of Reed-Solomon codes,” in *Proc. IEEE Workshop Signal Processing Syst.*, pp. 39-44, Oct. 2002.

[5] W. J. Gross, F. R. Kschischang, R. Koetter and P. G. Gulak, “Towards a VLSI architecture for interpolation-based soft-decision Reed-Solomon decoders,” to appear in *J. VLSI Signal Processing*, 2003.

[6] H. Xia and J.R. Cruz, “Application of soft-decision Reed-Solomon decoding on magnetic recording channels,” *IEEE Trans. Magn.*, vol. 40, pp. 3419-3430, Sept. 2004.

Chapter 5

Forward Recursive Algorithms for Soft-Decision Reed-Solomon Decoding

5.1 Introduction

In soft-decision RS decoding algorithms such as the KV algorithm [1], a bivariate polynomial $Q(x, y)$ will be generated to pass through a number of point pairs $\{x_i, y_i\}$ with different multiplicities m_i. Several algorithms [1]-[5] have been proposed to determine the points pairs $\{x_i, y_i\}$ as well as the corresponding multiplicity m_i. The way the multiplicity is determined is by maximizing the probability $\Pr\{y_i = j \mid x_i, \Pi\}$, given the received channel reliability information Π. From Chapter 4 we know that the interpolation step starts with a set of bivariate polynomials $\mathbf{Q}$, and after this set satisfies the N constraints, the one with the smallest weight degree polynomial will be selected as output $Q(x, y)$.

Let $\mathbf{Q}^{(i)} = \left(Q_0^{(i)}, Q_1^{(i)} \cdots, Q_{L-1}^{(i)}\right)^{\mathbf{T}}$ be a polynomial vector initialized as

$$\mathbf{Q}^{(0)} = \left(y^0, y^1, \cdots, y^{L-1}\right)^{\mathbf{T}} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} 1 \\ y \\ \vdots \\ y^{L-1} \end{bmatrix} \tag{5.1}$$

where $\mathbf{T}$ represents transpose, and $i = 1$ to N. The polynomial interpolation step can be implemented by generating a set of polynomials $\mathbf{Q}^{(i)}$ passing through the point pairs $\{x_i, y_i\}$ based on the linear operation of entries in $\mathbf{Q}^{(i-1)}$. Therefore, the interpolation step can be computed recursively as

$$\mathbf{Q}^{(i)} = \mathbf{W}^{(i)}\mathbf{Q}^{(i-1)}, \tag{5.2}$$

or explicitly as

$$\begin{bmatrix} Q_0^{(i)} \\ Q_1^{(i)} \\ \vdots \\ Q_{L-1}^{(i)} \end{bmatrix} = \begin{bmatrix} w_{0,0}(x) & w_{0,1}(x) & \cdots & w_{0,L-1}(x) \\ w_{1,0}(x) & w_{1,1}(x) & \cdots & w_{1,L-1}(x) \\ \vdots & \vdots & \vdots & \vdots \\ w_{L-1,0}(x) & w_{L-1,2}(x) & \cdots & w_{L-1,L-1}(x) \end{bmatrix} \begin{bmatrix} Q_0^{(i-1)} \\ Q_1^{(i-1)} \\ \vdots \\ Q_{L-1}^{(i-1)} \end{bmatrix}, \tag{5.3}$$

where $\mathbf{W}^{(i)} = [w_{l,t}(x)]$, which is initialized as the identity matrix $\mathbf{I}$ and $w_{l,t}(x)$ is a univariate polynomial with $l,t = 0$ to $L-1$.

Therefore,

$$\mathbf{Q}^{(N)} = \mathbf{W}^{(N)}\mathbf{W}^{(N\text{-}1)}\cdots\mathbf{W}^{(1)}\mathbf{Q}^{(0)} = \prod_{i=1}^{N}\mathbf{W}^{(i)}\mathbf{Q}^{(0)}, \tag{5.4}$$

where each $\mathbf{W}^{(i)}$ is related to an interpolation point pair with a certain multiplicity (constraint) m_i. The final output is selected from the vector $\mathbf{Q}^{(N)}$. Since each $\mathbf{W}^{(i)}$ is an $L \times L$ matrix, the order in which the multiplication of the various $\mathbf{W}^{(i)}$ is performed can be changed arbitrarily (that is the order of the point pairs can be changed arbitrarily), which make an efficient parallel implementation of the product $\prod_{i=1}^{N}\mathbf{W}^{(i)}$ becomes possible. Based on the observation above, Feng and Giraud shed some light with a divide-and-conquer approach [6] to the GS algorithm, which only fits for multiplicity $m = 1$. However, when multiplicity larger than one, we need to compute the Hasse derivative to generate $\mathbf{W}^{(i)}$ based on $\mathbf{W}^{(i-1)}$, the divide-and-conquer approach can not be used. More work remains to be done to achieve an efficient parallel implementation for multiplicities larger than one.

5.2 Reliability-Based Forward Recursive Algorithms

The KV algorithm [1], provides a method of generating the point pairs and their corresponding multiplicities based on the channel reliability information. As described above, the order of the interpolation points is arbitrary, but it is desired that the generated bivariate polynomial pass through more correct points $\{x_i,y_i\}$ with multiplicities m_i, than through erroneous points. We do not know which points $\{x_i,y_i\}$ are correct, however, we do know that more reliable symbols are more likely to be correct. Suppose that we generate a sequence of N point pairs $\{x_i,y_i\}$, and we order the sequence $\{x_i,y_i\}$ in terms of the decreasing reliability of y_i. Then while generating the bivariate polynomial passing through the sorted sequence $\{x'_i, y'_i\}_{i=1}^{N}$, we perform the factorization step every time we process an additional point in the sequence to determine if we have the correct decoding answer. The advantage of this approach, compared to the original KV algorithm, is that the more reliable points are processed first leading to an early convergence to a correct decoding answer.

The detailed forward recursive algorithm is described as follows. From the received channel output information, we can generate a sequence of point pairs $\{x_i, y_i\}_{i=1}^{N}$ with corresponding multiplicities m_i.

Step 1: Sort the sequence of point pairs $\{x_i, y_i\}_{i=1}^{N}$ in decreasing order of their reliability, to obtain a sorted sequence of point pairs $\{x'_i, y'_i\}_{i=1}^{N}$;

Step 2: Perform the recursive interpolation described above to generate a bivariate polynomial $Q(x, y)$ passing through the first p point pairs of $\{x'_i, y'_i\}_{i=1}^{N}$ with their corresponding multiplicities m'_t;

Step 3: Perform the factorization of this intermediate polynomial $Q(x, y)$ to get a codeword candidate $\hat{\mathbf{c}}$. If the stopping criterion has been satisfied, output $\hat{\mathbf{c}}$, go to End;

Step 4: Recursively let $Q(x, y)$ pass through one more point in $\{x'_i, y'_i\}_{i=1}^{N}$, and then go to Step 3 until all points have been used;

End.

The value of p is set to be larger than $k = \deg(f(x))$ since in order to fully determine a polynomial $y - f(x)$, we need to let the bivariate polynomial pass through $p > k$ points. The stopping criterion can be implemented using (2.24) by replacing $\mathbf{c}$ with the codeword candidate $\hat{\mathbf{c}}$, a CRC check, or a distance computation. The bivariate polynomial $Q(x, y)$ is sometimes denoted as $Q_M(x, y)$ to indicate that the polynomial is generated from the multiplicity or constraint matrix M. The crux of the algorithm proposed above is that instead of using reliability information once, as in the KV algorithm, we use it twice: once for multiplicity computation, and again for interpolation order determination.

The reliability-based forward recursive algorithm described, performs multiple factorization steps in contrast to the GS or KV algorithms, which perform only a single factorization step. However, as illustrated by the example shown in Fig. 5.1, the average number of trials is approximately $(n-k)/2$, and for large signal-to-noise ratios (SNRs), the number of trials becomes even smaller, in some cases as few as two. Given the substantial complexity reduction afforded by the recursive interpolation, since we do not need to let the bivariate polynomial pass through all the point pairs $\{x_i, y_i\}_{i=1}^{N}$ before we can find the correct answer, the slight increase in complexity due to the multiple factorizations is negligible. For low SNRs, since for each interpolation constraint, a factorization trial needs to be executed, it is required that the time complexity of the factorization step be less or equal than the time complexity of the interpolation for every constraint. The complexity of the polynomial interpolation proposed in [7], [8] is $O(n^3)$, which means that for each constraint the interpolation has time complexity of $O(n^2)$. For the factorization, the algorithm used in [9] gives a time complexity of $O((\ell \log^2 \ell)k(n + \ell \log q))$ with ℓ representing the highest degree of y in $Q(x, y)$. From the naïve interpolation algorithm using Gaussian elimination in [8] we can see that the interpolation time complexity for one constraint is close to the time complexity of the factorization algorithm [9]. However, more research needs to be done to find lower complexity interpolation algorithms [9], which will drive the need for lower complexity factorization algorithms to be used by these forward recursive algorithms, otherwise, extra hardware will be needed for their implementation. An alternate approach is to perform factorization only after several interpolations of the intermediate polynomial have been completed.

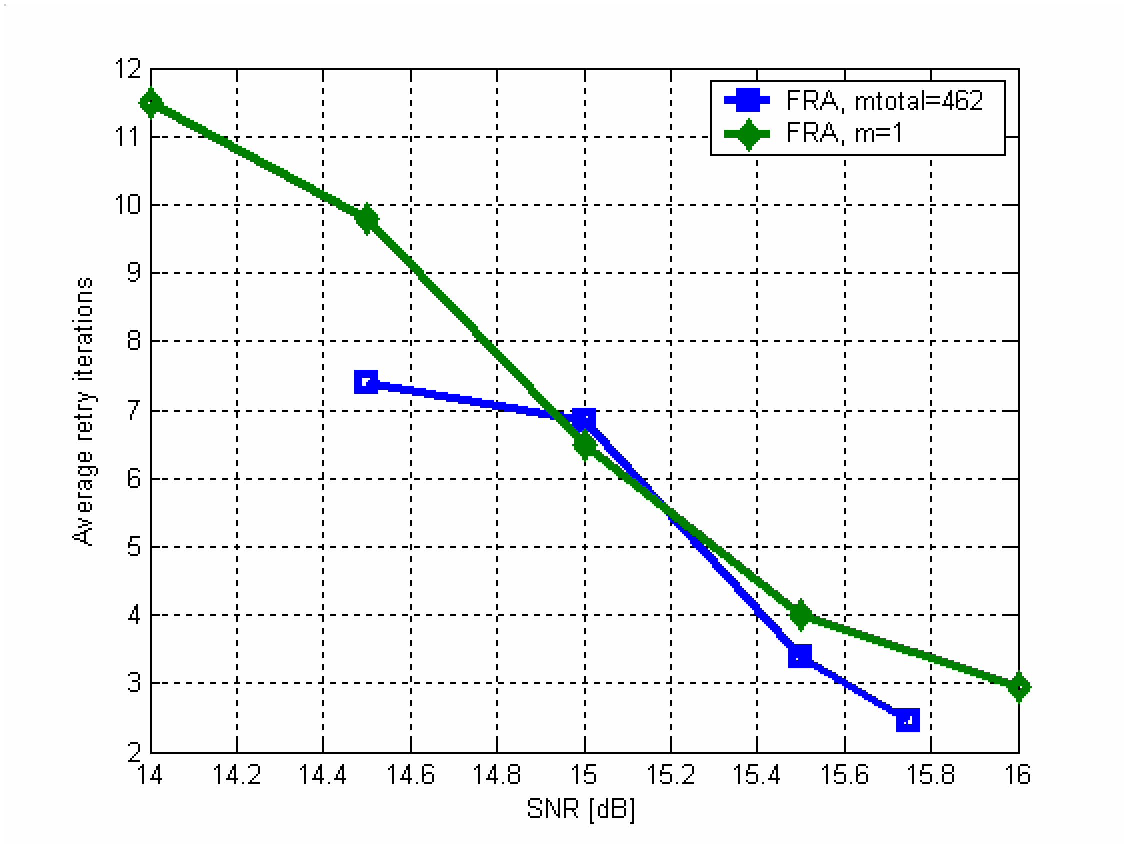


Fig. 5.1. Average number of retry iterations for the forward recursive algorithm and an RS (186, 172) code on an equalized MEEPR4 magnetic recording channel.

5.2.1 Chase-Type Variation

Other ways of utilizing channel reliability information for decoding, such as in the Chase algorithm [10], can be used to generate particular forward recursive algorithms. In the forward recursive algorithm described above, we can flip the least reliable symbols, which will lead to a forward recursive Chase-type algorithm. However, if the order of $GF(q)$ is very large, the flipping of symbols will be impractical, since a large number of test patterns would have to be generated. Instead of using conventional symbol flipping, we use bit flipping. The forward recursive algorithm can be modified as follows. First we find the $p^{'}$ least reliable bits in the received sequence. For notation simplicity and without loss of generality, we assume that each symbol contains only one of these least reliable bits. For those least reliable bits, we will "softly" flip them by assigning a large probability of being a "0" or a "1", which will generate $2^{p^{'}}$ different test patterns (in symbols). Then let the bivariate polynomial pass through $p = n - p^{'}$ points (symbols) to generate an intermediate polynomial. This intermediate polynomial can then pass through the rest of the symbols according to the test patterns, which will give $2^{p^{'}}$ decoding candidates, among which the most likely output will be selected as the decoding result. In [11], we refer to such a "soft" flipping algorithm as a soft Chase algorithm.

In summary, the sequence of points $\{(x_i, y_i), m_i\}_{i=1}^{N}$ for polynomial interpolation can be divided into three groups: Group 1 includes k points $\{(x_i, y_i), m_i\}$, which have the highest symbol reliabilities; Group 2 includes p' points $\{(x_i, y_i), m_i\}$, whose symbols contain bits involved in bit-flipping, and Group 3 contains the remaining symbols. Then an intermediate bivariate polynomial $Q'_M(x, y)$ is generated, which passes through the points in Groups 1 and 3, and is stored for future use. For each different test pattern we let $Q'_M(x, y)$ pass through the appropriate points in Group 2 to get $Q_M(x, y)$. Since the value of p' is usually very small, the complexity of the generation of the bivariate polynomial $Q_M(x, y)$ does not increase very much, while the performance improvement can be very significant.

5.2.2 Reduced-Complexity Implementation

Similar to the re-encoded Chase algorithm in Chapter 3, here the forward recursive algorithm and its variations described above can also be combined with the re-encoding algorithm [12] to further reduce the decoding complexity. The re-encoding algorithm divides the interpolation step into three sub-steps: re-encoding, reduced-complexity interpolation, and transformation. Afterwards, a factorization step is executed to find the decoding output. In [13], Ahmed *et al.* further reduced the decoding complexity by applying the factorization step without the transformation sub-step with the help of a conventional Berlekamp-Massey hard-decision decoder [14]. The motivation for combining the forward recursive algorithm with the re-encoding algorithm (in its original version in [5] as well as the reduced-complexity factorization version in [13]), is again based on the same requirement of first finding the k most reliable symbols from the channel output information. For these k symbols (Group 1), a reduced complexity interpolation algorithm can be used to generate an intermediate bivariate polynomial, and then a factorization step is executed to see if we can find the correct output. If not (which can be checked using a certain criterion), the forward recursive algorithm processes the remaining points.

If we consider the Chase variation, the bivariate polynomial is generated to pass through all symbols in decreasing reliability order until it reaches the p'-th least reliable symbol, from there on, the bivariate polynomial passes through different symbols (Group 2) according to the different test patterns, which generates an expanded list of decoding candidates, and the most reliable candidate is output as the decoding result. This reduced-complexity implementation of the interpolation algorithm reduces the time complexity to approximately $(n-k)/n$ of the conventional interpolation algorithm proposed in [1].

5.3 Connections to Other Decoding Algorithms

The forward recursive algorithm is an efficient general algorithm, which defaults to the GS and KV algorithms, with an appropriate selection of parameters. In addition, it can be viewed as a "forward" generalized minimum distance (GMD) algorithm.

5.3.1 Guruswami-Sudan Algorithm

Given the received channel reliability information Π, a hard-decision vector $\mathbf{y}$ can be generated. Guruswami and Sudan [7] proposed an algorithm which groups each entry y_i in the vector $\mathbf{y}$ with the corresponding x_i, and an assigned constant multiplicity value m, to generate a bivariate polynomial $Q(x,y)$ passing through the sequence of these n point pairs $\{x_i,y_i\}$ with multiplicity m, and performs a single factorization, to output a list of codeword candidates. If we set $p=n$ in our forward recursive algorithm, we obtain the GS algorithm as a special case. Of course, if we generate multiplicities according to the channel reliability Π, the forward recursive algorithm becomes the KV algorithm, which can be considered as a special case of the GS algorithm.

5.3.2 Generalized Minimum Distance Algorithm

Besides converting the channel reliability information into some multiplicity values as it was done in [1], there are other ways of using the channel information to assist the decoding process, as in the GMD algorithm proposed by Forney [15]. The basic idea behind Forney's GMD algorithm is that for hard-decision RS decoding algorithms, if we flag some unreliable symbols as erasures, we can further improve the decoding performance, given that the error-erasure correction capability is $2e + f < n - k + 1$, where e is the number of errors, and f is the number of erasures in a received channel output sequence. In the GMD algorithm, the number of erasures is increased at every step, i.e., each time we flag two more erasures in the received vector $\mathbf{y}$ if the previous trial did not give a correct output. In this context the GMD algorithm can be viewed as a "backward" algorithm.

The forward recursive algorithm we propose in this paper can be viewed as a "forward" GMD-type algorithm. By letting the bivariate polynomial pass through the p most reliable points, and perform the factorization step to find the correct decoding answer, we can think of this algorithm as a GMD-type algorithm with $N - p$ erasures. When we let the bivariate polynomial pass through one more point, we execute a GMD-type algorithm with $N-p-1$ erasures. The major difference between this forward recursive algorithm and the GMD algorithm is that the GMD algorithm increases the number of erasures at every step, while the forward recursive algorithm decreases the number of unreliable points. Also, besides using the channel reliability information to order the channel output symbol sequence, the forward recursive algorithm also uses the reliability information for generating variable multiplicity values for the interpolation, which further improves the decoding performance compared to the GMD or KV algorithms.

5.4 Performance Analysis

One of the key steps in algebraic soft-decision decoding of RS codes is the multiplicity computation, i.e., how to generate the best multiplicity matrix M given the channel reliability information. The nonzero entries in the multiplicity matrix are in fact the interpolation constraints which the generated bivariate polynomial $Q_M(x, y)$ should satisfy. However, because of the nature of polynomial interpolation, those constraints in matrix M are satisfied one at a time, and if we let the bivariate polynomial pass only through a subset of the non-zero points in M, then the rest of the symbols in the sequence will be taken as erasures. Therefore, interpolation based algorithms, such as the forward recursive algorithm described in this dissertation can be viewed as erasure-and-trial algorithms. In fact, if we start with a multiplicity matrix M_1 with a single nonzero entry and all other nonzero entries erased, each time the bivariate polynomial $Q_M(x, y)$ satisfies an interpolation constraint is equivalent to adding one more nonzero entry to the multiplicity matrix M_1, which we denote as matrix M_2. By carrying out this process we finally obtain matrix M with all the desired multiplicities for interpolation. We generate a set of matrices $\{M_i \mid M_i \subset M_{i+1}, i = 1 \; to \; N\}$ with $M_N = M$ (here the include symbol $\subset$ means the nonzero entries in M_i are a subset of the nonzero entries in M_{i+1}). Given a matrix M_i, $S_i = M - M_i$ is the erasure matrix whose nonzero entries correspond to the number of intermediate polynomial constraints that have not yet been satisfied. Therefore, for each step in the forward recursive algorithm, (2.24) becomes:

$$S_{M_i}(\mathrm{c}) > \deg_{1,k-1}\left(Q_{M_i}(x, y)\right), \tag{5.5}$$

where $S_{M_i}(\mathrm{c})$ is the score of an RS codeword **c** with given multiplicity matrix M_i and $\deg_{1,k-1}\left(Q_{M_i}(x, y)\right)$ is the weighted degree of bivariate polynomial $Q_M(x, y)$. Since $M_i = M - S_i$, from the definition of score we have

$$\begin{aligned} S_{M_i}(\mathrm{c}) &= \sum_{i=1}^{N} m_i - \sum_{j=1}^{e} e_j - \sum_{l=1}^{f} s_l \\ &= m_{total} - \sum_{j=1}^{e} e_j - \sum_{l=1}^{f} s_l, \end{aligned} \tag{5.6}$$

where $\sum_{i=1}^{N} m_i$ is equal to the total multiplicity m_{total}, which is the summation of the nonzero entries $\{m_i\}_{i=1}^{N}$ in M; $\sum_{l=1}^{f} s_l$ is the summation of all nonzero entries in

S_i, which can be denoted as $\{s_l\}_{l=1}^{f} \in S_i \subset M$. Among those nonzero entries in $M_i = M - S_i$, some of them correspond to erroneous points, and $\sum_{j=1}^{e} e_j$ is the summation of the nonzero erroneous entries $\{e_j\}_{j=1}^{e}$ in M_i. In addition, let us define another matrix R_i which contains all the "correct" entries in M_i, which is denoted as $\{r_t\}_{t=1}^{g}$. The right-hand side of (5.5) is given by:

$$\begin{aligned}\deg_{1,k-1}\left(Q_{M_i}(x,y)\right) &= \Delta(C-F) \\ &= \Delta\left(\frac{1}{2}\sum_{i=1}^{N} m_t(m_t+1) - \frac{1}{2}\sum_{l=1}^{f} s_l(s_l+1)\right) \\ &= \Delta\left(\frac{1}{2}\sum_{t=0}^{g} r_t(r_t+1) + \frac{1}{2}\sum_{j=1}^{e} e_j(e_j+1)\right).\end{aligned} \tag{5.7}$$

According to Corollary 5 in [1], (5.7) is bounded by $\Delta(C-F) < \sqrt{2(k-1)(C-F)}$, where C represents the cost of the whole multiplicity matrix, and F represents the cost of the erasure matrix S_i, that is $F = \frac{1}{2}\sum_{l=1}^{f} s_l(s_l+1)$. For correct decoding, the score given in (5.6), must be larger than the weighted degree given in (5.7), or

$$\begin{aligned}m_{total} - \sum_{j=1}^{e} e_j - \sum_{l=1}^{f} s_l &= \sum_{t=1}^{g} r_t \\ &> \sqrt{2(k-1)(C-F)} \\ &= \sqrt{(k-1)\left(\sum_{i=1}^{N} m_i(m_i+1) - \sum_{l=1}^{f} s_l(s_l+1)\right)} \\ &= \sqrt{(k-1)\left(\sum_{t=1}^{g} r_t(r_t+1) + \sum_{j=1}^{e} e_j(e_j+1)\right)}\end{aligned} \tag{5.8}$$

which can be further simplified to

$$\sum_{j=1}^{e} e_j\left(e_j+1\right) < \frac{1}{k-1}\left(\sum_{t=1}^{g} r_t\right)^2 - \sum_{t=1}^{g} r_t\left(r_t+1\right). \tag{5.9}$$

Given a multiplicity matrix M_i, as long as the "correct" entries r_t and "erroneous" entries e_j satisfy (5.9), correct decoding will occur.

Consider the error-correction capability of the GS algorithm with constant multiplicity m and error-only decoding, using (5.8), we have

$$\sum_{t=1}^{g} m > \sqrt{(k-1)\left(\sum_{j=1}^{e} m(m+1) + \sum_{t=2}^{g} m(m+1)\right)}, \tag{5.10}$$

that is

$$(n-e)m > \sqrt{(k-1)nm(m+1)},$$

$$e < n - \sqrt{(k-1)n(m+1)/m}, \tag{5.11}$$

so as $m \to \infty$, we obtain the error-correction bound in [2] as $e < n - \sqrt{(k-1)n}$, i.e., the maximum number of errors that can be corrected by the GS algorithm. When an error-and-erasure decoding algorithm is considered, the bound becomes $e < (n-f) - \sqrt{(k-1)(n-f)}$.

As originally discussed in [7], finding a good multiplicity for interpolation is always a problem. The performance improvement obtained by soft-decision RS decoding algorithms is a direct consequence of utilizing the channel output information to find the best multiplicity matrix, i.e., finding the *best multiplicity computation method*. In reality, we only have the received reliability matrix $\Pi = \lfloor \pi_{ij} \rfloor$, and an intuitive way to compute the multiplicity matrix is to use a large enough scalar λ to multiply reliability matrix, which leads to $\lfloor \lambda \Pi \rfloor = M$. Assume that π_{e_j} corresponds to the reliability of an "erroneous point" in the reliability matrix, and π_{r_t} corresponds to the reliability of a "correct point". Then (5.9) becomes

$$\sum_{j=1}^{e} \lfloor \lambda \pi_{e_j} \rfloor \left(\lfloor \lambda \pi_{e_j} \rfloor + 1 \right) < \frac{1}{k-1} \left(\sum_{t=1}^{g} \lfloor \lambda \pi_{r_t} \rfloor \right)^2 - \sum_{t=1}^{g} \lfloor \lambda \pi_{r_t} \rfloor \left(\lfloor \lambda \pi_{r_t} \rfloor + 1 \right), \tag{5.12}$$

and for $\lambda \to \infty$, we have

$$\lim_{\lambda \to \infty} \sum_{j=1}^{e} \lfloor \lambda \pi_{e_j} \rfloor \left(\lfloor \lambda \pi_{e_j} \rfloor + 1 \right)$$
$$< \lim_{\lambda \to \infty} \frac{1}{k-1} \left(\sum_{t=0}^{g} \lfloor \lambda \pi_{r_t} \rfloor \right)^2 - \sum_{t=1}^{g} \lfloor \lambda \pi_{r_t} \rfloor \left(\lfloor \lambda \pi_{r_t} \rfloor + 1 \right),$$

that is

$$\sum_{j=1}^{e} \left(\pi_{e_j} \right)^2 < \frac{1}{k-1} \left(\sum_{t=1}^{g} \pi_{r_t} \right)^2 - \sum_{t=1}^{g} \left(\pi_{r_t} \right)^2, \tag{5.13}$$

with $e = nq - n$, and $g = n$. Equation (5.13) corresponds to (28) in [2], and describes the asymptotic performance when the total multiplicity m_{total} goes to infinity, which requires an infinite decoding complexity. A lot of research [2]-[5] has been done on finding the *best multiplicity computation method* given a fixed moderate value for m_{total}, to maximize the correct decoding probability $\Pr\{S_M > \Delta(M) \mid \Pi, m_{total}\}$ or to minimize the decoding failure probability $\Pr\{S_M < \Delta(M) \mid \Pi, m_{total}\}$. Notice that as m_{total} becomes larger, we will include more and more "erroneous points" than "correct points" from the reliability matrix, no matter how good the multiplicity computation method is, since we have $nq - n$ "erroneous points" and only n "correct points" in Π. Therefore an increase in the total multiplicity does not always lead to a successful decoding, and error-and-erasure decoding algorithms such as the forward recursive algorithm are better suited to the nature of interpolation-based soft-decision RS decoding. The technique of increasing the multiplicity and trial, leads to the decoding bound

$$\Pr\{S_M < \Delta(M)\} = \prod_i \Pr\{S_{M_i} < \Delta(M_i) \mid \Pi, m_{total}\}, \tag{5.14}$$

regardless of which *multiplicity computation method* is used.

5.5 Performance Evaluation

The performance of the forward recursive algorithm is compared to hard-decision RS decoding and the KV algorithm on an equalized MEEPR4 channel for a shortened RS(186, 172) code, with rate R=0.925, and the GS algorithm decoding bound $n - \sqrt{n(k-1)}$, which is the same as the half minimum distance of the RS code. Fig. 5.2 shows that the forward recursive algorithm with constant multiplicity $m = 1$ performs 0.4-dB better than traditional hard-decision RS decoding algorithms at a frame-error rate (FER) of 10^{-4}, even better than the KV algorithm with total multiplicity $m_{total} = 462$, corresponding to an average multiplicity of two.

Furthermore, the performance of the forward recursive algorithm with the same multiplicity matrix generated by the KV algorithm with $m_{total} = 462$ is better than the asymptotic KV decoding bound [1], which corresponds to a total multiplicity of $m_{total} = \infty$. The performance improvement is obtained by erasing the least reliable entries in the multiplicity matrix, which in turn prevents the bivariate polynomial from passing through the corresponding unreliable points. The complexity increase for the forward recursive algorithm due to multiple factorization steps can be ignored since as Fig. 5.1 shows a decrease in the number of decoding trials at high SNR, is far outweighed by the complexity reduction because of fewer point pairs used in the interpolation of the bivariate polynomial $Q_M(x, y)$. We also show the decoding performance of the same RS code on an AWGN channel and the total number of decoding trials in Figs. 5.3 and 5.4, respectively, which is 0.1-dB better than the KV algorithm at FER=10^{-5}.

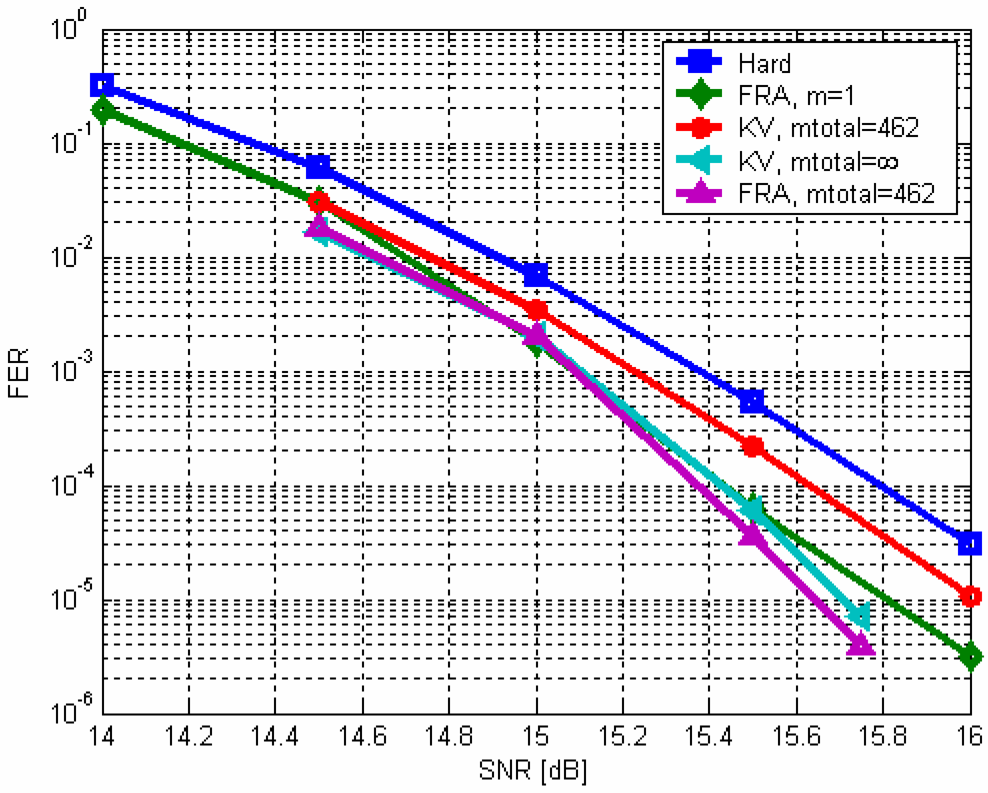


Fig. 5.2. Comparison of different decoding algorithms for the RS (186, 172) code on an equalized MEEPR4 magnetic recording channel.

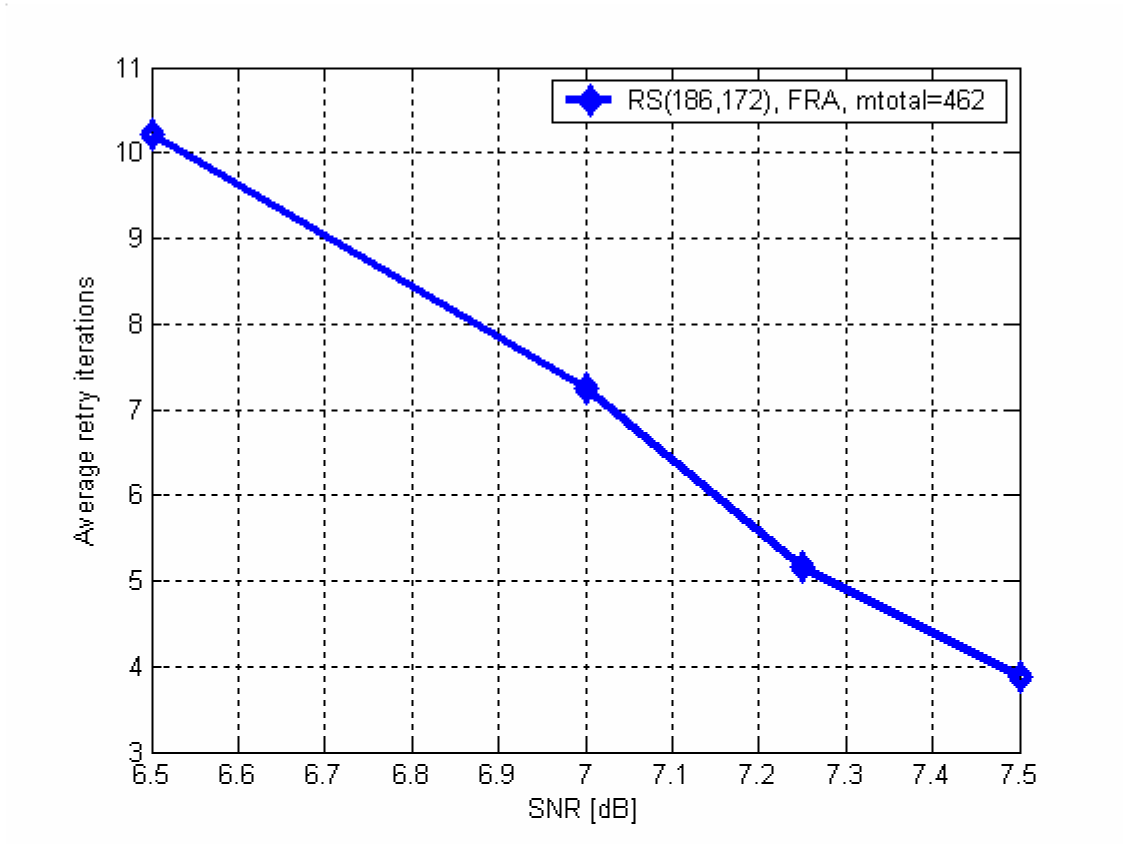


Fig. 5.3. Average number of retry iterations for the forward recursive algorithm on the RS (186, 172) code on an AWGN channel.

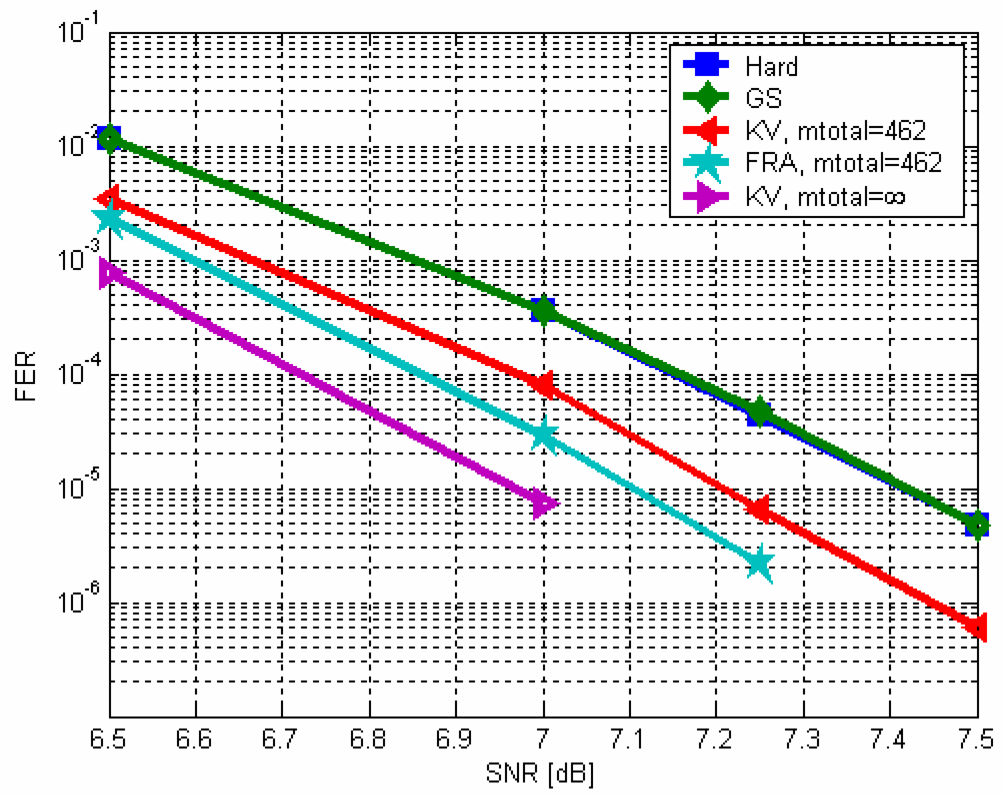


Fig. 5.4. Comparison of different decoding algorithms for the RS (186, 172) code on an AWGN channel.

5.6 Summary

In Chapter 5, by using the channel reliability information not only for generating the multiplicity matrix but also to determine the interpolation order, we developed an efficient reliability-based forward recursive algorithm and its variations for algebraic list-decoding of RS codes based on polynomial interpolation, which exhibit improved performance over the original GS and KV algorithms. Also, we reduced the decoding time latency by first processing the most reliable interpolation points leading to a fast convergence to the correct decoding answer. The proposed algorithm is particularly attractive for magnetic recording systems which operate at high SNR.

Bibliography

[1] R. Koetter and A. Vardy, “Algebraic soft-decision decoding of Reed-Solomon codes”, *IEEE Trans. Inform. Theory*, vol. 49, pp. 2809-2825, Nov. 2003.

[2] H. Xia and J. R. Cruz, “Soft-decision decoding of Reed-Solomon codes on magnetic recording channels with erasures,” in *Proc. IEEE Int. Commun. Conf.*, pp. 2909-2913, 2003.

[3] W.J. Gross, F. R. Kschischang, R. Koetter and P. G. Gulak, “Simulation results for algebraic soft-decision decoding of Reed-Solomon codes,” in *Proc. 21st Biennial Symp. Commun.*, pp. 356-360, 2002.

[4] F. Parvaresh, and A. Vardy, “Mulitplicity assignments for algebraic soft-decoding of Reed-Solomon codes,” in *Proc. IEEE Inter. Symp. Inform. Theory*, pp. 205, 2003.

[5] H. Xia and J. R. Cruz, “Application of soft-decision Reed-Solomon decoding to magnetic recording channels,” *IEEE Trans. Magn.*, vol. 40, pp. 3419-3430, Sept. 2004.

[6] G.-L. Feng and X. Giraud, “Fast algorithms in Sudan decoding procedure for Reed-Solomon codes,” submitted to *IEEE Trans. Inform. Theory*, 2001.

[7] V. Guruswami and M. Sudan. "Improved decoding of Reed-Solomon and algebraic-geometric codes," *IEEE Trans. Inform. Theory*, vol. 45, pp. 1757-1767, Sept. 1999.

[8] R. R. Nielsen and T. Hoholdt, "Decoding Reed-Solomon codes beyond half the minimum distance," in *Coding Theory, Cryptography and Related Areas*, J. Buchmann, T. Hoholdt, T. Stichtenoth, and H. Tapia-Recillas, Eds., Berlin, Germany: Springer-Verlag, 2000, pp. 221-236.

[9] R. M. Roth and G. Ruckenstein, "Efficient decoding of Reed-Solomon codes beyond half the minimum distance," *IEEE Trans. Inform. Theory*, vol. 46, pp. 246-258, Jan. 2000.

[10] D. Chase, "A class of algorithms for decoding block codes with channel measurement information," *IEEE Trans. Inform. Theory*, vol. 18, pp.170-182, Jan. 1972.

[11] H. Xia, C. Zhong and J. R. Cruz, "A Chase-type algorithm for soft-decision Reed-Solomon codes on Rayleigh fading channels," in Proc. IEEE Globe Commun. Conf., vol. 3, pp. 1751-1755, Dec. 2003.

[12] W.J. Gross, F.R. Kschischang, R. Koetter and P.G. Gulak, "A VLSI architecture for interpolation in soft-decision list decoding of Reed-Solomon codes," in *Proc. IEEE Workshop Signal Processing Syst.*, pp. 39-44, 2002.

[13] A. Ahmed, R. Koetter and N. R. Shanbhag, "VLSI architecture for soft-decision decoding of Reed-Solomon codes," submitted to *IEEE Trans. VLSI Systems,* Feb. 2003.

[14] R.E. Blahut, *Theory and Practice of Error Control Codes*, Reading MA: Addison-Wesley, 1983.

[15] G. D. Forney, Jr., “Generalized minimum distance decoding,” *IEEE Trans. Inform. Theory*, vol. 12, pp. 125-131, 1966.

Chapter 6

Nested Codes and Their Applications to Magnetic Recording Systems

6.1 Introduction

The basic idea of nested codes, which are also called integrated interleaving codes, was first proposed by Hassner *et al.* in [1]. The initial motivation for its introduction to storage applications is the fact that one additional random error in an interleave can cause the failure of the whole sector. The integrated interleaving scheme introduces some shared redundancy to Reed-Solomon (RS) codewords, so every codeword or interleave can use this redundancy if its original redundancy failed to decode the correct codeword. Because the error statistics in magnetic recording channels are bursty, nested codes are very attractive.

Let $\{C_i\}_{i=0}^{l}$ be $[n_i, k_i, d_i]$ linear codes, which satisfy the conditions $C_1 \supset C_2 \supset \cdots \supset C_l$ and $d_l > \cdots > \cdots > d_2 > d_1$ with n_i as the code length, k_i as the information length and d_i as the minimum distance of code C_i. For the codes which are most interesting for practical implementation, normally l is set to equal to two. The nested code in consideration, C, is defined as the set of $m \times n$ matrices whose ith row, denoted by $\mathbf{c}_i$, satisfies the following conditions:

$$\begin{aligned} C = \{\mathbf{c} \in GF(q)^{mn} : \mathbf{c}_i \in C_1, i = 0, \cdots, m-1 \; and \\ \sum_{i=0}^{m-1} \alpha^{bi} \mathbf{c}_i \in C_2, b = 0, 1, \cdots, B-1\}, \end{aligned} \tag{6.1}$$

where α is the primitive element of $GF(q)$.

So let a codeword $\mathbf{c} \in C$ with $\mathbf{c} = \{\mathbf{c}_0, \mathbf{c}_1, \cdots, \mathbf{c}_{B-1}, \mathbf{c}_B, \cdots, \mathbf{c}_{m-1}\}$ and another vector $\mathbf{c}' = \{\mathbf{c}'_0, \mathbf{c}'_1, \cdots, \mathbf{c}'_{B-1}, \mathbf{c}_B, \cdots, \mathbf{c}_{m-1}\}$, where $\mathbf{c}'_b \in C_2$ for $b = 0$ to $B-1$, $\mathbf{c}_j \in C_1$ for $j = B$ to $m-1$. Now let

$$\mathbf{c}'_b = \sum_{j=0}^{m-1} \alpha^{bj} c_j \qquad \text{for } b = 0, 1, \cdots, B-1, \tag{6.2}$$

and we have a matrix $\mathbf{A} = \left[\alpha^{bj}\right]$ that

$$\mathbf{c}'^{\mathbf{T}} = \mathbf{A}\mathbf{c}^{\mathbf{T}} = \begin{bmatrix} 1 & 1 & \cdots & 1 & \cdots & 1 & & 1 \\ 1 & \alpha^1 & \cdots & \alpha^{B-1} & \cdots & \alpha^{m21} & & \alpha^{m-1} \\ \vdots & \vdots & & \vdots & & \vdots & & \vdots \\ 1 & \alpha^{(B-1)} & \cdots & \alpha^{(B-1)(B-1)} & \cdots & \alpha^{(m-2)(B-1)} & & \alpha^{(m-1)(B-1)} \\ 0 \; 0 & 0 & \cdots & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 1 & 0 & 0 \\ \vdots & & \vdots & & & & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} \mathbf{c}_0 \\ \mathbf{c}_1 \\ \vdots \\ \mathbf{c}_{B-1} \\ \mathbf{c}_B \\ \vdots \\ \mathbf{c}_{m-1} \end{bmatrix}. \tag{6.3}$$

Since the matrix $\mathbf{A}$ is nonsingular [1], we can obtain the nested codeword $\mathbf{c}^{\mathbf{T}}$ by the encoding scheme $\mathbf{c}^{\mathbf{T}} = \mathbf{A}^{-1}\mathbf{c}'^{\mathbf{T}}$.

Considering that magnetic recording systems require high-rate codes, we let $B = 1$ and $m = 3$, so that a codeword $\mathbf{c}$ consists of three sub-codewords: $\mathbf{c}_1$, $\mathbf{c}_2$ and $\mathbf{c}_3 = \mathbf{c}_1 \oplus \mathbf{c}_2 \oplus \mathbf{c}_4$, where $\oplus$ represents modulo-2 addition, and $\mathbf{c}_1$, $\mathbf{c}_2 \in C_1$, and $\mathbf{c}_4 \in C_2$. The parity-check matrix of C_2 is generated such that $\mathbf{H}_2 = [\begin{smallmatrix}\mathbf{H}_1 \\ \mathbf{H}_{21}\end{smallmatrix}]$, with $\mathbf{H}_1$ being the parity-check matrix of C_1, thereby insuring that $C_1 \supset C_2$ and $\mathbf{c}_3 \in C_1$. For the case when $\mathbf{H}_{21}$ equals to all zero matrix, i.e., $\mathbf{H}_{21} = \boldsymbol{\varphi}$, C_1 and C_2 are the same code.

The sub-codewords are sent through the channel in a serial or interleaved way and decoded separately. And the received vectors are given as

$$\mathbf{r}_1 = \mathbf{c}_1 + \mathbf{e}_1, \quad \mathbf{r}_2 = \mathbf{c}_2 + \mathbf{e}_2, \quad \mathbf{r}_3 = \mathbf{c}_3 + \mathbf{e}_3 = \mathbf{c}_1 \oplus \mathbf{c}_2 \oplus \mathbf{c}_3 + \mathbf{e}_3,$$

where $\mathbf{e}_1$, $\mathbf{e}_2$, $\mathbf{e}_3$ are error vectors. If one of the sub-codewords fails to decode correctly, say codeword $\mathbf{c}_2$, for example, because the number of errors is beyond the error-correction capability of C_1, but we can correctly retrieve $\mathbf{c}_1$ and $\mathbf{c}_3 = \mathbf{c}_1 \oplus \mathbf{c}_2 \oplus \mathbf{c}_4$, then by adding these $\mathbf{c}_1$ and $\mathbf{c}_3$ to $\mathbf{r}_2$, that is,

$$\mathbf{r}_2 + (\mathbf{c}_1 \oplus \mathbf{c}_3) = \mathbf{c}_2 + \mathbf{e}_2 + (\mathbf{c}_1 \oplus \mathbf{c}_3) = \mathbf{c}_2 \oplus \mathbf{c}_1 \oplus \mathbf{c}_1 \oplus \mathbf{c}_2 \oplus \mathbf{c}_4 + \mathbf{e}_2 = \mathbf{c}_4 + \mathbf{e}_2,$$

given the operation + is same as operation $\oplus$ for operation between binary bits. Since codeword $\mathbf{c}_4 \in C_2$ is a more powerful code, it is more likely to be correctly decoded.

6.2 Message Passing between Decoders

Furthermore, if we can obtain soft information from channel output, the nested code described above can be modified to execute the message passing algorithm which will improve the decoding performance.

Given a received vector $\mathbf{r}$, the *a posteriori* probability that the jth bit of codeword $\mathbf{c}_i$ is given as $\Pr\left(c_i^{(j)} \mid \mathbf{r}\right)$. Let assume that $\mathbf{c}_1$, $\mathbf{c}_2$ and $\mathbf{c}_3$ are independent codewords, and $L_i^{(j)}$ be the likelihood ratio as $L_i^{(j)} = \Pr\left(c_i^{(j)} = 1 \mid \mathbf{r}\right) / \Pr\left(c_i^{(j)} = 0 \mid \mathbf{r}\right),\ i = 1,2,3,4$ for each bit, so the soft information for codeword $\mathbf{c}_4$ can be computed as following:

$$\begin{aligned}\Pr\left(c_4^{(j)} = 1 \mid \mathbf{r}\right) = \Pr\left(c_1^{(j)} + c_2^{(j)} + c_3^{(j)} = 1 \mid \mathbf{r}\right) \\ = \Pr\left(c_1^{(j)} = 0, c_2^{(j)} = 0, c_3^{(j)} = 1 \mid \mathbf{r}\right)\Pr\left(c_1^{(j)} = 0, c_2^{(j)} = 0, c_3^{(j)} = 1\right) \\ + \Pr\left(c_1^{(j)} = 0, c_2^{(j)} = 1, c_3^{(j)} = 0 \mid \mathbf{r}\right)\Pr\left(c_1^{(j)} = 0, c_2^{(j)} = 1, c_3^{(j)} = 0\right) \\ + \Pr\left(c_1^{(j)} = 1, c_2^{(j)} = 0, c_3^{(j)} = 0 \mid \mathbf{r}\right)\Pr\left(c_1^{(j)} = 1, c_2^{(j)} = 0, c_3^{(j)} = 0\right) \\ + \Pr\left(c_1^{(j)} = 1, c_2^{(j)} = 1, c_3^{(j)} = 1 \mid \mathbf{r}\right)\Pr\left(c_1^{(j)} = 1, c_2^{(j)} = 1, c_3^{(j)} = 1\right),\end{aligned}$$

and similarly we can obtain

$$\begin{aligned}\Pr\left(c_4^{(j)} = 0 \mid \mathbf{r}\right) = \Pr\left(c_1^{(j)} + c_2^{(j)} + c_3^{(j)} = 0 \mid \mathbf{r}\right) \\ = \Pr\left(c_1^{(j)} = 0, c_2^{(j)} = 0, c_3^{(j)} = 0 \mid \mathbf{r}\right)\Pr\left(c_1^{(j)} = 0, c_2^{(j)} = 0, c_3^{(j)} = 0\right) \\ + \Pr\left(c_1^{(j)} = 0, c_2^{(j)} = 1, c_3^{(j)} = 1 \mid \mathbf{r}\right)\Pr\left(c_1^{(j)} = 0, c_2^{(j)} = 1, c_3^{(j)} = 1\right) \\ + \Pr\left(c_1^{(j)} = 1, c_2^{(j)} = 0, c_3^{(j)} = 1 \mid \mathbf{r}\right)\Pr\left(c_1^{(j)} = 1, c_2^{(j)} = 0, c_3^{(j)} = 1\right) \\ + \Pr\left(c_1^{(j)} = 1, c_2^{(j)} = 1, c_3^{(j)} = 0 \mid \mathbf{r}\right)\Pr\left(c_1^{(j)} = 1, c_2^{(j)} = 1, c_3^{(j)} = 0\right).\end{aligned}$$

Since $\mathbf{c}_1$, $\mathbf{c}_2$ and $\mathbf{c}_3$ are independent codewords, and denote $L_i^{(j)} = \dfrac{\Pr\left(c_i^{(j)} = 1 \mid \mathbf{r}\right)}{\Pr\left(c_i^{(j)} = 0 \mid \mathbf{r}\right)}$, we can compute the likelihood ratio of $\mathbf{c}_4$ as

$$L_4^{(j)} = \frac{\Pr\left(c_4^{(j)} = 1 \mid \mathbf{r}\right)}{\Pr\left(c_4^{(j)} = 0 \mid \mathbf{r}\right)} = \frac{\Pr\left(c_1^{(j)} + c_2^{(j)} + c_3^{(j)} = 1 \mid \mathbf{r}\right)}{\Pr\left(c_1^{(j)} + c_2^{(j)} + c_3^{(j)} = 0 \mid \mathbf{r}\right)} = \frac{L_1^{(j)} + L_2^{(j)} + L_3^{(j)} + L_1^{(j)}L_2^{(j)}L_3^{(j)}}{1 + L_1^{(j)}L_2^{(j)} + L_1^{(j)}L_3^{(j)} + L_2^{(j)}L_3^{(j)}}.$$

(6.4)

The computed likelihood ratio $L_4^{(j)}$ for each bits of $\mathbf{c}_4$ can be used for soft decoding codeword $\mathbf{c}_4$, which, in turns, will generated an updated soft information ${L_{update}}_4^{(j)}$. Iteratively, the updated soft information of $\mathbf{c}_1$, $\mathbf{c}_2$ and $\mathbf{c}_3$ can be computed as

$$ {L_{update}}_1^{(j)} = \frac{L_2^{(j)} + L_3^{(j)} + {L_{update}}_4^{(j)} + L_2^{(j)} L_3^{(j)} {L_{update}}_4^{(j)}}{1 + L_2^{(j)} L_3^{(j)} + L_2^{(j)} {L_{update}}_4^{(j)} + L_3^{(j)} {L_{update}}_4^{(j)}}, \tag{6.5} $$

$$ {L_{update}}_2^{(j)} = \frac{L_1^{(j)} + L_3^{(j)} + {L_{update}}_4^{(j)} + L_1^{(j)} L_3^{(j)} {L_{update}}_4^{(j)}}{1 + L_1^{(j)} L_3^{(j)} + L_1^{(j)} {L_{update}}_4^{(j)} + L_3^{(j)} {L_{update}}_4^{(j)}}, \tag{6.6} $$

$$ {L_{update}}_3^{(j)} = \frac{L_1^{(j)} + L_2^{(j)} + {L_{update}}_4^{(j)} + L_1^{(j)} L_2^{(j)} {L_{update}}_4^{(j)}}{1 + L_1^{(j)} L_2^{(j)} + L_1^{(j)} {L_{update}}_4^{(j)} + L_2^{(j)} {L_{update}}_4^{(j)}}. \tag{6.7} $$

These updated likelihood ratios can in turn be used to generate inputs or extrinsic information to the respective decoders for the next iteration. Such iterative exchange of information between pairs of decoders (See in Fig. 6.1) is expected to improve the performance.

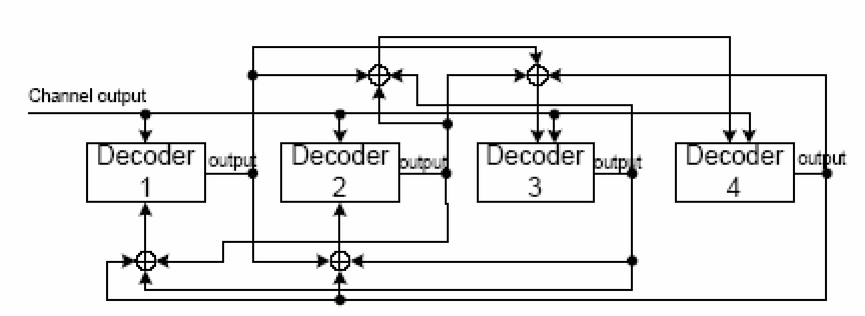


Fig. 6.1. Illustration of the iterative information exchange for two-level nested codes. Decoders 1, 2 and 3 operate on component code C_1, and Decoder 4 on code C_2.

6.3 Nested LDPC codes and its application

The purpose of nested RS codes proposed in [1] is to protect the data from bursty errors. The dominant noise in magnetic recording channels is bursty noise. In [2], the performance of LDPC codes on magnetic recording channels has been investigated for both random noise and bursty noise, which shows a large decoding gain of LDPC codes in random noise compared to RS codes. However, the decoding performance for LDPC codes for bursty noise is not as good as RS codes. Here, since the nested codes were invented for bursty noise channels, we address the nested code problem using LDPC codes as component codes to see if a better decoding performance can be obtained.

6.3.1 LDPC codes

LDPC codes are linear block codes, which were first proposed by Gallager [3], and rediscovered by MacKay *et al*. [4], [5]. Unlike "structured" codes such as the RS code, the parity-check matrix for an LDPC code is sparse, i.e., only a small portion of the entries are ones, the rest of them being zero, and the positions for the ones are selected at random. Originally, Gallager [3] required the parity-check matrix to have uniform column weight w_c as well as a uniform row weight w_r (the number of "1"s in a column or row), which we now call "regular" LDPC codes, while codes with non-uniform column weights and row weights are referred to as "irregular" LDPC codes. Recent work has shown the improved decoding performance of irregular codes over regular codes [6]. A parity-check matrix can be generated by trial and error. Ones are randomly placed in the parity-check matrix, following the requirements of the column and row weight distributions. Once the parity-check matrix $\mathbf{H}$ is constructed, Gaussian elimination and permutation of columns can convert the matrix H into a systematic form such as: $\mathbf{H} = \begin{bmatrix}\mathbf{I} & \mathbf{P}\end{bmatrix}$, and the generator matrix $\mathbf{G} = \begin{bmatrix}\mathbf{P} \\ \mathbf{I}_{\mathrm{k}\times\mathrm{k}}\end{bmatrix}$ can be obtained, where k represents the information length of a codeword. Also, LDPC codes can be represented by factor graphs, which contain two types of nodes: check nodes and bit nodes. Each bit node represents a bit in a given LDPC codeword, and each check node connects to several bit nodes (a row in the parity-check matrix $\mathbf{H}$), which represents a parity-check equation. Therefore, an n bit nodes, m check nodes factor graph for an LDPC code represents an $m \times n$ parity-check matrix $\mathbf{H}$. If a bit involves in the parity-check equation represents by a certain check node, an edge between the check node and the bit node will exist. An example of a parity-check matrix and its bipartite graph is given in Fig. 6.2.

$$\mathrm{H} = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{bmatrix}.$$

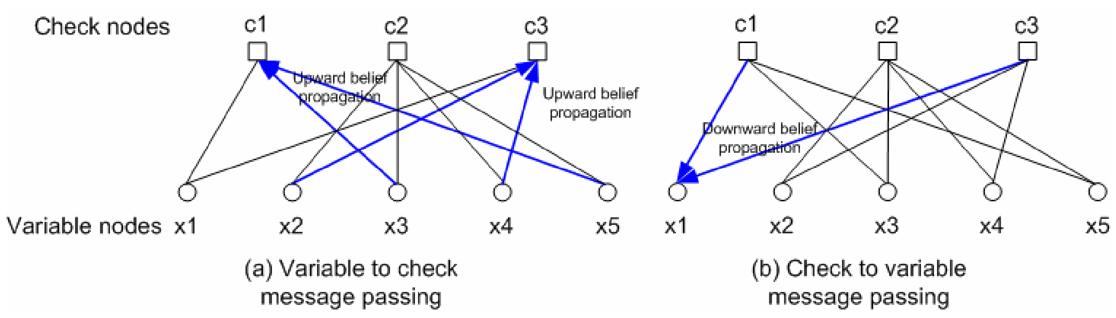

Check nodes
c1
c2
c3
Upward belief propagation
Upward belief propagation
Variable nodes x1
x2
x3
x4
x5
(a) Variable to check message passing
c1
c2
c3
Downward belief propagation
x1
x2
x3
x4
x5
(b) Check to variable message passing

Fig. 6.2. Bipartite graph of LDPC code.

From Fig. 6.2 we can see, bit x_1 involves into two check nodes c_1 and c_3, that is two check equations. In order to update the belief information of bit x_1, we need collect the belief information from check nodes c_1 and c_3, then send them to x_1 for updating. By iteratively exchanging soft probability information between check nodes and bit nodes, the LDPC decoder under certain conditions converges to a decoding answer [7]. The good performance of the LDPC codes stems from the randomness in the parity-check matrix, and as long as the parity check matrix does not contain any short cycles, such as cycle-4, etc., a good decoding performance can be expected. Fig. 6.3 shows a cycle with length 4, since the short cycle will make the probabilities (beliefs) in message passing algorithm highly correlated, which largely degrade the decoding performance.

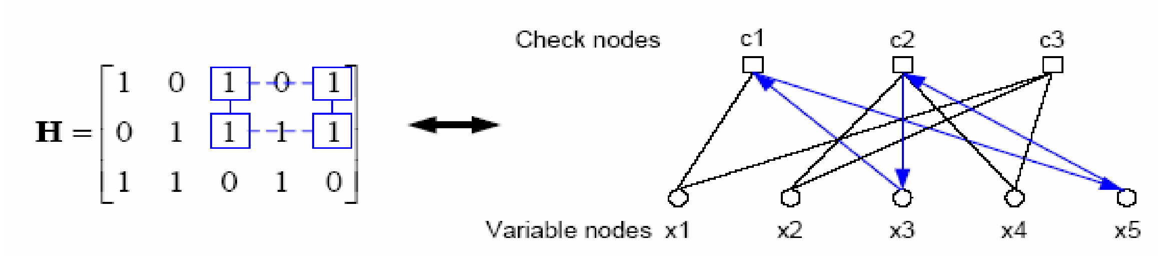

$\mathbf{H} = \begin{bmatrix} 1 & 0 & \boxed{1} & 0 & \boxed{1} \\ 0 & 1 & \boxed{1} & 1 & \boxed{1} \\ 1 & 1 & 0 & 1 & 0 \end{bmatrix}$
Check nodes
c1
c2
c3
Variable nodes x1
x2
x3
x4
x5

Fig. 6.3. Parity check matrix and its Tanner graph with short cycles.

6.3.2 RS-Based LDPC Codes

For an LDPC code, we hope there are no four-cycles in its parity check matrix to achieve good decoding performance. Also, in order to generate a nested code using LDPC codes, we want a parity check matrix $\mathbf{H} = [\mathbf{H}_1]$ for C_1 that can be expanded easily to $\mathbf{H}' = \begin{bmatrix} \mathbf{H}_1 \\ \mathbf{H}_{21} \end{bmatrix}$ for C_2. In consideration of these two issues, we use the RS-based LDPC codes proposed in [8], [9] for the component codes used for the nested LDPC encoding scheme.

Given a (shortened) RS $(n = \rho, k = 2, t = \rho - 1)$ code with two information symbols, according to the generator polynomial

$$\begin{aligned} g(x) &= (x-\alpha)(x-\alpha^2)\cdots(x-\alpha^{2t}) \\ &= g_0 + g_1 x + g_2 x^2 + \cdots + x^{2t} \end{aligned}$$

where $g_i \in GF(q)$ and the generator matrix is

$$G = \begin{bmatrix} g_0 & g_1 & g_2 & \cdots & 1 & 0 \\ 0 & g_0 & g_1 & g_2 & \cdots & 1 \end{bmatrix}. \tag{6.8}$$

The nonzero RS code generated by this generator matrix has two possible weights, ρ and $\rho - 1$, which indicates that there is at most one position in two different RS codewords which have the same symbol value.

Let us consider the q distinct elements in $GF(q)$, which are $0, \alpha, \alpha^2, \cdots, \alpha^{q-2}$ with α as the primitive element. If we denote $\mathbf{v}_i$ to be a vector that contains only a "1" at position i, and $q-1$ "0"s at the rest of positions, then we can map the distinct elements in $GF(q)$ to vectors $\mathbf{v}_i$ such as

$$\begin{array}{ccl} 0 & \rightarrow & \mathbf{v}_0 = (1 \quad 0 \quad 0 \quad \cdots \quad 0), \\ 1 & \rightarrow & \mathbf{v}_1 = (0 \quad 1 \quad 0 \quad \cdots \quad 0), \\ \alpha & \rightarrow & \mathbf{v}_2 = (0 \quad 0 \quad 1 \quad \cdots \quad 0), \\ \vdots & & \qquad\qquad \vdots \\ \alpha^{q-2} & \rightarrow & \mathbf{v}_{q-1} = (0 \quad 0 \quad 0 \quad \cdots \quad 1). \end{array}$$

So for an RS codeword $\mathbf{c} = (c_0, c_1, \cdots, c_{n-1})$, each symbol c_i in the codeword can be expanded into a vector $\mathbf{v}_j$ given $c_i = \alpha^j$. Therefore, the codeword $\mathbf{c} = (c_0, c_1, \cdots, c_{n-1})$ has been expanded into

$$\mathbf{c} = (c_0, c_1, \cdots, c_{n-1})\mathbf{c} = (c_0, c_1, \cdots, c_{n-1}) \quad \rightarrow \quad \mathbf{V} = (\mathbf{v'}_0 \ \mathbf{v'}_1 \cdots \mathbf{v'}_{n-1}).$$

Now, since two different codewords $\mathbf{c}_1$ and $\mathbf{c}_2$ have no more than one symbol in common, the expanded version of $\mathbf{c}_1$ and $\mathbf{c}_2$, $\mathbf{V}_1$ and $\mathbf{V}_2$ respectively, will not have two "1"s at the same location, so if we have a matrix $\mathbf{H} = \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{bmatrix}$, it is free of cycles of length four. And since we can generate more codewords $\mathbf{V}_3$, $\mathbf{V}_4$, etc. using the generator matrix in (6. 8), we can easily expand matrix $\mathbf{H}$ into

$$\mathbf{H} = \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \\ \mathbf{V}_3 \\ \mathbf{V}_4 \\ \vdots \end{bmatrix}.$$

These RS-based LDPC codes described above have no cycles of length four, and their performance has been shown to be almost as good as MacKay's randomly constructed LDPC codes. In addition, the rows of their parity-check matrix can be freely expanded by changing a certain parameter, which is very useful for designing nested LDPC codes.

6.4 Application of Nested Codes to Magnetic Recording Channels

6.4.1 Erasure Detection Techniques

For full erasures, the channel output consists only of noise, and even though the location of the erasure is not known, a channel detector such as the BCJR algorithm [10] outputs certain error patterns, which are fully determined by the combined trellis of the precoder and PR channel. For example, for the generalized PR (1.0, 1.72, 1.15, 0.33) target we get the alternating pattern "…101010…" with no precoder, the all-zeroes pattern with precoder $1/(1 \oplus D)$, and the all-ones pattern for precoder $1/(1 \oplus D^2)$. The location of full erasures can be determined by detecting these error patterns if the length of such error patterns is beyond a certain value. Then for RS decoding, the corresponding symbols are marked as erasures and an error-and-erasure decoding algorithm can be used. For LDPC decoding, the LLRs of the corresponding bits are set to zero, and input to the LDPC decoder [11]. The detection of the error patterns can also be combined with additional code constraints that maybe used in the system such as run-length-limits [12], [13].

The full erasure detection (FED) technique described above cannot be used for partial erasures. Instead, we use the structure of the nested codes to assist us in the location of a noise burst, which should be roughly the same for every interleave. So if we can correctly decode one codeword in a given sector, then we can flag the corresponding symbol errors at the output of the BCJR detector in the other interleaves as erasures and use an erasure decoding algorithm. Similar discussions can be found in [14], [15]. Furthermore, if we can correctly decode two codewords in a sector, the common error pattern in these two blocks can be used as burst error location information for the third incorrect block. Since we have a powerful code embedded in the interleaved codes, it is more likely to correctly decode at least one component code in a nested scheme, which in turn will supply burst location information to be used by the other interleaved blocks (See Fig. 6.4). This erasure location information sharing (ELIS) technique can be used for both RS and LDPC codes.

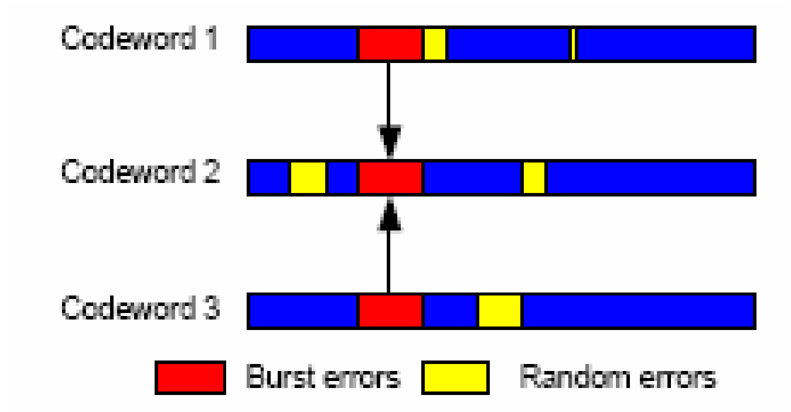


Fig. 6.4. Illustration of error location sharing in interleaved code scheme.

6.5 Performance Evaluation

In our simulations, the BCJR detector is used as the soft-decision channel detector on an equalized perpendicular magnetic recording channel. A generalized PR (1.0, 1.72, 1.15, 0.33) target [16] is used with channel density $K = 1.4$ and KV algorithm is used for soft-decision RS decoding [17].

In Fig. 6.5 a nested RS code with component codes RS(186, 170) and RS(186, 162), is compared with an interleaved RS(186, 168) code, with similar code rate, in the presence of both random and burst noise with L is the burst length in bits. The nested RS code performs better than the interleaved RS code by about 0.25-dB at a sector error rate (SER) of 10^{-4}, if the erasure location is known. From Fig. 6.6 we can observe that in general the introduction of the FED technique, where applicable, provides a coding gain of 1 dB, and that the performance is almost the same as the ideal case when the error location is known exactly. Results for an interleaved RS code with a single 10-bit/symbol sector-size RS(446, 403) code are also given in Fig.

6.5. It can be seen that the maximum performance that the two-level nested RS code scheme can achieve is the same as the sector-size RS code. However, the decoding complexity of a sector-size single RS code is substantially larger, which makes the nested RS coding scheme more attractive. Furthermore, in the presence of partial erasures, the single sector-sized RS code becomes much less attractive than the nested RS codes, since the erasure location information can be shared between component codes in the nested scheme. Fig. 6.6 shows that the performance improvement of nested RS codes afforded by the ELIS technique is 0.2 dB.

A nested LDPC code with component codes RS-based LDPC(2048, 1864) and RS-based LDPC(2048, 1765) (LDPC I) is compared with a non-nested case, which uses only code RS-based LDPC(2048, 1807) (LDPC II). First, we consider the three component codewords concatenated without interleaving, and observe that the nested code performs better than the non-nested LDPC code. In order to evenly distribute the bursts over the component codes we use three-way bit interleaving, and observe that the nested LDPC code shows a performance loss over the non-nested one. The performance of a single iteration of an iterative decoder is also shown in Fig. 6.7. However, since we only chose $B = 1, m = 3$, the nested code in fact is a very weak code, and the iterative improvement is very small. Finally, by nesting a single RS-based LDPC(2048, 1807) code, i.e., letting $\mathbf{H}_{21} = \phi$, we obtain a slight performance improvement without incurring any code rate loss. The erasure detection techniques used for RS codes can also be used for LDPC codes.

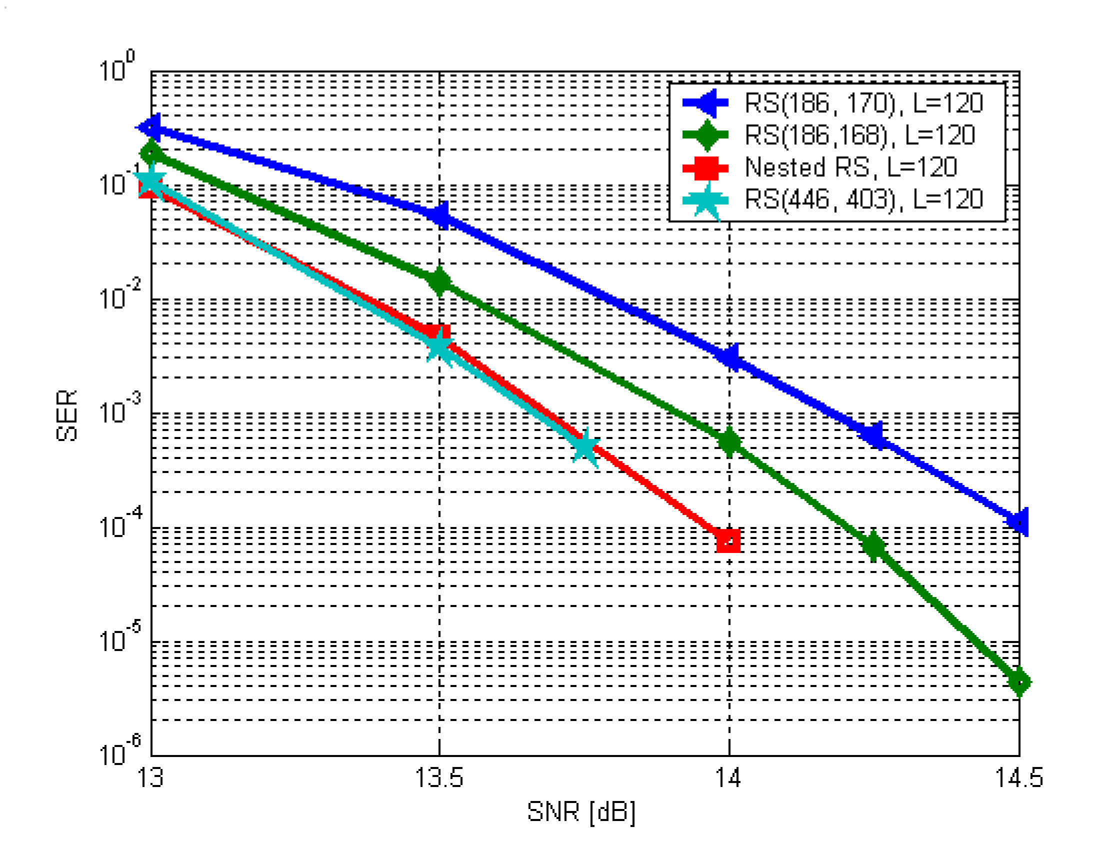


Fig. 6.5. Performance comparison of a nested RS code with the same overall code rate of a single RS code on a perpendicular recording channel, K=1.4, erasure length L =120 bits, with full erasures (η =1) and known location.

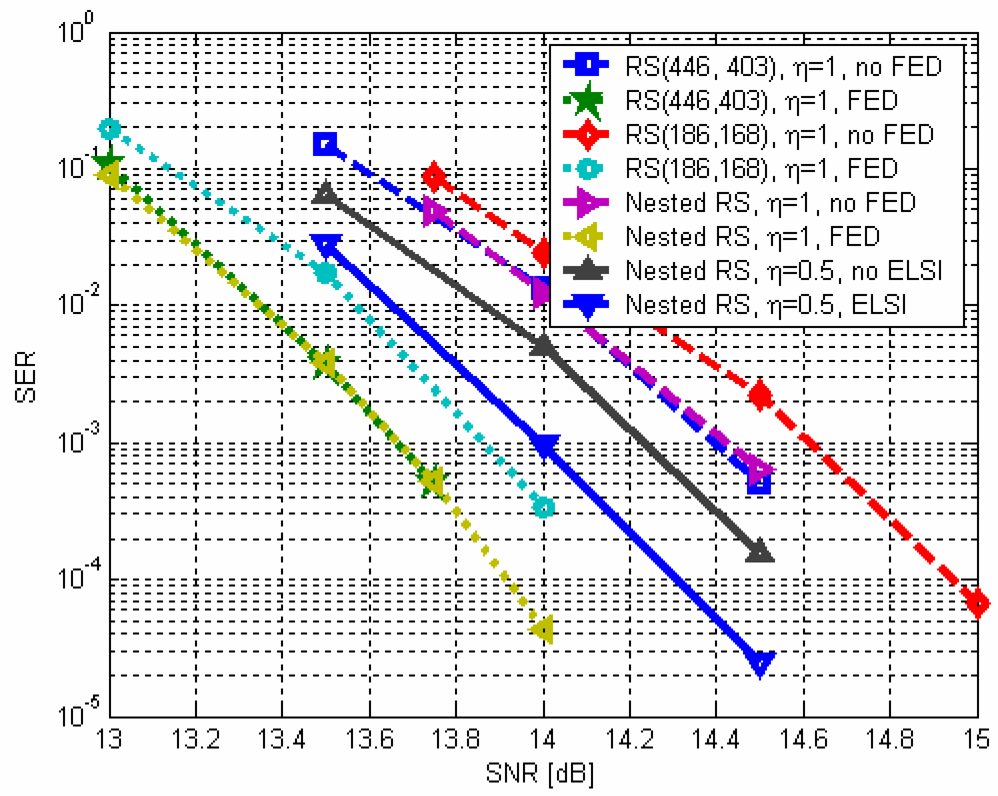


Fig. 6.6. Performance comparison of a nested RS code with the same overall code rate of a single RS code on a perpendicular recording channel, K=1.4, erasure length L =120 bits, with different erasure depths and unknown location.

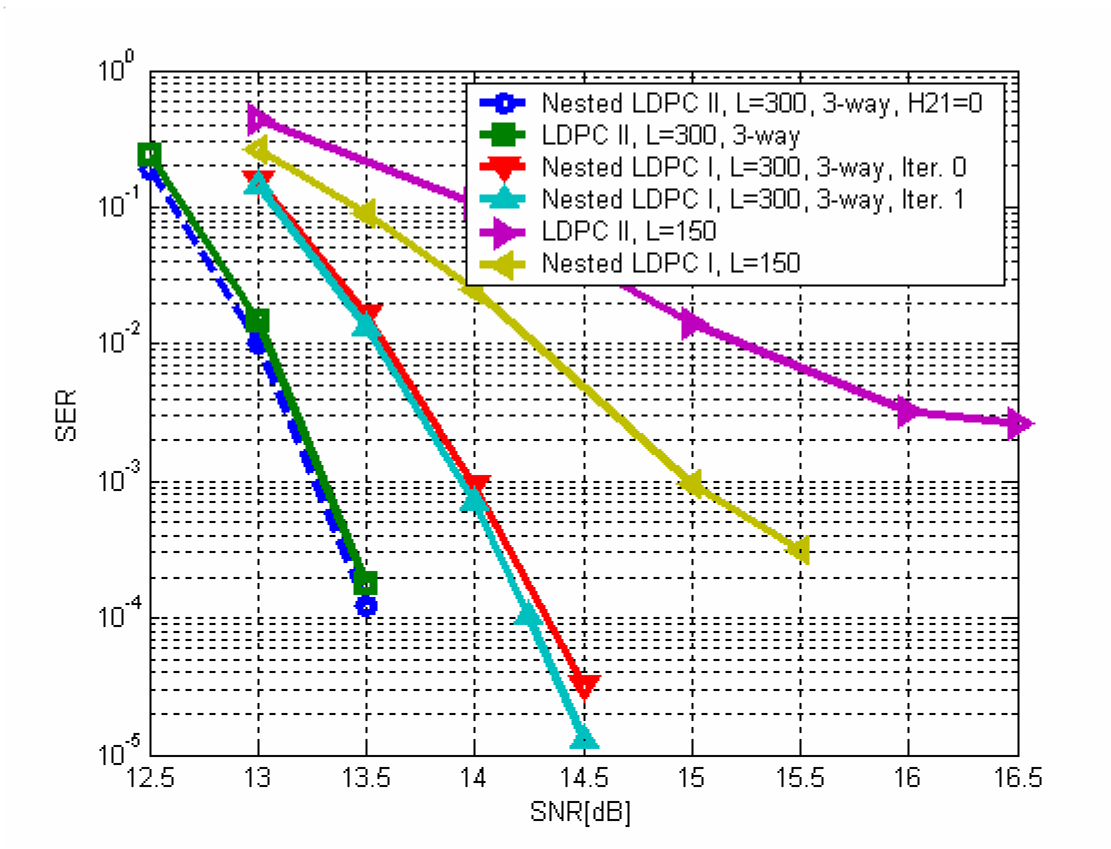


Fig. 6.7. Performance comparison of a nested LDPC code with the same overall code rate of a single LDPC code on a perpendicular recording channel, K=1.4 with full erasures (η =1) and known location.

6.6 Summary

In Chapter 6, a nested RS coding scheme was investigated in the presence of a mixture of random and burst noise on a perpendicular recording channel, and a performance improvement over a single RS code was observed. In addition, we described the nested LDPC codes design using RS-based LDPC codes and their performance was evaluated. Although the nesting of LDPC codes with interleaving led to a performance loss, the concept of iterative exchange of hard or soft information between each component decoder deserves further investigation.

Bibliography

[1] M. Hassner, K. Abdel-Ghaffar, A. Patel, R. Koetter and B. Trager, "Integrated interleaving – a novel ECC architecture," *IEEE Trans. Magn.,* vol. 37, pp. 773-775, Mar. 2001.

[2] H. Xia and J.R. Cruz, "On the performance of soft Reed-Solomon decoding for magnetic recording channels with erasures," *IEEE Trans. Magn*., vol. 39, pp. 2576-2578, Sept. 2003.

[3] R.G. Gallager, "Low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 8, pp. 21-28, Jan. 1962.

[4] D.J.C. MacKay and R.M. Neal, "Near Shannon limit performance of low density parity check codes," *Electronics Letters*, vol. 32, pp. 1645, Aug. 1996.

[5] D.J.C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 45, pp. 399-431, Mar. 1999.

[6] M.G. Luby, M. Mitzenmacher, M.A. Shokrollahi and D.A. Spielman, "Improved low-density parity-check codes using irregular graphs and belief propagation," in *Proc. IEEE Intl. Symp. Inform. Theory*, pp. 117, 1998.

[7] Z.-N. Wu, Coding and Iterative Detection for Magnetic Recording Channels. New York, NY: Kluwer Academic Publishers, 2000.

[8] I. Djurdjevic, J. Xu, K. Abdel-Ghaffar and S. Lin, “A class of low-density parity-check codes constructed based on Reed-Solomon codes with two information symbols,” *IEEE Commun. Letters*, vol. 7, pp. 317-319, July 2003.

[9] I. Djurdjevic, private communication.

[10] L. R. Bahl, J. Cocke, F. Jelinek and J. Raviv, “Optimal decoding of linear codes for minimizing symbol error rate,” *IEEE Trans. Inform. Theory*, vol. 20, pp. 284-287, Mar. 1974.

[11] M. Yang and W.E. Ryan, “Performance of (quasi-)cyclic LDPC codes in noise bursts on the EPR4 channel,” in *Proc. IEEE Global Telecom. Conf.*, vol. 5, pp. 2961-2965, 2001.

[12] T. Morita, Y. Sato and T. Sugawara, “ECC-less LDPC coding for magnetic recording channels,” *IEEE Trans. Magn.,* vol. 38, Sept. 2002.

[13] W. Tan, H. Xia and J. R. Cruz, “Erasure detection algorithms for magnetic recording channels,” *IEEE Trans. Magn.*, vol. 40, pp. 3099-3101, July 2004.

[14] D. Toumpakaris, W. Yu, J. M. Cioffi, D. Gardan, and M. Quzzif, “A byte-erasure method for improved impulse immunity in DSL systems using soft information from an inner code,” in *Proc. IEEE Inter. Commun. Conf.*, vol. 4, pp. 2431-2435, 2003.

[15] X. Tang, and R. Koetter, "On the performance of integrated interleaving coding schemes", in *Proc. Thirty-Sixth Asilomar Conf. on Signals, Systems, and Computers*, vol. 1, pp. 267-271, 2002.

[16] H. Sawaguchi, Y. Nishida, H. Takano and H. Aoi, “Performance analysis of modified PRML channels for perpendicular recording systems,” *J. Magnetism.Magnetic Materials*, vol. 235, pp. 265-272, 2001.

[17] R. Koetter and A. Vardy, “Algebraic soft-decision decoding of Reed-Solomon codes,” *IEEE Trans. Inform. Theory*, vol. 49, pp. 2809-2825, Nov. 2003.

Chapter 7

Iterative Soft-Decision Reed-Solomon Decoding using Reliability Information

7.1 Introduction

Although the interpolation-based soft-decision Reed-Solomon (RS) decoding algorithms [1]-[3] discussed in previous chapters can achieve large decoding gain, the decoding performance of random noise dominanted systems is not as good as LDPC codes [4]. The question that arises is whether we can use the decoding algorithm for LDPC codes to decode RS codes since both of them are block codes. The problem is that although LDPC and RS codes are both block codes, but they have very different parity-check matrices. RS codes have very dense parity-check matrices making them unsuitable for BP decoding due to error-propagation. In order to make the parity-check matrix of an RS code suitable for BP decoding, one would need to convert its parity-check matrix or at least parts of it into a low-density matrix. A lot of work [5]-[8] has been done on modifying the parity-check matrix of RS codes to permit effective BP decoding. Recently, Jiang and Narayanan [9] proposed an adaptive BP algorithm for decoding RS codes based on channel reliability information. By adaptively modifying the parity-check matrix of the RS code into a partially low-density matrix based on bit reliability, they have observed large decoding gains using the BP algorithm on AWGN channels.

In this chapter, the effectiveness and performance of the adaptive BP (ABP) algorithm on inter-symbol interference (ISI) channels as found in magnetic recording systems is investigated, and a modified version of the ABP algorithm is proposed to reduce its decoding complexity.

7.2 Iterative Soft-Decision RS Decoding

Let $GF(q=2^m)$ be a finite field with $q = n+1$ elements and a primitive element α. An RS (n,k) code over $GF(q)$ has a parity-check matrix $\mathbf{H}$, and every codeword $\mathbf{c}$ satisfies the following parity-check equation

$$\mathbf{Hc}^{\mathrm{T}} = \begin{bmatrix} 1 & \alpha & \alpha^2 & \cdots & \cdots & \alpha^{(n-1)} \\ 1 & \alpha^2 & \alpha^4 & \cdots & \cdots & \alpha^{2(n-1)} \\ \vdots & & & & & \vdots \\ 1 & \alpha^i & \alpha^{2i} & & & \alpha^{i(n-1)} \\ \vdots & & & & & \vdots \\ 1 & \alpha^{(\delta-1)} & \alpha^{2(\delta-1)} & \cdots & \cdots & \alpha^{(\delta-1)(n-1)} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_j \\ \vdots \\ c_{n-1} \end{bmatrix} = 0 \qquad (7.1)$$

where $\delta = n-k+1$, and $t = \lfloor (\delta - 1)/2 \rfloor$ is the error-correction capability of the code. The polynomial representation of every codeword $\mathbf{c}$, $c(x) = c_0 + c_1 x^1 + \cdots + c_{n-1}x^{n-1}$, has $\delta = 2t$ zeros, i.e., $c(\alpha) = c(\alpha^2) = \cdots = c(\alpha^{(\delta-1)}) = 0$. Every codeword symbol can be represented by an m-tuple binary expansion using a given basis and each entry in $\mathbf{H}$ can be represented by an $m \times m$ matrix, $\mathbf{M}_j^{\mathrm{T}}$ [10, p. 106].

The BP algorithm has been widely used for decoding low-density parity-check codes, and is responsible for decoding performances approaching the Shannon limit [11]. The binary expansion of the parity-check matrix of the RS code, hereby denoted by $\mathbf{H}_b$, is a high-density matrix containing many short cycles, which lead to correlation between the belief information generated from different check equations and poor performance of the BP algorithm. In [9], Jiang and Narayanan proposed an algorithm which can adaptively modify the parity-check matrix of the RS code into a partly low-density matrix based on bit reliability and effectively reduce error propagation in the BP algorithm still operating on a largely high-density matrix. A brief description of the ABP algorithm is given as follows:

Definitions: Let the log-likelihood ratio (LLR) of the channel output be $L(p_i) = \log \frac{\Pr(p_i = 0)}{\Pr(p_i = 1)}$ for each binary bit i in codeword $\mathbf{c}$, with $i = 1, 2, \cdots, nm$.

Initialization: Input the maximum number of iterations $j_{\max}$, and the channel LLR vector $\mathbf{L}^{(j)}, j = 0$.

Step 1. Sort the LLRs in ascending order of their absolute value and let $i_1, i_2, \cdots, i_j, \cdots, i_{nm}$ denote the position of the sorted codeword bits, i.e., i_1 is the least reliable bit, and i_{nm} is the most reliable bit.

Step 2. Reduce the i_1-th column of $\mathbf{H}_b$ to $[1\,0 \cdots 0]^{\mathrm{T}}$, then the i_2-th column to $[0\,1 \cdots 0]^{\mathrm{T}}$, etc. Since the parity-check matrix has $(n-k)m$ independent columns, we can reduce $(n-k)m$ columns among the nm columns of $\mathbf{H}_b$ to the identity matrix, but not necessarily the ones corresponding to the least reliable bits. If we are not able to reduce the i_j-th column to $\left[\underbrace{0 \cdots 0}_{j}\,1\,0 \cdots 0\right]^{\mathbf{T}}$, skip the i_j-th bit and go on to the next unreliable bit. We obtain $\mathbf{H_b}^{(j)} = \phi\left(\mathbf{H_b}, \mathbf{L}^{(j)}\right)$ where ϕ is the function which realizes the matrix reduction.

Step 3. Perform BP decoding based on $\mathbf{H_b}^{(j)}$ and $\mathbf{L}^{(j)}$ to generate the updated and extrinsic LLR vectors, $\mathbf{L}_u^{(j)}$ and $\mathbf{L}_{ext}^{(j)} = \mathbf{L}_u^{(j)} - \mathbf{L}^{(j)}$.

Step 4. Hard decision: $\hat{p}_i = \begin{cases} 0 & L_u^{(j)}(p_i) > 0 \\ 1 & L_u^{j}(p_i) < 0 \end{cases}$.

Step 5. Termination Criterion: If all the check equations are satisfied, End; else if $j = j_{max}$, declare a decoding failure.

Step 6. Update the LLR vector as

$$\mathbf{L}^{(j+1)} = \mathbf{L}^{(j)} + \lambda \mathbf{L}_{ext}^{(j)}, \tag{7.2}$$

where λ is a damping coefficient. Set $j = j+1$ and go to Step 1.

Example 7.1: Let α be the primitive element over $GF(q=2^3)$ with primitive polynomial as $p(x) = 1 + x + x^3$. With information polynomial as $f(x) = 4 + 2x + 6x^3 = \alpha^2 + \alpha x + \alpha^4 x^3$, an RS (7, 4) codeword is generated as $\mathbf{c} = (0\ 4\ 6\ 6\ 0\ 2\ 4)$ and its binary representation as $\mathbf{c}_b = \{l_i\}_{i=1}^{nm} = (0\,0\,0, 0\,0\,1, 0\,1\,1, 0\,1\,1, 0\,0\,0, 0\,1\,0, 0\,0\,1)$. According to (7.1) and [10, p. 106], the binary expansion of RS parity check matrix is given as

$$\mathbf{H}_b = \left[\begin{array}{ccc:ccc:ccc:ccc:ccc:ccc:ccc}
1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\
\hdashline
1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\
0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\
\hdashline
1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1
\end{array}\right]$$

and $\mathbf{H}_b\mathbf{c}_b^{\mathrm{T}} = \mathbf{0}$. Suppose the binary bit sequence $\mathbf{c}_b$ is sent out, and the received channel reliability information can be converted into an LLR vector as:

$$\mathbf{L} = (5.0,\ 4.9,\ 1.6,\ 4.3,\ 0.2,\ -8.3,\ 5.8,\ -3.1,\ -0.5,\ -3,\ -9,\ -0.3,\ 6.7,\ 3.6,\ -0.18,\ 6,\ -7.4,\ 1.1,\ 4,\ 5.4,\ 0.75)$$

and the sorted LLR vector $\tilde{\mathbf{L}}$ is

$$\tilde{\mathbf{L}} = (-\boxed{0.18},\ 0.2,\ -0.3,\ -0.5,\ \boxed{0.75},\ 1.1,\ 1.6,\ \boxed{-3,}\ -3.1,|\,3.6,\ 4,\ 4.3,\ 4.9,\ 5.0,\ 5.4,\ 5.8,\ 6,\ 6.7,\ -7.4,\ -8.3,\ 9).$$

The three erroneous bits are marked with red boxes, and the number of errors is beyond the hard-decision RS decoding correction capability. According to the sorted LLR vector, the binary expansion parity check matrix can be modified to

$$\tilde{\mathbf{H}}_b = \begin{array}{ccccccccc:cccccccccccc}
\boxed{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & \boxed{1} & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \boxed{1} & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\
l_{15} & l_5 & l_{12} & l_9 & l_{21} & l_{18} & l_3 & l_{10} & l_8 & l_{14} & l_{19} & l_4 & l_2 & l_1 & l_{20} & l_7 & l_{16} & l_{13} & l_{17} & l_6 & l_{11}
\end{array}.$$

Using the modified parity check matrix above, it is shown by simulation that with one iteration message passing, the number of erroneous bits is reduced to only one bit, which is within the hard-decision RS decoding correction capability. If the number of errors is still beyond the correction capability of the code, more iterations of message passing are needed.

7.3 Performance Discussion

The ABP algorithm described above divides the received sequence into two groups based on the absolute value of their LLRs, namely groups G_r and G_u containing reliable and unreliable bits, respectively. Fig. 7.1 shows an example which illustrates the relationship between bit error rate and reliability, making it clear that the $(n-k)m$ least reliable bits have a high probability of being in error. The columns in the parity-check matrix corresponding to these unreliable bits will be reduced to having a single non-zero entry, forcing that unreliable bit to be involved in a single check equation.

However, the remaining columns of the parity-check matrix corresponding to reliable bits are still dense, and each reliable bit is involved in more than one check equation. Fig. 7.2 shows examples of a modified parity check matrix $\mathbf{H}_\mathbf{b}$ with message passing to unreliable bit l_{15} and reliable bit l_{13} using a Tanner graph.

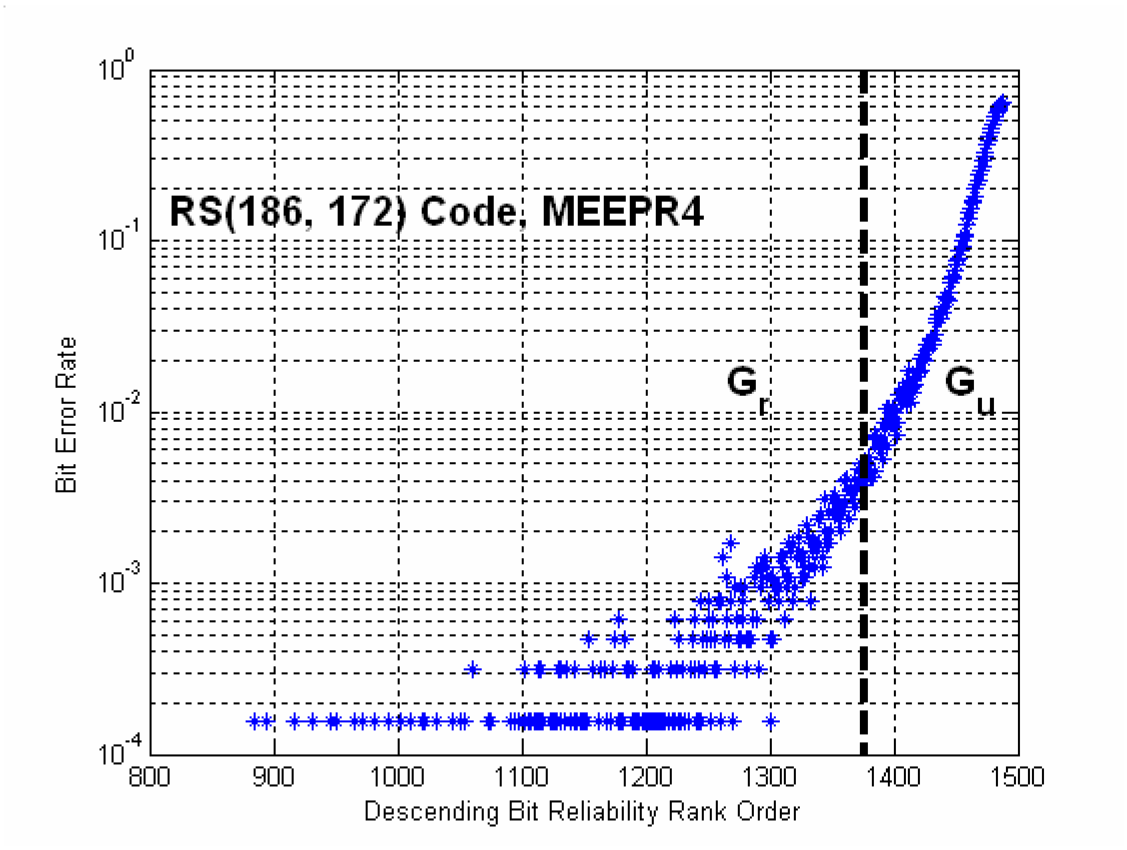


Fig. 7.1. Bit error rate as a function of reliability rank order for an RS(186, 172) code on an equalized MEEPR4 channel at SNR=14.5 dB.

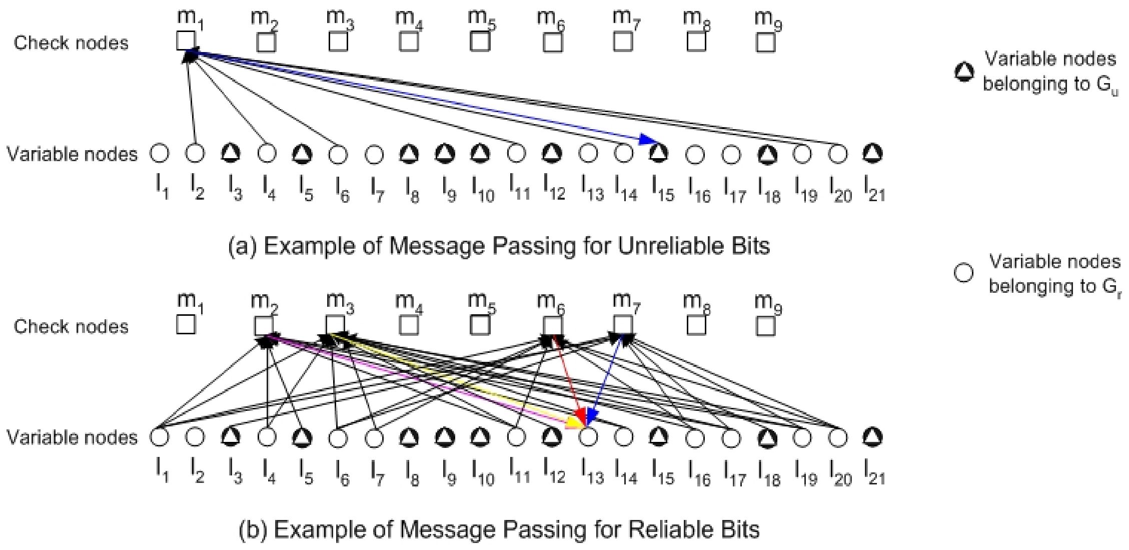

Check nodes
m1 m2 m3 m4 m5 m6 m7 m8 m9
Variable nodes
l1 l2 l3 l4 l5 l6 l7 l8 l9 l10 l11 l12 l13 l14 l15 l16 l17 l18 l19 l20 l21
(a) Example of Message Passing for Unreliable Bits
(b) Example of Message Passing for Reliable Bits
Variable nodes belonging to Gu
Variable nodes belonging to Gr

Fig. 7.2. Examples of message passing between variable and check nodes.

Using the same notation as in [12, p. 60], where l denotes variable nodes and m denotes check nodes, the checks-to-variables and the variables-to-checks updates for the modified parity-check matrix are given as

$$L(r_{m\to l}) = 2\tanh^{-1}\left(\prod_{l'\in \mathrm{L}(m)\backslash l} \tanh\left(L(q_{l'\to m})/2\right)\right), \tag{7.3}$$

$$L(q_{l\to m}) = \begin{cases} L(p_l) + \sum\limits_{m'\in \mathrm{M}(l)\backslash m} L(r_{m'\to l}) & l\in G_r \\ L(p_l) & l\in G_u \end{cases}, \tag{7.4}$$

and the updated LLR for each bit is

$$L(q_l) = \begin{cases} L(p_l) + \sum\limits_{m\in \mathrm{M}(l)} L(r_{m\to l}) & l\in G_r \\ L(p_l) + L(r_{m\to l}) & l\in G_u \end{cases}. \tag{7.5}$$

Equations (7.3)-(7.4) are initialized as $L(q_{l\to m}) = L(p_l)$, $L(r_{m\to l}) = 0$.

From (7.3)-(7.5) above we can see that bits in G_u are involved in only one check equation, and since the absolute value of their LLRs is small (compared to those in G_r), the LLR computed by (7.3) (using bits from G_r with LLRs with larger absolute values) will likely change the sign of bits in G_u if the check equation is not satisfied. Even if it cannot change the sign of a particular bit, the summation of LLRs in (7.5) will change the original absolute value of the LLR for this bit to a small value, which will definitely help its decoding in the next iteration of the BP algorithm. For bits in G_r, because of the large absolute value of their LLRs, (7.5) will not change their sign. Therefore, as long as the bits in G_r are correct, it is likely that the sign of the LLRs of most of the unreliable bits in G_u will be correctly changed, and the sign of the LLRs of the reliable bits in G_r will definitely not be changed. Furthermore, even if not all the signs of the LLRs for bits in G_r are correct, (7.4) will reduce the absolute value of their LLRs, which will help their decoding in the next iteration. Therefore, partially reducing the column weight according to the channel reliability information, limits the error propagation of unreliable bits, which improves the decoding performance, even though the parity-check matrix may still have a large number of short cycles.

For a parity-check matrix which is cycle-free, density evolution can be used for asymptotic performance analysis of message-passing decoding [13], [14]. But for parity-check matrices with many short cycles, no method is available for determining BP decoding performance. Here we exploit a special property of the adaptive BP algorithm, to provide an approximate decoding performance analysis. Since the BP algorithm might lead to error propagation when the parity-check matrix has short cycles, in a practical system, an outer algebraic RS decoder is required for good decoding performance. The performance of a decoding system utilizing both an algebraic decoder and a BP decoder can be expressed in terms of the probability of decoding error $P_d = P_{ad} \times P_{bpd}$, where P_{ad} is the error rate of the algebraic decoder given a received block and P_{bpd} is the error rate of the BP decoder given that the received block cannot be successfully decoded by the algebraic decoder. The error rate P_{ad} can be found either analytically [15, p. 250] or by simulation. The error rate P_{bpd} can be approximated on the basis of the following observation. The BP decoder operates on a set of reliable bits whose hard decision prior to BP decoding may or may not be correct. We have observed that if all reliable bits are correct, the BP decoder is always able to converge to the correct codeword or reduce the errors to within the RS algebraic decoder correction capability. The probability that the reliable bits are all correct can be estimated using simulations as $P_{G_r} = \Pr\left(all\, bits\, in\, G_r\, are\, correct\right)$, and even if not all bits in G_r are correct, by adaptively changing matrix $\mathbf{H}_b$ and do BP decoding, we might be able to find the correct answer with probability P_c. So an approximation to the probability of decoding error can be computed as

$$\begin{aligned} P_d &= P_{ad} \times \left(1 - P_{G_r} - P_c \times \left(1 - P_{G_r} \right) \right) \\ &= P_{ad} \times \left(1 - P_{G_r} \right) \times \left(1 - P_c \right) \\ &\le P_{ad} \times \left(1 - P_{G_r} \right) \end{aligned} \qquad (7.6)$$

In Table 7.1, we show simulation results for the values of the probabilities P_{ad}, P_{G_r}, P_c and the overall error rate P_d for a shortened RS (190, 172) code on a perpendicular recording channel with DC-full target (1.0, 1.72, 1.15, 0.33) and channel density K=1.4. It can be seen that the majority of decoding failures of the algebraic decoder can be correctly decoded by the BP decoder since most of those errors in an RS codeword will be included in G_u, and bits in G_r are more likely to be correct.

Based on the analysis above, it is paramount that all erroneous bits be grouped into G_u to prevent error propagation during BP decoding. A swap of bits between G_u and G_r (as shown in [9]) will further improve the decoding performance if all the erroneous bits can be swapped out of G_r . The best possible performance of the ABP algorithm would occur when we know all the erroneous bits in the received sequence, sort them in descending order of their probabilities, and assign the first $(n-k)m$ erroneous bits to G_u, (if there are more than $(n-k)m$ bits in error), then modify the parity check matrix according to G_u and G_r , and execute the BP decoding algorithm. The corresponding RS codeword error rate of such operation can be taken as the ML bound for this ABP decoding algorithm, which is also the best result of the swapping proposed in [9]. Also, since error propagation will occur if erroneous bits are assigned to G_r, after each BP iteration, the Hamming distance between the hard-decision BP decoder output and the hard-decision received vector can be computed to see if the difference is larger than a threshold θ; if it is larger, an indication of error propagation, the BP decoder output will be either discarded or its extrinsic information will be weighted less in the next iteration. This will mitigate the error propagation of the ABP algorithm. Also, error propagation is an indication that some erroneous bits might have been included in G_r, and a swap of bits between G_u and G_r may be helpful.

Table 7.1
Probabilities for an RS(190, 172) code on a DC-Full GPR4 channel with channel density K=1.4

SNR (dB)	P_{ad}	P_{G_r}	$P_c\left(1-P_{G_r}\right)$	P_d
12.5	0.211	70%	8.2%	0.046
13	0.0311	86.2%	4.82%	0.0028
13 .5	0.0025	95.09%	2.08%	$7\text{x}10^{-5}$

7.4 Modified Adaptive Belief Propagation Algorithm

The adaptive algorithm proposed in [9] is very effective when doing BP decoding of RS codes. However, the decoding complexity of the modified partially dense parity check matrix is still high. Based on the analysis and observation in Section 7.3, we slightly modify the adaptive BP algorithm [9], leading to a small degradation of the decoding performance, but the complexity reduction is very substantial.

The modified adaptive BP (MABP) algorithm is based on the observation that bits in G_u are more likely to be erroneous bits. The detailed algorithm is: Given a received bits sequence and corresponding LLR sequence, the bit sequence will be divided into two groups (G_u and G_r) according to the channel output reliability, and the RS parity check matrix will be modified correspondingly. The LLRs for bits in G_u is updated using

$$L(q_l) = L(p_l) + 2\tanh^{-1}\left(\prod_{l' \in \mathrm{L}(m)\setminus l} \tanh\left(L(q_{l' \to m})/2\right)\right) \tag{7.7}$$

and those bits in G_r will keep unchanged as

$$L(q_l) = L(p_l)\ . \tag{7.8}$$

The updated LLRs will be sent to soft/hard RS decoder for further decoding to see if the correct answer can be found. If not, a decoding failure will be claimed or the updated LLRs can be sent back to channel detector such as SOVA [16] or BCJR [17] algorithms for another iteration decoding trial. Since the message passing algorithm is used by only those bits involved in only one check equation, tremendous decoding complexity reduction can be expected.

Also, in order to further reduce the decoding complexity, the MABP algorithm can be further modified into a hard version message passing with negligiable performance loss. Let $H(p_l) \in \{0,1\}$ be the hard decision of bit l with probability p_l and the addition is a modulo-2 operation. A hard version message passing of (7.7) becomes

$$H(q_l) = \sum_{l' \in \mathrm{L}(m) \backslash l} H(p_{l'}) \tag{7.9}$$

So the update of hard decision of bit l belongs to G_u is give by $H(q_l)$, and bits belong to G_r is unchanged. Using the example shown in Section 7.2, the unreliable bits such as l_{15}, l_5, etc. can be computed as

$$H(p_{15}) = H(p_{14}) + H(p_{19}) + H(p_4) + H(p_2) + H(p_{20}) + H(p_6) + H(p_{11}),$$

$$H(p_5) = H(p_{14}) + H(p_{19}) + H(p_4) + H(p_1) + H(p_{13}) + H(p_{17}) + H(p_{11}),$$

$$\begin{aligned} H(p_{12}) = H(p_{14}) + H(p_{19}) + H(p_4) + H(p_1) + H(p_{20}) + H(p_7) + H(p_{16}) \\ + H(p_{13}) + H(p_{17}) + H(p_6) + H(p_{11}) \end{aligned},$$

……

$$H(p_8) = H(p_{14}) + H(p_4) + H(p_1) + H(p_{20}) + H(p_7) + H(p_{16}) + H(p_{17}).$$

The updated hard-decision of unreliable bits (G_u) will combine with other bits in G_r for hard-decision RS decoding. By doing this, the received bits need only to be sorted and the unreliable bits need to be found out, no LLRs need to be saved for each bit (which consumes a lot of memory), and the computation of (7.9) is very easy to implement. Also, instead of storing the non-zero entries of a binary parity check matrix in BP decoding, here we can store the zero entries of the binary parity check matrix of RS codes. Once the reliability-based modified parity check matrix is generated, a hard version message passing using (7.9) will flip the unreliable bits if a check equation is not satisfied. The hard version of the MABP (HMABP) algorithm is very attractive for hardware implementation. Also, by exhaustive swapping bits between two groups, the modified ABP algorithm can achieve its asymptotic performance.

Furthermore, it is interesting to note that the HMABP algorithm can be taken as a variation of the Chase algorithm [18] with $p = (n-k)m$ bits. In the Chase algorithm, the p least reliable bits are selected, and 2^p test patterns are generated by exhaustively flipping those unreliable bits. The HMABP algorithm determines which particular test pattern should be chosen instead of going through 2^p trials. From (7.7) and (7.9), the performance difference between the MABP and the HMABP algorithms should be negligible.

7.5 Implementation on Magnetic Recording Systems

Although the algorithm in [9] has the potential to deliver improved performance, simulations show that error propagation takes place when erroneous bits are included in G_r. In order to mitigate error propagation, a damping coefficient has been introduced in the updating of the LLRs. Also, an algebraic RS decoder is needed before and after the ABP decoder to increase the convergence rate as well as prevent error propagation. Since a hard-decision algebraic RS decoder is already used in current magnetic recording systems, a retry-mode ABP decoding scheme can be used whenever the algebraic RS decoder fails.

The retry mode decoding scheme for an RS coded system is given as follows:

Step 1: Execute an algebraic RS decoding algorithm (Decoder 1). If it fails go to *Retry Mode*.

Retry Mode: Step 2: Sort the received bit sequence based on the magnitude of the reliability information, $\left|\mathbf{L}^{(j)}(p_i)\right|$, divide it into two groups, G_u and G_r, and reduce the columns of $\mathbf{H}$ corresponding to bits in G_u to identity matrix form [9].

Step 3: Execute a BP algorithm (Decoder 2) followed by an algebraic decoding algorithm. If it fails, update the original LLRs with the extrinsic information generated by the BP decoder with a damping coefficient λ. If the maximum number of iterations has been is reached *End,* else go to Step 3.

7.5.1 Effect of Error Bursts

In magnetic recording channels, TA and MDs are the main source of burst errors but they can be declared as erasures. Erasure detection techniques [19], can be used to provide erasure location information to the ECC decoder. The modified adaptive BP algorithms described above as well as the original one in [9] can immediately assign these erasures to G_u without having to sort the sequence.

7.5.2 Hardware Architecture

The modified parity-check matrix changes every execution of the adaptive BP decoding algorithm according to the reliability order of each bit. It would be more efficient if the BP decoder circuit could adapt to the change of the parity-check matrix. Modern circuit design techniques allow a reconfigurable implementation as shown in Fig. 7.3. For each parity-check matrix generated, the data defining the edges between variable nodes and check nodes is input to the BP decoder to reconfigure the circuit, while the output pads are still unchanged. If decoding failure occurs after algebraic decoding, a retry signal and the updated LLRs are sent back for another execution of sorting, parity-check matrix updating and BP decoding until a correct answer is found or the maximum number of iterations is reached.

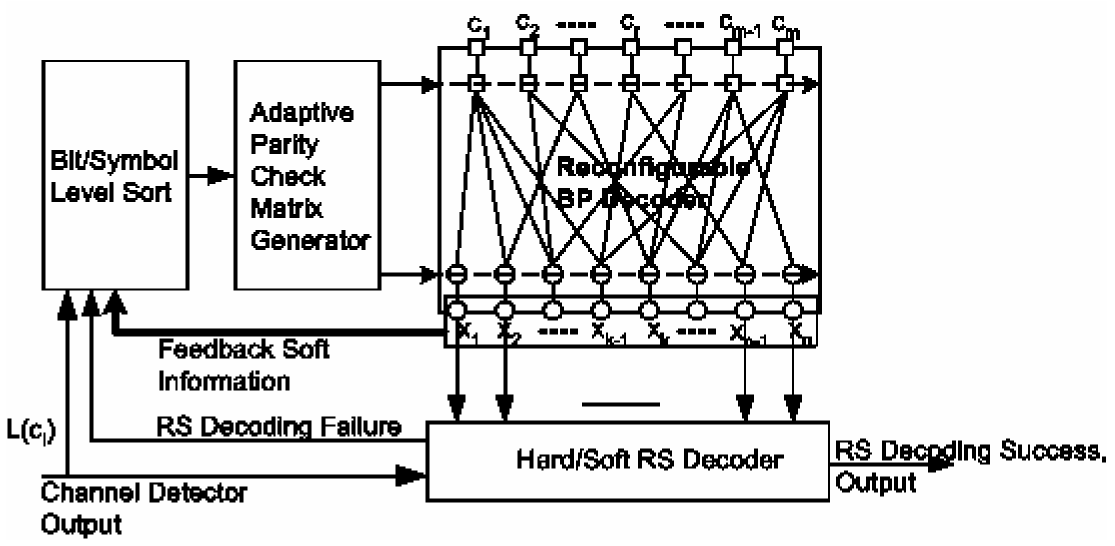

c_1 c_2 ---- c_i ---- c_{m-1} c_m
Bit/Symbol Level Sort
Adaptive Parity Check Matrix Generator
Reconfigurable BP Decoder
x_1 x_2 ---- x_{k-1} x_k ---- x_{n-1} x_n
Feedback Soft Information
RS Decoding Failure
$L(c_i)$
Hard/Soft RS Decoder
RS Decoding Success, Output
Channel Detector Output

Fig. 7.3. Diagram of an efficient implementation of RS decoding with an adaptive BP algorithm.

7.5.3 Decoding System of Concatenation with LDPC Code(s)

The adaptive BP decoding algorithm and its variations proposed in this paper can be applied to LDPC decoding, as well as to systems where an outer LDPC code is concatenated with one or several inner RS codes or a long outer RS code is concatenated with several short length inner LDPC codes [20]. In Fig. 7.4 we show a possible configuration for an LDPC coded magnetic recording system and its system architecture. Since the soft information generated by the RS decoder with the ABP algorithm and by the LDPC decoder are independent, an iterative exchange of the extrinsic information can provide an additional coding gain. Also, a reconfigurable circuit can be used for both LDPC decoding and ABP decoding of the RS code without the need of extra circuitry.

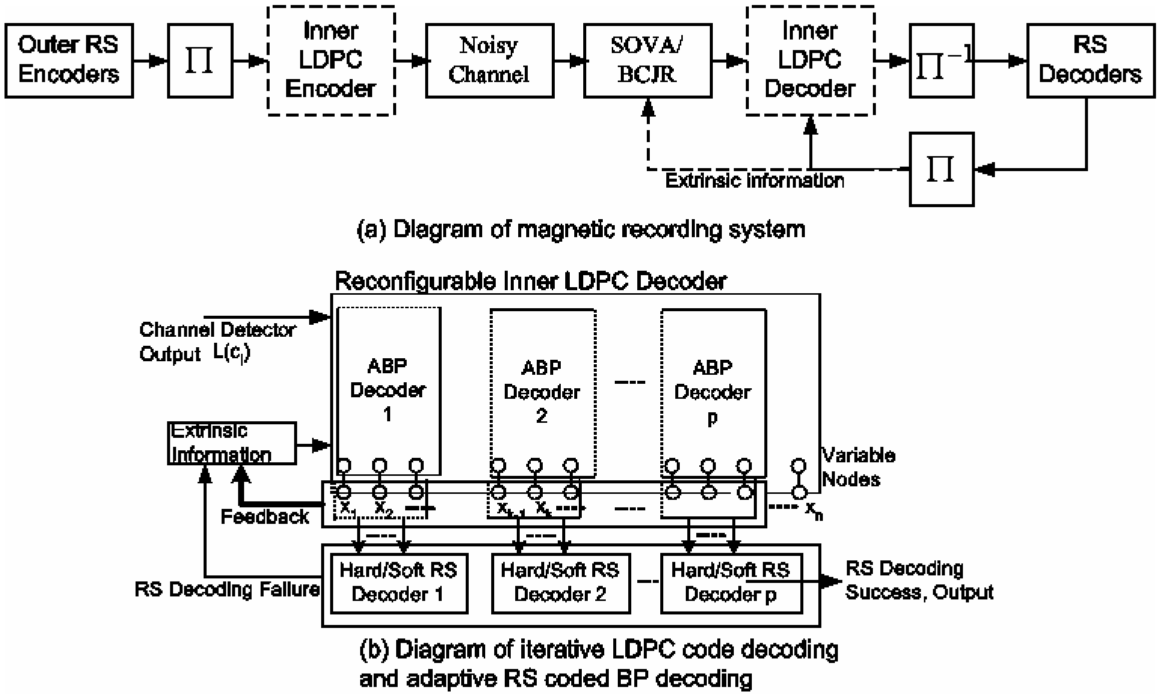


Fig. 7.4. Diagram of implementation of future magnetic recording systems.

7.5.4 Complexity

The adaptive BP algorithm can be divided into four steps: 1) Sorting of channel reliability information; 2) RS parity check matrix modification; 3) BP decoding; 4) Algebraic RS decoding.

Compared to algebraic hard-decision RS decoding algorithms used in current communication systems, ABP decoding is significantly more complex. However, by using the MABP algorithm proposed in this dissertation, the BP decoding (message passing) can be greatly simplified using (7.7). Furthermore, the hard version of message passing using (7.9) does not need to store the LLR of each received bit for BP decoding, making possible a fast and low complexity implementation of the algorithm.

7.6 Performance Evaluation

In our simulations, the BCJR algorithm is used as the soft-decision channel detector on equalized longitudinal magnetic recording channels with an MEEPR4 target [21] and on an equalized perpendicular recording channel with generalized targets [22]. All simulation results for RS codes are obtained using an actual implementation of the algebraic soft-decision decoding algorithm given in [2] or the forward recursive algorithm (FRA) [23], [24]. The maximum number of iterations of the ABP algorithm is set to $j_{\max} = 5$, and the number of iterations of the MABP algorithm is set to one.

Fig. 7.5 shows a performance comparison of different RS decoding algorithms. For a shortened RS (186, 172) code on a longitudinal equalized Lorentzian channel with MEEPR4 target (5, 4, -3, -4, -2) and channel density S_c=2.967, the soft-decision Koetter-Vardy (KV) algorithm shows only 0.3 dB gain compared to a traditional hard-decision RS decoding algorithm. The FRA algorithm, which uses channel reliability information during interpolation, provides a 0.3-dB extra gain over the KV algorithm. The adaptive BP algorithm in cooperation with hard-decision RS decoding gives a total 0.8-dB gain over hard-decision RS decoding at frame error rate FER=10^{-4}. The modified ABP algorithm proposed in this paper performs within 0.2-

dB of the original ABP algorithm. The ideal performance of the modified ABP is about 1.5-dB away from hard-decision decoding. In Fig. 7.6, the performance of a 10-bit/symbol RS (440, 410) code is given. The performance gain of the ABP algorithm compared to hard-decision RS decoding is about 0.25-dB at FER=10^{-3}, and the performance of the MABP algorithm is almost as good as the ABP algorithm. The performance of a similar rate Mackay LDPC (4376, 4096) code is also given as a reference [25]. The LDPC code performs better than the RS code by about 0.75-dB on AWGN, but for continuous burst errors, the RS code outperforms the LDPC code. In Fig. 7.6, an error floor can be observed for the LDPC code with full erasures of length L=128 bits, while the RS code with 130-bit erasures can still achieve good performance. No noise overestimation [25] is used in Fig. 7.6, and but even if noise overestimation is used, the performance of the LDPC (4376, 4096) code [26, Fig. 2] is still not as good as the RS (440, 410) code at high SNR. Also, in the presence of erasures the coding gain of the ABP algorithms increases.

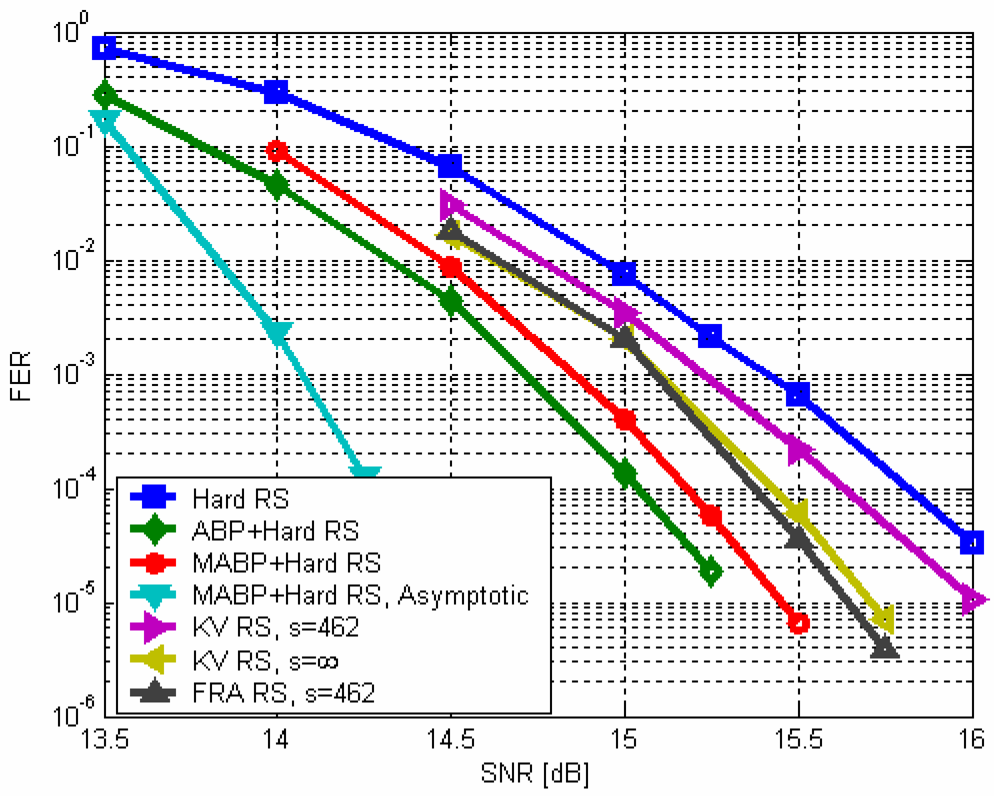


Fig. 7.5 Performance comparison of the RS (186, 172) code with different decoding algorithms on an equalized MEEPR4 channel, S_c=2.967, $\lambda = 0.1$, $j_{\max} = 5$.

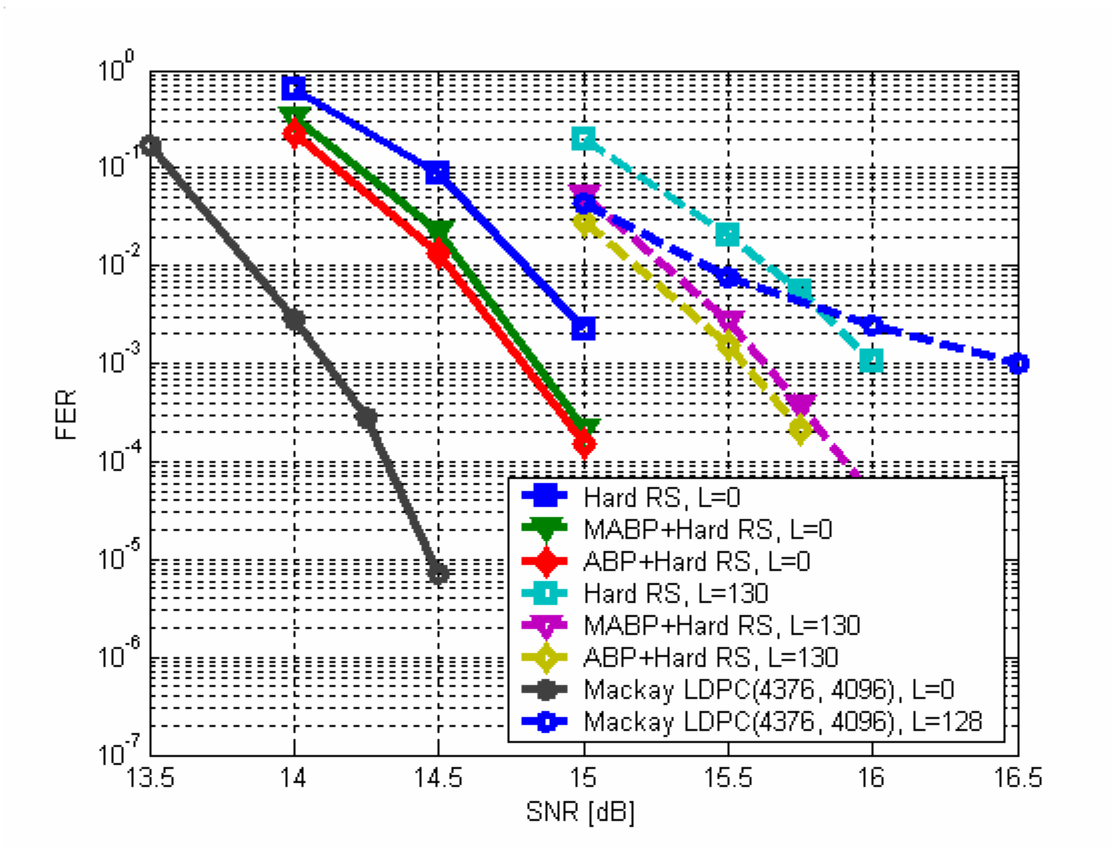


Fig. 7.6 Performance comparison of 10 bits/symbol RS (440, 410) code with different decoding algorithms on an equalized MEEPR4 channel for different erasure lengths L, S_c=2.967, $\lambda = 0.1$, $j_{\max} = 5$, without noise overestimation.

Figs. 7.7 and 7.8 show the performance of RS codes over perpendicular recording channels with a DC-full target (1.0, 1.72, 1.15, 0.33) and a DC-free target (1.0, 1.06, -0.37, -1.12, -0.57), and channel density S_c=1.4. In Fig. 7.7, a 0.75-dB gain is observed for a shortened RS (190, 172) code using the ABP algorithm compared to a hard-decision decoding algorithm, and a 0.5-dB gain over the KV algorithm with total multiplicity s=570. For a DC-full target in Fig. 7.8, the ABP algorithm is 0.6-dB better than hard-decision RS decoding, and the MABP is almost as good as the ABP algorithm.

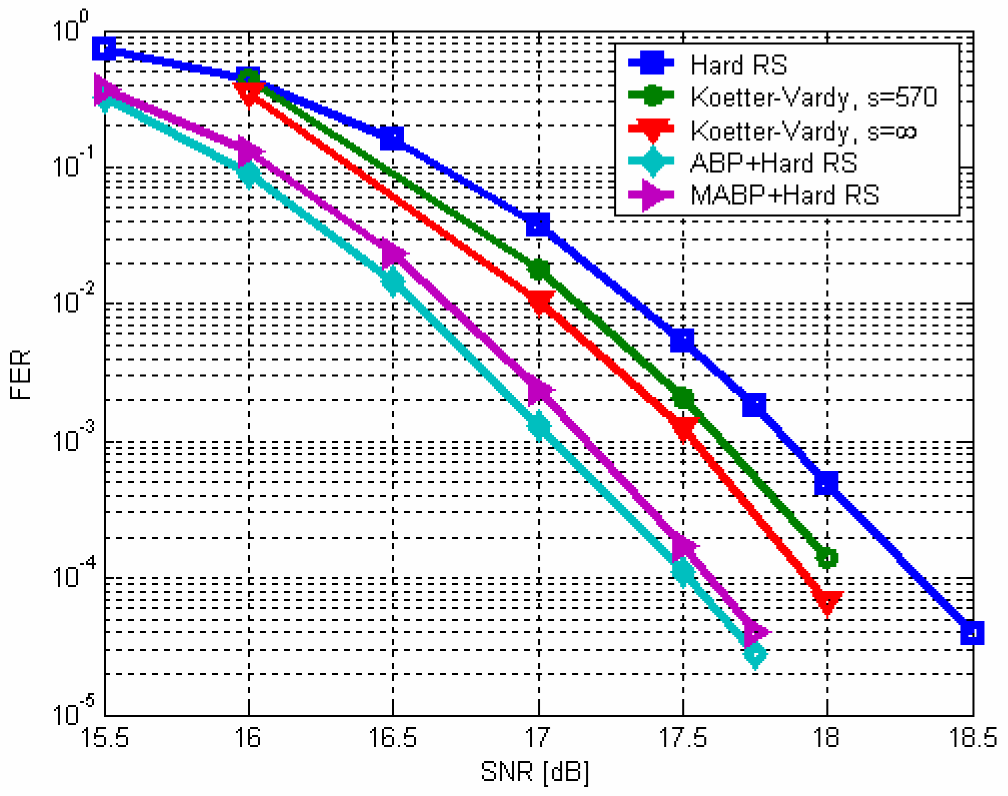


Fig. 7.7 Performance comparison of RS (190, 172) code with different decoding algorithms on a perpendicular recording channel with DC-free target (1, 1.06, -0.37, -1.12, -0.57), K=1.4, $\lambda = 0.1$, $j_{\max} = 5$.

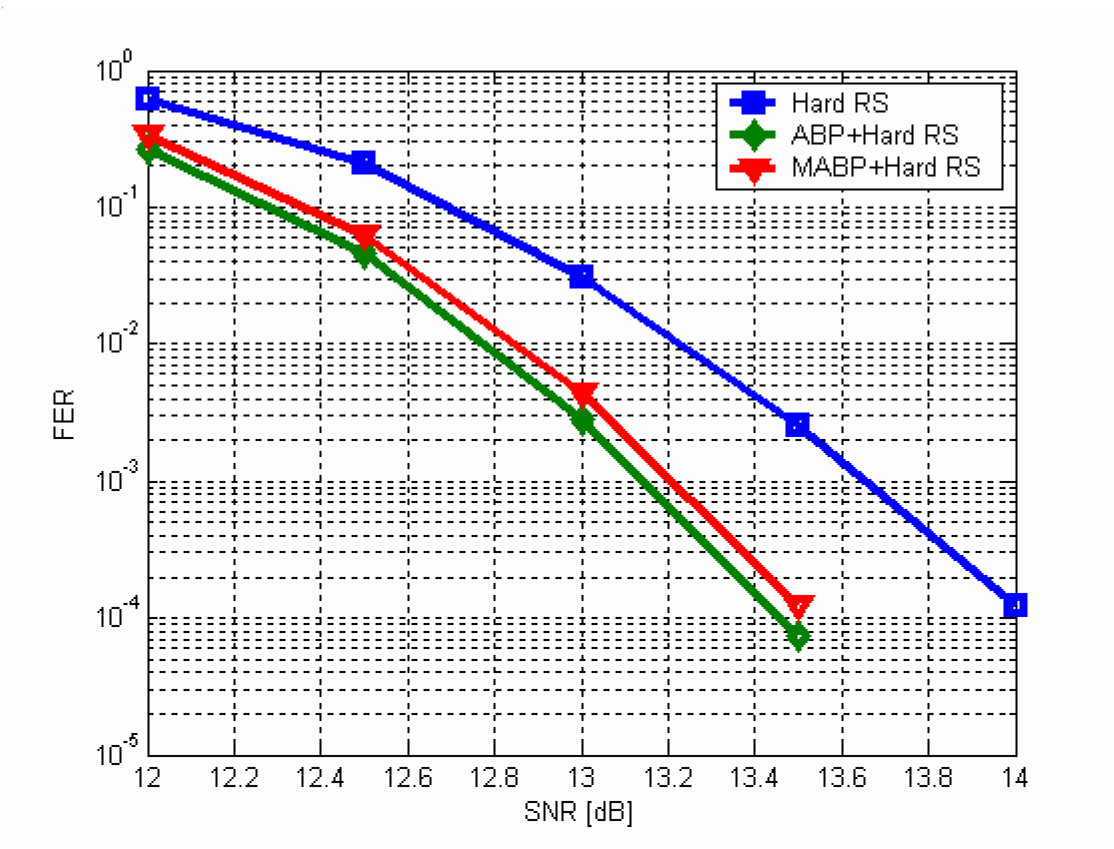


Fig. 7.8 Performance comparison of RS (190, 172) code with different decoding algorithms on an equalized perpendicular recording channel with DC-full target (1, 1.72, 1.15, 0.33), K=1.4, $\lambda = 0.1$, $j_{\max} = 5$.

In high density magnetic recording channels, jitter-like noise becomes the dominant noise. Let the noise at the channel input with total noise power σ_N^2 be the combination of AWGN and jitter noise [26], with noise powers σ_{AWGN}^2 and σ_J^2 respectively. In Fig. 7.9, we set the jitter-to-overall noise ratio to $\sigma_J^2 / \sigma_N^2 = 90\%$ for an equalized Lorentzian-Gaussian channel [27, p. 5] with an MEEPR4 target, and show that the ABP and MABP algorithms provide almost 0.5-dB performance improvement over hard-decision RS decoding.

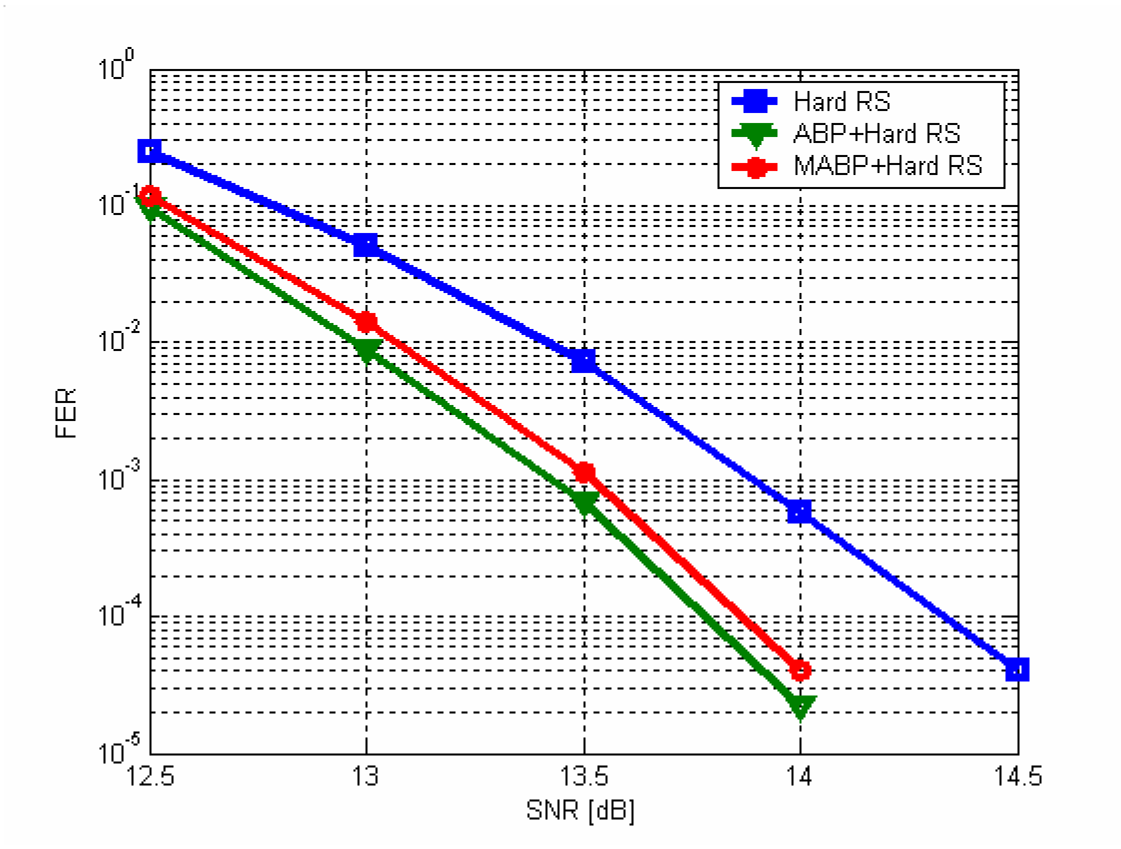


Fig. 7.9 Performance comparison of RS (186, 172) code with different decoding algorithms on an equalized Lorentzian-Gaussian channel, S_c=3.0127, $\lambda = 0.1$, $j_{\max} = 5$ with 90% jitter noise.

7.7 Summary

In Chapter 7, we have investigated the performance and implementation of iterative soft-decision RS decoding algorithms over magnetic recording channels, and have shown that large decoding gains can be achieved with acceptable latency and complexity. Since a given codeword is decoded by both the BP algorithm and a traditional hard-decision RS decoding algorithm, the concern with error floors associated with BP decoding of LDPC codes is not an issue. This makes ABP decoding a very attractive option for the next generation of magnetic recording systems in terms of both performance and hardware implementation.

Bibliography

[1] V. Guruswami and M. Sudan. "Improved decoding of Reed-Solomon and algebraic-geometric codes," *IEEE Trans. Inform. Theory*, vol. 45, pp. 1757-1767, Sept. 1999.

[2] R. Koetter and A. Vardy, "Algebraic soft-decision decoding of Reed-Solomon codes", *IEEE Trans. Inform. Theory*, vol. 49, pp. 2809-2825, Nov. 2003.

[3] H. Xia and J. R. Cruz, "Application of soft-decision Reed-Solomon decoding on magnetic recording channels," *IEEE Trans. Magn.*, vol. 40, pp. 3419-3430, Sept. 2004.

[4] R.G. Gallager, "Low-density parity-check codes", *IEEE Trans. Inform. Theory*, vol. 8, pp. 21-28, Jan. 1962.

[5] R. Lucas, M. Bossert and M. Breitbach, "On iterative soft-decision decoding of linear binary block codes and product codes," *IEEE Journal Selected Areas in Commun.*, vol. 16, pp. 276-296, Feb. 1998.

[6] J. S. Yedidia, J. Chen and M. Fossorier, "Generating code representations suitable for belief propagation decoding," in *Proc. Allerton'2002*, Oct. 2002.

[7] J. Jiang and K. R. Narayanan, "Iterative soft decoding of Reed Solomon codes," to appear in *IEEE Commun. Letters*, 2004.

[8] S. Mita, H. Matsui, M. Izumita and H. Sawaguchi, "Practical iterative decoding scheme using Reed-Solomon codes for magnetic recording channels," *IEEE Trans. Magn.*, vol. 40, pp. 219-224, Jan. 2004.

[9] J. Jiang and K. R. Narayanan, "Iterative soft decision decoding of Reed-Solomon codes based on adaptive parity check matrices," to appear in *Proc. IEEE Intl. Symp. Inform. Theory*, 2004.

[10] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. Amsterdam, The Netherlands: North-Holland, 1977.

[11] D.J.C. MacKay and R.M. Neal, "Near Shannon limit performance of low density parity check codes," *Electronics Letters*, vol. 32, pp. 1645, Aug. 1996.

[12] Z.-N. Wu, *Coding and Iterative Detection for Magnetic Recording Channels.* New York, NY: Kluwer Academic Publishers, 2000.

[13] M. Luby, M. Mitzenmacher, A. Shokrollahi, D. Spielman, and V. Stemann, "Practical loss-resilient codes," in *Proc. 29th Annu. ACM Symp. Theory of Computing*, pp. 150-159, 1997.

[14] T. Richardson and R. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, vol. 47, pp. 599-618, Feb. 2000.

[15] S. B. Wicker, *Error Control Systems for Digital Communication and Storage*, Upper Saddle River, NJ: Prentice Hall, 1995.

[16] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications," in *Proc. IEEE Global Telecommun. Conf.*, pp. 1680-1686, 1989.

[17] L. R. Bahl, J. Cocke, F. Jelinek and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. 20, pp. 284-287, Mar. 1974.

[18] D. Chase, "A class of algorithms for decoding block codes with channel measurement information," *IEEE Trans. Inform. Theory*, vol. 18, pp.170-182, Jan. 1972.

[19] H. Xia, W. Tan and J. R. Cruz, "Nested codes for perpendicular recording channels," to appear in *IEEE Trans. Magn.*, 2004.

[20] T. Morita, M. Ohta and T. Sugawara, "Efficiency of short LDPC codes combined with long Reed-Solomon codes for magnetic recording channels," to appear in *IEEE Trans. Magn*., July, 2004.

[21] T. Nishiya, K. Tsukano, T. Hirai, T. Nara, and S. Mita, "Turbo-EEPRML: An EEPR4 channel with an error-correcting post-processor designed for 16/17 rate quasi-MTR code," in *Proc. IEEE Global Telecommun. Conf.*, pp. 2868-2873, 1998.

[22] H. Sawaguchi, Y. Nishida, H. Takano and H. Aoi, "Performance analysis of modified PRML channels for perpendicular recording systems," *J. Magnetism.Magnetic Materials*, vol. 235, pp. 265-272, 2001.

[23] H. Xia and J. R. Cruz, “A Reliability-based forward recursive algorithm for algebraic soft-decision decoding of Reed-Solomon codes,” to appear in *Proc. Intl. Symp. Inform. Theory and its Applications*, 2004.

[24] H. Xia and J. R. Cruz, “Reliability-based forward recursive algorithms for algebraic soft-decision decoding of Reed-Solomon codes,” submitted to *IEEE Trans. Commun.* for publication, 2004.

[25] D. J. Mackay, available at: http://www.inference.phy.cam.ac.uk/mackay/codes/data.html.

[26] H. Xia and J. R. Cruz, “On the performance of soft Reed-Solomon decoding for magnetic recording channels with erasures,” *IEEE Trans. Magn.*, vol. 39, pp. 2576-2578, Sept. 2003.

[27] J. Moon, “Signal-to-noise ratio definition for magnetic recording channels with transition noise,” IEEE Trans. Magn., vol. 36, pp. 3881-3883, Sept. 2000.

[28] H. Song, Applications of Iterative Decoding to Magnetic Recording Channels, Ph.D. dissertation, The University of Oklahoma, 2002.

Chapter 8

Conclusions

Current magnetic recording systems use RS codes and hard-decision decoders. The demand for high capacity drives future magnetic recording systems to use high areal density magnetic recording techniques such as perpendicular recording methods. This, in turn, requires better coding schemes, equalization, and channel detection to maintain the same levels of reliability.

Large coding gains have been reported by replacing RS codes with LDPC codes for magnetic recording systems, and LDPC codes have been considered as an option for the next generation magnetic recording systems. However, the uncertainty of the performance of LDPC codes at high SNR is still an issue where burst noise is the dominant noise. Compared to LDPC codes, RS codes are well suited for correcting error bursts and the performance at high SNR is fully determined. Also, the replacement of current RS-coded magnetic recording system with an LDPC-coded system entails a complete change of the system architecture.

Therefore, in this dissertation we try to shed some light on using alternative ECC decoding techniques, namely soft-decision RS decoding, for future magnetic recording systems without requiring a complete change in the system architecture.

Soft-decision RS decoding algorithms and their performance on magnetic recording channels have been researched, and the algorithm implementation and hardware architecture issues have been discussed. Several variations of the KV algorithm such as the soft Chase algorithm, the re-encoded Chase algorithm and the forward recursive algorithm have been proposed. The performance of nested codes with RS and LDPC codes as component codes have also been investigated for bursty noise magnetic recording channels.

Chapter 1 gave an overview of magnetic recording systems.

Chapter 2 gave an overview of the decoding algorithms for RS codes which includes traditional hard-decision algorithms as well as new proposed interpolation-based soft-decision RS decoding algorithms such as the GS, KV, and re-encoding algorithms. A suboptimal method to compute the multiplicity for the KV algorithm was proposed and its performance was evaluated.

Chapter 3 evaluated the application of soft-decision RS decoding algorithms on magnetic recording systems, in terms of performance, reduced-complexity implementation and burst noise protection capability. Also a soft Chase algorithm and a re-encoded Chase algorithm were proposed for both performance improvement and decoding complexity reduction.

Chapter 4 investigated the hardware implementation and architecture of the major step in interpolation-based soft-decision RS decoding algorithms, and variations of the interpolation step incorporating the reduced-complexity re-encoding algorithm have also been discussed.

Chapter 5 proposed a new soft-decision RS decoding algorithm called the forward recursive algorithm, which by utilizing the channel reliability information to determine the interpolation orders in the KV algorithm, and improved the performance compared to the original KV algorithm. Another reliability-based algorithm proposed by Jiang and Narayanan has also been investigated here for magnetic recording channels, and a modified algorithm has been proposed to reduce the decoding complexity without large performance loss.

Chapter 6 investigated the performance of nested RS codes in the presence of a mixture of random and burst noise on a perpendicular recording channel, and the performance improvement over a single RS code was observed. In addition, we described the nested LDPC codes design using RS-based LDPC codes and their performance was evaluated.

Chapter 7 discussed the iterative RS decoding algorithm using the message passing scheme, and its performance has been evaluated on magnetic recording channels. Also a reduced-complexity modification has been proposed with a small degradation in performance.