The Hybrid List Decoding and Chase-Like Algorithm of Reed-Solomon Codes

Wei Jin

supervisor: Dr. HongJun Xu

co-supervisor: Prof. Fambirai Takawira

Submitted to the School of Electrical, Electronic and Computer Engineering
in fulfillment of the requirements for the degree of

Master of Science in Engineering

at the

University of KwaZulu-Natal, King George V Avenue, Glenwood
Durban, 4041, South Africa

June 8 2005

The Hybrid List Decoding and Chase-Like Algorithm of Reed-Solomon Codes

by

Wei Jin

Submitted for the degree of Master of Science in Engineering

Abstract

Reed-Solomon (RS) codes are powerful error-correcting codes that can be found in a wide variety of digital communications and digital data-storage systems. Classical hard decoder of RS code can correct $t = (d_{min} - 1)/2$ errors where $d_{min} = (n-k+1)$ is the minimum distance of the codeword, n is the length of codeword and k is the dimension of codeword. Maximum likelihood decoding (MLD) performs better than the classical decoding and therefore how to approach the performance of the MLD with less complexity is a subject which has been researched extensively. Applying the bit reliability obtained from channel to the conventional decoding algorithm is always an efficient technique to approach the performance of MLD, although the exponential increase of complexity is always concomitant. It is definite that more enhancement of performance can be achieved if we apply the bit reliability to enhanced algebraic decoding algorithm that is more powerful than conventional decoding algorithm.

In 1997 Madhu Sudan, building on previous work of Welch-Berlekamp, and others. discovered a polynomial-time algorithm for decoding low-rate Reed- Solomon codes beyond the classical error-correcting bound $t = (d_{min} - 1)/2$. Two years later Guruswami and Sudan published a significantly improved version of Sudan's algorithm (GS), but these papers did not focus on devising practical implementation. The other authors, Kotter, Roth and Ruckenstein, were able to find realizations for the key steps in the GS algorithm, thus making the GS algorithm a practical instrument in transmission systems. The Gross list algorithm, which is a simplified one with less decoding complexity realized by a reencoding scheme, is also taken into account in this dissertation. The fundamental idea of the GS algorithm is to take advantage of an interpolation step to get an interpolation polynomial produced by support symbols, received symbols and their corresponding multiplicities. After that the GS algorithm implements a factorization step to find the roots of the interpolation polynomial. After comparing the reliability of these codewords which are from the output of factorization, the GS algorithm outputs the most likely one. The support set, received set and multiplicity set are created by Koetter-Vardy (KV) front end algorithm. In the GS list decoding algorithm, the number of errors that can be corrected increases to $t_{GS} = n - 1 - \left\lfloor \sqrt{(k-1)\,n} \right\rfloor$. It is easy to show that the GS list decoding algorithm is capable of correcting more errors than a conventional decoding algorithm.

In this dissertation, we present two hybrid list decoding and Chase-like algorithms. We apply the Chase algorithms to the KV soft-decision front end. Consequently, we are able to provide a more reliable input to the KV list algorithm. In the application of Chase-like algorithm, we take two conditions into consideration, so that the floor cannot occur and more coding gains are possible. With an increase of the bits that are chosen by the Chase algorithm, the complexity of the hybrid algorithm increases exponentially. To solve this problem an adaptive algorithm is applied to the hybrid algorithm based on the fact that as signal-to-noise ratio (SNR) increases the received bits are more reliable, and not every received sequence needs to create the fixed number of test error patterns by the Chase algorithm. We set a threshold according to the given SNR and utilize it to finally decide which unreliable bits are picked up by Chase algorithm. However, the performance of the adaptive hybrid algorithm at high SNRs decreases as the complexity decreases. It means that the adaptive algorithm is not a sufficient mechanism for eliminating the redundant test error patterns.

The performance of the adaptive hybrid algorithm at high SNRs motivates us to find out another way to reduce the complexity without loss of performance. We would consider the two following problems before dealing with the problem on hand. One problem is: can we find a terminative condition to decide which generated candidate codeword is the most likely codeword for received sequence before all candidates of received set are tested? Another one is: can we eliminate the test error patterns that cannot create more likely codewords than the generated codewords? In our final algorithm, an optimality lemma coming from the Kaneko algorithm is applied to solve the first problem and the second problem is solved by a ruling out scheme for the reduced list decoding algorithm. The Gross list algorithm is also applied in our final hybrid algorithm. After the two problems have been solved, the final hybrid algorithm has performance comparable with the hybrid algorithm combined the KV list decoding algorithm and the Chase algorithm but much less complexity at high SNRs.

Preface

The research work presented in this dissertation was performed by Mr Wei Jin, under the supervision of Dr. HongJun Xu and Prof Fambirai Takawira, at the University of KwaZulu-Natal's School of Electrical, Electronic and Computer Engineering, in the Centre of Radio Access and Rural Technologies. This work was partially sponsored by Telkom South Africa Ltd and Alcatel as part of the Centre of Excellence programme.

The entire dissertation, unless otherwise indicated, is the author's original work and has not been submitted in part, or in whole, to any other university for degree purposes.

Acknowledgments

I would like to express my sincere appreciation to my supervisor, Dr HongJun Xu, for his excellent supervision and academic support during my research. Without his insightful advice and guidance, the accomplishment of this thesis would not have been possible. His brilliant ideas have directly contributed to the key chapters of this dissertation.

I would also like to thank Prof. F. Takawira for his helpful suggestions. Thanks for him to giving me a chance to study in this school, which is committed to academic excellence and innovation in research.

I am greatly indebted to my parents for their warm encouragement and steady support not only for my research but also for my life. Their love is the source of my inspiration and always invigorates me to actualize my personal value. They are the most important people in my life.

I also want to thank my colleagues for the memorable time when we work together. The communications with them are unforgettable for me. They all contributed to my development.

Finally, I am also grateful for the financial support received from Telkom SA Ltd and Alcatel S.A. during my MScEng.

Contents

List of Figures

List of Tables

List of acronyms

APP	A Posteriori Probability
ARQ	Automatic Repeat Request
ASK	Amplitude Shift Keying
AWGN	Additive White Gaussian Noise
BCH	Bose-Chaudhuri-Hocquenghem
BM	Berlekamp-Massey
ECC	Error-correcting Code
FEC	Forward Error Correction
FER	Frame Error Rate
FSK	Frequency Shift Keying
GF	Galois Field
GMD	Generalized Minimum Distance
GS	Guruswami-Sudan
HEC	Hybrid Error Control
Hybrid 1	The Hybrid KV List Decoding and Chase-Like Algorithm of Reed-Solomon Codes
Hybrid 2	The Hybrid Gross List Decoding and Chase-Like Algorithm of Reed-Solomon Codes
Hybrid 3	The Hybrid Gross List Decoding and Kaneko Algorithm of Reed-Solomon Codes
IEE	Industrial Electronic Engineers
IEEE	Institute of Electrical and Electronic Engineers
JCN	Journal of Communications and Networks
KV	Koetter-Vardy
MLD	Maximum Likelihood Decoding

RR	Roth-Ruckenstein
PSK	Phase Shift Keying
RS	Reed-Solomon
SNR	Signal-to-Noise Ratio

Chapter 1

Introduction

In the twenty-first century, a technological society, digital communication is present in every aspect of our lives. Computer file transfers, network transmissions, radio communications, cellular communications and satellite data transmissions transfer information through a channel where noise exists. In each case information or data is transmitted using a finite source alphabet encoder. For the reliability of the received sequence coming from the channel which distorts the transmitted binary data, error-correcting codes are applied to a communication system to solve this problem. Error-correcting codes are obtained after adding redundancy to a binary sequence before transmitting it, so that even if a corrupted sequence is received, the original message can be recovered using the redundancy.

This thesis studies soft-decision decoding algorithms for Reed-Solomon codes which are the most popular codes in practical use today with applications ranging from CD players to deep space communication.

1.1 Digital communication systems

In this section, a basic digital communication system model is introduced with the separate components. Due to the purpose of this dissertation, the channel coding techniques, which include channel encoding and decoding, are emphatically described following the overview. Then, the description is focused on the protagonist

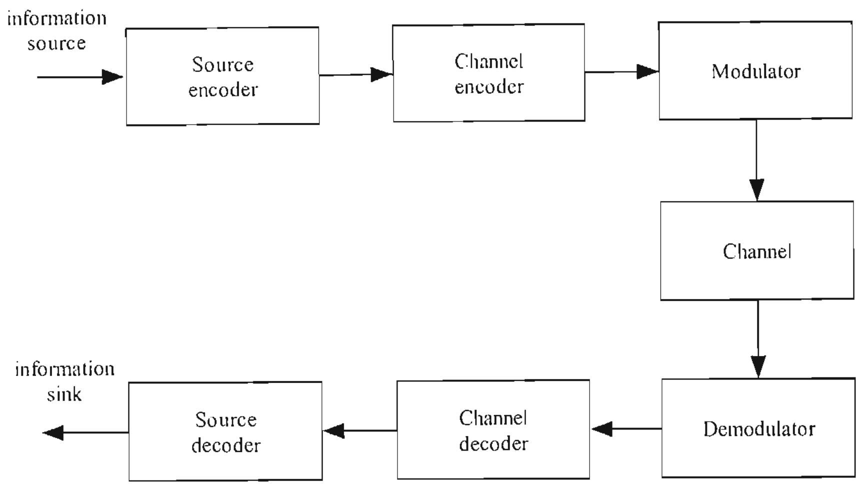

Figure 1.1: Block diagram of a communication system

of this dissertation, Reed-Solomon Codes, which play a significant role in digital communication and digital storage.

1.1.1 An overview

The intention of a communication system is to reliably transmit data without delay from a source to a receiver through an interferential channel. The existence of noise in a channel begets the transmission of information. then how to achieve accurate and real-time transmission is the reason and objective of digital communication. Figure 1.1 shows a basic point-to-point communication model which comprises source coding, channel coding, modulation, demodulation and channel.

The input of digital communication might be any source which contains information such as the English or Chinese alphabet, a voice signal from a telephone or mobile phone, the output of a sensor, photographs, a video waveform or binary symbols from a computer, etc. Whatever the source signal is, it will be formatted as a sample function of a random process.

A task for the source encoder is to convert the data from source to the digital sequence or bit prepared for the channel coding. The simplest source coding techniques involve simply applying blocks of bits to label the source signal that is

represented by a sequence of symbols from some finite alphabet. Morse code is an example that encodes the 26-symbol English letters and 10 decimal numbers to 30 combinations of "." and "-". So Morse code is an early version of variable length source encoding. An example of fixed-length source encoding is the Baudot code that was proposed by Emile Baudot in 1875. The Baudot code maps every English letter into a block of 5 binary bits. The word "communication" is expressed as follows by Baudot code,
01110 11000 11100 11100 00111 01100 00110 01110 00011 10000 00110 11000 01100.
For transmission efficiency, the other objective for source encoding is to compress the data into as few bits as possible by reducing the redundancy.

Due to the noise in a channel, channel encoding is an indispensable process for reliable transmission. The channel encoder has the function of one-to-one mapping the compressed binary sequence at the channel interface into channel outputs through adding redundancy. Reduction of the interference is obtained through enlarging the distance among the binary sequence produced by source encoding. For example, if several people living in one room are each put into a single room the possibility of their meeting each other is much lower than if they all live in one room. The particular description for channel coding is given in the following section.

The modulation schemes such as Frequency Shift Keying (FSK), Phase Shift Keying (PSK), Amplitude Shift Keying (ASK) transform the bandwidth of the data into a desired pass band region for easy transmission. After modulation, the binary sequence is replaced by an analog signal, then the channel can be shared with other-signals.

The channel is defined as the medium between the transmitted and received data such as, cable, optical fiber and magnetic disks, etc. Different types of distortions occur on the channel, such as ISI generally, additive white Gaussian noise (AWGN). The channel is that part of the communication component that is not under the control of the designer. To a source code designer, the channel might be a digital channel with capacity restriction; to a telephone-line modem designer, it might be a 4 KHz voice channel with certain bandwidth restrictions.

For the receiver, the demodulation is used to recover the digital data; the channel decoding is used to correct the errors caused by channel and to rebuild the information sequence; finally the source decoding produces the original analog signals.

1.1.2 Channel coding

Since Shannon published his classical production in 1948, which is an important theory in error correction. The theory describes the maximum attainable efficiency of an error-correcting scheme versus the levels of noise interference expected. A research subject, information theorem, was instituted. Many researchers have contributed their investigation in this field such as Kotelnikov, Wozencraft, Jacobs, Hamming, etc. Through their endeavors, many important codes have been discovered such as the Hamming code by Hamming in 1950, BCH code by Hocquenghem in 1959 and Bose and Ray-Chaudhuri in 1960, Reed-Solomon code by Reed and Solomon in 1960, Groppa code by Groppa in 1970, etc. Due to the discovery of error-correcting codes, channel coding has been ensured to be an indispensable part of digital communication system. Many error-correcting methods through using error-correcting codes are presented as Automatic Repeat Request (ARQ), Forward Error Correction (FEC), and Hybrid Error Control (HEC). Which code will be adopted in a communication system depends on the available decoding method since the final purpose of communication is to recover the information transmitted. For instance, the syndrome decoding is used for Hamming codes, the Berlekamp-Massey decoding for BCH and RS codes. Also, some soft decoding algorithm through using the soft information in a channel are proposed to assist the algebraic decoder to enhance the decoding performance, such as generalized minimum distance decoding (GMD) in [5], Chase algorithm in [25], etc. The optimal decoding method is also detected as Maximum-Likelihood-Decoding, which is infeasible in practice due to the acceptable decoding complexity. So how to achieve the performance with Maximum likelihood decoding (MLD), but lower complexity, is the dedicated objective for the decoding reacher.

The leading player for this dissertation is Reed-Solomon codes which are the most basic and well-studied codes in the literature and also used in industry for magnetic storage, optical storage, data transmission channels, satellite and even deep-space communications. In 1960, Reed and Solomon published a paper in the Journal of the Society for Industrial and Applied Mathematics, which described a new class of error-correcting codes that are now called Reed-Solomon codes. The main advantage of these codes is their high capability of correcting both random and burst errors. Many soft-decision decoding algorithms were developed for decoding Reed-Solomon codes, notably [24], [28], [31]-[33].

1.2 Motivation of research

A signal transmission with no errors is always the dream of communication companies. The most important process for this aim is the decoding performance of a channel decoder. Although the performance is exchanged by using soft information in the channel, the communication companies are always afraid to envisage the decoding complexity increased with the number of iterations. The main purpose of this dissertation is to obtain high performance with the complexity as low as possible. Our research is based on Reed-Solomon codes and an efficient algebraic decoding algorithm, whose error-correcting capability is better than classical $t = (tmin - 1)/2$, is adopted. Due to the expectation of high performance, the Chase algorithm, which is a route to achieve the MLD, is applied to the algebraic decoder. The increase in complexity is expected and is the particular problem that we want to solve. An adaptive scheme that uses a threshold value to weed out the "unreal" unreliable bits attempts to reduce the decoding complexity. From the simulation results, although the effect of an adaptive scheme is obvious, performance is reduced since the elimination of "real" unreliable bits. The severe situation encourages us to find other ways to solve the problem. Two problems before we reducing the decoding complexity should be firstly considered. After the two problems are settled by the Kaneko algorithm and Ruling out algorithm respectively, the decoding performance is the same as using the Chase algorithm, but, fortunately, the complexity is largely reduced at high signal-to-noise ratios (SNR).

1.3 Dissertation overview

This section is given as a guide for the rest of this dissertation. The background of Reed-Solomon, which includes the two encoding methods and the convention algebraic decoding algorithm, is given in chapter 2. The list decoding algorithms, which contains the Guruswami-Sudan list algorithm, the Koetter-vardy list algorithm, the Gross list algorithm, are also introduced with examples in chapter 2. Chapter 3 describes two hybrid list decoding and Chase-like algorithms with respect to the application of different list algorithms. Chapter 4 proposes an adaptive list decoding and Chase-like algorithm, which apply an adaptive scheme to achieve the comparable performance with the hybrid algorithm in chapter 3 but lower decoding complexity. In chapter 5, the Kaneko algorithm and Ruling out algorithm cooperate with the Gross algorithm to be the new hybrid algorithm with notable reduction of decoding complexity. The simulation results are distributed into respective chapters corresponding to derivation for each proposed algorithm. The conclusion is drawn in chapter 6 with the prospect of future work.

1.4 Original contributions in this dissertation

The original contributions of this dissertation include:

- In chapter 2, we present the binary weight distributions for (7,5) and (15,7) RS codes, which is the prerequisite for computing the upper bound of MLD.

- In chapter 3, we apply the Chase algorithm to two list algorithms to create two hybrid algorithms. In simulation results, the hybrid algorithm using the Gross list algorithm has a lower decoding complexity than the hybrid algorithm using the Koetter-Vardy list algorithm but with slight performance loss. A criterion for producing test error patterns for the list algorithm is also proposed.
- In chapter 4, an adaptive algorithm is applied to our hybrid algorithm to reduce the complexity after using the Chase algorithm. The simulation results are also accomplished to explain the effectiveness of the adaptive hybrid algorithm. The complexity of the hybrid algorithm in chapter 3 is reduced with respect to the better orientation for unreliable bits.
- In chapter 5, two problems that obstruct the road to achieving better performance but lower complexity, are presented. The solutions for these two problems are also given. Therefore, a new advanced hybrid algorithm which contains the Kaneko algorithm and Ruling out algorithm is created. Simulation results show that the last hybrid algorithm has the best performance but lowest complexity among the algorithms proposed in this dissertation.

Parts of the work in this dissertation have been presented and submitted for the following conferences and journals:

- Wei Jin, HongJun Xu and Fambirai Takawira, "A Hybrid List Decoding and Chase-Like Algorithm of Reed-Solomon Codes", in Proceedings of the 4th International Symposium on Information and Communication Technologies, pp. 87-92, 03 - 06 January 2005.
- Wei Jin, HongJun Xu and Fambirai Takawira, "An Adaptive Hybrid List Decoding and Chase-Like Algorithm of Reed-Solomon Codes", in Proceedings of the 12th International Conference on Telecommunications, 03 - 06 May 2005.

- Wei Jin, HongJun Xu and Fambirai Takawira, "A Hybrid Gross List Decoding and Chase-Like Algorithm of Reed-Solomon Codes", in Proceedings of the Southern African Telecommunication Networks and Applications Conference, 11 - 14 September 2005.
- Wei Jin, HongJun Xu and Fambirai Takawira, "An Adaptive Hybrid List Decoding and Chase-Like Algorithm of Reed-Solomon Codes", submitted to the Journal of South African Institute of Electrical Engineers.
- Wei Jin, HongJun Xu and Fambirai Takawira, "A Hybrid List Decoding and Kaneko Algorithm of Reed-Solomon Codes", accepted by the IEE proceedings.
- Wei Jin, HongJun Xu and Fambirai Takawira, "An Adaptive Hybrid Gross List Decoding and Chase-Like Algorithm of Reed-Solomon Codes", submitted to the JCN.

Part I

The Background

Chapter 2

Reed-Solomon Codes and the List Decoding Algorithm

This chapter includes a comprehensive description of Reed-Solomon codes, which are powerful error-correcting codes in digital communication and digital data-storage system. Two encoding methods for Reed-Solomon codes are introduced in this chapter and the corresponding binary weight distributions are also presented. The introduction of conventional decoding algorithm for Reed-Solomon codes is one part of the indispensable materials for the background.

The list algorithm and its practical implementation version, which is Koetter-Vardy (KV) list algorithm, is given in this chapter. The Gross list algorithm (Gross), which is a simplified version of the list algorithm, is also explained. The comparison of decoding performance and complexity between the KV algorithm and the Gross algorithm is a requisite for choosing the better algebraic decoding algorithm and some necessary figures are provided to serve as the illustration.

2.1 The encoding for Reed Solomon codes

The Reed-Solomon codes were discovered by Reed and Solomon in 1960. In [1], the authors defined the Reed-Solomon codes as the primary BCH codes in Galois Field (GF) (q). So it is obvious that the Reed-Solomon codes are the subset of the

BCH codes. Two encoding schemes are introduced in the following subsections.

2.1.1 Systematic encoding

The RS codes are the subset of the BCH codes. Consequently, the encoding of RS codes is introduced based on the encoding of BCH codes. Consider an (n, k) RS code and the minimum distance d_{min} is known as $n-k+1$ since the RS codes are minimum-distance-separable. The generator polynomial is obtained as follows:

$$g(x) = (x-\alpha)\left(x-\alpha^2\right)\cdots\left(x-\alpha^{d_{min}-1}\right) \tag{2.1.1}$$

where α is the primary element in GF(q). According to k independent codeword polynomials that are as follows:

$$\begin{aligned} g(x) &= g_{n-k}x^{n-k} + g_{n-k-1}x^{n-k-1} + \cdots + g_1x + g_0 \\ xg(x) &= g_{n-k}x^{n-k+1} + g_{n-k-1}x^{n-k} + \cdots + g_1x^2 + g_0x \\ &\vdots \\ x^{k-1}g(x) &= g_{n-k}x^{n-1} + g_{n-k-1}x^{n-2} + \cdots + g_1x^k + g_0x^{k-1}, \end{aligned} \tag{2.1.2}$$

we obtain the generator matrix as follows:

$$G = \begin{pmatrix} g_{n-k} & g_{n-k-1} & \cdots & g_1 & g_0 & 0 & \cdots & \cdots & 0 \\ 0 & g_{n-k} & \cdots & g_2 & g_1 & g_0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & g_{n-k} & g_{n-k-1} & \cdots & \cdots & g_1 & g_0 \end{pmatrix}. \tag{2.1.3}$$

Then the parity check matrix is achieved as follows:

$$H = \begin{pmatrix} h_0 & h_1 & \cdots & h_k & 0 & \cdots & \cdots & 0 \\ 0 & h_0 & h_1 & \cdots & h_k & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & h_0 & h_1 & \cdots & h_k \end{pmatrix} \tag{2.1.4}$$

where is the check polynomial given by:

$$h(x) = \frac{(x^n-1)}{g(x)}. \tag{2.1.5}$$

It is easy to achieve the following equation:

$$G \cdot H^{\mathrm{T}} = 0. \tag{2.1.6}$$

For the systematic encoding where T stands for transposition of H, we need to convert the generator matrix to the following formal:

$$G = [I_k P] \tag{2.1.7}$$

where I is a $k \times k$ identity matrix. Based on the generator matrix in this formal, the corresponding codeword is as the following format:

$$C(x) = f(x)\, x^{n-k} + r(x) \tag{2.1.8}$$

where $f(x)$ is an information polynomial that all coefficients come from the information sequence and $r(x)$ is a parity check polynomial that all coefficients come from the parity check sequence. The two polynomials are shown as:

$$f(x) = f_{k-1}x^{k-1} + f_{k-2}x^{k-2} + \cdots + f_1 x + f_0 \tag{2.1.9}$$

and

$$r(x) = r_{n-k-1}x^{n-k-1} + r_{n-k-2}x^{n-k-2} + \cdots r_1 x + r_0. \tag{2.1.10}$$

Using Equation(2.1.8), we get the parity check polynomial as follows:

$$r(x) = C(x) + f(x)\, x^{n-k} \equiv f(x)\, x^{n-k} (\mathrm{mod}\ g(x)). \tag{2.1.11}$$

We define

$$p_i(x) = x^{n-k}x^{k-i}\,(\mathrm{mod}\ g(\mathrm{x})) = x^{n-i}\,(\mathrm{mod}\ g(\mathrm{x})) \qquad i = \{1, 2, \cdots, k\}\,. \tag{2.1.12}$$

So the generator matrix is converted to the following format:

$$G = \begin{pmatrix} 1 & 0 & \cdots & \cdots & 0 & \tilde{p}_1(x) \\ 0 & 1 & 0 & \cdots & 0 & \tilde{p}_2(x) \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & \cdots & 1 & \tilde{p}_k(x) \end{pmatrix} = [I_k P]\,. \tag{2.1.13}$$

It is obvious that we have the following equation:

$$r(x) = \sum_{i=0}^{k-1} f_i \, \tilde{p}_{k-i}(x). \tag{2.1.14}$$

The parity check matrix is also achieved as follows based on the $p_i(x)$:

$$H = \left[P^{\mathrm{T}} I_{n-k} \right] = \left[\begin{array}{ccccc} \tilde{p}_1(x)^{\mathrm{T}} & \tilde{p}_2(x)^{\mathrm{T}} & \cdots & \tilde{p}_k(x)^{\mathrm{T}} & I_{n-k} \end{array} \right]. \tag{2.1.15}$$

Then the codeword is obtained using following equation:

$$C = (f_{k-1}, f_{k-2}, \cdots, f_1, f_0) \begin{pmatrix} 1 & 0 & \cdots & \cdots & 0 & \tilde{p}_1(x) \\ 0 & 1 & 0 & \cdots & 0 & \tilde{p}_2(x) \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & \cdots & 1 & \tilde{p}_k(x) \end{pmatrix}. \tag{2.1.16}$$

For easy understanding, the codeword polynomial is also obtained in the following way:

$$C(x) = f(x)\, x^{n-k} + r(x) = f(x)\, x^{n-k} + f(x)\, x^{n-k} \, (\mathrm{mod}\; g(x)). \tag{2.1.17}$$

2.1.2 Evaluation map encoding

The evaluation map encoding is to create the generator polynomial based on the information sequence.

Suppose we want to transmit a message $(f_0, f_1, \cdots, f_{k-1})$ using an (n, k) RS code. Then the generator polynomial is produced in the following convenient way:

$$g(x) = g_0 + g_1 x + \cdots + g_{k-1} x^{k-1}. \tag{2.1.18}$$

The codeword is obtained as follows:

$$C = \left(g\left(\alpha^0\right), g\left(\alpha^1\right), \cdots, g\left(\alpha^{n-1}\right) \right). \tag{2.1.19}$$

We also can use the generator matrix to produce the codeword. The generator matrix G of the evaluation map encoder is shown as follows:

$$G = \begin{pmatrix} \alpha^{(k-1)(n-1)} & \cdots & \alpha^{2(n-1)} & \alpha^{(n-1)} & 1 \\ \vdots & & \vdots & \vdots & \vdots \\ \alpha^{2(n-1)} & \cdots & \alpha^4 & \alpha^2 & 1 \\ \alpha^{(n-1)} & \cdots & \alpha^2 & \alpha & 1 \\ 1 & \cdots & 1 & 1 & 1 \end{pmatrix} \tag{2.1.20}$$

and the codeword is consequently achieved as the following equation:

$$C = (f_{k-1}, f_{k-2}, \cdots, f_1, f_0) \begin{pmatrix} \alpha^{(k-1)(n-1)} & \cdots & \alpha^{2(n-1)} & \alpha^{(n-1)} & 1 \\ \vdots & & \vdots & \vdots & \vdots \\ \alpha^{2(n-1)} & \cdots & \alpha^4 & \alpha^2 & 1 \\ \alpha^{(n-1)} & \cdots & \alpha^2 & \alpha & 1 \\ 1 & \cdots & 1 & 1 & 1 \end{pmatrix}. \tag{2.1.21}$$

The RS code obtained through the evaluation map encoding is not a systematic code.

After making the comparison between these two encoding methods, the conclusion is evident that the evaluation map encoding has less complexity than the systematic encoding. The advantage of the systematic encoding is embodied in transferring the codeword to information sequence. As a result that all the k input symbols to the encoder explicitly appear in the k consecutive positions of encoded codeword, the information sequence is read off directly from the codeword.

The two encoding methods produce the different codewords aiming at the same information sequence. An example is given to explain this conclusion.

Example 1: We consider the (7,5) RS code, which the minimum distance $d_{min} = 3$ and the conventional error-correcting capability is 1. Based on the format $(f_0, f_1, \cdots, f_{k-2}, f_{k-1})$, the message that we want to transmit is $(\alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5)$. The generator polynomial of the systematic encoder is $g(x) = (x - \alpha)(x - \alpha^2) = \alpha^3 + \alpha^4 x + x^2$ while the generator polynomial of the evaluation map encoder is $g(x) = \alpha + \alpha^2 x + \alpha^3 x^2 + \alpha^4 x^3 + \alpha^5 x^4$. The systematic encoder generates code-

Table I: Binary weight distribution of (7,5) RS code

A_0, A_{21}	A_1, A_{20}	A_2, A_{19}	A_3, A_{18}	A_4, A_{17}	A_5, A_{16}	A_6, A_{15}	A_7, A_{14}	A_8, A_{13}	A_9, A_{12}	A_10, A_{11}
1	0	0	28	84	273	924	1956	2982	4340	5796

Table II: Binary weight distribution of (15,7) RS code

A_1, A_{59}	A_2, A_{58}	A_3, A_{57}	A_4, A_{56}	A_5, A_{55}	A_6, A_{54}	A_7, A_{53}	A_8, A_{52}	A_9, A_{51}	A_{10}, A_{50}
0	0	0	0	0	0	0	0	0	0
A_{11}, A_{49}	A_{12}, A_{48}	A_{13}, A_{47}	A_{14}, A_{46}	A_{15}, A_{45}	A_{16}, A_{44}	A_{17}, A_{43}	A_{18}, A_{42}	A_{19}, A_{41}	A_{20}, A_{40}
120	280	1260	4590	12796	33030	89160	226320	472080	913404
A_{21}, A_{39}	A_{22}, A_{38}	A_{23}, A_{37}	A_{24}, A_{36}	A_{25}, A_{35}	A_{26}, A_{34}	A_{27}, A_{33}	A_{28}, A_{32}	A_{29}, A_{31}	A_{30}
1864640	3511560	5458080	7872220	12072048	17266800	20466880	22638345	26671800	29284628

word $(\alpha^6, 1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5)$ while the evaluation map encoder generates codeword $(\alpha^2, \alpha^3, \alpha^3, \alpha^5, \alpha^2, \alpha^5, \alpha)$.

Example 1 shows that the codewords obtained from different encoders are distinct. Although the codewords obtained from two encoders for the same information sequence are different, the binary weight distributions, which are important for computing the performance of Maximum-Likelihood-Decoding, are the same. It means that the two encoding methods use the same q^k combinations in total q^n combinations to be the codewords. The alternate way for proving this conclusion is to consider whether the RS code generated by an evaluation map encoder is cyclic. If it is cyclic, then we can use the standard shift register encoder and generator strictly systematic code like the code produced by systematic encoding method. The consideration is proved in [19].

Because analytical methods for calculating the weight distribution of their binary images are not known, we only can obtain binary weight distribution for short RS codes through computation. We define A_w as the number of codewords with binary Hamming weight w. The binary weight distributions for (7,5) and (15,7) RS codes are shown in Table I and II, respectively.

As the results of that the q for GF is the power of 2 usually, the derived binary

RS codes are easily obtained through transferring the GF (q) into the binary field. The (n, k) RS codes in GF(q) are transferred to (mn, mk) binary RS codes, where $q = 2^m$. For example, the codeword $(\alpha^6, 1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5)$ in GF(8) is transferred to the codeword $(101, 001, 010, 100, 011, 110, 111)$ in the binary field. The derived binary RS codes not only have the capability to correct $t = (d_{min} - 1)/2$ random bit errors that is the same as symbol error-correcting capability of q-ary RS codes, but also can correct the B burst bit errors with any error patterns, where B is given as:

$$B \leq (t-1)m + 1. \tag{2.1.22}$$

If a codeword that has suffered error sequence with the length greater than B is received, then the conventional decoder probably has no ability to output the correct codeword because there is the probability that the span of error sequence exceeds the conventional symbol error-correcting capability of RS codes in GF(q). For example, if we consider the (7,3) RS code that can correct 2 symbol errors, the (21,9) derived binary RS code has the capability to correct 2 random bit errors and 4 burst bit errors. If the codeword has the error pattern with 5 burst bit errors, then the conventional decoder cannot produce the correct codeword if the following situation happens:

codeword :	$(\alpha^6, 1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5)$
codeword in GF(2) :	$(101, 001, 010, 100, 011, 110, 111)$
error pattern :	$(000, 000, 001, 111, 100, 000, 000)$
hard decision :	$(101, 001, 011, 011, 111, 110, 111)$
hard decision in GF(8) :	$(\alpha^6, 1, \alpha^3, \alpha^3, \alpha^5, \alpha^4, \alpha^5)$.

The conventional decoder cannot produce the correct codeword for the above example because the hard decision has 3 symbol errors.

The two encoding methods are introduced in this section and the comparison between two encoding methods is also accomplished. To calculate the performance of MLD, we give the binary weight distribution for (7,5) RS code and (15,7) RS code respectively in Table I and Table II. The derived binary RS codes, which are powerful error-correcting codes especially for burst errors, are presented at the end of this section.

2.2 The conventional decoding algorithm for Reed-Solomon codes

Whether the code is widely applied in practice depends on whether the decoder is simple, swift, economic and has low error probability. Peterson gave the theoretic solution for binary BCH codes in 1960; afterwards the decoding was extended to the non-binary codes by Gore and Zierler. In 1966, Berlekamp proposed an iteration decoding algorithm [2] that was further simplified by Massey in 1969 [3]. Thereafter many researchers presented new decoding methods for the BCH codes, for instance the Euclid's algorithm [4].

The Berlekamp-Massey (BM) algorithm, which is regarded as the classical decoding algorithm for BCH codes and RS codes and has the most extensive application in the communication field, is introduced in this section. Founded on the decoding procedure of linear block codes, the decoding of RS codes also includes three steps. First step is to obtain the syndrome S from hard decision R (received word) that is obtained from received sequence. The second step is to find the error pattern E based on the S. The last step is to obtain the codeword C by adding of R and E. Readers can refer to [2] for more details.

2.3 The Guruswami-Sudan list algorithm

The BCH decoder is an algebraic decoder, which exploits the potential algebraic structure of the code to generate codeword by the assistance of finite field arithmetic. The error-correcting capability of BCH decoder for (n, k) RS code is known as $t = (d_{min} - 1)/2$ where $d_{min} = n - k + 1$ is the minimum distance of the code. The origin of the conventional error-correcting capability is explained as below. The conventional decoder is thinking to search the Hamming sphere centered at the received word for codeword. If the sphere contains a unique codeword, that is the output of decoder. Otherwise the decoder reports failure. It means that the sphere cannot contain more than one codeword. We consider the situation that the received word appears in the midpoint of two codewords that has the minimum distance $d_{\min}$. The distances of the received word from two codewords are the same, which is $d_{min}/2$. So the radius of a sphere must be less than $d_{min}/2$ according to the successful decoding's condition that the sphere contains only one codeword. Therefore the error-correcting capability of conventional decoder is $t = (d_{min} - 1)/2$. From this standpoint, it seems to be unfeasible to use an algebraic decoder to correct more errors than t. Nevertheless, we may reach a different conclusion if we examine the probability that the decoding sphere will contain multiple codewords.

Consider the (32,8) RS code in GF(32), with $d = 25$ and $t = 12$. Now, we set the radius for decoding sphere to 13, which exceeds the errors that can be corrected by conventional algebraic decoder, and the transmitted codeword suffers 13 errors. The decoding sphere has high probability of containing more than one codeword as discussed above. The decoding sphere may contain two codewords: the transmitted codeword and one other, a codeword at a distance of 12 or 13 from the received word. We assume that all error patterns of weight 13 are equally likely, the probability of this unfavorable happening is 2.08437×10^{-12}. It means that the sphere usually contains only one codeword, and withal, the unique codeword is the most probable to be the transmitted codeword. In short, the code is capable of correcting virtually all patterns of 13 errors, despite having a conventional error-correcting capability of only 12.

The example indicates that it might be possible to design a decoding algorithm for RS codes, which is more formidable than the conventional decoder since a greater capability is anticipated. The Guruswami-Sudan (GS) list decoding algorithm is proposed in [9] due to this fact and based on research done before such as [6], [7] and [8]. The GS list algorithm is proved to have the greater capability than conventional decoding algorithm for most of codes. The number of errors that can be corrected by the GS list algorithm has up to t_{GS} as below:

$$t_{GS} = n - 1 - \left\lfloor \sqrt{(k-1)\,n} \right\rfloor . \tag{2.3.1}$$

With the increase of multiplicity that is one of the determinants of the complexity of the GS algorithm, the error-correction capability of the GS algorithm is also increased, and we denote it as t_m. The main idea of GS list algorithm is to reconstruct the generator polynomial for received sequence through using a polynomial interpolation, which is associated with the reliability contributed by soft information. Some basic concepts and background are indispensable for comprehending the GS algorithm and need to be introduced first. Consider a bivariate polynomial constructed in finite field as follows:

$$P(x,y) = \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} p_{i,j} x^i y^j \tag{2.3.2}$$

where $p_{i,j}$ are the coefficients of the polynomial. We set w_x and w_y to be non-negative real numbers. The $(w_x, w_y)-$weighted degree of $P(x,y)$, which we denote it as $\deg^{(w_x,w_y)}(P)$, is defined as the maximum value over all the numbers iw_x+jw_y. To define the $(u,v)'$th formal derivative of a polynomial $P(x,y)$ over GF (2^q), we use the *Hasse* derivative in [15]. The $(u,v)'$th *Hasse* derivative of a bivariate polynomial $P(x,y)$ is shown as:

$$P^{[u,v]}(x,y) = \sum_{i\geq u \cup j\geq v} \begin{pmatrix} i \\ u \end{pmatrix}\begin{pmatrix} j \\ v \end{pmatrix} p_{i,j} x^{i-u} y^{j-v} \tag{2.3.3}$$

where $u, v \geq 0$. As a reason for using polynomial interpolation to build the polynomial, we need to ensure that the bivariate polynomial passes through the definite points. We define that a bivariate polynomial $P(x,y)$ passes through a point (x_i, y_j) with multiplicity m if $P^{[u,v]}(x_i, y_j) = 0$ for all integers u, v, such that $u + v < m$. Consider a codeword C for (n,k) RS code is transmitted and the hard decision is R, therefore $R = C + E$, where E is the error pattern. The codeword is created by the evaluation map encoding based on Equation(2.1.9) as

$c_i = g(x_i), 0 \leq i < n$. After the R is obtained, a list of points is formed by the x_i and the received symbol y_i as follows:

$$L = \{(x_0, y_0), (x_1, y_1), \cdots, (x_{n-1}, y_{n-1})\}. \quad (2.3.4)$$

If $E = 0$, then we have $y_i = c_i = g(x_i), 0 \leq i < n$. The polynomial interpolation is used to restructure the bivariate polynomial, $P(x, y) = y - g(x)$, which passes through all the points in L with their respective multiplicities m. Guruswami and Sudan imply that the utilization of soft information has the possibility to be magnified in [9] by allowing each point on the polynomial interpolation to have its own multiplicity. The multiplicity is an adjustable integer parameter and reflects the reliability of received symbol. The multiplicity can decide the decoding radius t_m, which influences the error-correcting capability of the GS algorithm. Given a received word, the GS algorithm returns a list which includes all codewords with distance t_m or less from the received word. So the m is also regarded as a user-selectable complexity parameter. The error-correcting capability of the GS list algorithm increases as the m increases, but unfortunately, the decoding complexity is also increased. The method for attaining multiplicities from a received sequence is diversiform and still being looked at by researchers. As a result of the employment of soft information in the decision of multiplicities, the GS list algorithm is not an unsophisticated algebraic decoding algorithm but an algebraic soft-decision decoding algorithm. After passing through all the points in L, an interpolation polynomial is acquired as the ultimate bivariate polynomial. The GS list algorithm ensures that under certain conditions, the codeword generator polynomial lives inside the interpolation polynomial. So the interpolation-based list decoding algorithm can be the alias of the GS algorithm. The GS list algorithm, which is the version that admits the different multiplicities for each point, concludes two main steps as below:

1. Interpolation Step: Build the list of points L, compute the interpolation polynomial $P(x, y) = y - g(x)$ of minimal $(1, k - 1)$ – weighted degree that

passes through all the points in the list with their respective multiplicities m.

2. Factorization Step: Given the interpolation polynomial $P(x,y)$, identify all the factors of $P(x,y)$ as the form $y-f(x)$ with $\deg f(x) < k$. The algorithm produces a list of codewords according to these factors.

It is fortunate that the complete factorization of $P(x,y)$ is not necessary since the degree of generator polynomial is less than k.

The GS list algorithm, which is capable of correcting more errors than conventional algebraic decoding algorithm, is introduced in this section as an algebraic soft-decision decoding algorithm. Some notation and facts are given for preparatory data structure for the algorithm. Finally, the GS list algorithm is concluded with two main steps: interpolation step and factorization step.

2.4 The Koetter-Vardy list algorithm

Since the GS list algorithm has the more powerful error-correcting capability than the conventional decoding algorithm, it has been actively researched. It is a pity that the main focus of [8] and [9] is to establish the existence of the interpolation-based list decoding algorithm, and not to devising practical implementations. However, several later authors, particularly Ralf Kotter, Alexander Vardy, R. M. Roth and G. Ruckenstein find the realizations for the important steps in the GS list algorithm, thus promoting the GS algorithm to be a practical implementation for digital communication and data-storage system.

The Koetter-Vardy list algorithm proposed by Kotter and Vardy in [13], which includes the method for the acquirement of unequal multiplicities related to their reliability, and the realization for interpolation step, is recommended in this section. The contribution of R. M. Roth and G. Ruckenstein for the actualization of the factorization step is indispensable and deservedly occupies a position in this section. We divide the applied Koetter-Vardy list algorithm into four steps since every step accomplishes an unattached function: reliability matrix, multiplicity matrix, the Koetter-Vardy's interpolation step and the Roth-Ruckenstein solution for the factorization step.

2.4.1 Reliability matrix

The reliability matrix created by all possible transmitted and received symbol pairs including the hard decision is the preparation of a multiplicity matrix. Consider a codeword of (n, k) RS code in GF(q) is transmitted through a memoryless channel. The reliability for a received symbol β_j according to the symbol α_i was sent is the a posteriori probability (APP) as follows:

$$\pi_{i,j} = p_r\left(\alpha_i \text{ sent} \mid \beta_j \text{ received}\right) \tag{2.4.1}$$

where $0 \leq i < q$ and $0 \leq j < n$. Related to the noise probability density function (PDF) is $\eta\left(\beta_j \mid \alpha_i\right)$, the $\pi_{i,j}$ is calculated as follows by Bayes's Rule:

$$\pi_{i,j} = p_r\left(\alpha_i \mid \beta_j\right) = \frac{\eta\left(\beta_j \mid \alpha_i\right)}{\sum_{\alpha_k \in GF(q)} \eta\left(\beta_j \mid \alpha_k\right)}. \tag{2.4.2}$$

The reliability matrix Π is constructed by $\pi_{i,j}$ and has the size of $(q \times n)$ and columns sum are unity, which is judged by the property of $\pi_{i,j}$. The hard decision vector $r = (r_0, r_1, \cdots, r_{n-1})$ also stay in the matrix with the following value:

$$r_j = \max\left(\pi_{i,j}\right) \quad i = \{0, 1, \cdots, n\} \ . \tag{2.4.3}$$

2.4.2 Multiplicity matrix

The multiplicity matrix is translated from the reliability matrix and the procedure is similar to the reliability matrix and is quantized to some positive real number. The different approaches of quantization process dissimilar multiplicity matrix. However, the diversified multiplicity matrixes corresponds to the discrepant reflection degree for the reliabilities of $(q \times n)$ symbols despite all of multiplicity matrixes having the ability to mirror the soft information in received sequence. Two algorithms for constructing the multiplicity matrix are introduced in this section.

Algorithm 1

Algorithm 1 proposed in [13] is a soft-decision front end for generating the multiplicity matrix M from Π subject to the constraint s, which is the summation of the value for non-zero items in M. The adjustable parameter s, which is one of two factors that decide the complexity of the algorithm, and the other one is the length of the codeword, is calculated as below:

$$s = \sum_{i=0}^{q-1} \sum_{j=0}^{n-1} m_{i,j}. \tag{2.4.4}$$

Algorithm 1 is shown as follows according to the constraint s.

Multiplicity Algorithm 1 for calculating Multiplicity Matrix from reliability matrix subject to complexity constraint s

---

```
Choose a desired value for s = $\sum_{i=0}^{q-1} \sum_{j=0}^{n-1} m_{i,j}$
$\prod^* \leftarrow \prod$; $M \leftarrow 0$
while $s > 0$ do
    Find the position $(i, j)$ of the largest entry $\pi^*_{i,j}$ in $\prod^*$
    $\pi^*_{i,j} \leftarrow \frac{\pi_{i,j}}{m_{i,j}+2}$; $m_{i,j} \leftarrow m_{i,j} + 1$; $s \leftarrow s - 1$
end while
```

The complexity of algorithm 1 is considerable since it has to search through a $(q \times n)$ reliability matrix s times. Consider the performance of the list decoder is the same as the hard-decision decoder we need $s \geq n$. Therefore the complexity of algorithm 1 is $O((n+1)(n)(n)) = O(n^3)$.

Algorithm 2

Algorithm 2 proposed in [14] is a simplified version of algorithm 1 and has the lower complexity $O(n^2)$. Algorithm 2 is derived based on the situation that the reliability matrix Π has the entry that is equal to 1 for almost every column at high SNRs and it wastes the memory and computational resources to search and store every zero entries in Π. Algorithm 2 is constructed by a tunable parameter λ with a signal pass through Π.

Algorithm 2 is shown as follows according to the parameter λ.

Multiplicity Algorithm 2 for calculating Multiplicity Matrix from reliability matrix Π subject to parameter Π

```
for $i = 1$ to $n$ do
    for $j = 1$ to $q$ do
        $m_{i,j} \leftarrow \lfloor \lambda \pi_{i,j} \rfloor$
    end for
end for
```

Although algorithm 2 has less complexity than algorithm 1, algorithm 1 is still worthwhile because it has the optimal search method for performance. After the comparison between two algorithms, the multiplicity matrix constructed by algorithm 1 has more accurate reflection for reliability matrix than algorithm 2 since algorithm 1 arranges the symbols in Π to a queue in turn. So algorithm 1 has a better performance than algorithm 2 in the condition that the complexities of two algorithms are the same. Unless specifically indicated otherwise, algorithm 1 is applied in this dissertation.

2.4.3 Interpolation step

The interpolation step is the key step for the GS list algorithm and was firstly realized by Koetter. The Koetter's solution for interpolation problem is introduced and applied in this thesis based on the consistency of the KV soft-decision front end although some new solutions are proposed by other authors, particularly the Feng-Tzeng algorithm in [16].

As mentioned in the GS list algorithm, the purpose of the interpolation step is to create a bivariate polynomial that passes through all symbol pairs given by algorithm 1 with their respective multiplicity. Then the final polynomial must satisfy the following equation:

$$P^{[u,v]}(x_j, y_i) = 0 \quad u+v < m_{i,j} \tag{2.4.5}$$

for all symbol pairs in L. The complexity of KV interpolation step is measured by the number of the equations or constraints that need to be satisfied for the interpolation polynomial. Based on the $(q \times n)$ multiplicity matrix, the complexity of the KV interpolation step, which is also called the cost of interpolation step, is derived as:

$$C = \frac{1}{2}\sum_{i=0}^{q-1}\sum_{j=0}^{n-1} m_{i,j}(m_{i,j}+1). \tag{2.4.6}$$

Based on the GS list algorithm, the maximum number of generator polynomials that can be factorized from the interpolation polynomial is restricted as following value:

$$d_y = \left\lfloor \frac{1+\sqrt{1+\frac{8C}{k-1}}}{2} \right\rfloor - 1. \tag{2.4.7}$$

This value is also the restriction of the maximum degree of y in interpolation polynomial. Then the maximum degree of x in interpolation polynomial is limited according to the d_y as follows:

$$d_x = \left\lfloor \frac{C}{d_y+1} + \frac{d_y(k-1)}{2} \right\rfloor. \tag{2.4.8}$$

At the end of every interpolation iteration, the KV interpolation step produces $d_y + 1$ polynomials that all satisfy the constraints. As the reason for reducing the complexity, the polynomial has the smallest $(1, k-1)$ – weighted degree is chosen to be the final interpolation polynomial. Obviously, the complexity of this step is decided by the multiplicity $m_{i,j}$ and the length of the codeword. From the Equation(2.4.7), we get the conclusion that $m_{i,j}$ also has the ability to determine the maximum degree of y in interpolation polynomial, which can be explained to the number of factors in interpolation polynomial. It means that as the multiplicity increases the more candidate codewords are obtained. As mentioned in section 2.3, the decoding radius is assuredly extended and the error-correcting performance is improved. So one hesitates to choose the multiplicity to balance the complexity and the performance.

The algorithm of the KV interpolation step is shown as follows.

Interpolation Algorithm

```
$G \leftarrow \left\{g_0 = 1, g_1 = y, g_2 = y^2, \cdots, g_{d_y} = y^{d_y}\right\}$
$\deg(g_0) = 0, \deg(g_1) = k-1, \deg(g_2) = 2(k-1), \cdots, \deg(g_{d_y}) = d_y(k-1)$
for each point $(x_j, y_i)$ with multiplicity $m_{i,j} > 0$ do
  for $\alpha \leftarrow 0$ to $m_{i,j} - 1$ do
    for $\beta \leftarrow 0$ to $m_{i,j} - \alpha - 1$ do
      $f \leftarrow \min_{\deg(1,k-1)} \left\{g \in G \text{ such that } g^{[\alpha,\beta]}(x_j, y_i) \neq 0\right\}$
      for $g \in G$ such that $g \neq f$ do
        $g \leftarrow g \cdot f^{[\alpha,\beta]}(x_j, y_i) - f \cdot g^{[\alpha,\beta]}(x_j, y_i)$
      end for
      $f \leftarrow (x - x_j) f, \deg(f) + +$
    end for
  end for
end for
$P(x,y) = \min_{\deg(1,k-1)} \{G\}$
```

2.4.4 Factorization step

The intention of the factorization step is to find all generator polynomials that exist in the interpolation polynomial $P(x,y)$. The Roth-Ruckenstein (RR) algorithm in [23], which is the most efficient algorithm currently known for solving the factorization problem, is introduced as one significant portion of the KV list algorithm.

The algorithm of RR factorization step is shown as follows.

Factorization Algorithm

```
begin (Given$P(x,y), k$)
$f_a(0) = \text{Null}, \deg(0) = -1, P_0(x,y) = P(x,y), t = 1, u = 0$
$DFS(u)$
end
/* $DFS(u)$: Depth – first – search beginning at $u$ */
if $P_u(x,0) = 0$
    output $f_u(x) = Coeff(u)\, x^{\deg(u)} + Coeff(f_a(u))\, x^{\deg(f_a(u))} + \cdots$
    else if $\deg(u) < k$
      $R$ = Chien search $(P_u(0,y))$
      for $r \in R$ do
        $v = t, t = t + 1$
        $f_a(v) = u, \deg(v) = \deg(u) + 1, Coeff(v) = r$
        $P_v(x,y) = \langle\langle P_u(x, xy + r)\rangle\rangle$
        $DFS(v)$
      end for
    end if
end if
```

The $\langle\langle P(x,y)\rangle\rangle$ is defined as:

$$\langle\langle P(x,y)\rangle\rangle = \frac{P(x,y)}{x^d} \tag{2.4.9}$$

and d is the maximum value that satisfies $x^d | P(x,y)$. The depth-first-search algorithm adopted in factorization step is described in [17]. Figure 2.1 shows us the procedure of the KV list algorithm.

An example in small field is necessary to demonstrate how the KV list algorithm works.

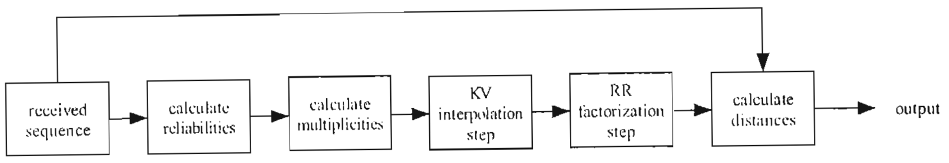

Figure 2.1: The Koetter-Vardy list algorithm

Example 2: Consider a (7,5) RS code, which has the minimum distance $d_{\min} = 3$ and the conventional decoder which has capability to correct single symbol error. The information sequence that we want transmitted is:

$$f = \left(1, \alpha^5, 1, \alpha^2, \alpha^3\right).$$

Then the generator polynomial is $f(x) = 1+\alpha^5 x+x^2+\alpha^2 x^3+\alpha^3 x^4$ if the evaluation map encoding is adopted. Consequently the codeword is:

$$C = \left(0, \alpha^4, \alpha, \alpha, 1, \alpha^4, 0\right).$$

The codeword is transmitted bit-by-bit using BPSK modulation over an AWGN channel and the reliability matrix is built through the received sequence as follows:

$$\Pi = \begin{pmatrix} 0.8244 & 0.0000 & 0.0116 & 0.6032 & 0.0000 & 0.0000 & 0.7021 \\ 0.1752 & 0.0000 & 0.0000 & 0.0000 & 0.2481 & 0.0000 & 0.2955 \\ 0.0002 & 0.0001 & 0.9813 & 0.3881 & 0.0000 & 0.0070 & 0.0004 \\ 0.0002 & 0.0242 & 0.0001 & 0.0053 & 0.0000 & 0.0017 & 0.0013 \\ 0.0000 & 0.0000 & 0.0002 & 0.0000 & 0.0007 & 0.0005 & 0.0002 \\ 0.0000 & 0.9274 & 0.0068 & 0.0034 & 0.0000 & 0.9211 & 0.0000 \\ 0.0000 & 0.0472 & 0.0000 & 0.0000 & 0.0020 & 0.0696 & 0.0000 \\ 0.0000 & 0.0012 & 0.0000 & 0.0000 & 0.7493 & 0.0001 & 0.0005 \end{pmatrix}.$$

The hard decision r, which is composed by the maximum reliability values in every column, is obtained as:

$$r = \left(0, \alpha^4, \alpha, 0, \alpha^6, \alpha^4, 0\right).$$

The conventional decoding algorithm is incapable of decoding this received sequence and this conclusion is testified by checking the hard decision that has two errors from the transmitted codeword. However, the KV list decoding algorithm exhibit it's strength in this example. The multiplicity matrix is achieved in the first instance by running multiplicity algorithm 1 with $s = 12$:

$$M = \begin{pmatrix} 2 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

Then the cost of interpolation step is calculated as:

$$C = \frac{1}{2}\sum_{i=0}^{7}\sum_{j=0}^{6} m_{i,j}\left(m_{i,j}+1\right) = 16$$

deservedly, the $d_y = 2$ and $d_x = 9$. The soft information of multiplicity matrix is summarized as three sets: support set, received set and multiplicity set as:

$$\begin{array}{ll} \text{support set :} & (1, \alpha, \alpha^2, \alpha^3, \alpha^3, \alpha^4, \alpha^5, \alpha^6) \\ \text{received set :} & (0, \alpha^4, \alpha, 0, \alpha, \alpha^6, \alpha^4, 0) \\ \text{multiplicity set :} & (2, 2, 2, 1, 1, 1, 2, 1). \end{array}$$

The interpolation polynomial, which has the minimum $(1,4)$ – weighted degree of $P(x,y)$ among the $(d_y + 1) = 3$ bivariate polynomials, restricted by d_y and d_x is produced by using interpolation algorithm according to the three sets as:

$$\begin{aligned} P(x,y) = {} & \alpha^5x^5y + \alpha x^9 + \alpha y^2 + \alpha^2x^3y + \alpha^4x^2y + \alpha^3x^6 + \alpha^6xy + \alpha x^5 \\ & + \alpha^5x^4 + \alpha^6x^3 + \alpha^4x^2 + \alpha^6x + \alpha. \end{aligned}$$

Next, We use RR factorization to find the roots of interpolation polynomial $P(x,y)$ with the form $y - f(x)$ where $\deg(f(x)) < k$. There are two candidates for $f(x)$ in the list:

$$\begin{aligned} \hat{f}_1(x) &= 1 + \alpha x^2 + \alpha^4x^3 + \alpha^3x^4 \\ \hat{f}_2(x) &= 1 + \alpha^5x + x^2 + \alpha^2x^3 + \alpha^3x^4. \end{aligned}$$

Then the corresponding codewords are obtained as:

$$\hat{C}_1 = (\alpha^4, \alpha, \alpha^3, \alpha^5, \alpha^4, \alpha^2, \alpha^3)$$
$$\hat{C}_2 = (0, \alpha^4, \alpha, \alpha, 1, \alpha^4, 0)\,.$$

After checking the probability of each candidate codeword through reliability matrix Π, the KV decoder output the $\hat{C}_2$ that is more likely and then the information sequence $\hat{f}_2$ is locked. So the KV list decoding algorithm is visible in this example.

The KV list decoding algorithm, which is a practical implementation for the GS list algorithm, is illustrated in this section. The interpolation and factorization step are realized in practice and the function of the GS algorithm is accomplished.

2.5 The Gross list algorithm

As the interpolation and the factorization step are the most time-consuming component of the KV list algorithm and the complexity, which is indicated as $O\left(d_y C^2\right)$ for interpolation step, is unacceptable if the large multiplicity is used to a long RS code , Warren J.Gross et al have published a reduced-complexity KV list decoding algorithm in [19]-[21]. The systematic encoding and reencoding scheme are used to reduce the complexity of the interpolation step and the factorization step in the KV list algorithm through reducing the number of symbols prepared for the two above steps. The systematic encoding is adopted in the Gross algorithm. The introduction of the Gross list decoding algorithm is given below.

2.5.1 Reencoding

After transmitting the codeword c over an AWGN channel, the hard decision is obtained as $r = c + e$, where e is the error sequence. The idea of reencoding is to reduce the number of received symbols based on their reliability. The first step is to partition the hard decision r into two sets, an unreliable set (U) that includes $n - k$ unreliable symbols and the reliable set (R) that includes k reliable symbols. Then apply the arbitrary-position systematic encoding, which the information appears in arbitrary positions and can be implemented by an erasures-only RS decoder which has been proposed in [22], to the R set to get a reencoded codeword ψ. Taking the difference between r and ψ we get:

$$r^{'} = r - \psi = (c + e) - \psi = (c - \psi) + e \tag{2.5.1}$$

which is a codeword that has the same error pattern with r. However, the new codeword $r^{'}$ only includes at most $n - k$ non-zero symbols due to the application of systematic encoding, and the k reliable symbols in $r^{'}$ are zeros. These k reliable symbols with zero value contribute an interpolation polynomial $v(x)^{m}$, where m is the multiplicities of these symbols and $v(x)$ is a simple univariate interpolation polynomial as follows:

$$v(x) = \prod_{i \in R} \left(x - \alpha^{i}\right) \tag{2.5.2}$$

because these symbols correspond to k interpolation points with a zero y-component. In [19], Gross sets the multiplicities of these k reliable symbols with the maximum multiplicity.

2.5.2 The simplified interpolation step

To implement the simplified interpolation, consider the original set of polynomials $G = \left\{1\ , y, \cdots, y^{d_y}\right\}$ where d_y is the maximum degree of y. Based on the univariate polynomial created by the symbols in the set R, the starting polynomial set for simplified interpolation is:

$$\begin{aligned} G^{'} &= \left\{v(x)^{m}, v(x)^{m-1} y, \cdots, v(x)^{m-d_y} y^{d_y}\right\} \\ &= v(x)^{m} \left\{1\ , \frac{y}{v(x)}, \left(\frac{y}{v(x)}\right)^{2}, \cdots, \left(\frac{y}{v(x)}\right)^{d_y}\right\}. \end{aligned} \tag{2.5.3}$$

After changing variables $\tilde{y} = y/v(x)$, we get a new starting set $\tilde{G} = \left\{1\ , \tilde{y}, \cdots, \tilde{y}^{d_y}\right\}$ and the weighted degree of the new variable $\tilde{y}$ is:

$$\deg^{(1,k-1)}(\tilde{y}) = \deg^{(1,k-1)}(y) - \deg^{(1,k-1)}(v(x)) = (k-1) - k = -1. \tag{2.5.4}$$

Correspondingly, the new y-coordinates of the interpolation points are acquired as:

$$\tilde{y}_i = \frac{y_i^{'}}{v(x_i)} \tag{2.5.5}$$

where $y_i^{'}$ are the $y-$ components in $r^{'}$. It is concluded that the simplified interpolation step starts from the set $\tilde{G}$ and applies the KV interpolation step to the $n-k$ unreliable symbols $(x, \tilde{y})$ with respect to the $(1, -1)-$ weighted degree. After the simplified interpolation step the reduced interpolation polynomial is:

$$\tilde{P}(x, \tilde{y}) = \sum_{j=0}^{d_y} w_j(x)\, \tilde{y}^j. \tag{2.5.6}$$

2.5.3 The simplified factorization step

The simplified factorization step is realized by applying the Roth-Ruckenstein algorithm directly to the reduced polynomial $\tilde{P}(x, \tilde{y})$. After the factorization step, a sequence $\{s_0, s_1, \cdots, s_{l-1}\}$ which are the coefficients of

$$s(x) = \frac{f^{'}(x)}{v(x)} \tag{2.5.7}$$

is obtained, where $f^{'}(x)$ is the generator polynomial corresponding to $c^{'} = c - \psi$. From $r^{'} = c^{'} + e$ and $r_i^{'} = 0\,(i \in R)$ we get:

$$f^{'}\left(\alpha^i\right) = e_i\,(i \in R). \tag{2.5.8}$$

If no error in position $i \in R$ then $e_i = 0$ and $f^{'}(\alpha^i) = 0$. Therefore $(x - \alpha^i)$ is a root of $f^{'}(x)$, and for all non-error positions in R we get:

$$f^{'}(x) = \prod_{i \in R s.t. e_i = 0} \left(x - \alpha^i\right)\Omega(x). \tag{2.5.9}$$

Therefore,

$$s(x) = \frac{f^{'}(x)}{v(x)} = \frac{\prod_{i \in R s.t. e_i = 0} (x - \alpha^i)\,\Omega(x)}{\prod_{i \in R}(x - \alpha^i)} = \frac{\Omega(x)}{\prod_{i \in R s.t. e_i \neq 0} (x - \alpha^i)}. \tag{2.5.10}$$

We denote the denominator as $\Lambda(x)$, which is an error-locating polynomial that can be efficiently reconstructed by the BM algorithm. The roots of $\Lambda(x)$ indicate the error positions in set R. Based on the fact that most errors are occurred in the $n-k$ unreliable positions, we only need to correct a few errors in R. Consequently fewer coefficients than the requirement of the KV list algorithm are demanded.

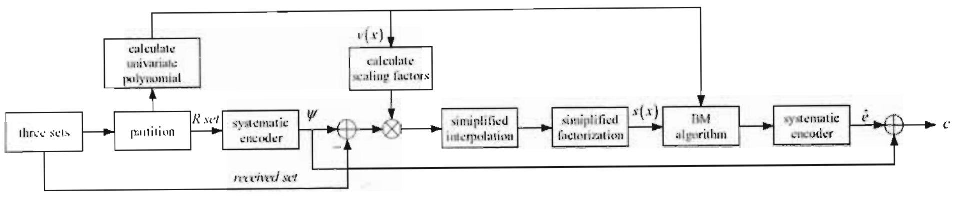
calculate univariate polynomial
v(x)
calculate scaling factors
three sets
partition
R set
systematic encoder
ψ
−
received set
simiplified interpolation
simiplified factorization
s(x)
BM algorithm
systematic encoder
ê
c

Figure 2.2: The Gross list algorithm

In the Gross algorithm, only $l = 2\lceil (k/n)\, t \rceil$ coefficients are given after simplified factorization step.

To get error values we use rule to the following equation:

$$f'(x) = \frac{\Omega(x)\, v(x)}{\Lambda(x)} \tag{2.5.11}$$

and we get:

$$f'\left(\alpha^i\right) = \frac{\Omega(\alpha^i)\, v^{(1)}(\alpha^i)}{\Lambda^{(1)}(\alpha^i)} \tag{2.5.12}$$

where $v^{(1)}(x)$ and $\Lambda^{(1)}(x)$ are the formal derivatives of $v(x)$ and $\Lambda(x)$. To put error values in their respective positions, we can get the error pattern for set R. The error pattern for the whole n positions is easily obtained by using the arbitrary-position systematic encoding and we denote it as $\hat{e}$. The finally estimated codeword can be found by adding $\hat{e}$ and ψ.

Figure 2.2 shows us the procedure of Gross list algorithm.

In the following example, we use the same information sequence of (7,5) RS code in the KV list algorithm to illustrate the Gross list algorithm.

Example 3: Consider the information sequence $f = (1, \alpha^5, 1, \alpha^2, \alpha^3)$ and the corresponding codeword is $C = (\alpha, \alpha^4, 1, \alpha^5, 1, \alpha^2, \alpha^3)$ if the systematic encoding is used. The codeword is transmitted through AWGN channel with BPSK modulation and the reliability matrix is constructed by received sequence as:

$$\Pi = \begin{pmatrix} 0.0087 & 0.2714 & 0.0043 & 0.0000 & 0.0003 & 0.5370 & 0.0001 \\ 0.0001 & 0.0027 & 0.9914 & 0.0027 & 0.9168 & 0.0037 & 0.0284 \\ 0.8422 & 0.0604 & 0.0000 & 0.0016 & 0.0000 & 0.0030 & 0.0032 \\ 0.0014 & 0.5386 & 0.0000 & 0.0001 & 0.0000 & 0.4154 & 0.0000 \\ 0.0123 & 0.0006 & 0.0011 & 0.1752 & 0.0707 & 0.0000 & 0.9678 \\ 0.1333 & 0.1198 & 0.0000 & 0.0072 & 0.0000 & 0.0022 & 0.0000 \\ 0.0020 & 0.0012 & 0.0000 & 0.8009 & 0.0009 & 0.0000 & 0.0005 \\ 0.0000 & 0.0053 & 0.0032 & 0.0124 & 0.0113 & 0.0027 & 0.0000 \end{pmatrix}.$$

The multiplicity algorithm 1 is applied with $s = 12$ and the multiplicity matrix is as:

$$M = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 2 & 0 & 2 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

The multiplicity matrix is also expressed as following by ignoring the zero entries in matrix:

$$\begin{array}{ll} \text{support set}: & (1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^5, \alpha^6) \\ \text{received set}: & (\alpha, \alpha^2, 1, \alpha^5, 1, 0, \alpha^2, \alpha^3) \\ \text{multiplicity set}: & (2, 1, 2, 1, 2, 1, 1, 2)\,. \end{array}$$

The conventional decoding algorithm also report failure for this case because the hard decision is $r = (\alpha, \alpha^2, 1, \alpha^5, 1, 0, \alpha^3)$, which has symbol distance 2 from the transmitted codeword. Next, the reencoding scheme is applied to compress the list of interpolation points through choosing k received symbols with distinct support symbols in the received set as:

	R set					U set		
support set	1	α^2	α^3	α^4	α^6	α	α^5	
received set	α	1	α^5	1	α^3	α^2	0	α^2
multiplicity set	2	2	1	2	2	1	1	1

and the reencoded codeword ψ is as $(\alpha, \alpha^4, 1, \alpha^5, 1, \alpha^2, \alpha^3)$, where the received symbols in reliable set appear in the related positions. In this example we use reencoding scheme to the received set not the hard decision since the one support symbol includes two received symbols. After subtracting with the related symbols from ψ, the unreliable set becomes:

$$\text{unreliable support set}: \quad (\alpha, \alpha^5, \alpha^5)$$
$$\text{unreliable received set}: \quad (\alpha, \alpha^2, 0).$$

The $v(x)$, which is the univariate initial interpolation polynomial for k reliable symbols in simplified interpolation step, is obtained as:

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

The final received symbols that are sent to the Gross interpolation step are using $\tilde{y}_i = \frac{y_i'}{v(x_i)}$:

$$\text{final support set}: \quad (\alpha, \alpha^5, \alpha^5)$$
$$\text{final received set}: \quad (\alpha, \alpha^6, 0).$$

The reencoding technique does not change the values in multiplicity set. Then the interpolation step is performed with final support and received set that includes only three symbols based on the $(1, -1)-$weighted degree. After $C' = 3$ iterations, the simplified interpolation polynomial is as follows:

$$\tilde{P}(x, \tilde{y}) = \alpha^5 x \tilde{y}^2 + \alpha^6 \tilde{y} + \alpha \tilde{y}^2.$$

After calculating $l = 2\lceil (k/n)\, t \rceil = 2$, we obtain the factors from $\tilde{P}(x, \tilde{y})$, which is also the syndrome polynomials for BM algorithm, as:

$$s_1(x) = 0$$
$$s_2(x) = \alpha^5 x + \alpha^2.$$

The error patterns are achieved when the $\Omega(x)$ and $\Lambda(x)$ are unfolded as:

$$\hat{e}_1 = (0,0,0,0,0,0,0)$$
$$\hat{e}_2 = (0,\alpha,0,\alpha^3,0,\alpha^2,0)\,.$$

Then the Gross list decoder outputs a list which includes the following two codeword:

$$C_1 = (\alpha,\alpha^4,1,\alpha^5,1,\alpha^2,\alpha^3)$$
$$C_2 = (\alpha,\alpha^2,1,\alpha^2,1,0,\alpha^3)\,.$$

C_1 is chosen after calculating its distance from received sequence using the reliability matrix and the information sequence is easily discovered as $\hat{f} = (1,\alpha^5,1,\alpha^2,\alpha^3)$ due to the implementation of systematical encoding.

This example shows us that the Gross list algorithm requires less computation in interpolation and factorization step than the KV list decoding algorithm.

The idea of the Gross list algorithm is similar to the Ordered statistic algorithm that was proposed in [24]. Construct a codeword from the reliable received symbols and apply the soft information in the other received symbols to adjust the constructed codeword. This notion is actualized as the result that the probability of reliable symbol occurs the error is decreased as the signal-to-noise-ratios increase. In [19], Gross compared the complexities between the original interpolation step and the simplified one, he indicated that $O\left(\left(\frac{n}{n-k}\right)^2\right)$ speedup is achieved in the simplified interpolation step. Also, the speedup in the simplified factorization step is palpable as $O\left(\frac{d_y kn}{d_y 2\lceil (k/n)t\rceil n}\right) \approx O\left(\frac{n}{n-k}\right)$. The following simulation results show us the Gross list algorithm has the same performance compared to the KV list algorithm.

We simulate both the KV and the Gross list algorithms in order to show the differences between their performances and complexities using (7,5) and (15,7) RS codes. $s = 12$, and $s = 25$ are used for (7,5) and (15,7) RS codes in our simulation, respectively. Although list algorithm only can correct single error for (7,5) RS code due to $t_{GS} = 1$, which is the error-correcting capability of conventional decoding algorithm, the performance of the KV and the Gross list algorithm is still

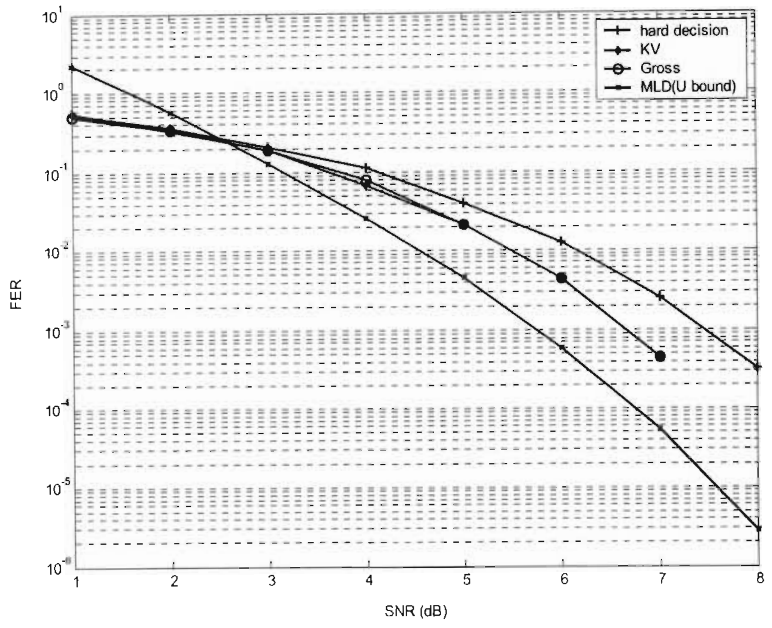

Figure 2.3: The simulation results of the KV and the Gross algorithm for (7,5) RS code

better than the conventional algorithm. This is explained by d_y that is decided by C. After checking the most received sequence, the KV list decoder produces two candidate codewords that is one more than the output of conventional decoder. It means that the search scope is enlarged since the t_m is greater than t_{GS} for $s = 12$. For (15,7) RS code, the list algorithm can correct one more error than conventional decoding algorithm, which corresponds to $t_{GS} = 5$. Figure 2.3 and Figure 2.4 show that the performance of the Gross algorithm is exactly the same as the KV algorithm for both codes.

The Gross list algorithm, which incorporates the BM decoding algorithm and the KV list algorithm with a reencoding scheme, is introduced in this section. The complexities and performances between the Gross and the KV algorithm are compared with numerical and simulation result.

Summary

This chapter introduces two decoding methods of RS codes. The BM decoding algorithm, which is the conventional decoding algorithm for RS codes, is also

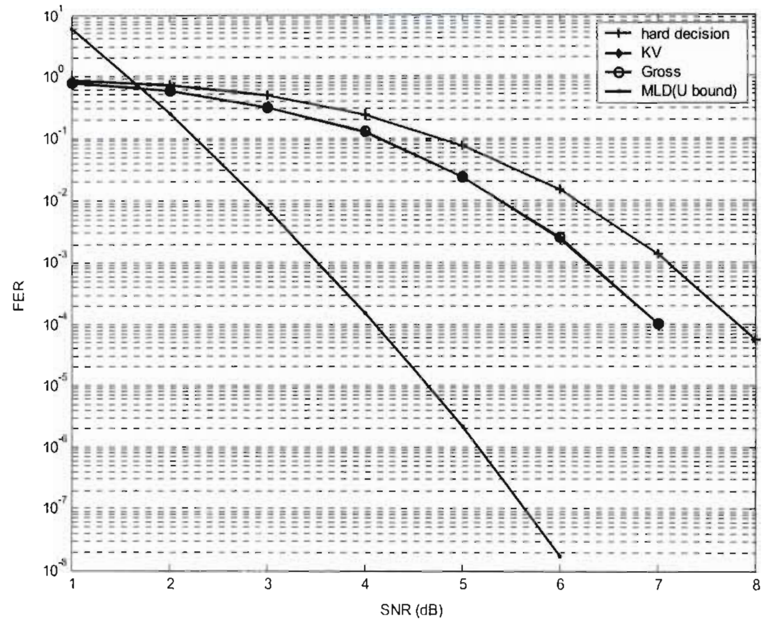

Figure 2.4: The simulation results of the KV and the Gross algorithm for (15,7) RS code

illustrated using an example. The evolution process of the list algorithm beyond the classical bound $t = (d_{\min} - 1)/2$ is represented by introducing the GS list algorithm, the KV list algorithm and the Gross list algorithm. The GS list algorithm theoretically proves the existence of the list algorithm, while the KV list algorithm is the practical implementation including the KV soft decision front end, the KV interpolation step and the RR factorization step. The Gross list algorithm is a simplified version through imposing the BM decoding algorithm.

Part II

The Hybrid Algorithm Based on the Chase Algorithm

Chapter 3

The Hybrid List Decoding and Chase-Like Algorithm of Reed-Solomon Codes

Although the list algorithm has better performance than the conventional decoding algorithm, there is still a considerable gap between the performance of the list algorithm and MLD. In [33], the author proposed a soft-decision decoding algorithm which applied the Chase algorithm to list decoding algorithm. Better performance was achieved through iteration. In this chapter, we present a hybrid list decoding and Chase-like algorithm. A criterion for producing test error patterns for the KV list decoding algorithm is proposed to differentiate our hybrid algorithm from the hybrid algorithm proposed in [33]. We apply different Chase algorithms to the KV soft-decision front end. Consequently, we are able to provide a more reliable input to the GS list algorithm, and achieve better performance.

3.1 The Chase algorithm

The Chase algorithm [25] was proposed in 1972. Many authors have attempted to improve the Chase algorithm and part of their contributions have been proposed in [26] and [27]. The main idea of the Chase algorithm is to create test error patterns for hard decision according to the reliability of the received sequence and then obtain the candidate codewords through adding the test error patterns to the hard decision. Finally, after comparing the distance of all candidates of codeword from the received word, the Chase algorithm outputs the codeword with the minimum distance from hard decision. Due to the different methods that produce the test error patterns, the Chase algorithms has three classes: Chase-1, Chase-2 and Chase-3. For binary case, the Chase algorithm finds several unreliable bits below a given number, and then inverses the different bits to create the corresponding candidate received word. For non-binary case, the Chase algorithm needs to change the chosen unreliable symbol to another symbol based on the bit reliability or symbol reliability. In Chase algorithms, we need to consider the minimum distance of the codeword because the maximum errors that can be corrected are $(d_{\mathsf{min}} - 1)/2$.

The Chase algorithm is known as an efficient way to improve the decoding performance based on the reliability drawn from a received sequence. In [28]-[32], the authors utilize the reliability to create more candidates of received sequence, in an attempt to achieve better performance. In our algorithm, we attempt to bridge the Chase algorithm and the KV soft-decision front end. We apply the Chase algorithms to the received bit based on their reliabilities. In our algorithm, we do not need to take the minimum distance into account due to the list algorithm. We imagine that the application of the Chase algorithms is to alter the position of the center of the search circle if the list algorithm is considered to create a decoding sphere.

3.2 The hybrid KV list decoding and Chase-like algorithm

Applying Chase-like algorithms in linear block codes is an efficient approach to improve the decoding performance. First, let us look at an example of the KV list decoding algorithm over a small field.

Example 4: Consider a (7, 5) RS code over GF (8). A classical hard-decision decoder is able to correct one symbol error. Suppose the message is $f = (\alpha, \alpha, \alpha^3, 0, \alpha)$, the generator polynomial is $f(x) = \alpha + \alpha x + \alpha^3 x^2 + \alpha x^4$ and the corresponding codeword using the evaluation map encoding is $(1, \alpha^4, \alpha^2, \alpha^6, 0, \alpha^3, \alpha^5)$. The codeword is transmitted bit-by-bit using BPSK modulation over an AWGN channel. After the KV soft-decision front end with s=12, we get a support set, received set and multiplicity set that are required by the KV list decoder as follows:

$$\begin{array}{ll} \text{support set :} & (1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6, \alpha^6) \\ \text{received set :} & (\alpha^3, \alpha^4, 0, \alpha^6, 0, \alpha^3, \alpha^3, \alpha^5) \\ \text{multiplicity set :} & (1, 2, 1, 2, 2, 2, 1, 1)\,. \end{array}$$

After the interpolation and the factorization, we obtain two candidates:

$$\begin{aligned} f_1(x) &= \alpha^4 + \alpha^2 x + \alpha^4 x^2 + x^4 \\ f_2(x) &= \alpha^6 + \alpha^4 x + \alpha^6 x^2 + \alpha^3 x^3 + \alpha^5 x^4. \end{aligned}$$

It is obvious that the KV list decoder cannot output the correct generator polynomial because the number of symbol errors exceeds the error-correcting ability based on $s = 12$. In this case the correct codeword polynomial "lives outside" the interpolation polynomial. In terms of geometric meaning, the KV list decoder cannot factorize the right curve because too many incorrect or disturbing points exist.

Now let us look at a test. We are able to obtain the reliability based on the absolute value of the received sequence. At first we search the most unreliable bit in the received set except for the received symbol whose support set is α^6. Then we find that the bit in the received symbol, whose support symbol is 1, is the most unreliable. We invert this bit to get a new received symbol. Now the new input of the KV list decoder is

$$\begin{array}{ll} \text{support set :} & (1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6, \alpha^6) \\ \text{received set :} & (1, \alpha^4, 0, \alpha^6, 0, \alpha^3, \alpha^3, \alpha^5) \\ \text{multiplicity set :} & (1, 2, 1, 2, 2, 2, 1, 1)\,. \end{array}$$

We use "multi-points" to define the received symbols whose support symbol is the same, and use "low-multiplicity-points" to define the received symbols whose multiplicities are less than the maximum multiplicity except for multi-points, and use "high-multiplicity-points" to define the received symbols whose multiplicities are equal to the maximum multiplicity. We can also call the high-multiplicity-points reliable points. Then we call the other received symbols, except for reliable points, unreliable points.

In this example, the multi-points are α^3 and α^5, the low-multiplicity-points are 1 and 0, the high-multiplicity-points or reliable points are $\alpha^4, \alpha^6, 0$ and α^3.

After that, we get another two candidate polynomials:

$$f_1(x) = \alpha + \alpha x + \alpha^3 x^2 + \alpha x^4$$
$$f_2(x) = \alpha^4 + \alpha^2 x + \alpha^4 x^2 + x^4.$$

After we change one unreliable bit in the received word, the KV list decoder will output the accurate generator polynomial. The number of error symbols is less than the error-correcting ability based on $s = 12$.

Through this test, we conclude that we can change the received set based on their bit reliabilities using the Chase algorithm. We obtain the separate candidate codeword polynomials from these received sets, respectively. That is, the search scope of the KV list decoding algorithm is enlarged.

In the previous example, we do not consider the multi-points because the KV list decoding algorithm has already taken them into account. In the other words, the list algorithm pays more attention to multi-points than the other points. If the KV list algorithm fails, the probability of errors coming from multi-points is very small.

The hybrid KV list decoding and Chase-like algorithm contains the following steps:

1. Implement the KV soft-decision front end to obtain support set, received set and multiplicity set.

2. Calculate the number of multi-points in the output of the KV soft-decision front end. As long as we find that several received symbols, whose support symbols are the same, the number of multi-points increases by one. We denote it as N_{multi}.

3. Calculate the number of low-multiplicity-points in the output of the KV soft-decision front end. We denote it as N_{low}.

4. If $N_{multi} + N_{low} \leq t_{GS}$, use the Chase-like algorithm for high-multiplicity-points.

5. Else use the Chase-like algorithm for both low-multiplicity-points and high-multiplicity-points.

6. Output all the received set, the corresponding support set and multiplicity set to interpolation step.

7. Interpolation step.

8. Factorization step.

9. Compare the probability of all candidate codewords created by different received set and output the most likely one.

In the above algorithm, the output of the KV soft-decision front end leads to two different scenarios, $N_{multi} + N_{low} \leq t_{GS}$ and $N_{multi} + N_{low} > t_{GS}$. We will discuss them separately.

If $N_{multi} + N_{low} \leq t_{GS}$, it means that the number of unreliable points does not exceed the error-correcting capability. In other words, these unreliable symbols are taken care of by the list decoding algorithm. Even if all the unreliable symbols are incorrect, the list decoding algorithm can still generate the right codeword polynomial. In this scenario, the errors coming from reliable symbols are the main reason for the decoding failure. So we apply the idea of the Chase algorithm to reliable symbols in order to obtain more correct received sets corresponding to the same multiplicity.

If $N_{multi} + N_{low} > t_{GS}$, it means that the number of unreliable symbols exceeds the error-correcting capability. If all these symbols are incorrect, the list decoding algorithm cannot output the correct codeword. For this scenario we have to concentrate on both low-multiplicity-points and high-multiplicity-points. Because of the reason we have mentioned before, we drop the multi-points. So we apply the Chase algorithm to both kinds of points. Before we invert the unreliable bits, we must make it clear which kind of symbols we should change in the first place, low-multiplicity-points or high-multiplicity-points. We can extend the search scope into high-multiplicity-points by changing the unreliable bits. In order to improve the search scope, changing the unreliable bits in high-multiplicity-points is better than in low-multiplicity-points. This implies that we can obtain more candidates if we choose high-multiplicity-points. It seems that we should change unreliable bits in high-multiplicity-points. But at high SNRs, we draw different conclusions. As the SNR increases, the high-multiplicity-points (reliable points) become more and more "reliable". The probability that the reliable points are transmitted correctly is very large. The performance improvement is marginal even if we invert the bits in the reliable points. In this paper, we pay attention to low-multiplicity-points first at high SNRs.

Another example that is parallel with the previous example is provided to give us a deeper insight into the above conclusion.

Example 5: Consider a (7,5) RS code, suppose the information sequence is $f = (\alpha, \alpha, 0, \alpha, 1)$, the codeword is $C = (\alpha^3, \alpha^4, \alpha, 0, \alpha^2, \alpha^4, \alpha^5)$. The codeword is transmitted bit-by-bit using BPSK over an AWGN channel. The received sequence is:

$$\begin{aligned} R_e = (&-1.0674\ , -0.7512, 1.0186, 0.4176, 0.8236, -1.2177, -1.7698, \\ &1.3855, -0.8407, -1.2332, -0.4071, 0.0489, 0.2635, -1.0172, \\ &-1.7381, 1.4990, 1.3324, -1.3438, 0.6284, 0.4993,\ 1.6153) \end{aligned}$$

through AWGN channel with the BPSK modulation. After applying the KV soft-decision front end, the three sets are shown as:

$$\begin{aligned} &\text{support set :} && (1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6) \\ &\text{received set :} && (1, \alpha^4, \alpha, 1, \alpha^2, \alpha^4, \alpha^5) \\ &\text{multiplicity set :} && (2, 2, 2, 1, 1, 2, 2)\,. \end{aligned}$$

Because two errors occurred in the received word and one of them has the maximum multiplicity, the KV list decoder cannot decode this received sequence. If we apply the Chase algorithm, we get the new received set which is given as:

$$\text{New received set: } (1, \alpha^4, \alpha, 0, \alpha^2, \alpha^4, \alpha^5).$$

The new received set is also useless for successful decoding because the error with maximum multiplicity still exists.

We examine the received set in example 5 again. If we create the test error pattern only focusing on the high-multiplicity-points, the Chase algorithm is also useless for decoding as all high-multiplicity-points are correct. Example 5 mostly happens at low SNRs. As the SNR increases, errors occurred in received set are similar to the example 4. Example 4 confirms $N_{multi} + N_{low} > t_{GS}$ and example 5 confirms $N_{multi} + N_{low} \leq t_{GS}$. So our method for choosing unreliable bit is optimal.

Figure 3.1 shows the results of our hybrid algorithm. We simulated 4-bit hybrid algorithm and 1-bit hybrid algorithm. For comparison, the simulation results of two different methods for choosing the unreliable bit are also shown in Figure 3.1. To apply the Chase algorithm, we denote the method which focuses on the high-multiplicity-points as contrast 1, the method focuses on all points as contrast 2. Due to more codewords being generated if we only focus to the high-multiplicity-points, the performance of contrast 1 is better than our hybrid algorithm at low SNRs. As the SNR increases, the high-multiplicity-points also become more reliable, the enhancement through using contrast 1 decreases. So the performance of contrast 1 approaches our hybrid algorithm at high SNRs, eventually the performance of our hybrid algorithm exceeds the contrast 1. The performances of contrast 2 are always inferior to the hybrid algorithm since the larger decoding

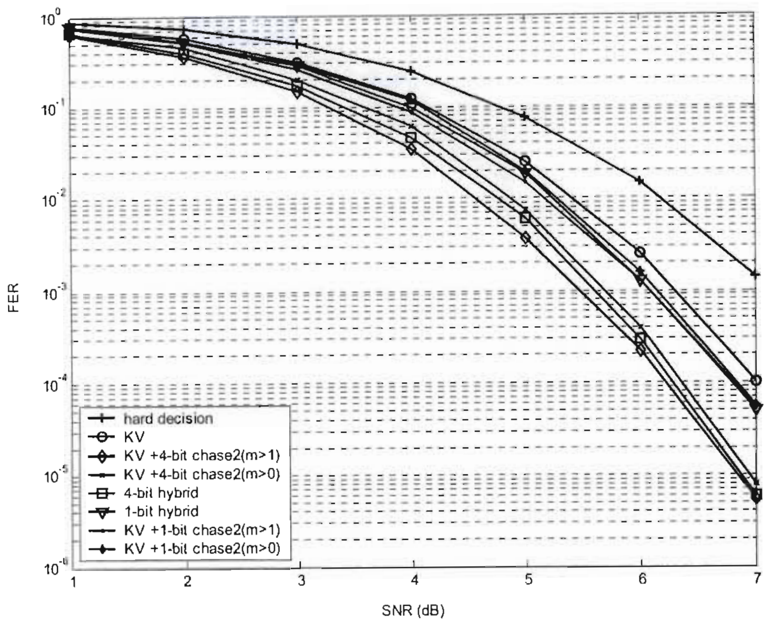

Figure 3.1: The simulation results of the hybrid KV list decoding and Chase-like algorithm for (15,7) RS code

sphere is reached in the hybrid algorithm by using different selective methods from contrast 2 if $N_{multi} + N_{low} \leq t_{GS}$.

In this section, a hybrid KV list decoding and Chase-like algorithm are presented. The basic idea of this algorithm is to invert the unreliable bits in the received set. In the proposed algorithm, we discuss two situations: the number of unreliable symbols exceeds and does not exceed the error-correcting capability of the list decoding algorithm and those kinds of points with priorities for changing unreliable bits corresponding to every situation. After using the Chase algorithm to the KV soft-decision front end, we obtain more received sets and accordingly obtain more candidates. As the search scope is extended, the transmitted codeword is easily recovered.

3.3 The hybrid Gross list decoding and Chase-like algorithm

Although the application of the Chase algorithm brings a large decoding gain, the complexity also exponentially increases because the KV algorithm has to produce interpolation polynomials after C iterations which are calculated by Equation 2.4.6 for each candidate of received set. In [33], the author also attempted to combine the Chase algorithm and the Gross list decoding algorithm for reduction of complexity. Assume the received set has the single multiplicity m_s, if b bits in different symbols are chosen, then the number of constraints for the interpolation step is given as:

$$C_{KV} = \frac{1}{2} \cdot 2^b \sum_{i=0}^{q-1} \sum_{j=0}^{n-1} m_{i,j} \left(m_{i,j} + 1\right). \tag{3.3.1}$$

In this section we discuss how to combine the Gross list decoding algorithm with the Chase algorithm. The reencoding scheme in the Gross algorithm may reduce the total iterations.

The hybrid Gross list decoding and Chase-like algorithm includes the following steps:

1. Implement the KV soft-decision front end to obtain support set, received set and multiplicity set.

2. Apply the reencoding scheme to divide the received set into two sets: reliable set (U) that we also denote it as set 1, and unreliable set $(\bar{U})$.

3. Apply the Chase algorithm to the unreliable set. Then the unreliable set can be further subdivided into two sets: the set that includes all bits that are chosen by the Chase algorithm, which we denote as set 2; and the set includes the remaining entries in unreliable set, which we denote as set 3.

4. Start the simplified interpolation step to obtain the polynomials for set 3 and store them. Accomplish the remaining iterations through a Tree scheme that will be proposed later.

5. Simplified Factorization step.

6. Compare the probability of all candidate codewords created by different received set and output the most likely one.

In the proposed algorithm, the hard decision is substituted by the received set since the situation that a single support symbol matches multi received symbols, which we denote as multi-points, exists in practices.

In the proposed hybrid algorithm, we apply the Chase algorithm after the reencoding step. It means that the Chase algorithm is applied to only $n-k$ symbols, and not all n received symbols. Actually, the test error patterns created in this method are less reliable than firstly using the Chase algorithm to find the unreliable bits among the overall received bits. If we chose the latter method to produce the test error patterns, then we have to build $v(x)$ and ψ for every candidate received set. Simulation results in section 3.4 show that the performance gap between two methods is very small. So we give the reencoding step priority to tradeoff performance with decoding complexity.

The tree scheme applied in the proposed algorithm is a technique by using the memory to minimize the number of iterations. Consider Chase-2 algorithm is employed and set the number of bits that are chosen by Chase-2 algorithm to b, thus, the subsets for set 2 are created. We denote an ideal situation that assumes these bits are distributed in diverse received symbols and the multiplicities for the symbols in unreliable set are all equal to the minimum value m_{min}. If we simply combine Chase-2 algorithm and the Gross algorithm under the condition that the received set does not contain the multi-points, the total cost of decoding is as follows:

$$C_{Gross} = 2^{b-1}(n-k)\, m_{\min}(m_{\min}+1). \tag{3.3.2}$$

Although C_{Gross} is much smaller than C_{KV} since the k reliable symbols do not get involved in the iteration creating the interpolation polynomial for each candidate received set, the cost still needs to be reduced if the rate of code is small or b is large. The proposed hybrid algorithm reduces the iterations in the interpolation

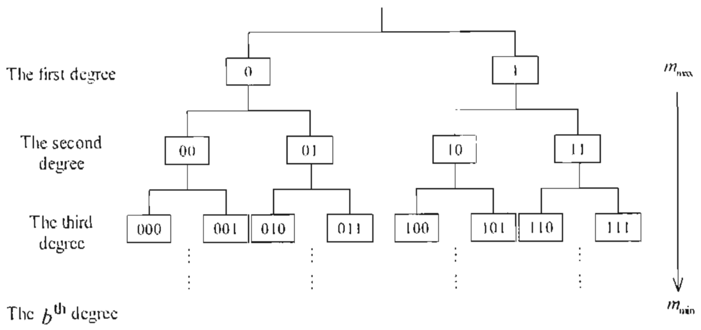

Figure 3.2: The tree scheme

step through the intrinsic relations among 2^b candidates of the received set. The polynomials produced by symbols in set 3 to accommodate all the 2^b candidates of received set because set 3 is the same among all of 2^b sets. So the hybrid algorithm firstly processes set 3 and stores the polynomials created by interpolation iteration. The remaining iterations are compressed by the tree scheme, which is shown in Figure 3.2. In Figure 3.2, 0 means the original symbol in the received set. Consequently the more reliable symbol obtained through changing one bit is expressed by 1. After the iterations for one symbol in set 2, we store d_y polynomials to prepare for the remaining candidates of the received set. Then the cost of interpolation step is reduced to:

$$C_{Gross+T} = \frac{1}{2}\left((n-k-b) + \left(2^{b+1}-2\right)\right) m_{\min}\left(m_{\min}+1\right). \qquad (3.3.3)$$

An example is shown below to explain the success of the proposed hybrid algorithm. Consider that a (255,127) RS code is adopted and 10-bit Chase 2 algorithm is realized by the Gross list decoder. Under the ideal circumstance, $65536 m_{\min}(m_{\min}+1)$ iterations are required for simply jointly using Chase-2 algorithm and the Gross algorithm but only $1082 m_{\min}(m_{\min}+1)$ iterations are needed in the proposed hybrid algorithm. The iterations for the proposed hybrid algorithm take only 1.651% proportion for C_{Gross} in this case. It is obvious that the proposed hybrid algorithm largely reduces the iterations for the interpolation step, especially for low rate codes.

In fact, the multiplicities of symbols in set 2 are usually not identical, and

Table V: The comparison of costs

Algorithm	Cost
The hybrid 1	$C_{KV} = \frac{1}{2} \cdot 2^b \sum_{i=0}^{q-1} \sum_{j=0}^{n-1} m_{i,j} (m_{i,j} + 1)$
Gross+Chase	$C_{Gross} = 2^{b-1} (n-k) m_{\min} (m_{\min} + 1)$
The algorithm in [33]	$C_{Xia} = \frac{1}{2} \left((n-k-b) + 2^b \cdot b\right) m_{\min} (m_{\min} + 1)$
The hybrid 2	$C_{Gross+T} = \frac{1}{2} \left((n-k-b) + \left(2^{b+1} - 2\right)\right) m_{\min} (m_{\min} + 1)$

thus how to decide the order of symbols in the tree scheme also needs to be taken into account. After checking the frequency of symbols for various degrees, the conclusion is drawn that the reduction for iterations is reduced as the degree of the tree scheme increases. The reduction of iterations is given as:

$$S_a = 2^{(b-1)} - 2^{(\deg -1)}. \tag{3.3.4}$$

Then it means that the iterations brought by the symbols in the first degree get the maximum reduction and no reduction is achieved for the symbols in the final degree. This opinion is tested through checking the iterations for every symbol in each degree. For each symbol in the first degree, it contributes single iteration for interpolation step for all of 2^b candidates of received set, but $2^{(b-1)}$ iterations for each symbol in the last degree. In order to further reduce the iterations, the symbols in set 2 are placed into the tree scheme in conformity with the regulation that the symbol with the larger multiplicity is arranged into more anterior degree. We denote the hybrid KV list decoding and Chase-like algorithm as hybrid algorithm 1, and the hybrid algorithm using the Gross decoder as hybrid algorithm 2. The costs for different hybrid algorithms including the list decoding algorithm and b-bit Chase 2 algorithm are shown in Table V under the ideal situation. The hybrid algorithm proposed in [33], which contains the Gross algorithm and the Chase algorithm, is similar to the proposed algorithm but without the tree scheme. In fact, the cost for the proposed hybrid algorithm is greater than the value in Table V because the multiplicities of symbols in set 2 do not entirely equal to m_{min}. The hybrid algorithm spares more iterations as the multiplicities of symbols in set

2 increase from m_{min} since the tree scheme arranges the symbols based on the multiplicity order that from maximum to minimum.

The hybrid Gross list decoding and Chase-like algorithm is presented in this section for the simplification of the Gross algorithm. The priority between the Chase algorithm and reencoding step is discussed. A tree scheme is applied to further reduce the number of iterations through utilizing the mutual relationship among 2^b candidates of the received set.

3.4 The simulation results

This section mainly explains the frame error rate (FER) performances and complexity of the two hybrid algorithms using (7,5) and (15,7) RS codes.

For (7,5) RS code, the conventional decoder can correct one error and the maximum likelihood decoder can correct two errors. So we apply Chase-1 to improve the performance of the KV list decoding algorithm. We implement 4-bit Chase-1 algorithm to received set. We choose 4-bit Chase-1 (4 times KV interpolation and RR factorization step) algorithm because the complexity of the generation of the interpolation polynomial does not increase very much. Simulation results are shown in Figure 3.3. From Figure 3.3, we see that the performance of the hybrid algorithm 1 approaches the performance of MLD. In this short code, the hybrid algorithm 2 is not applied because the complexity is acceptable by using the KV decoding algorithm. The tunable parameter s is chosen to be 12 for the hybrid algorithm 1.

For (15,7) RS code, we choose Chase-2 algorithm to enhance the performance. We select 4 unreliable bits and then correspondingly, we attain 2^4 candidate received sets. Simulation results are shown in Figure 3.4. Simulation results in Figure 3.4 show that the hybrid algorithm 1 is able to correct more than 6 errors at 6 dB. Because of the complexity (it has 16 times interpolation and factorization steps), we have to increase the maximum multiplicity of the list decoder if we want to achieve better performance. The performance of the hybrid algorithm 2 is also

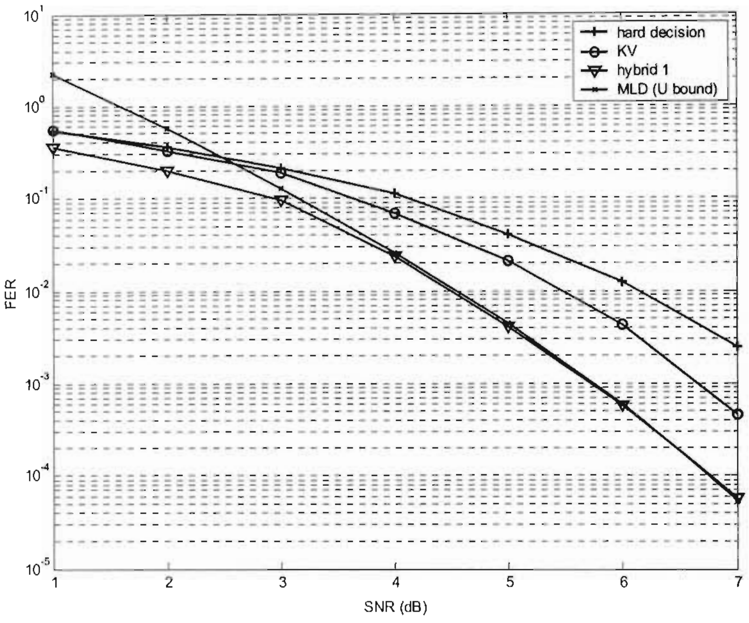

Figure 3.3: The simulation results of the hybrid KV list decoding and Chase-like algorithm for (7,5) RS code

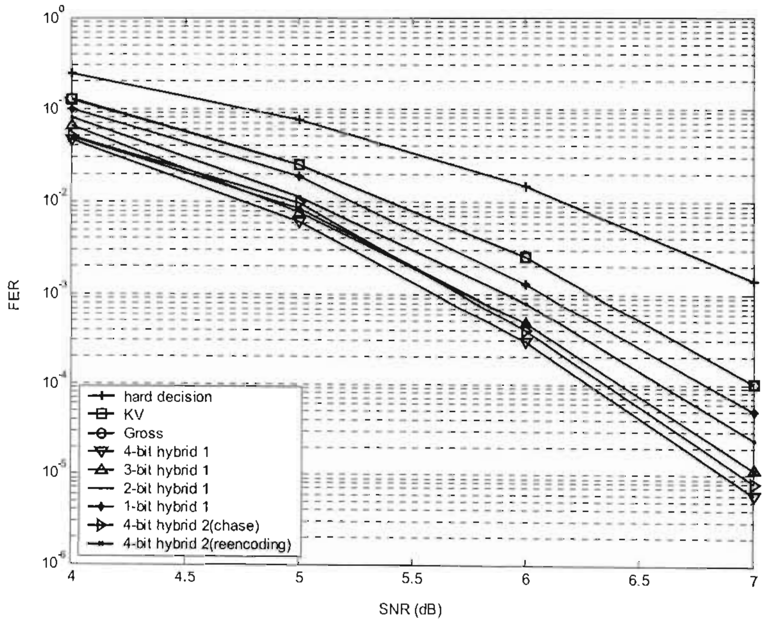

Figure 3.4: The simulation results of the hybrid Gross list decoding and Chase-like algorithm for (15,7) RS code

Table VI: The costs of simulation

Algorithm	4-bit	3-bit	2-bit	1-bit
The hybrid 1	$(3 \times 10 + 5) \times 2^4$ $= 560$	280	140	70
Gross+Chase	$(3 \times 3 + 5) \times 2^4$ $= 224$	112	56	28
The algorithm in [33]	$(3 \times 3 + 1) + 4 \times 2^4$ $= 74$	35	20	15
The hybrid 2	$(3 \times 3 + 1) + 2^5 - 2$ $= 40$	25	18	15

shown in Figure 3.4. As discussed in section 3.3 and proved by simulation results, the performance gap between the hybrid algorithm 2 that firstly uses Chase algorithm, which is denoted as hybrid 2(Chase) in Figure 3.4, and the hybrid algorithm 2 that firstly uses the reencoding step, which is denoted as hybrid 2(reencoding), is very small. The decoding costs of (15,7) RS code under the situation that no multipoints in received set and $s = 25$ are calculated in Table VI. Although the performance of the 4-bit hybrid algorithm 2 is the same as the 3-bit hybrid algorithm 1, 40 iterations are required for the former algorithm but 280 iterations for the latter. Figure 3.4 also shows that the performance of the hybrid algorithm 2 is worse than the hybrid algorithm 1 despite that the Gross algorithm has a completely uniform performance with KV algorithm. Then why are the performances different after applying the same Chase algorithm to two algebraic decoders that have the same performance? Actually, the difference of the performances between the hybrid algorithm 1 and 2 is caused by the simplified factorization step. As discussed in section 2.5, only $l = 2\lceil (k/n)\, t \rceil$ coefficients produced by simplified factorization step, it is a hint that the number of errors that occur in reliable set must be equal to $\lceil (k/n)\, t \rceil$ or less, otherwise the Gross list algorithm cannot successfully decode the received sequence even though the amount of errors that occur in the whole received set does not exceed the classical decoding bound $t = (d_{min} - 1)/2$. Since the probability that more than $\lceil (k/n)\, t \rceil$ errors occur in k reliable symbols is very low and decreases as the SNR increases, then it is acceptable that the performance of the Gross algorithm is same as KV algorithm. However if we apply the Chase algorithm to Gross algorithm with large erasures, the decoding fails because more than $\lceil (k/n)\, t \rceil$ errors occur in the k reliable symbols take a large of proportion for the reduction of performance. As the SNR increases, the symbols in set 2 with increasing reliabilities, cause the performance of hybrid algorithm 2 to approach the hybrid algorithm 1. This situation is proved by the simulation results. To change this situation, the simplified factorization step needs to unceasingly produce more coefficients, but the complexity also undoubtedly increases.

The costs of decoding for different hybrid algorithms are analyzed in this section. From the comparisons of simulation results and the costs of decoding, the hybrid algorithm 2 has the less complexity than the hybrid algorithm 1 and the hybrid algorithm proposed in [33] but with a slight performance loss corresponding to the hybrid algorithm 1.

Summary

This chapter discusses two hybrid algorithms based on the KV list algorithm and the Gross list algorithm. The particular steps for the two hybrid algorithms are given in this section. The costs of decoding for each hybrid algorithm are also discussed. The kernel of this chapter is how to achieve the maximum reduction of complexity by using the Gross algorithm. The hybrid algorithm 2 has a lower complexity because of the tree scheme and the concept of [33]. The discussion in section 3.4 confirms the effectiveness of the hybrid algorithm 2.

Chapter 4

The Adaptive Hybrid List Decoding and Chase-Like Algorithm of Reed-Solomon Codes

Although the complexity of the hybrid algorithm 2 is lower than the hybrid algorithm 1, its complexity increases exponentially with the number of bits used in the Chase-like algorithm. So the complexity after applying the Chase-like algorithm needs to be still further reduced. Actually, as the SNR increases, the received bits become more and more reliable. So it is not necessary to apply the Chase algorithm to every received sequence in the KV list decoding algorithm. To further reduce complexity at high SNRs, we propose an adaptive hybrid algorithm based on using the KV list decoding with an adaptive Chase-like algorithm in this chapter. The basic idea of the adaptive hybrid algorithm is to use a reliability threshold to exclude more reliable bits from being processed by the KV list decoding algorithm.

4.1 The adaptive algorithm

Mahran and Benaissa proposed an adaptive Chase algorithm for linear block codes in [34]. In the adaptive Chase Algorithm, a l-bit quantizer is used to classify the received bits by their reliability. A brief overview of the adaptive Chase algorithm is given below.

As errors are more likely to occur in the η least reliable positions of the received sequence R, the Chase algorithm is applied in those positions. A reliability threshold function or confidence level, r, is used in the adaptive Chase algorithm. The higher the confidence level the more Chase-like erasures. The threshold function is shown as:

$$T = S \times r \times \sqrt{\frac{0.5}{\left(\frac{E_b}{N_o}\right) \cdot R_c}} \tag{4.1.1}$$

where R_c is the code rate. E_b/N_0 is the bit SNR. S is a scalar constant that depends on the number of quantization levels which is given as:

$$\frac{0.45}{2^{l-3}} \leq S \leq \frac{0.7}{2^{l-3}}. \tag{4.1.2}$$

The threshold is used to decide which bits should be processed by the Chase algorithm. Let the reliability of received sequence be α_j, $j = 1, 2, \ldots, [d_{\min}/2]$. The bits will be used in the adaptive Chase algorithm only if their reliabilities satisfy the following condition:

$$-T \leq \alpha_j \leq +T \quad j = 1, 2, \ldots, [d_{\min}/2]. \tag{4.1.3}$$

If a bit that does not satisfy the above condition it can be ignored by the Chase algorithm even if it is the most unreliable bit in the received bits. If no bits satisfy the condition, the Chase algorithm is not used.

4.2 The adaptive hybrid KV list decoding and Chase-like algorithm of Reed-Solomon codes

The application of the Chase algorithm to the KV soft-decision front end can improve the performance of the KV list decoding algorithm. The complexity can be further reduced by using an adaptive scheme.

Since the adaptive hybrid KV list decoding and Chase-like algorithm is proposed based on the hybrid algorithm 1, we only give the steps that are different from the hybrid algorithm 1:

- If $N_{multi} + N_{low} \leq t_{GS}$
  use the Chase-like algorithm for high-multiplicity-points. The threshold obtained from Equation (4.1.1) with appropriate scale constant and confidence level is used to finally decide if the unreliable bits are picked up by the Chase algorithm or not

- else
  use the Chase-like algorithm for both of low-multiplicity-points and high-multiplicity-points. The unreliable bits selected by the Chase algorithm must also satisfy Equation 4.1.3.

Figures 4.1- 4.4 show different threshold values for different confidences. Figure 4.1 and Figure 4.2 show the threshold values for code rate with $R_c = 0.467$ while the code rate for Figure 4.3 and Figure 4.4 is $R_c = 0.677$. The scalar constant in Figure 4.1 and Figure 4.2 is 0.225, which is the minimum of a 4-bit quantizer. The scalar constant in Figure 4.3 and Figure 4.2 is 0.35, which is the maximum of the same 4-bit quantizer.

It is obvious that we can change the confidence level to get different thresholds. As the confidence level increases, the number of bits that can be ignored by the Chase algorithm decreases. The confidence level is adjusted to fit the KV list decoding algorithm. It is ideal that the performance of the adaptive hybrid algorithm is comparable with the performance of the hybrid algorithm 1 that is proposed in section 3.2, but with lower complexity.

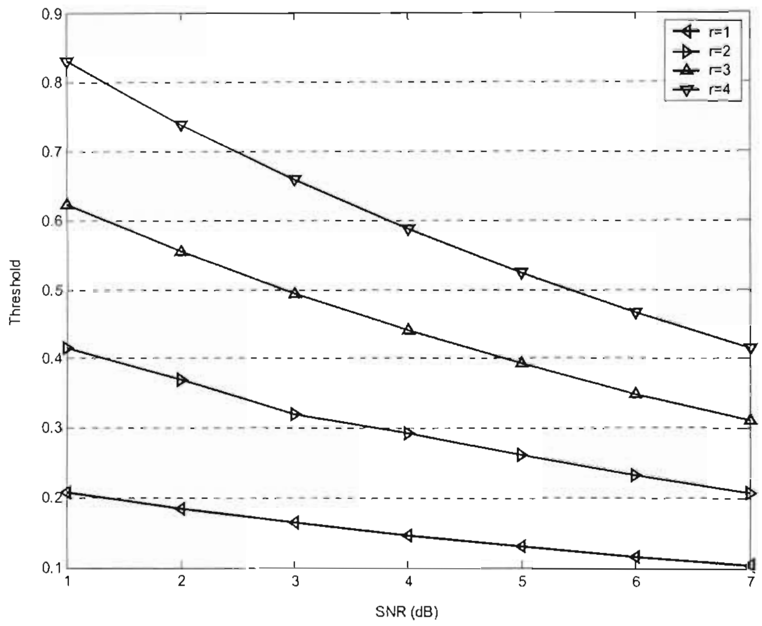

Figure 4.1: The threshold values with minimum scalar constant in 4-bit quantizer for codes with $R_c = 0.467$

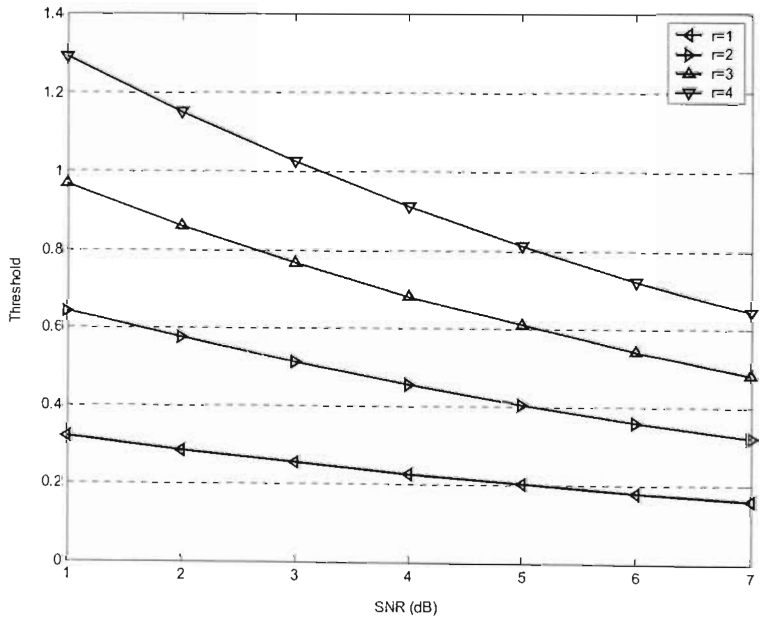

Figure 4.2: The threshold values with maximum scalar constant in 4-bit quantizer for codes with $R_c = 0.467$

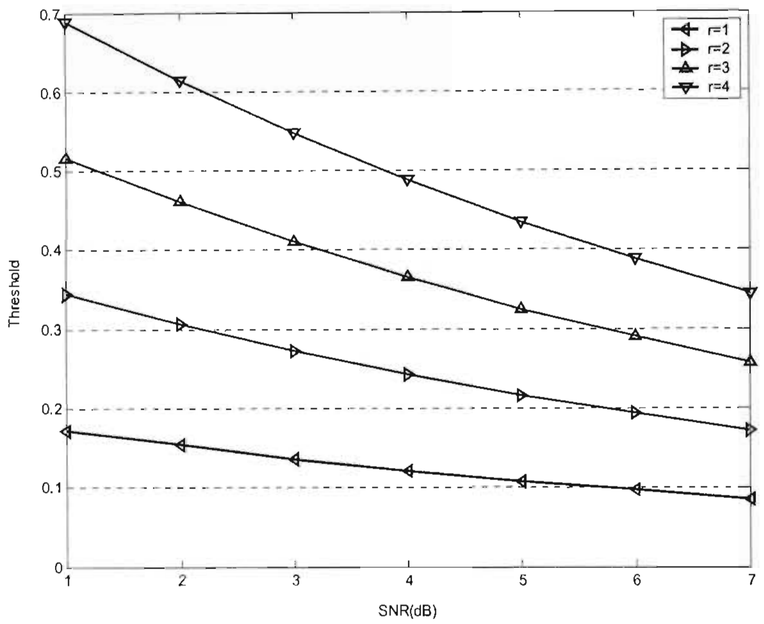

Figure 4.3: The threshold values with minimum scalar constant in 4-bit quantizer for codes with $R_c = 0.677$

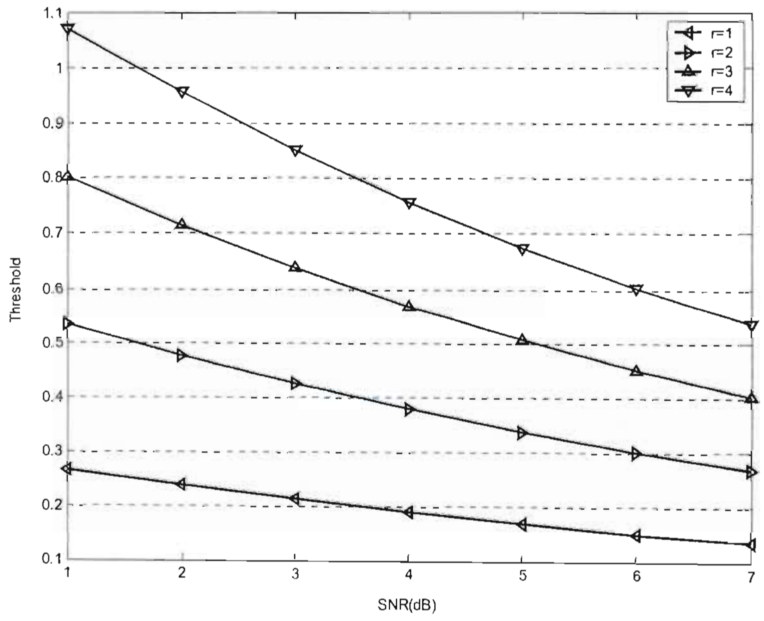

Figure 4.4: The threshold values with maximum scalar constant in 4-bit quantizer for codes with $R_c = 0.677$

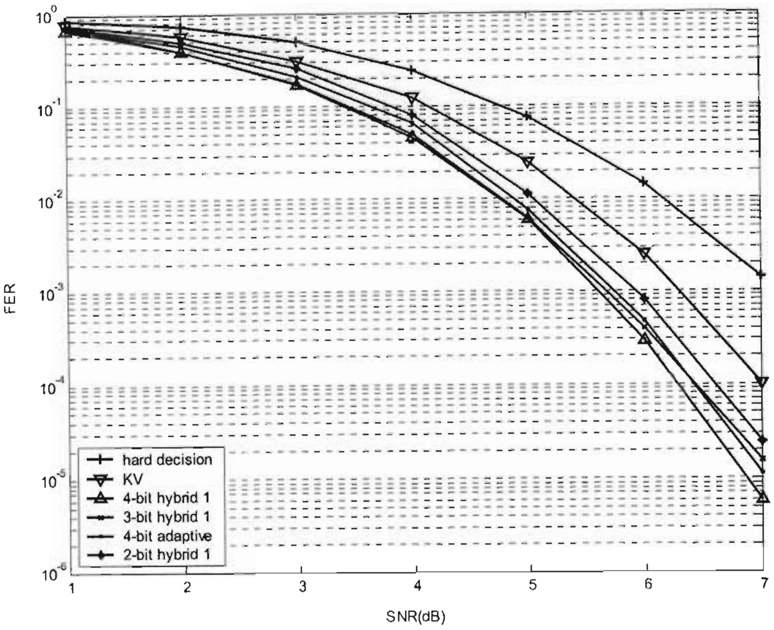

Figure 4.5: The FER performance of the adaptive hybrid KV list decoding and Chase-like algorithm for (15,7) RS code

4.3 The simulation results

All simulations are performed in an AWGN channel and assume BPSK transmission. For comparison, we simulate the conventional hard decoding, the KV list decoding, the hybrid algorithm 1 proposed in section 3.2 and the adaptive hybrid algorithm. For (15,7) RS code, we use Chase-2 algorithm to improve the performance. We select 4 unreliable bits based on the hybrid algorithm and input them to the 4-bit quantizer with the confidence level 1 to 8. In the simulation of (15,7) RS code we choose $S = 0.26$ and $r = 3$, which are suitable for the KV list decoding algorithm with maximum multiplicity 2. The FER performance is shown in Figure 4.5, and the corresponding complexity is shown in Figure 4.6. The simulation results of 2-bit hybrid algorithm, 3-bit hybrid algorithm and 4-bit hybrid algorithm are also shown in Figure 4.5 for comparison. Based on the fact that the interpolation step and factorization step take almost 95% of total decoding time, we use the time taken by the interpolation step and factorization step to be the measure of

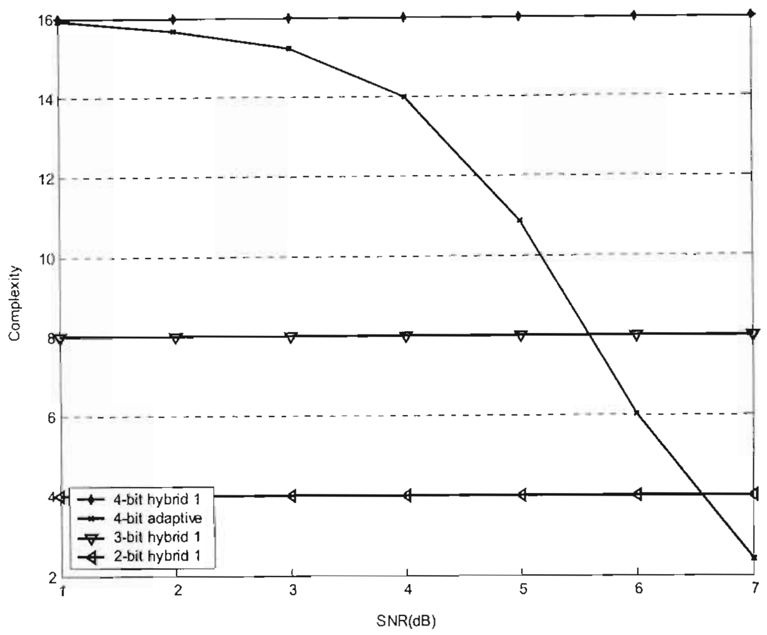

Figure 4.6: The complexity of the adaptive hybrid KV list decoding and Chase-like algorithm for (15,7) RS code

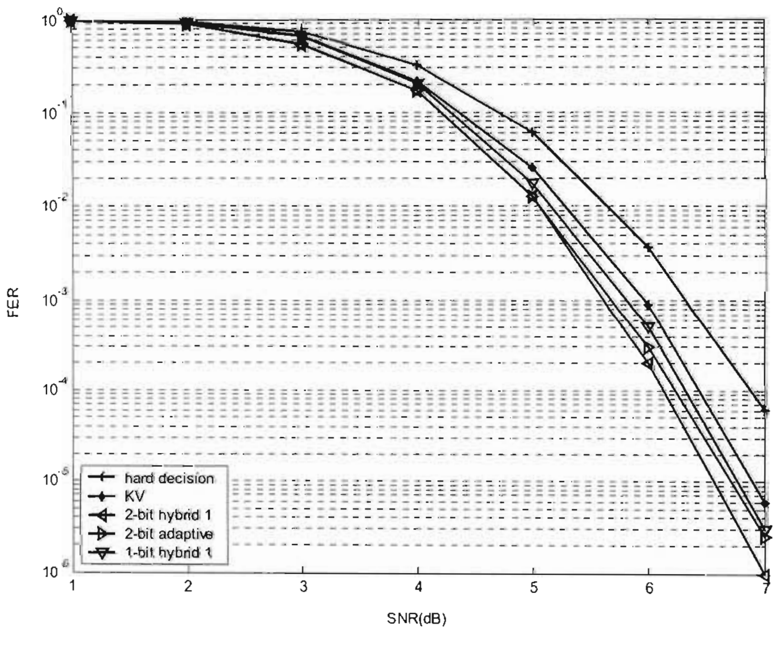

Figure 4.7: The FER performance of the adaptive hybrid KV list decoding and Chase-like algorithm for (31,21) RS code

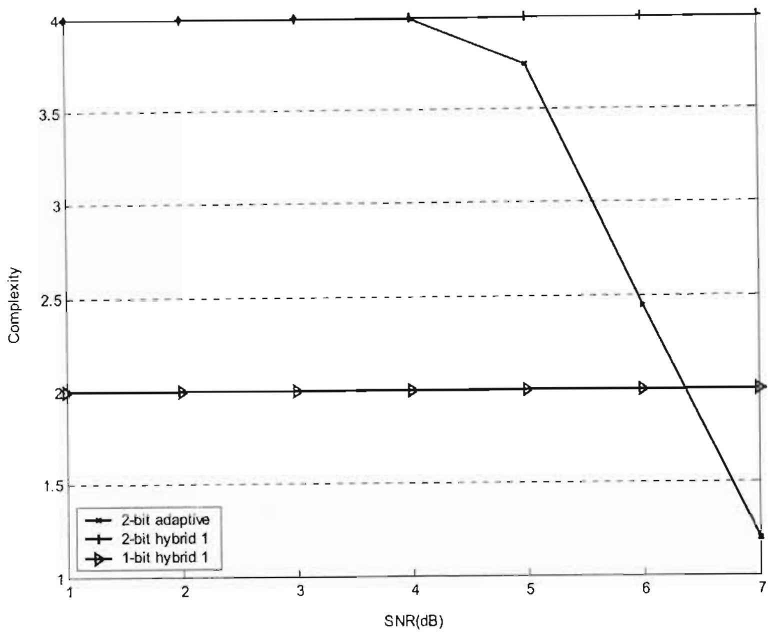

Figure 4.8: The complexity of the adaptive hybrid KV list decoding and Chase-like algorithm for (31,21) RS code

the complexity. Simulation results in Figure 4.6 show that the complexity of the adaptive hybrid algorithm decreases as the SNR increases. The complexity of the adaptive hybrid algorithm at 7dB is 2.3912, which is almost the same complexity as the 1 bit Chase-2 algorithm applied to the KV list decoding algorithm. The gap between the FER performance of the adaptive hybrid algorithm and the hybrid algorithm 1 is imperceptible.

For (31,21) RS code, we still use Chase-2 algorithm to improve the performance. A 4-bit quantizer is also used with $S = 0.225$ and $r = 3$. The FER performance is shown in Figure 4.7, and the corresponding complexity is shown in Figure 4.8. The simulation results of the 1-bit hybrid algorithm and 2-bit hybrid algorithm are also shown for comparison. The simulation results in Figure 4.5 and Figure 4.6 show us that the adaptive algorithm can reduce the complexity with a small or marginal performance penalty. The complexity of the adaptive hybrid algorithm in Figure 4.8 approaches the KV list decoding algorithm without the Chase algorithm at high SNRs.

Summary

In this chapter, an adaptive hybrid KV list decoding and Chase-like algorithm is presented. The basic idea of the adaptive hybrid algorithm is based on using a reliability threshold to exclude the more reliable bits from being processed by the KV list decoding algorithm. After using the adaptive Chase algorithm to the KV soft-decision front end, some bits that are not really "unreliable" are filtered out by the adaptive scheme therefore we obtain more accurate but less received sets than hybrid algorithm 1 and accordingly obtain more accurate but less candidates of codeword polynomial. Simulation results show that the FER performance of the proposed adaptive hybrid algorithm for both (15,7) and (31,21) RS codes is comparable with the performance of the hybrid algorithm 1, but the complexity is much lower, specially at high SNRs.

Part III

The Hybrid Algorithm Based on the Kaneko Algorithm

Chapter 5

The Hybrid Gross List Decoding and Kaneko Algorithm of Reed-Solomon Codes

In chapter 4, the adaptive algorithm is applied in order to reduce the decoding complexity through estimating whether the bit selected by the Chase algorithm is "really" unreliable or not. Although the complexity of the proposed adaptive hybrid algorithm is less than the hybrid algorithm that we proposed in chapter 3, a loss of performance is also detected. It means that some bits determined by the adaptive algorithm as reliable bits contain errors. The adaptive algorithm is not a sufficient condition for reducing the number of iterations of the Chase algorithm. So we have to find another approach to cut down the complexity without loss of performance through decreasing the number of candidates of a received set.

We will consider the two following problems before dealing with the approach on hand. One problem is to find a terminative condition to decide which generated candidate codeword is the most likely codeword for the received sequence before all candidates of a received set are tested. The other one is to eliminate the test error patterns that cannot create the more likely codeword than the generated codewords. After the two problems are solved, the number of iterations is reduced when the Chase algorithm is applied. Consequently, the complexity of the decoding algorithm is reduced without the loss of performance.

The first problem is settled by an efficient MLD algorithm [35] with algebraic decoder that was proposed by Toshimitsu Kaneko et al in 1994. The Kaneko algorithm generates a large set of candidates when a noisy sequence is received, and a smaller set of candidates when a clear sequence is received, and thus reduces the average decoding complexity without loss of performance for MLD. The main process of the Kaneko algorithm is to apply an optimality lemma to decide which generated candidate codeword is the most likely codeword for the received sequence before all candidates of the received set are tested. Obviously, the complexity of the Kaneko algorithm depends on the property of the received sequence.

The tool that can solve the second problem is a ruling out scheme for the simplified KV list algorithm that is also proposed in this paper. The ruling out scheme can further reduce the complexity of our hybrid algorithm by eliminating some useless test error patterns after forecasting their minimum distance for the received sequence.

5.1 The Kaneko algorithm

The Kaneko algorithm that was proposed in [35] is an MLD algorithm with lower complexity than Chase-2 algorithm. The main components in the algorithm are two lemmas. One lemma is used to find the most likely codeword for the received sequence and terminate the decoding. The other one is used to generate test error patterns.

Consider a RS codeword whose symbols are transmitted across a memoryless channel. The reliability matrix of the received sequence is calculated as:

$$\pi_{i,j} = p_r\left(\alpha_i \mid \beta_j\right) = \frac{\eta\left(\beta_j \mid \alpha_i\right)}{\sum_{\alpha_k \in GF(q)} \eta\left(\beta_j \mid \alpha_k\right)}. \tag{5.1.1}$$

A hard decision vector $r = (r_0, r_1, \cdots, r_{n-1})$ can be extracted from Π where

$$r_j = \max\left(\pi_{i,j}\right), 0 \leq j \leq n-1. \tag{5.1.2}$$

Let A be the all positions of a codeword, i.e, $A = \{1, 2, \cdots, n\}$. We divide the set A into S_x and S_x^c according to a codeword x. If $x_i = r_i$ then the position i belongs to S_x, otherwise the position i belongs to S_x^c, i.e, $S_x = \{i \,|x_i = r_i, i \in A\}$ and $S_x^c = \{i \,|x_i \neq r_i, i \in A\}$. Obviously, $A = S_x + S_x^c$.

The maximum-likelihood metric $L(r, x)$ for q-ary codes is calculated as:

$$L(r, x) = \sum_{i \in A} \pi_{i,x_i} = \sum_{i \in A} \pi_i^{1st} - \sum_{i \in S_x^c} \left(\pi_i^{1st} - \pi_{i,x_i}\right) \tag{5.1.3}$$

where π_i^{1st} is the maximum symbol reliability for ith received symbol. So $L(r, x)$ depends only on

$$l(r, x) = \frac{1}{2} \cdot \sum_{i \in S_x^c} \left(\pi_i^{1st} - \pi_{i,x_i}\right). \tag{5.1.4}$$

So our purpose is to find the codeword x from r which minimizes the value of $L(r, x)$. If there exists x such that

$$l(r, x) < \min_{x' \neq x} l\left(r, x'\right) \tag{5.1.5}$$

then we can determine that x is the most likely codeword for r.

We denote the number of S as $\|S\|$. To find the $\min_{x' \neq x} l\left(r, x'\right)$, we need to find the minimum $\left\|S_{x'}^c\right\|$ and the smallest values for $\pi_i^{1st} - \pi_{i,x_i}$. From $d_s\left(x', x_0\right) \geq d_{\min}$ where x_0 is the output of the algebraic decoder when the hard decision r is the input, we have

$$\left\|S_{x'}^c\right\| + \left\|S_{x_0}^c\right\| > d_{\min}. \tag{5.1.6}$$

After we denote the $\left\|S_{x_0}^c\right\|$ as m_0, we can get one of the lower bounds on $\left\|S_{x'}^c\right\|$,

$$\left\|S_{x'}^c\right\| \geq d_{\min} - m_0. \tag{5.1.7}$$

Based on the following equation,

$$\min\left(\pi_i^{1st} - \pi_{i,x_i}\right) = \pi_i^{1st} - \pi_i^{2nd} \tag{5.1.8}$$

and $m = \|S_x^c\|$ we reorder the elements $S_x^{(i)}$ in S_x as:

$$\pi_{A(1)}^{1st} - \pi_{A(1)}^{2nd} \leq \pi_{A(2)}^{1st} - \pi_{A(2)}^{2nd} \leq \cdots \leq \pi_{A(n-m)}^{1st} - \pi_{A(n-m)}^{2nd}. \tag{5.1.9}$$

Then one of the lower bounds is derived as:

$$\min_{\substack{x' \neq x \\ x' \neq x_0}} l\left(r, x'\right) \geq \sum_{i=1}^{d-m_0} \left(\pi_{A(i)}^{1st} - \pi_{A(i)}^{2nd}\right). \tag{5.1.10}$$

Consequently we have the following lemma 1.

Lemma 1: *If the codeword x satisfies the following condition*

$$l\left(r, x\right) < \sum_{i=1}^{d-m_0} \left(\pi_{A(i)}^{1st} - \pi_{A(i)}^{2nd}\right) \tag{5.1.11}$$

then there is no codeword which is more likely than x.

For the derivation of lemma 2 that can give us the smallest scope with which the transmitted codeword is likely to be incorporated, we select all sequences as the test error patterns e, which are located in the i positions with the lowest value of $\pi_{A(i)}^{1st} - \pi_{A(i)}^{2nd}$. We define a set L_j as the set of codewords that are outputs of the algebraic decoder when $r + e$ are inputs. Lemma 2 is stated as.

Lemma 2: *If $j \in A$ satisfies*

$$l\left(r, x\right) < \sum_{i=1}^{d_{min}-m_0-t-1} \left(\pi_{A(i)}^{1st} - \pi_{A(i)}^{2nd}\right) + \sum_{i=1}^{t+1} \left(\pi_{A(j+i)}^{1st} - \pi_{A(j+i)}^{2nd}\right) \tag{5.1.12}$$

then $C_x \subseteq L_j$ where C_x is the set of codewords which are more likely than x.

If j is set to a fixed value then all sequences in L_j are test error patterns. If $j = 0$, the Kaneko algorithm has the same performance with hard decision decoder and if $j \geq 1$ the performance is the same as Chase 2 algorithm. The simulation results in [35] show that the performances of the Kaneko algorithm for BCH codes and Golay codes are always the same as Chase 2 algorithm, but the complexities are decreased.

5.2 The hybrid Gross list decoding and Kaneko algorithm of Reed-Solomon codes

Although lemma 1 of the Kaneko algorithm can largely reduce the complexity of decoding through using a minimum value of codeword to terminate the decoding process, the complexity of the Kaneko algorithm is still high if we set j to a large value. A ruling out scheme for Gross list decoding algorithm is applied to further reduce the complexity.

The idea of a ruling out scheme is presented in [37]. The description of the ruling out scheme is given below.

Lemma 1 of the Kaneko algorithm can find the most likely codeword for the received sequence, but before the most likely one is found we have to create the codeword for every test error pattern. However, some test error patterns cannot create a more likely codeword than the generated codewords. The ruling out scheme is used to eliminate the useless test error pattern, thus we do not need to process every test error pattern. Then the complexity can be further reduced.

Consider that we have a received sequence and the corresponding hard decision is r. A test error pattern is created as $\hat{e} = (\hat{e}_1, \hat{e}_2, \cdots, \hat{e}_n)$ and a candidate of hard decision is obtained as $\hat{z} = r + e$, correspondingly, the codeword created by the Gross list algorithm is $\hat{c}$ when $\hat{z}$ is input. We denote $E = \{i \,|\hat{e}_i \neq 0\}$ and $||E|| \leq d_{\min}/2$. We partition the hard decision r into two sets as mentioned in the reencoding scheme, unreliable set (U) and reliable set (R). After we apply the systematic encoding to the set R of the $\hat{z}$, we can get a codeword $\hat{c}^r$. Let A be all positions of a codeword and we divide A into four sets as follows:

$$D_U = \{i \mid i \in U \; and \; \hat{e}_i = 0\} \tag{5.2.1}$$

$$D_U^c = \{i \mid i \in U \; and \; \hat{e}_i \neq 0\} \tag{5.2.2}$$

$$D_R = \{i \mid i \in R \; and \; \hat{e}_i = 0\} \tag{5.2.3}$$

$$D_R^c = \{i \mid i \in R \; and \; \hat{e}_i \neq 0\}\,. \tag{5.2.4}$$

We reorder the elements in U and R as:

$$\pi^{1st}_{U(1)} - \pi^{2nd}_{U(1)} \leq \pi^{1st}_{U(2)} - \pi^{2nd}_{U(2)} \leq \cdots \leq \pi^{1st}_{U(n-k)} - \pi^{2nd}_{U(n-k)} \tag{5.2.5}$$

$$\pi^{1st}_{R(1)} - \pi^{2nd}_{R(1)} \leq \pi^{1st}_{R(2)} - \pi^{2nd}_{R(2)} \leq \cdots \leq \pi^{1st}_{R(k)} - \pi^{2nd}_{R(k)}. \tag{5.2.6}$$

Lemma 3, which has been proved in [37], is given to define a new candidate codeword.

Lemma 3: *For a nonzero test error pattern $\hat{e}$. $\hat{c}$ is a new and prospective candidate codeword which is different from all generated codeword, only $\hat{c}$ satisfies the following conditions:*

$$\hat{c}_i = \hat{z}_i \,(i \in E) \tag{5.2.7}$$

$$\|\hat{c}_i \neq \hat{z}_i\| = t_m \,(i \notin E)\,. \tag{5.2.8}$$

Based on the d_{min} and the reencoding scheme, a new and prospective candidate codeword $\hat{c}$ for the Gross list algorithm is required to satisfy one more condition which is shown in the following lemma.

Lemma 4: *If the following situation is occurred*

$$\|\hat{z}_i \neq \hat{c}_i^r\| \geq 1 \,(i \in U) \tag{5.2.9}$$

then the new and prospective candidate codeword $\hat{c}$ created by $\hat{e}$ must satisfy the following condition:

$$1 \leq \|\hat{c}_i \neq \hat{c}_i^r\| \leq 2\left\lceil (k/n)\, t\right\rceil \,(i \in D_R)\,. \tag{5.2.10}$$

Based on the lemma 3 and lemma 4, we can get the following **condition** for $\hat{e}$ that can create a new and prospective codeword $\hat{c}$:

$$\|D_U^e\| \leq \|U\| + 2\left\lceil (k/n)\, t\right\rceil - t_m. \tag{5.2.11}$$

We denote the minimum maximum-likelihood metric for $\hat{c}$ created by $\hat{e}$ as $L_{\min}(r, \hat{c})$. The following theorem can be deduced.

Theorem 1: *Suppose c_b is the most likely codeword for received sequence among those generated codewords and $\hat{e}$ is the next test error pattern. Then $\hat{e}$ is useless and can be ruled out if the following condition holds:*

$$L_{\min}(r, \hat{c}) \geq L(r, c_b)\,. \tag{5.2.12}$$

Now, our purpose is to find the $L_{\min}(r, \hat{c})$ and compare it with $L(r, c_b)$.

Based on the lemma 3, lemma 4 and the condition for $\hat{e}$, we can get the $L_{\min}(r, \hat{c})$ as follows. We denote

$$l(r, \hat{e}) = \sum_{i=1}^{\|E\|} \left(\pi_{E^{(i)}}^{1st} - \pi_{E^{(i)},\, \hat{e}_i} \right). \tag{5.2.13}$$

If $\|\hat{z}_i \neq \hat{c}_i^r\| = 0 \, (i \in U)$, then

$$\begin{aligned} L_{\min}(r, \hat{c}) = l(r, \hat{e}) &+ \sum_{i=1}^{\min\left(\|U\| - \left\|D_U^e\right\|,\, t_m\right)} \left(\pi_{D_U^{(i)}}^{1st} - \pi_{D_U^{(i)}}^{2nd} \right) \\ &+ \sum_{i=1}^{t_m - \left(\|U\| - \left\|D_U^e\right\|\right)} \left(\pi_{D_R^{(i)}}^{1st} - \pi_{D_R^{(i)}}^{2nd} \right) \end{aligned} \tag{5.2.14}$$

else

$$\begin{aligned} L_{\min}(r, \hat{c}) = l(r, \hat{e}) &+ \sum_{i=1}^{\min\left(\|U\| - \left\|D_U^e\right\|,\, t_m - 1\right)} \left(\pi_{D_U^{(i)}}^{1st} - \pi_{D_U^{(i)}}^{2nd} \right) \\ &+ \sum_{i=1}^{\max\left(t_m - 1 - \left(\|U\| - \left\|D_U^e\right\|\right),\, 0\right) + 1} \left(\pi_{D_R^{(i)}}^{1st} - \pi_{D_R^{(i)}}^{2nd} \right). \end{aligned} \tag{5.2.15}$$

Our ruling out scheme can give a sufficient condition to rule out some useless test error patterns without the loss of performance. This conclusion is proved by the simulation results shown in the next section.

The hybrid decoding algorithm for an (n, k) RS code whose elements are drawn from $GF(q)$ contains the following steps:

1. Implement the KV soft-decision front end to obtain the support set, received set and multiplicity set.

2. Use the Gross list decoding algorithm to produce the first candidate codeword c_0 and get the $L(r, c_0)$. Set $c_b = c_0$ and $L(r, c_b) = L(r, c_0)$. Compute the value of m_0 and store it.

3. Use the lemma 1 to check if c_b is optimal. If yes, go to step 10, otherwise go to step 4.

4. Use the lemma 2 to get T which is the smallest value of j satisfying Equation 5.1.12 or give a fixed value to T.

5. Check if all the $\left(q^T - 1\right)$ test error patterns have been generated. If not, go to step 6, otherwise go to step 10.

6. Use the criterion proposed in section 3.2 to generate a new test error pattern $\hat{e}$ with $\|E\| \leq T$.

7. Calculate the $L_{\min}(r, \hat{c})$ and use theorem 1 to decide if the test error pattern is useless or not. If it is useless, go to step 5, else go to step 8.

8. Use the Gross list algorithm to produce the codeword $\hat{c}$ and compute its maximum-likelihood metric $L(r, \hat{c})$.

9. Check whether $L(r, \hat{c}) < L(r, c_b)$. If yes, set $c_b = \hat{c}$ and $L(r, c_b) = L(r, \hat{c})$, then go to step 3, otherwise, go to step 4.

10. Terminate the decoding and output the c_b.

In the above proposed algorithm, we use t_m, which is the error-correcting capability of the Gross list algorithm, to substitute the classical t. The criterion used in step 5 is an efficient way to generate the test error pattern based on reliability and multiplicity.

A flow chart is given to illustrate the procedure of the proposed hybrid algorithm. We also can adjust T to a fixed value, in which the same performance as Chase 2 algorithm is achieved.

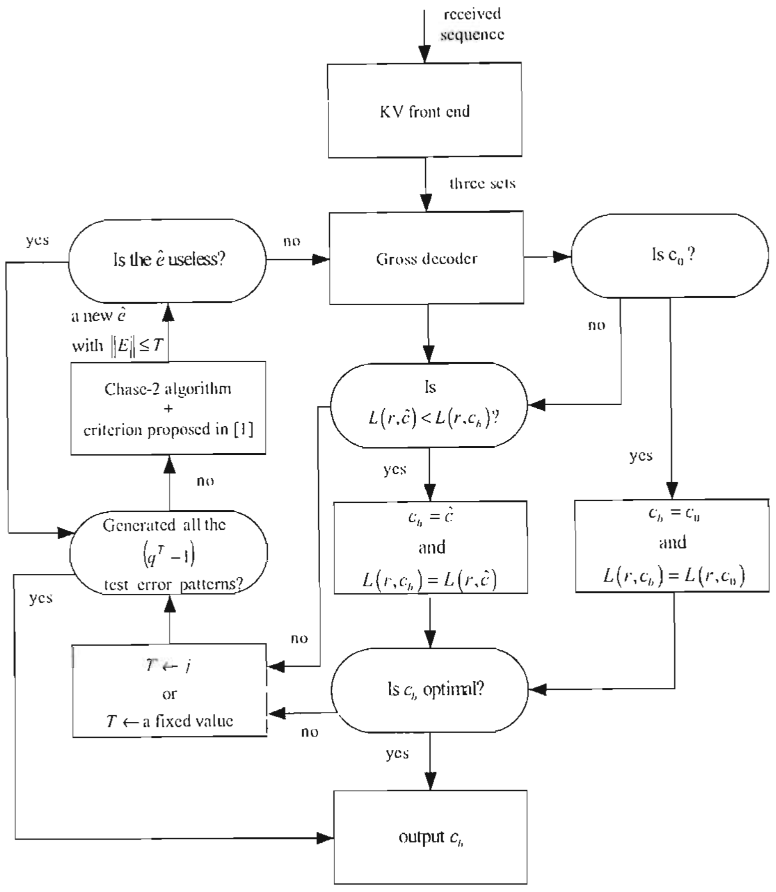
received sequence
KV front end
three sets
Gross decoder
Is c_0?
no
yes
Is the $\hat{e}$ useless?
yes
no
a new $\hat{e}$ with $\|E\| \le T$
Chase-2 algorithm + criterion proposed in [1]
no
Generated all the $(q^T - 1)$ test error patterns?
yes
Is $L(r,\hat{c}) < L(r,c_b)$?
yes
$c_b = \hat{c}$ and $L(r,c_b) = L(r,\hat{c})$
$c_b = c_0$ and $L(r,c_b) = L(r,c_0)$
no
$T \leftarrow j$ or $T \leftarrow$ a fixed value
Is c_b optimal?
no
yes
output c_b

Figure 5.1: The hybrid Gross list decoding and Kaneko algorithm

5.3 The simulation results

In this section, we compare the performance and complexity among three decoding algorithms: the proposed decoding algorithm in this chapter, which we denote as hybrid algorithm 3, the hybrid algorithm 1, and the adaptive hybrid algorithm proposed in chapter 4. In the simulation, we create the test error patterns in the way as 3^T for hybrid algorithm 3. It means that we only create 2 different symbols for a chosen unreliable symbol. For (15,7) RS code, we set the total multiplicity s of the Gross algorithm to 25 then we get $t_m = 5$. We choose the simulation of hybrid algorithm 1 with 4-bit Chase-2 algorithm and the simulation of the adaptive hybrid algorithm composed by the KV algorithm and the adaptive algorithm with

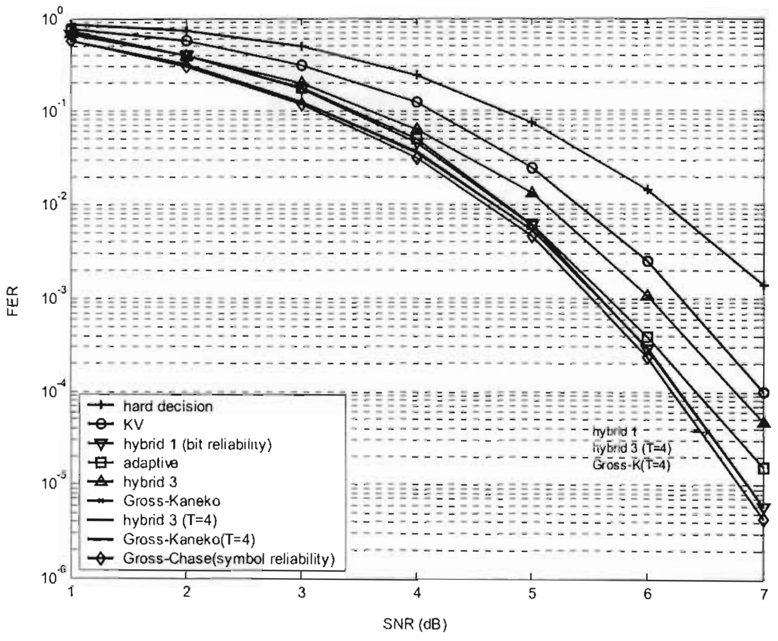

Figure 5.2: The FER performance of the hybrid Gross list decoding and Kaneko algorithm for (15,7) RS code

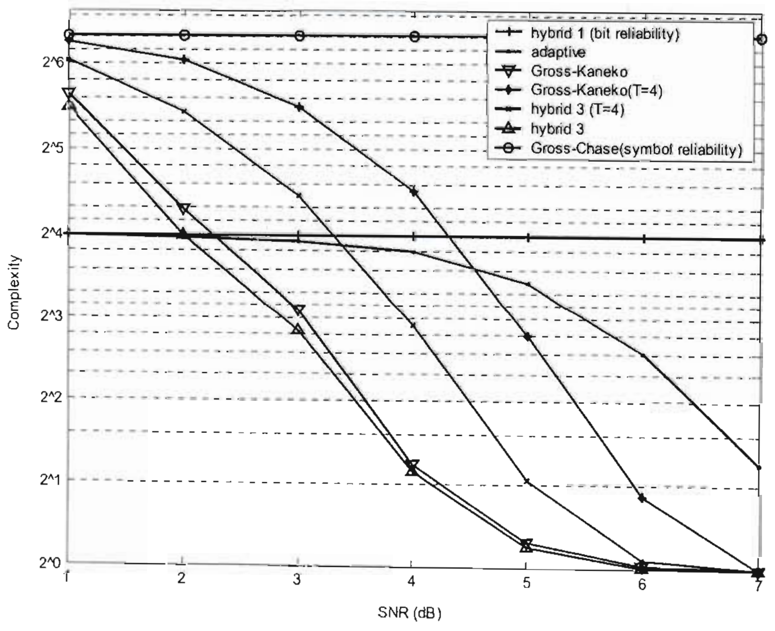

Figure 5.3: The complexity of the hybrid Gross list decoding and Kaneko algorithm for (15,7) RS code

$S = 0.26$ and $r = 3$ to compare with the hybrid algorithm 3. We also perform the simulation with setting $T = 4$ of the hybrid algorithm 3. So the hybrid algorithm 3 with $T = 4$ should have the same performance as the decoding process that creates 81 test error patterns. It is shown in Figure 5.2 that the performance of the hybrid algorithm 3 with $T = 4$ is the same as the 4-bit hybrid algorithm 1 that produces 16 test error patterns. It means that some loss of performance occurred in hybrid algorithm 3. It is proved, by the comparison between the simulation results of the Gross-Chase algorithm that applied the Chase algorithm to the Gross algorithm with symbol reliability and the 4-bit hybrid algorithm 1, that the application of the Gross algorithm and symbol reliability in the Kaneko algorithm are the main reason for the loss of performance. The performance difference between the hybrid algorithm 3 and the Gross-Chase algorithm is made by the symbol maximum-likelihood metric in the Kaneko algorithm. Despite of the loss of the performance in Figure 5.2 and Figure 5.3, it is shown that our algorithm with $T = 4$ has the best performance among the three algorithms and requires less complexity than the others at high SNRs.

For (31,21) RS code, we set total multiplicity to 56. 2-bit hybrid algorithm 1 and the adaptive hybrid algorithm with $S = 0.225$ and $r = 3$ are also compared in the simulation results. Figure 5.4 and Figure 5.5 show that the hybrid algorithm 3 has the best performance and the least complexity at high SNRs.

We also simulate the hybrid algorithm 3 without the ruling out scheme. It is obvious that the ruling out scheme further reduces the complexity of the hybrid algorithm 3 without the loss of the performance.

Summary

In this chapter, a hybrid Gross list decoding and Kaneko algorithm has been presented. A ruling out scheme for the Gross list decoding algorithm is also proposed. The idea of this hybrid algorithm is to reduce the complexity of a decoding algorithm through an optimality lemma and a ruling out scheme. The optimality lemma is thought to reduce the average complexity and the ruling out scheme is

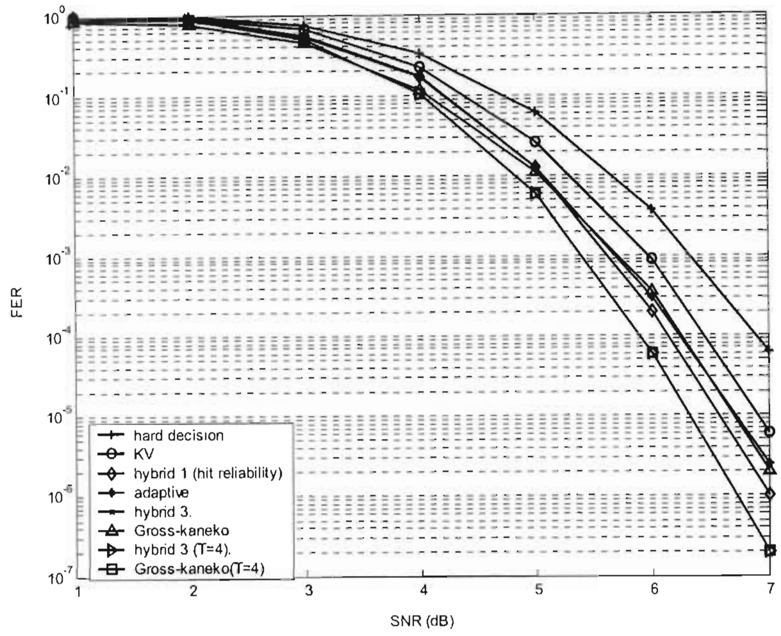

Figure 5.4: The FER performance of the hybrid Gross list decoding and Kaneko algorithm for (31,21) RS code

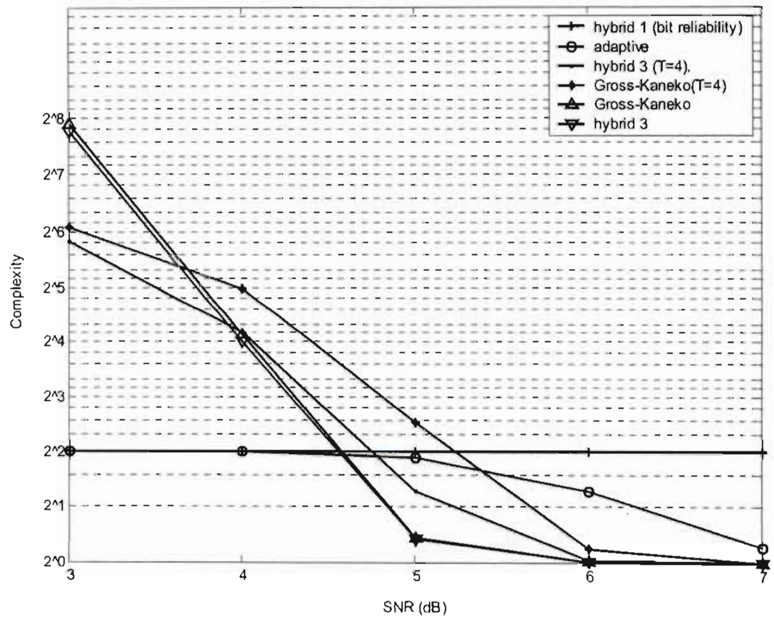

Figure 5.5: The complexity of the hybrid Gross list decoding and Kaneko algorithm for (31,21) RS code

thought to reduce the complexity in every single decoding process. The simulation results show that the complexity of hybrid algorithm 3 are much less than the others at high SNRs, though the performance of the hybrid algorithm 3 is always superior to the others.

Chapter 6

Conclusion and Future Work

6.1 Conclusion for dissertation

Three hybrid decoding algorithms and one adaptive hybrid algorithm for RS codes were investigated in this dissertation. The KV and the Gross list decoding algorithms, which have the error-correcting capability beyond the classical $t = (d_{min} - 1)/2$, were used to substitute the conventional algebraic decoding algorithm. The Chase, Adaptive, Kaneko and Ruling out algorithms have been taken into account to achieve the optimal decoding performance and low complexity.

The first hybrid algorithm comprises the KV list decoding and Chase algorithm. Due to the characteristics of list decoding, we divided the received symbols into multi-points, low-multiplicity-points and high-multiplicity-points with respect to their multiplicities. A criterion giving which points the priority to contribute the unreliable bits was proposed based on two situations. The simulation results attested to this deduction as the hybrid algorithm using this criterion has better performance than the hybrid algorithm using Chase-2 algorithm to focus on the whole symbols. From the simulation results, though the hybrid algorithm that forces the Chase algorithm to only focus on the high-multiplicity-points which we denote it as contrast 1 has the better performance than our hybrid algorithm at low SNRs, the enhancement through using contrast 1 decreases as the SNR increases.

So the criterion proposed in our hybrid algorithm is the optimal method to create error test patterns.

The Gross list decoding algorithm replaced the KV algorithm to build the second hybrid algorithm. The Chase algorithm was still applied to improve system performance. The second hybrid algorithm is provided with lower complexity than the first hybrid algorithm since normally, only $n - k$ symbols need to be processed by the interpolation step and $l = 2\lceil (k/n)\, t \rceil$ symbols need to be exported by the factorization step. A tree scheme is used to further reduce the number of symbols entangled into iterations in the interpolation step, which takes the largest portion of decoding time. The tree scheme utilizes the correlation among candidates of the received set to store the fractional interpolation polynomial that is created by the symbols in previous candidate of the received set but also exists in the current candidate of the received set. Due to the communal interpolation polynomial, the total cost of the interpolation step was reduced, which is shown in Table V, accordingly, the complexity of the second hybrid algorithm is less than the first hybrid algorithm. The loss of performance is detected in simulation results, and it was caused by the Gross list algorithm. Even the performance of second hybrid algorithm with 4 erasures is the equivalent of 3-bit first hybrid algorithm. Compared to the first hybrid algorithm, only 1/7 of the cost is required for the second hybrid algorithm.

Although the second hybrid algorithm largely reduced the decoding complexity, the number of iterations was still large if more unreliable bits were chosen by the Chase algorithm. Most of decoding time was wasted dealing with the received sequence where no error occurred during transmission. At high SNRs, the probability that the unreliable bits selected by the Chase algorithm are correct cannot be overlooked. Thus it indicated that we can find a solution to reduce the complexity in this idea. An adaptive algorithm is considered to reduce the total complexity through throwing off the bits that were estimated to be unreliable bits by the Chase algorithm. Therefore, there are not the changeless 2^b test error patterns for every received sequence if the b-bit Chase-2 algorithm was applied. The algebraic decoder used in this adaptive hybrid algorithm is the KV decoder, but the performance of the adaptive hybrid algorithm is not as good as the first hybrid algorithm using the same algebraic decoder, though the complexity was reduced. It means that a part of the bits released by the adaptive scheme was "really" unreliable. The loss of performance occurred in the situation that received a sequence that contains the errors but can be accurately restituted if we use the Chase algorithm. To solve this problem, we have to place a threshold with a larger value, however the complexity of decoding also increased since the participation of more bits in one decoding process.

The reduction of the decoding performance through using adaptive scheme motivated us to find other means to achieve the purpose of reducing the complexity but keeping the performance. The two problems were considered before we proposed the third hybrid algorithm. One problem is to find a terminative condition to decide which generated candidate codeword is the most likely codeword for received sequence before all candidates of a received set are tested, another is to eliminate the test error patterns that cannot create the more likely codeword than the generated codewords. After the two problems were solved, the number of iterations was reduced when the Chase algorithm is applied. Consequently, the complexity of the decoding algorithm is reduced without the loss of performance. In our third hybrid algorithm, the first problem was solved by the Kaneko algorithm, and the second task was taken over by the Ruling out algorithm. The Kaneko algorithm used an optimality lemma to decide whether the current generated candidate codeword is the most likely codeword for the received sequence or not before all candidates of received set are tested. The ruling out scheme eliminated some useless test error patterns after forecasting their minimum distance for the received sequence to reduce the complexity. From the simulation results, after the two problems were solved, the third hybrid algorithm has a better performance than the first hybrid algorithm and the adaptive hybrid algorithm, but the lowest complexity for both

(15,7) and (31,21) RS codes.

The loss of performance of the third hybrid algorithm was also detected and the peace breakers were not only the Gross algorithm but also the application of symbol reliability in the Kaneko algorithm. The symbols which have the minimum values with $\pi^{1st} - \pi^{2nd}$ (π is the symbol reliability) were taken into account by the Kaneko algorithm in the first place, nevertheless, the Chase algorithm confirmed the erasures in the whole received sequence corresponding to the bit reliability. The symbol reliability is created by bit reliability, so the soft information is lost in the second manipulation for bit reliability. This is the reason that the bit reliability was more reliable than the symbol reliability, and was also the cause for reduction of performance in third hybrid algorithm.

6.2 Further work

Future work will cover the two aspects of enhancing the performance and reducing the complexity.

Performance

Enhancement of the performance can be achieved through increasing the value of constraint s to enlarge the search scope in order to obtain more candidates of codeword, though the cost increases exponentially with the length of code and the value of constraint s.

Based on the simulation results of the third hybrid algorithm, the orientation is doable that achieving the performance of MLD but a much lower complexity through solving the two problems. The amelioration should be focused on applying the bit reliability to supersede the symbol reliability. It means that another optimal lemma to decide whether the current generated candidate codeword is the most likely codeword for the received sequence, should to be produced with a view to bit reliability. The algorithm proposed in [31] is the embodiment of this idea. In their algorithm, the Ordered statistic algorithm replaced the Chase algorithm to create the candidates of received sequence. Nevertheless, it is a pity that the necessary algebraic decoder used in their algorithm is still the conventional algebraic decoder. Thus, how to combine the list decoding algorithm with the optimal lemma respected to bit reliability, for the elevation of performance, is part of further work.

Complexity

In our third hybrid algorithm, we create the test error patterns in turn through the Chase algorithm even the candidate codeword produced by previous candidate of received set is less likely than the current optimal codeword. Kaneko proposed an improved version in 1997 [36] to amend this situation. In their improvement, they select the next test error pattern depending on the comparison between the $L(r, \hat{c})$ and $L(r, c_b)$, instead of creating it in turn. This technique can also be future work for reducing the decoding complexity.

Bibliography

[1] S. B. Wicker and V. K. Bhargava, An introduction to Reed-Solomon codes, in Reed-Solomon Codes and Their Applications (S. B. Wicker and V. K. Bhargava, eds.), ch. 1, pp. 1C16, New York, New York: IEEE Press, 1994.

[2] E. B. Berlekamp, *Algebraic Coding Theory.* New York: McGraw 1968

[3] J. L. Massey, "Shift register synthesis and BCH decoding," IEEE Inform. Theory, vol. IT-15, pp.122-127, Jan.1969

[4] Y. Sugiyama, M. Kasahara. S. Hirawawa, T. Namekawa, A Method for Solving Key Equation for Decoding Goppa Codes, Information and Control, 27, 87-99, 1975.

[5] G. D. Forney, Jr., "Generalized Minimum Distance Decoding," IEEE Transactions on Information Theory, vol. IT-12, pp. 125-131, April 1966.

[6] E. R. Berlekamp, "Bounded distance+1 soft-decision Reed-Solomon decoding", IEEE Trans. Inform. Theory vol. 22 no. 3 (May 1996), pp. 704-720.

[7] Peter Elias, "Error-Correcting Codes for List Decoding" IEEE Trans Infor vol 37, No 1, Jan 1991.

[8] M. Sudan, "Decoding of Reed Solomon codes beyond the error-correcting bound", J. Complexity, vol. 13, pp.180-193, March, 1997.

[9] V. Guruswami and M. Sudan, "Improved decoding of Reed-Solomon codes and algebraic geometry codes", IEEE Trans. Inform. Theory vol. 45 no. 6, pp. 1757-1767, Sept. 1999

[10] R. Koetter,"Fast generalized minimum-distance decoding of algebraic-geometry and Reed-Solomon codes," IEEE Transaction on Information Theory, vol. 42 no. 3, pp. 721-736, May 1996.

[11] R. Koetter, J. Ma, A. Vardy, and A. Ahmed, Efficient interpolation and factorization in algebraic soft-decision decoding of Reed-Solomon codes. Submitted to ISIT 2003.

[12] R. Koetter and A. Vardy, A complexity reducing transformation in algebraic list decoding of Reed-Solomon codes, in Proceedings of ITW2003, (Paris, France), March 31 - April 4 2003.

[13] R. Koetter and A. Vardy, "Algebraic soft-decision decoding of Reed-Solomon codes," Proceedings of the IEEE International Symposium on Information Theory, pp. 61, June, 2000.

[14] W. J. Gross, F. R. Kschischang, R. Koetter, and P. Gulak, "Simulation results for algebraic soft-decision decoding of Reed-Solomon codes," in Proceedings of the 21'st Biennial Symposium on Communications, pp. 356-360, June 2-5 2002.

[15] H. Hasse, Theorie der hoheren differentiale in einem algebraischen funktionenkorper mit vollkommenem konstantenkorper bei beliebiger charakteristik, in J. Reine. Ang. Math.,, vol. 175, pp. 50C54, 1936.

[16] G. Feng and K. Tzeng, A generalization of the Berlekamp-Massey algorithm for multisequence shift-register synthesis with applications to decoding cyclic codes, IEEE Transactions on Information Theory, vol. 37, pp. 1274C1287, September 1991.

[17] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms.* Cambridge, Mass.; MIT Press, 1990.

[18] G. L. Feng, "Two fast algorithms in the Sudan decoding procedure" in Proc, 37th Annual Allerton Conf. Communication, Control and Computing, 1999, pp 545-554.

[19] Warren J. Gross, Frank R. Kschischang, Ralf Koetter, and P. Glenn Gulak, "Towards a VLSI Architecture for Interpolation-Based Soft-Decision Reed-Solomon Decoders", Journal of VLSI Signal Processing. Vol. 39 Nos. 1-2, January-February 2005. pp. 93-111

[20] W. J. Gross, F. R. Kschischang, R. Koetter, and P. Gulak, A VLSI architecture for interpolation in soft-decision list decoding of Reed-Solomon codes, in Proceedings of the 2002 IEEE Workshop on Signal Processing Systems (SIPS02), (San Diego, CA), pp. 39C44, October 16-18 2002.

[21] Warren J. Gross, Frank R. Kschischang, Ralf Koetter, and P. Glenn Gulak, "Application of Algebraic Soft-Decision Decoding of Reed-Solomon Codes", submitted to Trans Paper for IEEE Trans Comm.

[22] S. B. Wicker, Error Control Systems for Digital Communication and Storage. Upper Saddle River, New Jersey: Prentice Hall, 1995.

[23] R. M. Roth and G. Ruckenstein, "Efficient decoding of Reed-Solomon codes beyond half the minimum distance," IEEE Transactions on Information Theory, vol. 46, pp. 246-257, January 2000.

[24] M. P. C. Fossorier and S. Lin, "Soft-decision decodings of linear codes based on ordered statistics algorithm," IEEE Trans. Inform. Theory, vol. 41, pp. 1379-1396, Sept. 1995.

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

[26] M. Fossorier and S. Lin, "'Chase-Type and GMD Coset Decodings," IEEE Transactions on Communications, vol. COM-48, March 2000.

[27] G. Arico and J. H. Weber, "Limited-trial Chase decoding", IEEE Trans. on Inform. Theory, vol. 49, no. 11, pp. 2972-2975, November 2003.

[28] Heng Tang, Te Liu, Marc Fossorier and Shu Lin,"On Combining Chase-2 and GMD Decoding Algorithms for Nonbinary Block Codes" IEEE Communication letters, vol. 5, NO. 5, pp. 209-211 May 2001

[29] Marc P. C. Fossorier, Shu Lin and Jakov Snyders, "Reliability-Based Syndrome Decoding of Linear Block Codes" IEEE Trans Infor vol 44, No 1, Jan 1998.

[30] M. P. C. Fossorier and S. Lin,"Complementary reliability-based decoding of binary block codes," IEEE Trans. Infor. Theory, vol.43, pp. 1667-1672.

[31] Ta-Hsiang Hu and Shu Lin, "An Efficient Hybrid Decoding Algorithm for Reed-Solomon Codes Based on Bit Reliability," IEEE Trans.Commu.vol.51, no.7 (July 2003), pp. 1073-1081.

[32] Marc Fossorier and Antoine Valembois, "Reliability-Based Decoding of Reed-Solomon Codes Using Their Binary Image". IEEE.Commu Letter.vol.8, no.7 (July 2004), pp. 452-454.

[33] Haitao 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.

[34] A. Mahran, M. Benaissa, "Adaptive combined Chase-GMD algorithms for block codes", IEEE Commun. Lett., vol.8, no.4, pp.235-237,April, 2004.

[35] Toshimitsu Kaneko, Toshihisa Nishijima, Hiroshige Inazumi, and Shigeichi Hirasawa, "An Efficient Maximum-Likelihood-Decoding Algorithm for Linear Block Codes with Algebraic Decoder", IEEE Trans. Inform. Theory vol. 40, no. 2, pp. 320-327, Mar. 1994.

[36] Toshimitsu Kaneko, Toshihisa Nishijima, and Shigeichi Hirasawa, "An Improvement of Soft-Decision Maximum-Likelihood Decoding Algorithm Using Hard-Decision Bounded-Distance Decoding", IEEE Transactions on Information Theory VoL. 43 No. 4, pp. 1314-9, July 1997.

[37] T. Koumoto, T. Kasami and S. Lin, "A Sufficient Condition for Ruling Out Some Useless Test Error Patterns in Iterative Decoding Algorithms", IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E81-A, No. 2, pp. 321-326, February 1998.