EQUALIZATION AND CODING FOR THE TWO-DIMENSIONAL INTERSYMBOL INTERFERENCE CHANNEL

By

TAIKUN CHENG

A dissertation submitted in partial fulfillment of
the requirements for the degree of

DOCTOR OF PHILOSOPHY

WASHINGTON STATE UNIVERSITY

School of Electrical Engineering and Computer Science

DECEMBER 2007

To the Faculty of Washington State University:

The members of the Committee appointed to examine the dissertation of TAIKUN CHENG find it satisfactory and recommend that it be accepted.

_____
Chair

_____

_____

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisor, Dr. Benjamin Belzer for his invaluable guidance and consistent support through out the time it took me to complete this research and write the dissertation in Washington State University. The program was one of the most important and formative experiences in my life. Without his help this dissertation would not have been possible.

The members of my dissertation committee, Dr. Thomas Fischer and Dr. Krishnamoorthy Sivakumar, have generously given their time and expertise to better my work. I thank them for their contribution and their good-natured support.

The work in this dissertation was partially supported by the National Science Foundation under grant CCR-0098357 and CCF-0635390. I want to express my appreciation to NSF for their financial support. Also, I want to thank Advanced Hardware Architectures, Inc., of Moscow, ID, for experience and financial support they provided through a summer internship in 2006.

I also wish to thank all the staff of the School of Electrical Engineering and Computer Science, for their help and support.

Finally, I would like to express my deepest gratitude to my parents, my brother and my wife, for their love, support, patience and encouragement.

EQUALIZATION AND CODING FOR THE TWO-DIMENSIONAL

INTERSYMBOL INTERFERENCE CHANNEL

Abstract

by Taikun Cheng, Ph.D.
Washington State University
December 2007

Chair: Benjamin Belzer

This dissertation addresses the problems of detection and error correction for binary data corrupted by two-dimensional (2D) intersymbol interference (ISI) and additive white Gaussian noise (AWGN). The 2D-ISI channel occurs in proposed next-generation optical disk systems and holographic storage systems, which store data in 2D pages, and are subject to 2D-ISI at high data densities.

This dissertation presents a novel iterative row-column soft decision feedback algorithm (IRCSDFA) which exchanges weighted soft information between row and column maximum-a-posteriori (MAP) detectors. Each MAP detector exploits soft-decision feedback from previously processed rows or columns. The new algorithm gains about 0.3 dB over the previously best published results for the 2×2 averaging mask. For a non-separable 3×3 mask, the IRCSDFA gains 0.8 dB over a previous soft-input/soft-output iterative algorithm which decomposes the 2D convolution into 1D row and column operations. This dissertation also shows how to adapt the IRCSDFA for non-equiprobable binary sources. And this dissertation proposes an approach to optimize the iteration schedule inside the

IRCSDFA by using EXIT charts.

This dissertation addresses the error floor problem of low-density parity check (LDPC) codes on the binary-input AWGN channel, by constructing a serially concatenated code consisting of two systematic irregular repeat-accumulate (IRA) component codes connected by an interleaver. The interleaver is designed to prevent stopping-set error events in one of the IRA codes from propagating into those of the other code. Simulations with two 128-bit rate 0.707 IRA component codes show that the proposed architecture achieves a much lower error floor at higher SNRs, compared to a 16384-bit rate 1/2 IRA code, but incurs an SNR penalty of about 2 dB at low to medium SNRs. Experiments indicate that the SNR penalty can be reduced at larger blocklengths.

This dissertation also demonstrates an iterative joint detection/decoding scheme for the 2D ISI channel by employing IRCSDFA and LDPC codes. Simulation results show that about $1/3$ of the gain achieved by IRCSDFA over the previous best equalization algorithm is retained when the two equalizers are combined with LDPC codes.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Fast growing information technologies such as high-speed internet, ultra wideband communication, and multi-media applications need to exchange and save huge amounts of digital information. Traditional data storage systems read/write data on a disk or tape along one dimensional tracks spaced sufficiently far apart to avoid inter-track interference. Even third generation optical disk storage systems (e.g., the BlueRay standard) also are still based on 1D tracks. Some recently proposed systems, such as the "TwoDos" project sponsored by Phillips [1], aim to achieve densities at least two times greater than BlueRay, and data rates at least ten times greater than BlueRay, by decreasing the inter-track spacing and reading the data from many (typically, 10 or more) tracks simultaneously. Decreased inter-track spacing causes inter-track interference when reading the disk. The inter-track interference, plus the intra-track interference between same-track bits, can be modeled as two dimensional ISI.

The emerging technology of holographic storage (HS) aims to achieve densities more

than thirty times that of BlueRay by encoding bits as laser interference patterns in light-sensitive disks [2]. The disks store bits in layers of (approximately square) 2D pages, with typically millions of bits on a page. Future HS systems may even suffer from inter-page interference, leading to 3D-ISI.

Without effective equalization, 2D and 3D ISI will cause unacceptablely high bit error rates (BERs) in proposed next-generation optical and magnetic storage systems. Multi-dimensional ISI equalization is thus a key enabling technology for such systems.

To achieve a very low BER on the data storage systems (typically $10^{-12} - -10^{-15}$), error correction coding (ECC) is always used together with the equalizer. The low density parity check (LDPC) codes, with their near-capacity performance in the waterfall region of the BER vs. SNR curve, have been the subject of much recent research, and are of interest for next-generation data storage systems.

This dissertation proposes a 2D equalization approach and a novel serial concatenated LDPC coding structure with low error floor. The performance of combined 2D-ISI equalization and LDPC coding is also studied.

1.1 System Model and Background of 2D ISI Equalization

An $M \times N$ binary 2D data set **f** with elements $f(m,n) \in \{-1,1\}$ corrupted by 2D ISI and AWGN can be modeled as received data set **r** with elements

$$r(m,n) = \sum_k \sum_l h(k,l) f(m-k, n-l) + w(m,n), \tag{1.1}$$

where $h(k,l)$ is a finite-impulse-response 2D blurring mask, the $w(m,n)$ are zero mean independent and identically distributed (i. i. d.) Gaussian random variables (r. v. s) with variance σ_w^2, and the double sum is computed over the mask support region $\mathcal{S}_h = \{(k,l) : h(k,l) \neq 0\}$. It is assumed that a boundary of -1 elements surrounds the data set. In the following, we refer to 2D data set as "images," and their elements as "pixels."

Direct maximum likelihood (ML) detection of $f(m,n)$ from $r(m,n)$ requires comparison of $r(m,n)$ with 2^{N^2} candidate transmitted images, and is therefore impractical for typical image dimensions of $N \geq 128$. The standard Wiener filtering solution is significantly inferior to ML detection, especially at high SNR. Hence, it is desirable to develop a low-complexity 2D detection algorithm that achieves or approximates the performance of 2D ML detection. For one dimensional signals, the Viterbi algorithm provides an efficient method for ML detection of ISI-corrupted data [7]. Unfortunately the VA does not generalize to two or higher dimensions. For 2D ISI, issues of scan-order, adjacency, and causality must be considered in construction of the trellis, and the mapping between input pixel sequences and trellis paths is not always one-to-one. However, union bounds on

the performance of 2D ML detection have been developed in [10]; these ML bounds are tight at high SNR, and are useful in assessing the performance of sub-optimal 2D detection algorithms.

A number of 2D decision-feedback VA (DFVA) algorithms have been constructed, based on a row-by-row raster scan ordering of the image pixels (e.g. [3, 4, 11]); the best of these ([4],[11]) attain or approximate the performance of ML detection at high SNR by using hard decisions in a fixed number of previously decoded rows. Performance tends to be mask dependent. For example, the algorithm in [11], when applied to binary images corrupted by ISI from the 3×3 averaging mask, achieves a bit error rate (BER) of about 3×10^{-4} at an SNR about 3 dB higher than the ML bound curve, yet attains ML performance when the ISI is due to a length-five horizontal averaging mask.

To our knowledge, [11] employed the first iterative algorithm for 2D ISI reduction; the DFVA was run on rows and columns, and bits which agreed in both directions were fixed for subsequent iterations. This scheme effectively exchanged hard decisions between row and column DFVAs. Subsequent work has employed the turbo principle (after turbo coding [33]); i.e., detection reliability can be greatly improved by exchanging soft estimates of the detected bits between two or more estimators. In [15], the 2D convolution operation is decomposed into two 1D computations, and an iterative decoding algorithm exchanges soft information between SISO detectors corresponding to each 1D operation; decision feedback is not necessary under this approach. In [9], mask separability is exploited to construct an interactive row-column detector for LDPC coded binary images, in which ex-

trinsic information is exchanged between a non-binary column SISO detector, a binary row detector, and a LDPC decoder. In [16, 17], soft information is exchanged between MAP detectors operating on multiple rows and multiple columns; this scheme avoids decision feedback by making decisions on multiple rows/columns, rather then one row/column at a time, and also handles non-separable masks.

Generalized belief propagation (GBP) is developed for 2D ISI equalization and related problems in [5, 6]. GBP uses exact inference over the sub-region of the image covered by the ISI mask, and then passes messages between adjacent sub-regions. GBP works well, but (in [5]) has been demonstrated only on small (20×20 or smaller) images; such cases are easy to handle for 2D equalizers because the nearby boundary conditions greatly aid the estimation. Also, [5] only considers ISI masks where the amplitude ratio between the bit being estimated and the ISI bit is greater than one, whereas other publications consider more challenging masks (e.g., the $m \times m$ averaging mask $h(k, l) = 1/m^2$).

The computational complexities of the equalizers discussed in this dissertation are linear in the total number of image pixels $M \times N$, but exponential in the mask size. This complexity is typical of the above-mentioned equalizers, except for GBP, which appears to have somewhat higher complexity because messages are passed to and from all four sides of a sub-region, instead of along a trellis defined by a single scan direction. The other exception is the separable algorithm of [9], which has computational complexity that is exponential in the square root of the mask size. However, the algorithm of [9] can be used only with separable masks.

1.2 Background on LDPC Coding

Low density parity check (LDPC) codes, introduced by Gallager in the early 1960s [27], have received great interest since researchers in the late 1990s and early 2000s ([28, 30, 31]) showed that they can perform within less than 0.1dB of the Shannon limit for a number of important communication channels, including the binary erasure channel and the binary-input AWGN channel. However, for the above-cited codes, near-capacity performance typically holds only above bit error rates (BERs) of 10^{-5} or 10^{-6}; at lower BERs, the nearly vertical (and highly negative) slope of the BER vs. SNR curve levels off into an "error floor" with a smaller magnitude slope.

As there are several important applications that require BERs of 10^{-12} or lower (e.g., mass storage, broadband satellite communications), a number of recent publications have proposed LDPCs specially designed to reduce the error floor. IRA codes, introduced in [25] by Jin, Khandekar, and McEliece, feature a section H_2 of the parity check matrix H that contains only weight-two columns (except for one weight-1 column), and consists of "1"s down the main diagonal and the diagonal just below it. A lemma proved in [32] shows that if the H_2 section contains all the weight two columns of H, then it helps lower the error floor because H_2 contains the maximum number of degree-two variable nodes without a cycle among them. Extended IRA (e-IRA) codes, introduced in [32], are a generalization of systematic IRA codes wherein the remaining section ("H_1") of the H matrix assumes a more general form; design rules for lowering the error floor of e-IRA codes by optimizing

the degree distributions of H_1 are given in [32]. IRA codes and e-IRA codes have the low decoding complexity characteristic of LDPC codes, and the low encoding complexity characteristic of turbo codes [25, 32, 33].

LDPC error floors are caused by connected sets of cycles called "stopping sets" [34]. Codes with larger stopping sets generally have lower error floors. The design technique in [35] attempts to maximize stopping set size by maximizing the average number of connections leading outside small cycles, referred to as the ACE distance d_{ACE}; simulations showed that LDPC codes with larger d_{ACE} had lower error floors. More recently, the authors of [36] proposed a method of directly estimating the variable and check nodes in the smallest stopping sets, along with a code design algorithm to directly maximize the size of these sets. The design algorithm in [36] resulted in codes with significantly lower error floors than those designed according to [35].

1.3 Main Contributions

The primary contribution of this dissertation is a new iterative soft-decision feedback MAP detection algorithm for reduction (or elimination) of 2D ISI. For the 3×3 averaging mask, our algorithm achieves about 1.5 dB of SNR improvement over that of [11]. For a more rapidly decaying 3×3 mask, our algorithm achieves about 0.8 dB gain over that of [15]. For the 2×2 averaging mask, the IRCSDFA gains about 0.3 dB over the separable algorithm of [9] (without coding), the previously best published result for that mask. A key innovation

of our approach is the use of *soft decision feedback* (SDF) from previously processed rows (or columns) of the image; we believe this is the first application of SDF to the 2D-ISI problem. (We note, however, that an iterative algorithm with SDF was proposed earlier for the 1D channel equalization problem [20].)

The EXIT chart, introduced in [21] is a good method to analyze the convergence properties of iterative algorithms; EXIT charts have typically been applied to analyze iterative decoding of error correction codes. In this dissertation, the convergence properties of the IRCSDFA are analyzed by using EXIT charts. To prevent the IRCSDFA system from converging too fast, we apply a weight schedule to the exchanged soft information between the row and column detectors. The design of the weights schedules was mainly based on repeated simulations in prior work (e.g., [44]), which is not efficient. In this dissertation, we propose a method to optimize the weight schedules by using EXIT charts.

Another main contribution of the dissertation is a method for design of serially concatenated IRA codes that achieve lower error floors than single IRA codes of equivalent rate and block size. Two systematic component codes, with block length and rate equal to the square roots of those of a comparable full-length IRA code, are connected in series, with an interleaver between them. This architecture is similar to that of turbo product codes [37], except that, rather than employing the row-column interleaver of product codes, we design the interleaver to avoid the convergence problems that lead to error floors. We use the method of [36] to estimate the stopping sets of the component codes. Then the stopping set data is used to design the interleaver so that, as much as possible, stopping set error events

of one of the component codes are not mapped into stopping set variable nodes of the other code. Since each component code has the ability to successfully decode the other code's non-convergent blocks, convergence problems are greatly reduced, resulting in a lowered error floor at high SNR. Because of the IRA component codes, the concatenated system has relatively low encoding complexity compared to a general irregular LDPC code. The decoding complexity is about twice that of the comparable full-length IRA code, due to the need for outer iterations between the component codes.

Based on the algorithms discussed above, we design a concatenated system including the IRCSDFA and a LDPC code, similar to the turbo equalization structure in 1D ISI case [23]. The simulation results show that the combined IRCSDFA and LDPC system is better than the previous best joint equalization/decoding system [9] under the same channel environment, when identical LDPC codes are used in both systems. In particular, about one-third of the SNR gain of the IRCSDFA over the separable equalizer employed in [9] is retained when these equalizers are combined with LDPC codes.

1.4 Dissertation Outline

Chapter 2 introduces the IRCSDF algorithm after a brief review of the BCJR algorithm, and then extends the IRCSDFA from equiprobable source data to non-equiprobable source data. Simulation experiments under different conditions, and performance comparisons with previous approaches are also discussed. Factor graphs of IRCSDFA are presented and

discussed at the end of this chapter. The material of this chapter was partially presented at the $38th$ Conference on Information Science and Systems [40] on March 2004, $39th$ Conference on Information Science and Systems [41] on March 2005, $43rd$ Annual Allerton Conference on Communication, Computing, and Control [42] on September 2005, $44th$ Annual Allerton Conference on Communication, Computing, and Control [43] on September 2006, and the journal articles of "IEEE Signal Processing Letters" [44] on July 2007, "IEEE Transactions on Signal Processing" [46] on Nov. 2007.

Chapter 3 analyzes the convergence properties of IRCSDFA by using EXIT charts. An iteration and weight schedule optimization method is also proposed. An example of numerical optimization of the IRCSDFA is given in this chapter.

Chapter 4 describes the serially concatenated IRA code, which can achieve a lower error floor compared with a single IRA code of the same rate and block length. The main steps of this code design are the stopping sets detection of the component codes, and the interleaver design to separate these stopping sets. The content of this chapter was presented at the 45th annual Allerton Conference on Communication, Computing, and Control [45], in September 2007.

The construction and performance of combined IRCSDFA detection and LDPC decoding is presented in chapter 5. The conclusions of this dissertation are given in chapter 6.

Chapter 2

Iterative Row-Column Soft-Decision Feedback Algorithm (IRCSDFA)

In this chapter, we will introduce a novel Iterative Row-Column Soft-Decision Feedback Algorithm (IRCSDFA) for the detection of binary 2D source data corrupted by two-dimensional intersymbol interference (2D ISI) and additive white Gaussian noise (AWGN). The algorithm described in this chapter exchanges soft information between row and column maximum a-posterior (MAP) detectors, each detector exploits *soft-decision feedback (SDF)* from previously processed rows or columns. The exchanged soft information, extrinsic information, is weighted to slow down the iteration convergence and give some performance improvement.

2.1 IRCSDFA Trellis Definition

We have shown the system model of 2D ISI on the first chapter, here just briefly re-summarize it for convenience: an $M \times N$ binary 2D data set **f** with elements $f(m,n) \in \{-1,1\}$ corrupted by 2D ISI and AWGN can be modeled as received data set **r** with ele-

ments

$$r(m,n) = \sum_k \sum_l h(k,l) f(m-k, n-l) + w(m,n), \tag{2.1}$$

where $h(k,l)$ is a finite-impulse-response 2D blurring mask, the $w(m,n)$ are zero mean independent and identically distributed (i. i. d.) Gaussian random variables (r. v. s) with variance σ_w^2, and the double sum is computed over the mask support region $\mathcal{S}_h = \{(k,l) : h(k,l) \neq 0\}$.

The simplest 2D intersymbol interference channel, 2×2 mask, is shown in Figure 2.1(a). The 2D convolution can be viewed as the inner product of the original image $f(m,n)$ with the inverted mask $h(-k,-l)$ as shown in Figure 2.1(b), where mask coefficient h_{00} is at pixel position (m,n). The binary image pixels $f(m,n)$ take values -1 or $+1$. The inverted mask scans through the image pixels $f(m,n)$ in the row-by-row and column-by-column pattern respectively.

Figure 2.1: (a) 2×2 mask (b) 2×2 mask inverted for 2D convolution.

For the row-by row case, we use Miller et. al.'s method [11] to define the IRCSDFA trellis construction as shown in Figure 2.2: the 4-state trellis has 4 branches entering and leaving each state resulting in a fully connected structure. At each position (m, n) the trellis branch output is a vector consisting of two 2×2 inner products between the inverted mask and the pixel values defined by the trellis; the upper inner product, named $x(m, n)$, uses the feedback pixels and the lower one, named $x(m + 1, n)$, just uses received pixels. The branch metric is the squared Euclidean distance between the branch output and the real received pixel vector [$r(m, n), r(m + 1, n)$]:

$$[r(m, n) - x(m, n)]^2 + [r(m + 1, n) - x(m + 1, n)]^2. \tag{2.2}$$

The column-by-column case is similar to the row-by-row case.

Trellis generation for the 3×3 mask on the mth image row is initiated by placing the input marked (m, n) in Fig. 2.3 at the left end of the row, where the initial values of the six state pixels are -1 due to the boundary conditions, and the vector of three input pixels can take eight different values. The entire state/input block is then shifted right to pick up the next three input pixels, and the previous three input pixels become the middle three state pixels. The trellises for each row are terminated at the right end of the row by extra shifts into the boundary pixels. For the 3×3 mask, the 64-state trellis has 8 branches entering and leaving each state, with no parallel branches. It's not a fully connected trellis like 2×2 case. At each position (m, n), the trellis branch output vector consists of three 3×3 inner

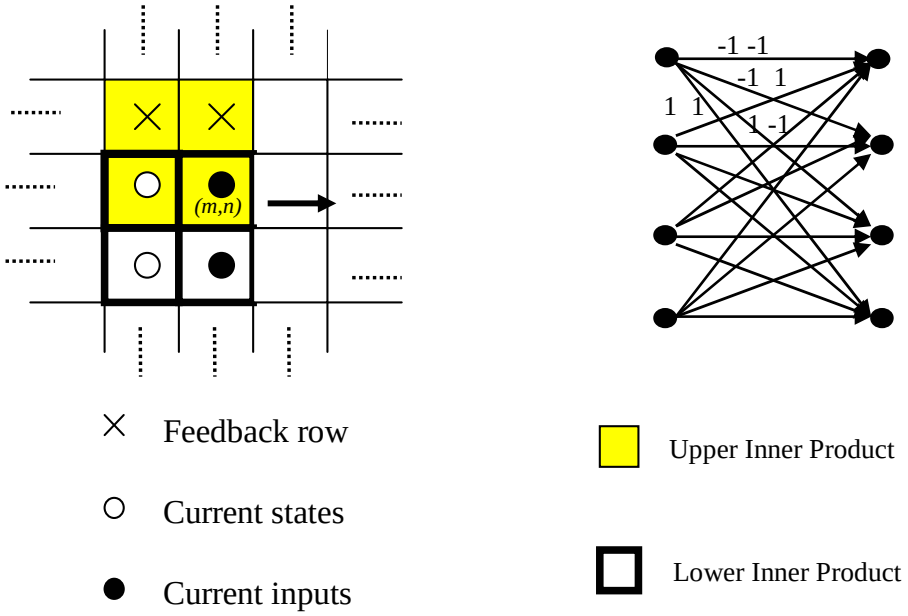

Figure 2.2: 2×2 mask trellis structure

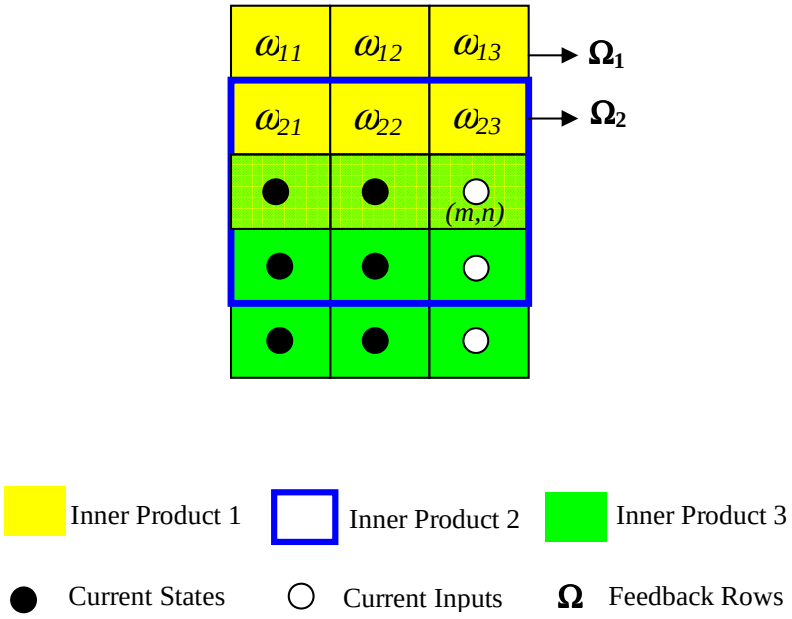

Figure 2.3: 3×3 mask trellis structure

products between the inverted mask and the pixel values defined by the trellis; the upper inner product uses two feedback rows, the middle uses one feedback row, and the lower uses received pixels only. The branch outputs and metrics are computed similarly to the 2×2 mask, but with three 3×3 inner products rather than 2 because of 2 feedback rows.

We can improve system performance by extending the state and input blocks shown in Fig. 2.2 by one or more rows. For the 2×2 mask, adding one more row gives an 8 state trellis with 8 branches per state, and three (rather than two) inner products per branch metric. The column-by-column case is similar to the row-by-row case.

As the image pixels are i.i.d., the above-described trellis constructions impose the Markov condition that, given the current trellis state, subsequent states and branch out-

puts are independent of past states or outputs. This Markov condition allows the use of a modified BCJR [18] algorithm for detection.

2.2 IRCSDF Algorithm Description

Fig. 2.4 shows a block diagram of the algorithm. The basic element is a *soft decision feedback, soft-input soft-output* (SDF-SISO) detector. Each SDF-SISO processes received image **r**, which is corrupted by 2D-ISI and by AWGN. The SDF-SISOs use a modified BCJR [18] algorithm, in which soft estimates of branch outputs from earlier trellis stages are used as SDF to aide computation of the current pixel's log-likelihood ratio (LLR).

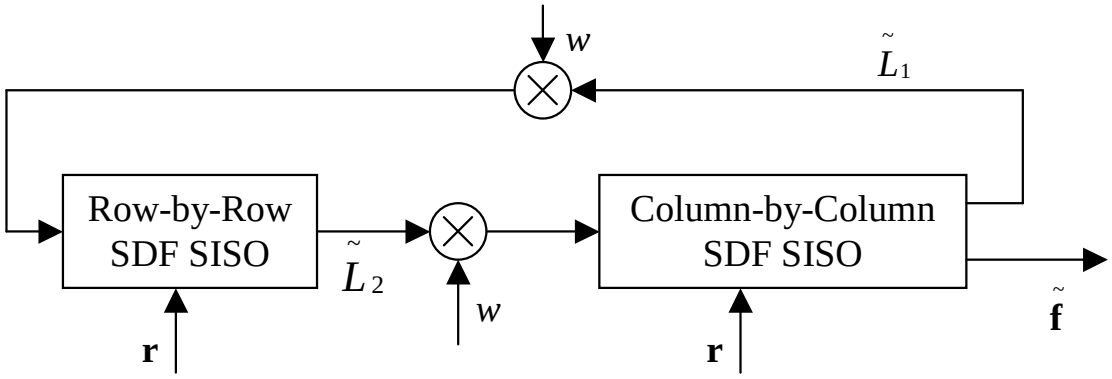

Figure 2.4: IRCSDFA block diagram

The row and column SDF-SISOs exchange weighted soft information. The SDF-SISOs assume that their decision feedback is correct, but in fact it contains errors. Decision feed-

back errors cause error propagation; an example is shown in Fig. 2.5(d), which shows the error map from a Monte-Carlo simulation of the IRCSDFA on the i.i.d. binary source image shown in Fig. 2.5(a). When the information weights w are set to one (the non-weighted case), the algorithm converges in two to three iterations. By employing the weight schedule $w = 0.008 \times (3 \times k^2 + 1)$, where k, $1 \leq k \leq 6$, is the iteration number, we slowed the convergence of the algorithm to six iterations, at all SNRs tested. (SNR gains from two additional iterations were limited to at most 0.01 dB, and they occurred at low SNR.) For the 2×2 averaging mask at high SNR, we observed that this weight schedule gave us an SNR gain of about 0.5 dB over the non-weighted case, at a bit error rate (BER) of 2×10^{-5}. (This particular weight schedule was arrived at experimentally after a non-exhaustive search, and must therefore be considered sub-optimal.)

A theoretical analysis of the weight schedule and its optimization by using EXIT chart will be discussed in chapter 3. Based on observations, we hypothesize that decision feedback errors make the output LLRs larger than their true values, and that the weights move the LLRs closer to their true reliabilities, thereby preventing the algorithm from converging too quickly.

2.2.1 BCJR Algorithm Review

The BCJR algorithm is an algorithm for maximum a posteriori (MAP) decoding of error correcting codes defined on trellises. The algorithm is named after its inventors: Bahl, Cocke, Jelinek and Raviv [18]. This algorithm is critical to modern iteratively-decoded

error-correcting codes such as turbo codes.

Standard BCJR algorithm works as following: suppose we received a signal sequence R_1^N from channel with length N, to decode the k^{th} bit, d_k, $1 \leq k \leq N$, we can define the log-likelihood ratio (LLR) of d_k as:

$$LLR(d_k) = log\left(\frac{p(d_k = +1|R_1^N)}{p(d_k = -1|R_1^N)}\right) \tag{2.3}$$

where

$$p(d_k = i|R_1^N) = \frac{p(d_k = i, R_1^N)}{p(R_1^N)}, \quad i \in \{-1, +1\}.$$

$p(d_k = i, R_1^N)$ could be written as:

$$p(d_k = i, R_1^N) = \sum_s \sum_{s'} p(d_k = i, S_k = s, S_{k-1} = s', R_1^N), \tag{2.4}$$

s and s' are the states of the trellis. Since R_1^N are Markov, by using Bayes' rule, we can

have

$$
\begin{aligned}
&p(d_k = i, S_k = s, S_{k-1} = s', R_1^N) \\
&= p(R_{k+1}^N | S_k = s, S_{k-1} = s', R_1^{k-1}, R_k, d_k = i) \cdot p(S_k = s, S_{k-1} = s', R_1^{k-1}, R_k, d_k = i) \\
&= p(R_{k+1}^N | S_k = s, S_{k-1} = s', R_1^{k-1}, R_k, d_k = i) \cdot p(S_k = s, R_k, d_k = i | S_{k-1} = s', R_1^{k-1}) \\
&\cdot p(S_{k-1} = s', R_1^{k-1}) \\
&= p(R_{k+1}^N | S_k = s) \cdot p(S_k = s, R_k, d_k = i | S_{k-1} = s') \cdot p(S_{k-1} = s', R_1^{k-1}).
\end{aligned}
\tag{2.5}
$$

So, by defining

$$
\begin{aligned}
&\alpha_k(s) = p(S_k = s, R_1^k) \\
&\beta_k(s) = p(R_{k+1}^N | S_k = s) \\
&\gamma_k^i(R_k, s', s) = p(d_k = i, S_k = s, R_k | S_{k-1} = s'),
\end{aligned}
\tag{2.6}
$$

and

$$
\lambda_i^k(s) = \sum_{s'} \alpha_{k-1}(s') \cdot \gamma_k^i(R_k, s', s) \cdot \beta_k(s), \tag{2.7}
$$

we can calculate the LLR as

$$
LLR(d_k) = log\left(\frac{\sum_s \lambda_{+1}^k(s)}{\sum_s \lambda_{-1}^k(s)} \right) \tag{2.8}
$$

The values of $\alpha_k(s)$ and $\beta_k(s)$ could be updated by following equations:

$$\begin{aligned}\alpha_k(s) &= \sum_{s'}\sum_{i}\gamma_k^i(R_k, s', s)\alpha_{k-1}(s') \\ \beta_k(s) &= \sum_{s'}\sum_{i}\gamma_k^i(R_{k+1}, s, s')\beta_{k+1}(s'),\end{aligned} \tag{2.9}$$

the initial values of α and β are

$$\alpha_0(0) = 1 \qquad \alpha_0(s) = 0 \quad if\ s \neq 0;$$

and

$$\beta_N(0) = 1 \qquad \beta_N(s) = 0 \quad if\ s \neq 0.$$

2.2.2 IRCSDF Algorithm

IRCSDFA is a modified BCJR algorithm, the key modification is the SDF branch output calculation: computing LLRs for inner products between the mask and candidate binary estimates $\hat{f}(m, n)$ of the image pixels. To illustrate the SDF LLR calculation on row scans, without losing generality, assume the 3×3 averaging mask $h_{kl} = 1/9$ is used to compute the convolution

$$c(m, n) = \sum_{k=0}^{2}\sum_{l=0}^{2}\hat{f}(m-k, n-l)/9.$$

For pixel (m, n) at the kth trellis stage, $k \in \{0, 1, \ldots, N\}$, the corresponding received pixel vector is

$$\mathbf{r} = [r(m, n), r(m + 1, n), r(m + 2, n)],$$

and the input vector is

$$\hat{\mathbf{f}} = [\hat{f}(m, n), \hat{f}(m + 1, n), \hat{f}(m + 2, n)],$$

as shown in Fig. 2.3. To simplify, let

$$\mathbf{y}_k = [y_{k0}, y_{k1}, y_{k2}] = \mathbf{r},$$

and

$$\mathbf{u} = [u_{k0}, u_{k1}, u_{k2}] = \hat{\mathbf{f}}.$$

The LLR is

$$L_i(k) = \log\left(\frac{P(u_{k0} = +1|\mathbf{y}_k, \tilde{\mathbf{u}}_i)}{P(u_{k0} = -1|\mathbf{y}_k, \tilde{\mathbf{u}}_i)}\right), \tag{2.10}$$

where $\tilde{\mathbf{u}}_i$ is the extrinsic estimate of $\mathbf{u}$ passed to detector i, $i \in \{1, 2\}$, from the other detector. Detector i's extrinsic LLR input is

$$\tilde{L}_i(k) = \log\left(\frac{P(u_{k0} = +1|\tilde{\mathbf{u}}_i)}{P(u_{k0} = -1|\tilde{\mathbf{u}}_i)}\right),$$

and the extrinsic LLR output to the next detector is

$$\tilde{L}_{\mathrm{next}(i)}(k) = L_i(k) - \tilde{L}_i(k),$$

where $\mathrm{next}(1) = 2, \mathrm{next}(2) = 1$. By using the input extrinsic information, we can compute the conditional probability of the input pixel:

$$\begin{aligned} P(u_{k0} = +1|\tilde{\mathbf{u}}_i) &= \frac{e^{\tilde{L}_i(k)}}{1 + e^{\tilde{L}_i(k)}} \\ P(u_{k0} = -1|\tilde{\mathbf{u}}_i) &= \frac{1}{1 + e^{\tilde{L}_i(k)}}. \end{aligned} \tag{2.11}$$

Given trellis state S_k, input vector $\mathbf{u}$, and received vector $\mathbf{y}_k$, define

$$\lambda_k^{\mathbf{i}}(s) = P(\mathbf{u}_k = \mathbf{i}, S_k = s, \mathbf{y}_k),$$

where $\mathbf{i} = [i_0, \ldots, i_{n_b}]$, $i_m \in \{-1, +1\}$, and n_b is the number of input bits per trellis stage. We can then compute the *a posteriori*

$$P(\mathbf{u}_k = \mathbf{i}|\mathbf{y}_k) = \sum_s \lambda_k^{\mathbf{i}}(s)/P(\mathbf{y}_k). \tag{2.12}$$

as in [18], by setting

$$\begin{aligned}
&\alpha_k(s) = P(S_k = s, \mathbf{y}_k) \\
&\beta_k(s) = P(\mathbf{y}^{k+1} | S_k = s) \\
&\gamma_{\mathbf{i}}(\mathbf{y}^k, s', s) = P(\mathbf{u}_k = \mathbf{i}, S_k = s, \mathbf{y}^k, \widetilde{\mathbf{u}}_1^N | S_{k-1} = s'),
\end{aligned} \tag{2.13}$$

where $\mathbf{y}^k, \mathbf{u}_k, \mathbf{i}, \widetilde{\mathbf{u}}_1^N$ are vectors because there are more than one row in the trellis, The SDF output LLRs can be incorporated into the pixel transition probabilities $\gamma_{\mathbf{i}}(\mathbf{y}_k, s', s)$. The modified γ is the product of a modified conditional channel PDF $p'(\cdot)$, trellis transition probabilities, and extrinsic information from the other detector:

$$\begin{aligned}
\gamma_{\mathbf{i}}(\mathbf{y}_k, s', s) &= p'(\mathbf{y}_k | \mathbf{u} = \mathbf{i}, S_k = s, S_{k-1} = s') \\
&\times P(\mathbf{u} = \mathbf{i} | s, s') \times P(s | s') \times P(\tilde{\mathbf{u}} | \mathbf{u} = \mathbf{i}).
\end{aligned} \tag{2.14}$$

For the given states s', s and input $\mathbf{u}$, $P(\mathbf{u} = \mathbf{i}|s, s')$ is 0 or 1 and $P(S_k = s | S_{k-1} = s')$ is 2^{-n_b} based on the trellis. The extrinsic information can be computed as:

$$P(\tilde{\mathbf{u}} | \mathbf{u} = \mathbf{i}) = \frac{P(\mathbf{u} = \mathbf{i} | \tilde{\mathbf{u}}) P(\tilde{\mathbf{u}})}{P(\mathbf{u} = \mathbf{i})}, \tag{2.15}$$

where $P(\mathbf{u} = \mathbf{i}|\tilde{\mathbf{u}})$ comes from (2.11), and $P(\tilde{\mathbf{u}}) = P(\mathbf{u}) = 2^{-n_b}$.

The modified channel PDF sums over the values of inner products c_{sdf} associated with

state transition $s' \rightarrow s$ that are affected by past decisions:

$$\begin{aligned} p'(\mathbf{y}_k|\mathbf{u}=\mathbf{i}, S_k=s, S_{k-1}=s') &= P\big(y_{k2}|u_{k0}, u_{k1}, u_{k2}, s, s'\big) \\ &\times \Bigg[\sum_{\Omega_2} P(\Omega_2) P\big(y_{k1}|u_{k0}, u_{k1}, s, s', c_{\mathrm{sdf2}}(\Omega_2), \Omega_2\big) \\ &\times \sum_{\Omega_1} P(\Omega_1) P\big(y_{k0}|u_{k_0}, s, s', c_{\mathrm{sdf1}}(\Omega_1, \Omega_2), \Omega_1, \Omega_2\big)\Bigg] \end{aligned} \tag{2.16}$$

where $\mathbf{\Omega}$ denotes feedback rows, inner product $c_{\mathrm{sdf}j}(\Omega)$ depends on the feedback pixels, and the row probabilities

$$P(\Omega_j = \omega_{j0}, \omega_{j1}, \omega_{j2}) = \prod_{l=0}^{2} P(\omega_{jl}),$$

where $P(\omega_{jl})$ are feedback pixel probabilities. For the 3×3 averaging-mask ISI channel, inner products $c_{\mathrm{sdf1}}(\Omega_1, \Omega_2)$ and $c_{\mathrm{sdf2}}(\Omega_2)$ are nine-pixel averages of the pixels labeled "inner product 1" and "inner product 2" in the 3×3 mask of Fig. 2.3. The $P(\omega_{jl})$ are computed by using feedback LLRs from previously processed rows (or columns) during the current iteration. Since the original image is subject to AWGN, the

$$p'(\mathbf{y}_k|\mathbf{u}=\mathbf{i}, S_k=s, S_{k-1}=s', c_{\mathrm{sdf}j}(\Omega))$$

are normal PDFs with means $c_{\mathrm{sdf}j}(\Omega)$ and variances σ_w^2.

Since we have vector inputs and received pixels, to estimate the pixel located on (m, n),

we sum the λs over $(m+1,n)$ and $(m+2,n)$:

$$\lambda_k^{i0}(s) = \sum_{i1,i2} \lambda_k^{i0,i1,i2}(s). \tag{2.17}$$

The pixel LLR is computed as:

$$L(k) = \log\left(\frac{\sum_s \lambda_k^{i0=+1}(s)}{\sum_s \lambda_k^{i0=-1}(s)}\right). \tag{2.18}$$

If $L(k) > 0$, we decide that pixel (m,n) is $+1$; otherwise, it is detected as -1.

2.3 IRCSDFA for Non-Equiprobable Sources

We can modify IRCSDF to estimate the non-equiprobable source data blurred by 2D ISI. If we define the $\lambda_k^i(s)$ and LLRs as in equiprobable case, equations (2.13) and (2.14) still hold. The difference is the γ calculating: $P(S_k = s|S_{k-1} = s')$ in equation (2.14) will have different values for different given states values, i.e., the transition probability from s to s' is different.

By finishing the non-equiprobable modification, IRCSDFA could be applied to any source data such as the correlated image. On this case, the source image need to be interleaved before being passed through the 2D ISI channel because we assume the source data are i.i.d. in IRCSDFA. The corresponding work could be found on [43].

2.4 IRCSDFA Simulation Results

In this section, we present Monte Carlo simulation results for the IRCSDFA on the random binary image $f(m,n)$ with pixel alphabet $\{-1,+1\}$. The plots in this section show the bit error rate (BER) of the estimated binary input image, versus signal noise ratio (SNR). The SNR is defined as in [11]:

$$\text{SNR} = 10\log_{10}\left(\text{var}\left[f * h\right]/\sigma_w^2\right), \tag{2.19}$$

where $*$ denotes 2D convolution, , σ_w^2 is the variance of the Gaussian r.v.s $w(m,n)$ in (2.1). To compute received image $r(m,n)$, we assume a boundary of -1 pixels around the original image $f(m,n)$; the receiver uses this known boundary condition to simplify the trellis at image edge pixels.

Fig. 2.6 shows IRCSDFA simulation results on a random 128×128 binary image blurred by the 2×2 averaging mask and AWGN. We plot results using two, three, and four rows in the 2×2 trellis state and input block of Fig. 2.3, which shows the basic two-row case. The maximum likelihood estimator (MLE) union upper bound of [10] is also plotted. In addition, we plot results for the row-by-row MAP with SDF (but without column extrinsic information), and iterative row-column MAP with extrinsic LLR exchange but with HDF (rather than SDF) on past rows (IRCHDF); these additional results are based on the basic (2 row) trellis definition. Plots marked "Opt. Weights" use the six-iteration weight schedule given in chapter 4; plots marked "Unit Weights" fix the weights at 1.0.

at BER 2×10^{-5}	
4 trellis rows IRCSDFA with opt. weights	0.6 dB
3 trellis rows IRCSDFA with opt. weights	0.6 dB
seperable algorithm of [9]	0.9 dB
2 trellis rows IRCSDFA with opt. weights	1.1 dB
3 trellis rows IRCSDFA with unit weights	1.2 dB
2 trellis rows IRCSDFA with unit weights	1.7 dB
2 trellis rows IRCHDFA with opt. weights	2.0 dB
2 trellis rows row-by-row SDF	2.5 dB

Table 2.1: SNR gap to the ML upper bound for 2×2 averaging mask

At high SNR, row-column iteration (with the weight schedule) gives about 1.5 dB SNR gain over the rows-only method. Row-column SDF gives high-SNR gains of about 1 dB over row-column HDF. The three-row IRCSDFA gives about 0.5 dB gain over the two-row algorithm at high SNR. The four-row IRCSDFA performs as well as the three-row at high SNR, and offers gains of up to 0.3 dB at lower SNRs. Additional state/input rows allow the algorithm to correct larger error patterns, which occur more frequently at low SNR. The SNR gaps between the ML upper bound and different algorithms at BER 2×10^{-5} are listed in Table 2.1.

For comparison, we plot simulation results for the separable algorithm of [9]. (Because [9] reports results only for one iteration of the separable equalization algorithm without LDPC coding, we implemented the equalization algorithm and tested its multi-iteration performance. We found that two iterations of the separable algorithm achieve almost all available performance gain, so two iterations are used in all separable-algorithm results pre-

sented here.) At a BER of about 1×10^{-5}, the 3-row IRCSDFA achieves about 0.3 dB gain over the separable algorithm; at a BER of 6×10^{-6}, the 3-row IRCSDFA performs within about 0.6 dB of the ML bound. Another advantage of the IRCSDFA is it works for general 2D masks, whereas the separable algorithm must use the closest separable approximation to a non-separable mask, which leads to an error floor in many cases.

We also compare to recent results by Marrow and Wolf [16, 17]. These results use the 2×2 mask with first row $1, \alpha$ and second row $\alpha, 0$, where α varies between 0 and 1. The Marrow-Wolf (M-W) algorithm reported in [16, 17] also employs row/column MAP decoding with soft information exchange, and their trellis (like our two-row version in Fig. 2.3) is fully connected with 4 states and 4 branches/state. Marrow-Wolf intentionally avoid decision feedback by estimating two rows, rather than one row at a time. Based on a comparison of Fig. 2.7, which is the result of 2-row IRCSDFA with unit weight, to a corresponding figure in [17] (the zoom-in graph for $\alpha = 0.7$ is also presented in [17], both results use 5×5 binary input images cause [17] just presents the result for such a simple experiment.),we believe the two-row version of our algorithm with unit weights is essentially equivalent to the 2×2 mask M-W algorithm. In Fig. 2.6, the two-row IRCSDFA with unit weights is about 1 dB worse than the three-row curve with optimized weights at BER 2×10^{-5}; hence, we believe our optimized algorithm's performance is about 1 dB better than the M-W algorithm, for the 2×2 averaging mask. The IRCSDFA in this dissertation, while similar to that of [16], was developed independently, and has several key differences. First, we make decisions one row at a time and use SDF, rather than making

decisions two or more rows at a time and using "feed-forward" [16]. Second, we weight the extrinsic information passed between SISOs, and increase the weights with each iteration; the weight schedule significantly improves the algorithm's performance. Third, we achieve additional gains by adding rows (respectively, columns) to the state and input pixel blocks of the row (column) SISOs. And fourth, we demonstrate performance with both 2×2 and 3×3 masks on 128×128 and 64×64 images, whereas the maximum source image size considered in [16, 17] is 5×5.

The improved performance of the three- and four-row IRCSDFAs comes at a complexity cost relative to the two-row version, and relative to the separable algorithm of [9]. The number of operations per pixel for the two, three, and four row IRCSDFAs, and the separable algorithm, are as follows: add/subtract, 423, 1935, 8735, and 480; multiply/divide, 742, 3398, 15366, and 943; exp/log, 87, 391, 1543, and 75. We note that the two-row IRCSDFA complexity is roughly equal to that of the separable algorithm.

Fig. 2.8 shows IRCSDFA simulation results on a random 64×64 binary image blurred by the 3×3 averaging mask ($h_{ij} = 1/9$, $(i, j) \in \{0, 1, 2\}$) and AWGN; here we use the three-row trellis definition shown in Fig. 2.3. The MLE upper bound for this mask is also plotted. We also plot simulation results of the iterative row-column hard-decision feedback algorithm. The IRCHDF and IRCSDF results shown in Fig. 2.8 were run with the weight schedule described in chapter 4. At high SNR, the IRCSDFA requires about 1.2 dB more SNR than the MLE. By comparison, an earlier iterative algorithm by Miller et. al. [11] is about 3 dB away from the ML bound. Also, by comparing to the 2×2 averaging mask

case, we can say that more feedback rows result in a larger SNR gain due to SDF.

We also simulate a random 128×128 image blurred by the 3×3 mask (named Channel B) defined by Chen and Chugg in [15]:$h(0,0) = h(0,2) = h(2,0) = h(2,2) = 0.0993$; $h(0,1) = h(1,0) = h(1,2) = h(2,1) = 0.352$; $h(1,1) = 1$. The results are shown in Fig. 2.9. (Chen and Chugg's original curve in [15] has been left-shifted by 3 dB, to account for differing SNR definitions.) The IRCSDFA gives about 0.8 dB of gain compared to [15]; also, Chen-Chugg's curve diverges from the ML bound, whereas the IRCSDFA is parallel to it.

Fig. 2.10 shows the performance of the Non-equal-probal IRCSDFA for different source pixel distribution, $P(+1) = 0.7$, $P(+1) = 0.9$ and $P(+1) = 0.99$, blurred by the 2×2 mask. We also simulated the original equal-probal IRCSDF for these 3 sources for comparing. We can see from the figure, the higher the difference between the probabilities of the binary source, the bigger gain the modified IRCSDFA could give us. This is very helpful to restore some non-uniform source such as correlated data cause most of them are non-equal-probal.

2.5 Factor Graphs of IRCSDFA

As a modified BCJR algorithm, this proposed IRCSDFA could be analyzed by using factor graphs. Following the steps in section IV of [19], we plotted the factor graphs of the IRCSDFA in Fig. 2.11. For comparison, we also plotted the factor graphs of the separable

algorithm of [9] in Fig. 2.12. All these factor graphs are based on an example: restoring the 4×4 source data blurred by the 2×2 mask.

In Fig. 2.11 we can see, there is no cycle inside the Row or Column SISO detector; the shortest length of the cycle between these two detectors is $4m + 6$, where $m = N + 1$ is the number of stages of the detector trellis for a $N \times N$ source data block. Therefore, even for a very small size source block, there is no short cycle in IRCSDFA. For the separable algorithm of [9] in Fig. 2.12, we can see there are some cycles inside its sub-detectors, and that the shortest cycle length is $2m + 2$. For a general source block size, we can still say there are no short cycles in the separable algorithm. The example shortest cycles of IRCSDFA and separable algorithm are marketed by red lines in Fig. 2.11 and Fig. 2.12 respectively.

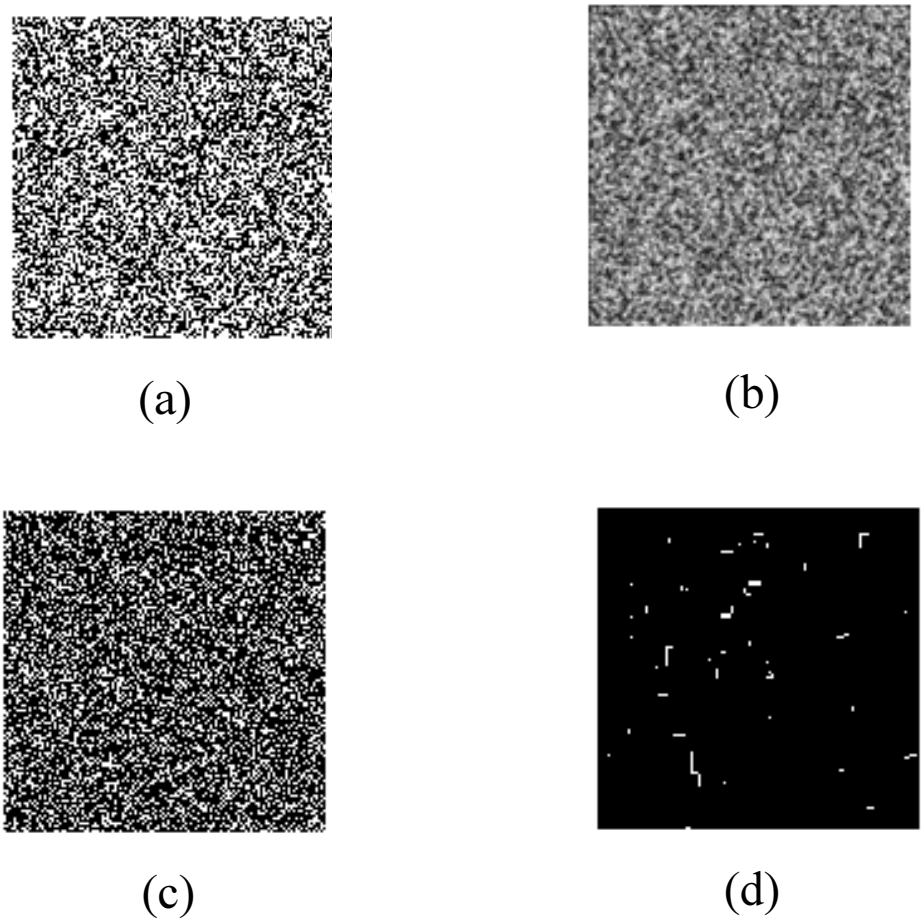

Figure 2.5: (a) Source image; (b) Source with 2D ISI and noise; (c) error image from hard decoding of image (b) (white pixels are errors, black are correct); (d) error image from IRCSDFA algorithm.

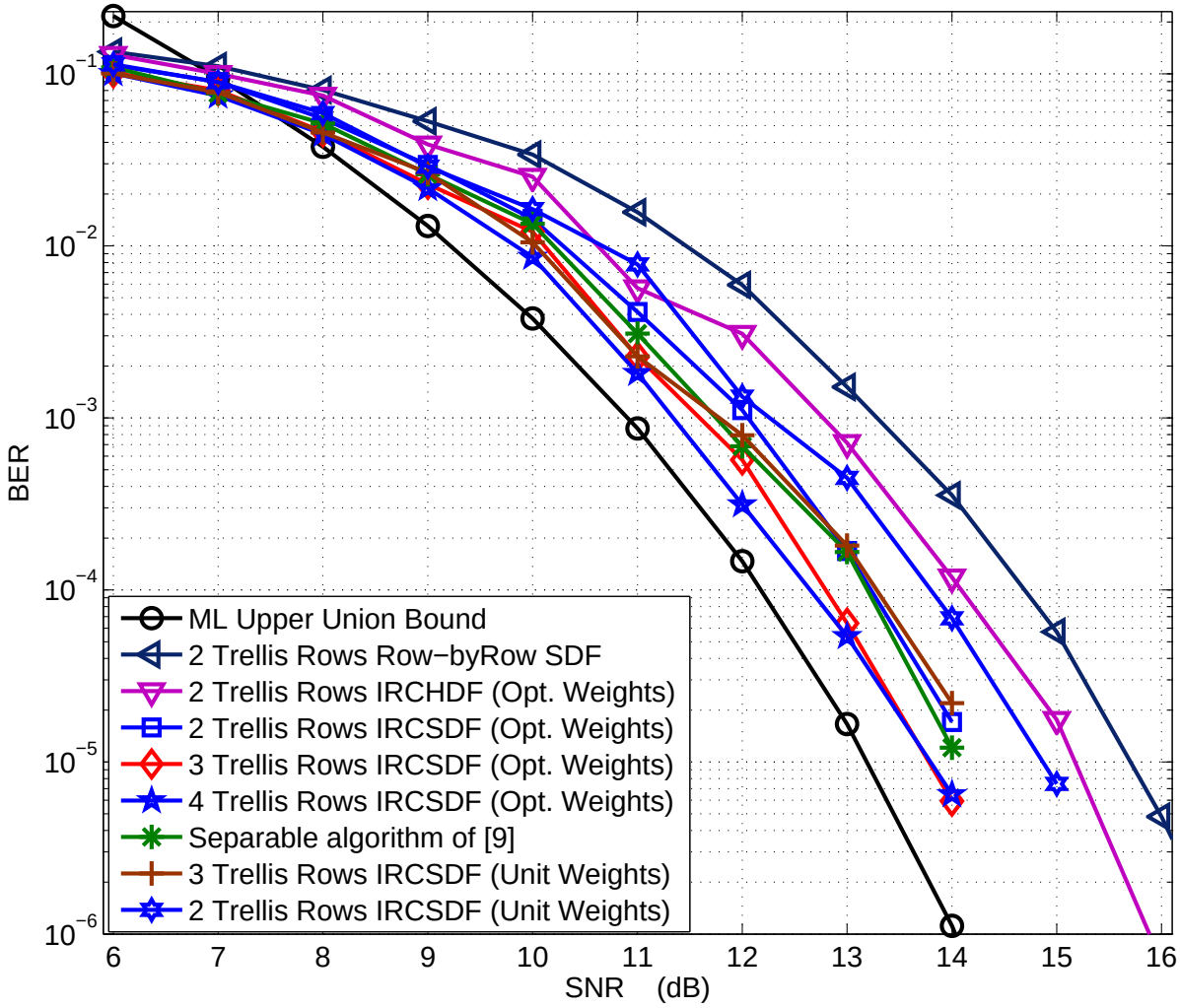

Figure 2.6: Simulation results for 128×128 binary input image and 2×2 averaging mask.

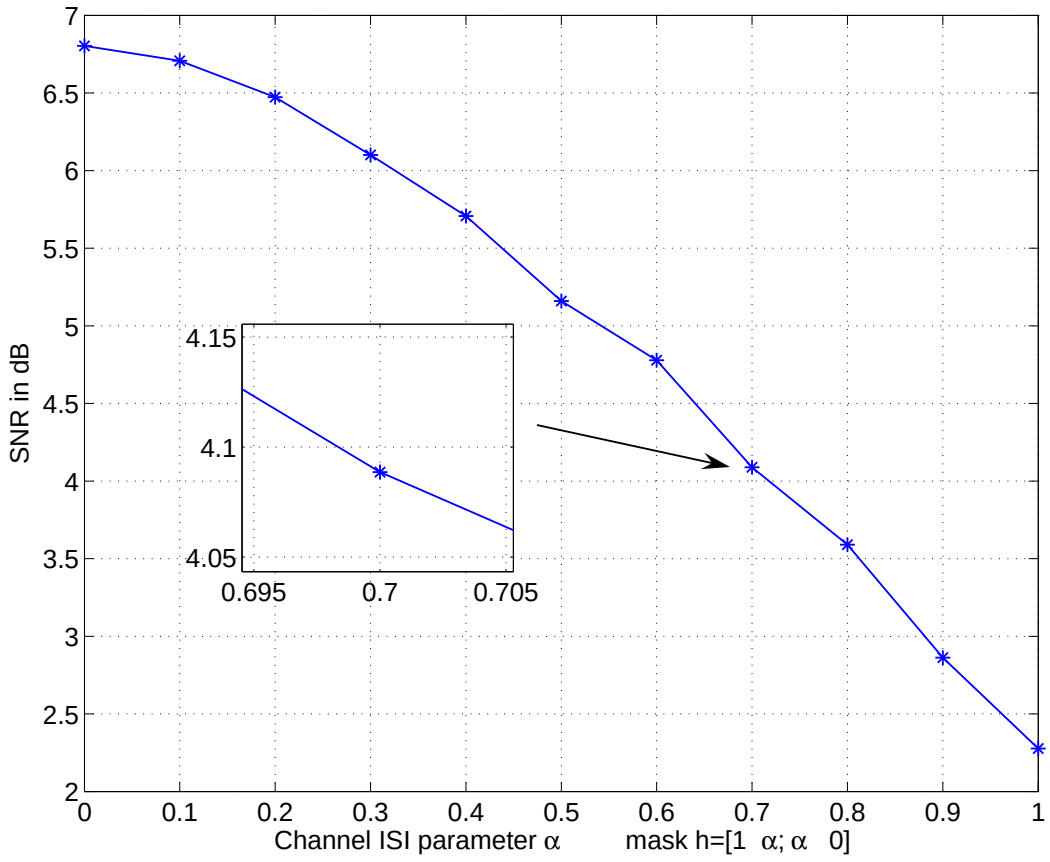

Figure 2.7: Results of 2-row IRCSDFA algorithm under the same condition as Marrow-Wolf. The vertical axis shows the SNR necessary to achieve a BER of 0.001, and the horizontal axis is the ISI parameter α.

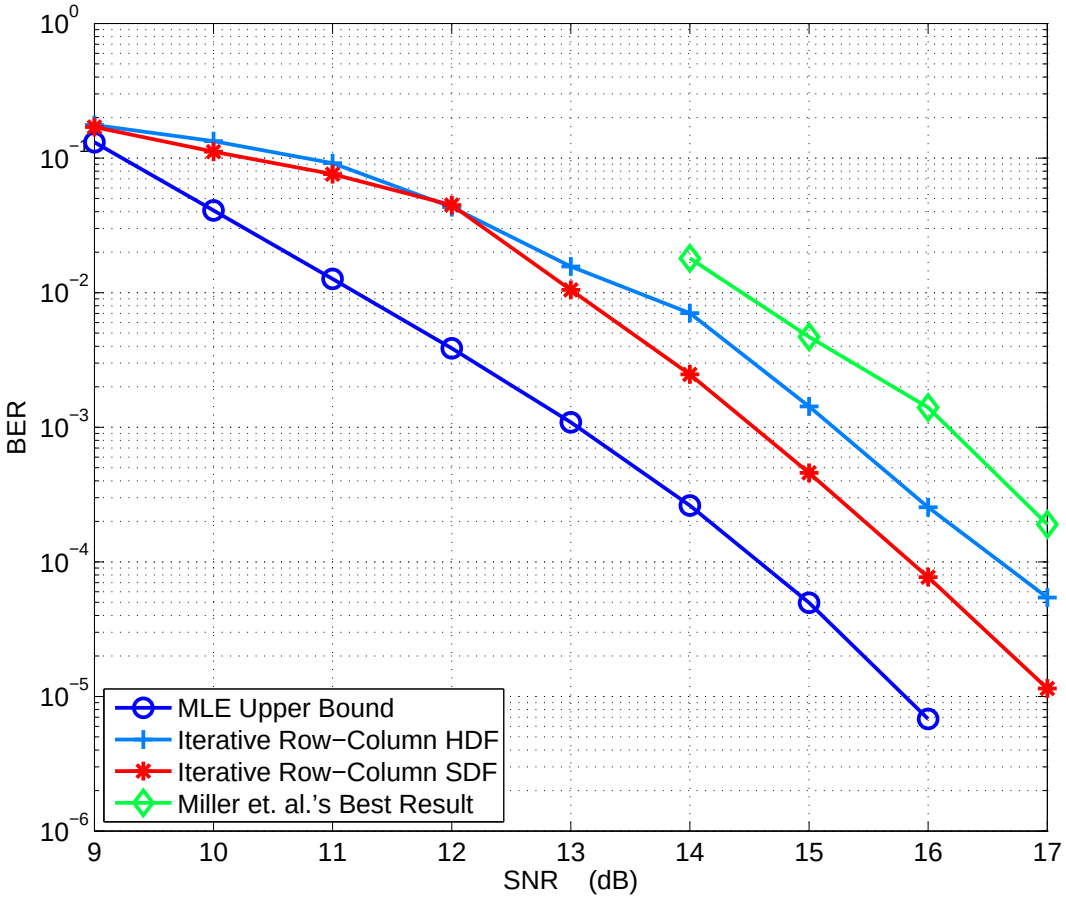

Figure 2.8: Simulation results for 3×3 averaging mask.

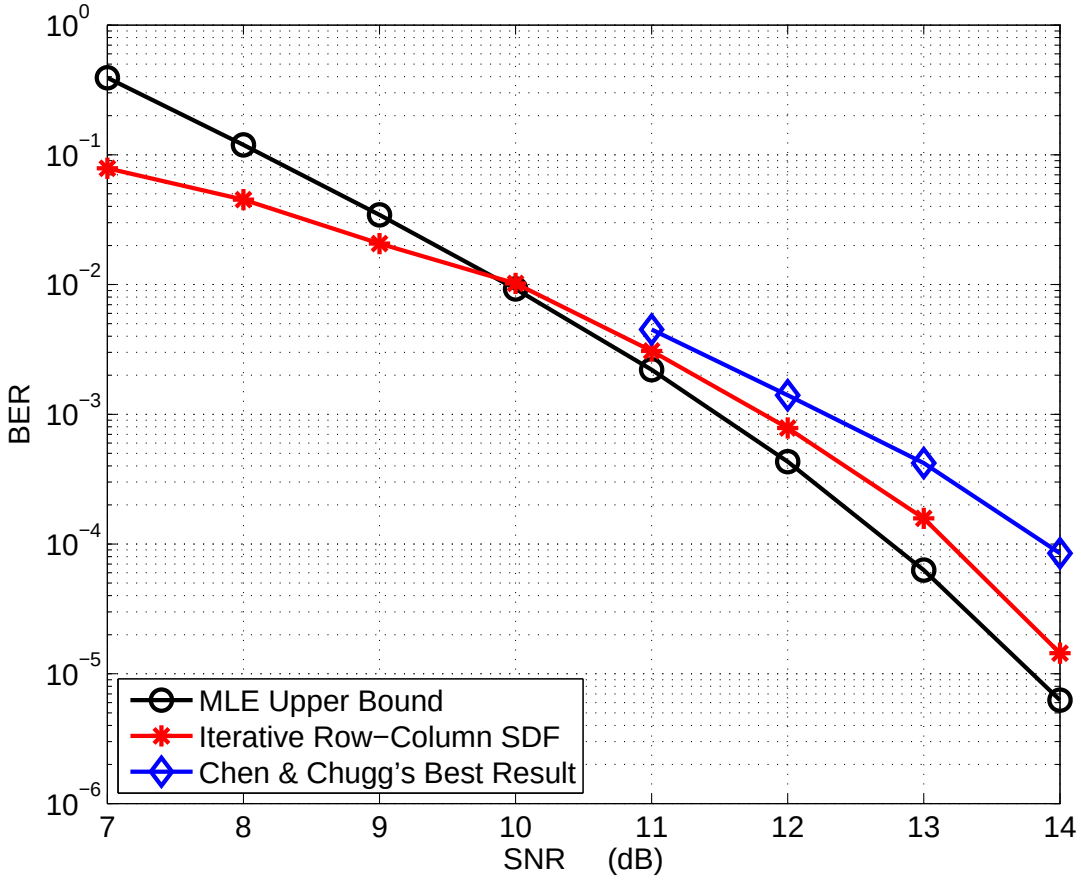

Figure 2.9: Simulation results for channel B.

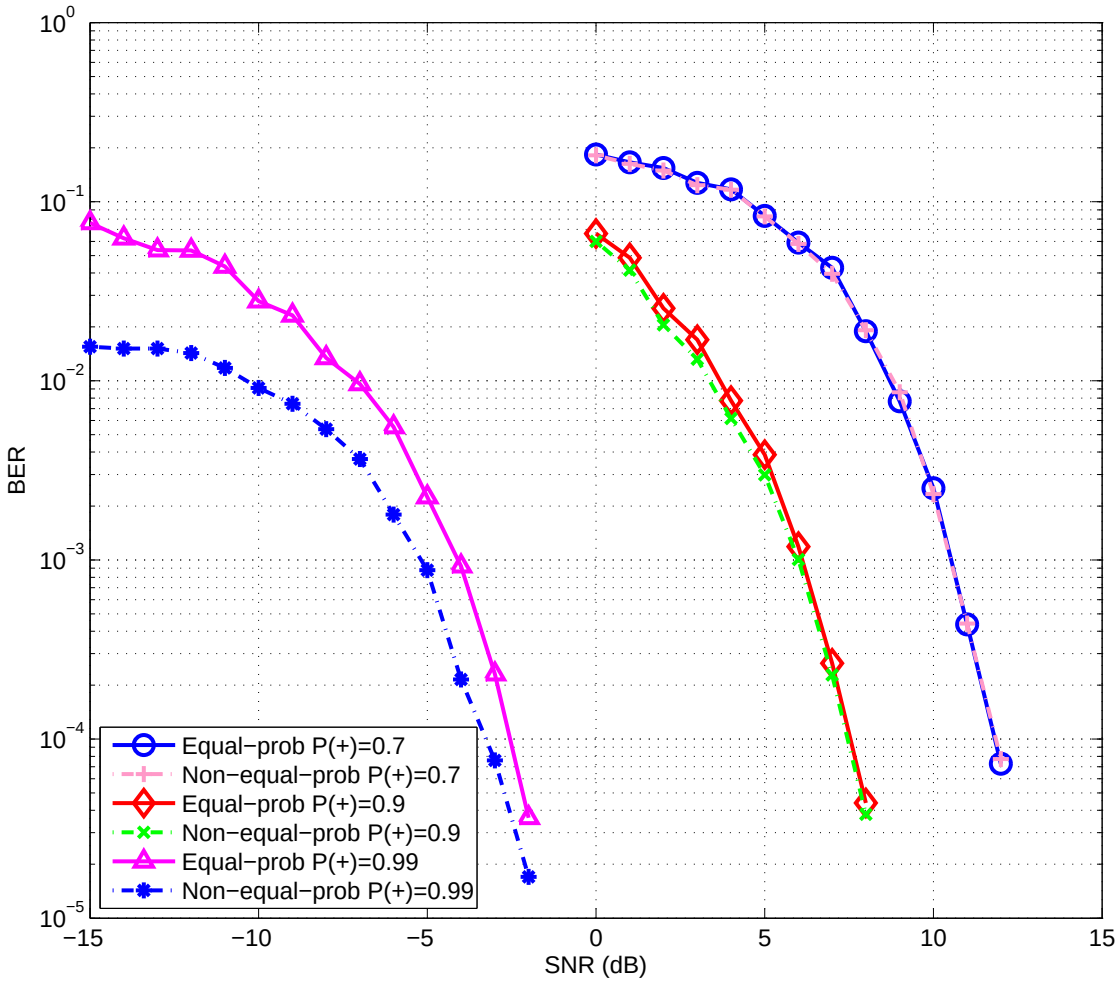

Figure 2.10: Simulation results for Non-equal-probal IRCSDFA.

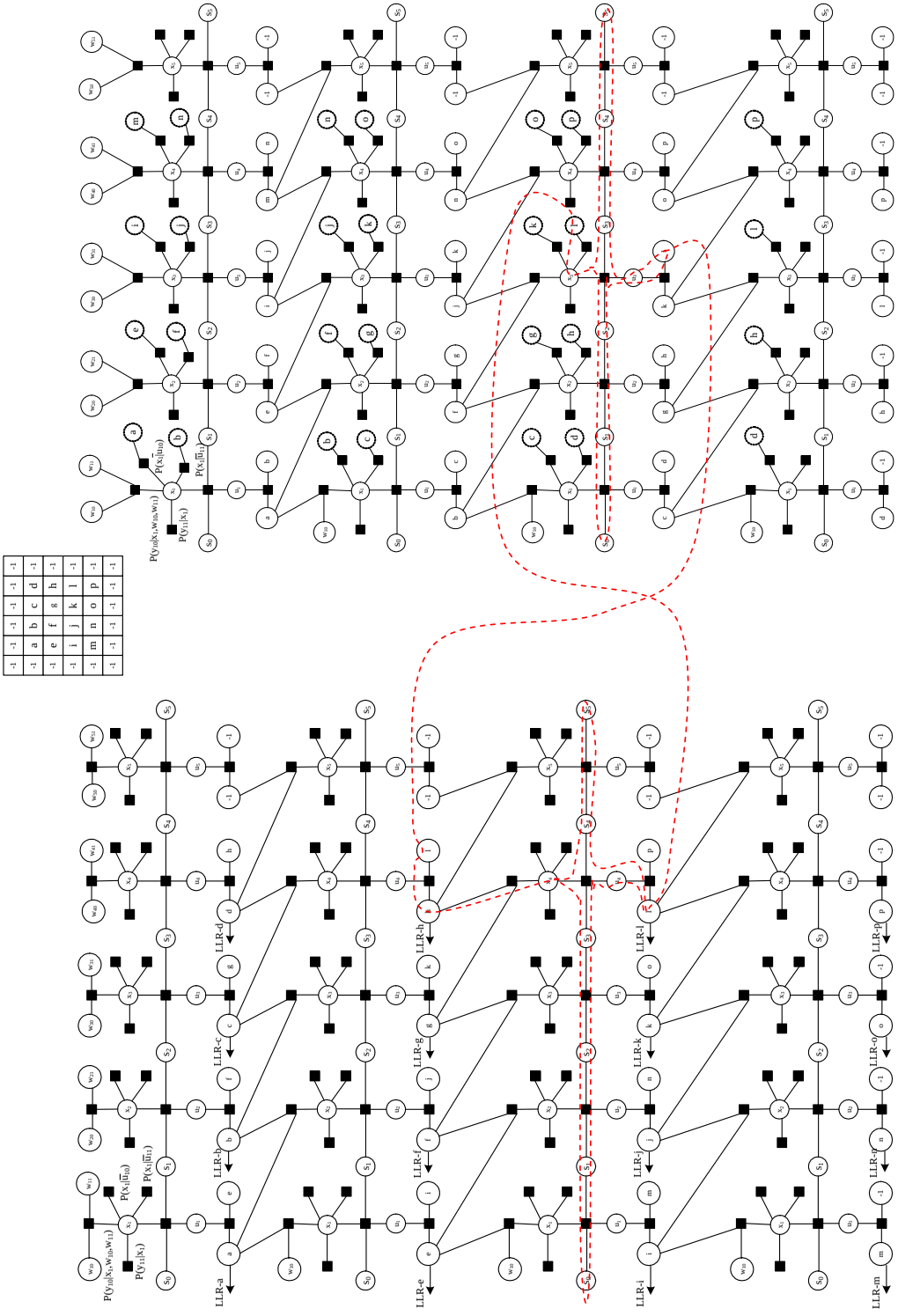

Figure 2.11: Factor Graphs of IRCSDFA.

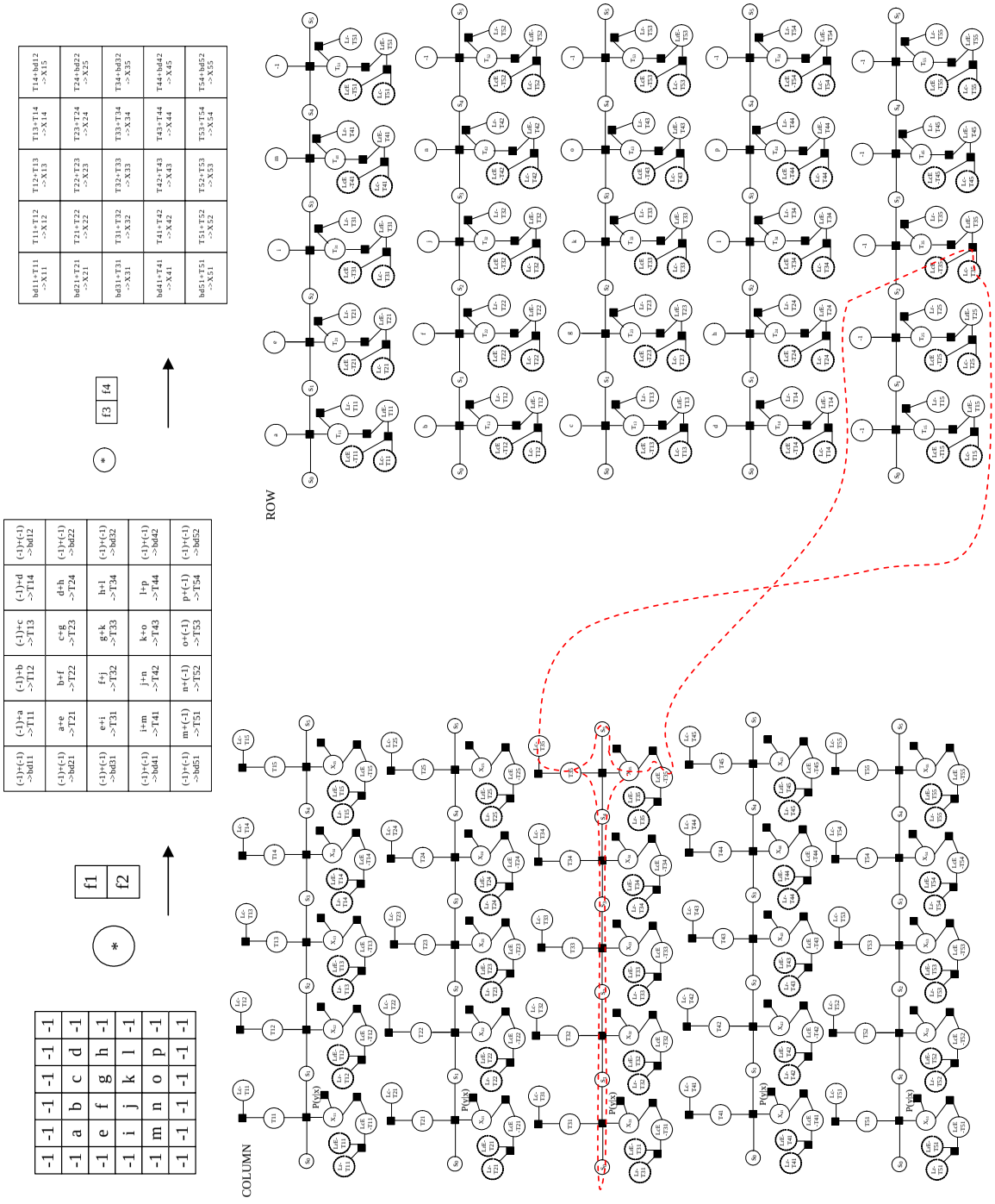

Figure 2.12: Factor Graphs of the Separable algorithm.

Chapter 3

Analyzing And Optimizing IRCSDFA by Using EXIT Chart

We can analyze the convergence properties of IRCSDFA by using the EXIT chart as in [21], since IRCSDFA also has a similar iterative structure. The EXIT charts also help to optimize the weights between row and column detectors during the iterations.

3.1 The Probability Distribution of The Extrinsic Information

The input information to the detectors of IRCSDFA are independent Gaussian random variables; as mentioned in [22], the output can be tightly approximated by a Gaussian random variable. Also, when the iteration number increases, the pdf of the extrinsic information should be more and more Gaussian (observed in [24]). We did the same experiment and observed that the pdf of the extrinsic information output by the row/column detector is well approximated by a Gaussian pdf, especially after several iterations.

Similar to [21], we derived the relationship between the mean and variance of the LLR for the IRCSDFA system. To simplify, all the derivations are based on the 2×2 ISI chan-

nel. Suppose we want to estimate the pixel on $\{m, n\}$. The received signal from the 2D ISI+AWGN channel on this position is $y = z + n$, where n is the zero mean Gaussian random variable with variance σ_n^2, and z is the 2-D convolution between the original pixels and the 2D ISI channel, which can be expressed as the inner product of the pixel x_0, x_1, x_2, x_3 and mask f_0, f_1, f_2, f_3:

$$z = x_0 \cdot f_0 + x_1 \cdot f_1 + x_2 \cdot f_2 + x_3 \cdot f_3. \tag{3.1}$$

The original pixel we want to estimate is x_3. Given $x_3 = x, x \in \{+1, -1\}$, the conditional probability density function (*pdf*) can be written as:

$$p(y|x_3 = x) = \frac{e^{-\frac{(y-z)^2}{2\sigma_n^2}}}{\sqrt{2\pi}\sigma_n}. \tag{3.2}$$

The corresponding *L*-values extrinsic information LLR are calculated as:

$$LLR = \ln \frac{p(y|x_3 = +1)}{p(y|x_3 = -1)}, \tag{3.3}$$

which can be simplified to:

$$\begin{aligned} LLR &= \ln\left(e^{-\frac{(y-z(+1))^2-(y-z(-1))^2}{\sqrt{2\pi}\sigma_n}}\right) \\ &= \frac{2[z(+1)-z(-1)]\cdot y + [z(+1)^2 - z(-1)^2]}{\sqrt{2\pi}\sigma_n} \\ &= \frac{2yf_3 - 2f_3\cdot(x_0f_0 + x_1f_1 + x_2f_2)}{\sigma_n^2} \\ &= \frac{2f_3\cdot(x_3f_3+n)}{\sigma_n^2} \end{aligned} \tag{3.4}$$

where $z(+1), z(-1)$ represent the inner products given the conditions that x_3 is +1 or -1. Therefore, we can write LLR as: $LLR = \mu_L \cdot x_3 + n_L$, with $\mu_L = \frac{2f_3^2}{\sigma_n^2}$, and n_L being a zero mean Gaussian r.v. with variance $\sigma_L^2 = \frac{4f_3^2}{\sigma_n^2}$. Thus, $\mu_L = \frac{\sigma_L^2}{2}$.

Based on these conclusions, we derived the mutual information between extrinsic information and the source pixel, which has the same form as the equation (19) in [21]. What's more, since we put some weight schedule on the LLR in each detector, we proved the weights will not change the mutual information. Then, we can plot the EXIT chart for IRCSDFA and analyze its properties.

3.2 EXIT Charts of IRCSDFA With Different Trellis Rows

From the simulation results of the 2×2 mask above, we found including some additional rows in the trellis can improve the system performance. An EXIT chart analysis shows this is reasonable.

Fig. 3.1 shows the mutual information between the SISO detector output extrinsic in-

formation and the signal to be restored (I_{out}) v.s. the mutual information between the input LLR and the source signal (I_{in}). We plotted 2 trellis-rows, 3 rows and 4 rows for channel SNR from 0 dB to 10 dB with the step of 2 dB. The lower group of lines correspond to the lower SNR. We can see, at very low SNR, the additional rows give very little gain; but at higher SNRs, additional rows do give some gain, especially the first added row. At the high SNR level, adding more than one row will not give too much gain compared to adding just one row because the I_{out} is already on a high level. This figure is consistent with the Monte Carlo simulation results we presented before, and it can help us to design the trellis structure for different SNR range.

3.3 Using EXIT Chart to Design The Weights Schedule

In [21], a method is presented to use EXIT chart to analyze the convergence properties of an iterative system. However, we proved the weights will not change the I_{out} for the SISO detector in the section 3.1. In order to make use of this method, we modified the EXIT chart: if we think of the connected row detector and column detector as a whole, the weight w put on the LLR from the row-SISO to the column-SISO is an inner parameter of this bigger detector. Based on simulation we found this weight w did make the I_{out} from the column-SISO detector lower by driving the LLR input to the column detector lower. Therefore we can get different EXIT charts of this bigger detector for different weights, then by analyzing the convergence behavior we can design a good weight schedule.

We start from 2 weight schedules, $w_1 = 0.045(k+1)$ and $w_2 = 0.008(3k^2+1)$, where k is the iteration time. From the Monte Carlo simulation we know w_2 has a better BER performance than w_1 after 6 iterations at the same SNR level. The EXIT charts of w_1 and w_2 are shown in Fig. 3.2 and Fig. 3.3. Both figures are plotted based on the 3 trellis rows of the 2×2 mask IRCSDFA at channel SNR 6 dB. As mentioned before, the inner parameter w will change the I_{out}, so there are 6 groups of the curves corresponding to 6 iterations; the lower/left curves are for the earlier iterations. The trajectories connect the kth and $(k+1)th$ lines. Comparing these two figures, we can see at the end of the $6th$ iteration, the final I_{out} of w_2 is bigger than the one of w_1; that's why w_2 has a better performance than w_1. Also, during the BER simulation we found w_1 made the system almost converge after 4 iterations while w_2 didn't converge until 6 iterations. By reading these two figures, we can see the length of each trajectory in Fig. 3.3 is more uniform than that in Fig. 3.2. The length of the trajectory represents the volume of the changed mutual information between the input/output extrinsic information and the source data; the higher the change in mutual information, the bigger the gain the detector gave during this iteration. Since the system of w_1 made the most change during the first 4 iterations while system w_2 didn't, the w_1 system of course converged faster.

By studying these results, we can propose some rules for weight schedule design:

1. Try to make the end I_{out} as close to the upper limit as possible. This limit can be found by finding the interception point of 2 symmetric weight 1 systems (such as the interception point of the solid lines in Fig. 3.2 and Fig. 3.3).

2. Try to make the length of the trajectories as uniform as possible to avoid early converge.

3. Select an appropriate iteration time based on the trade off between the performance requirement and the detecting speed requirement.

We designed a set of weights $w_3 = [0.0001, 0.005, 0.03, 0.06, 0.12, 0.25, 0.4, 0.6, 0.8, 1]$ by following these rules. Fig. 3.4 is the EXIT chart on same condition as Fig. 3.2 and Fig. 3.3. Simulation results show it has a 3.45% lower BER than w_1 system and 2.31% lower BER than w_2 system at 6 dB channel SNR.

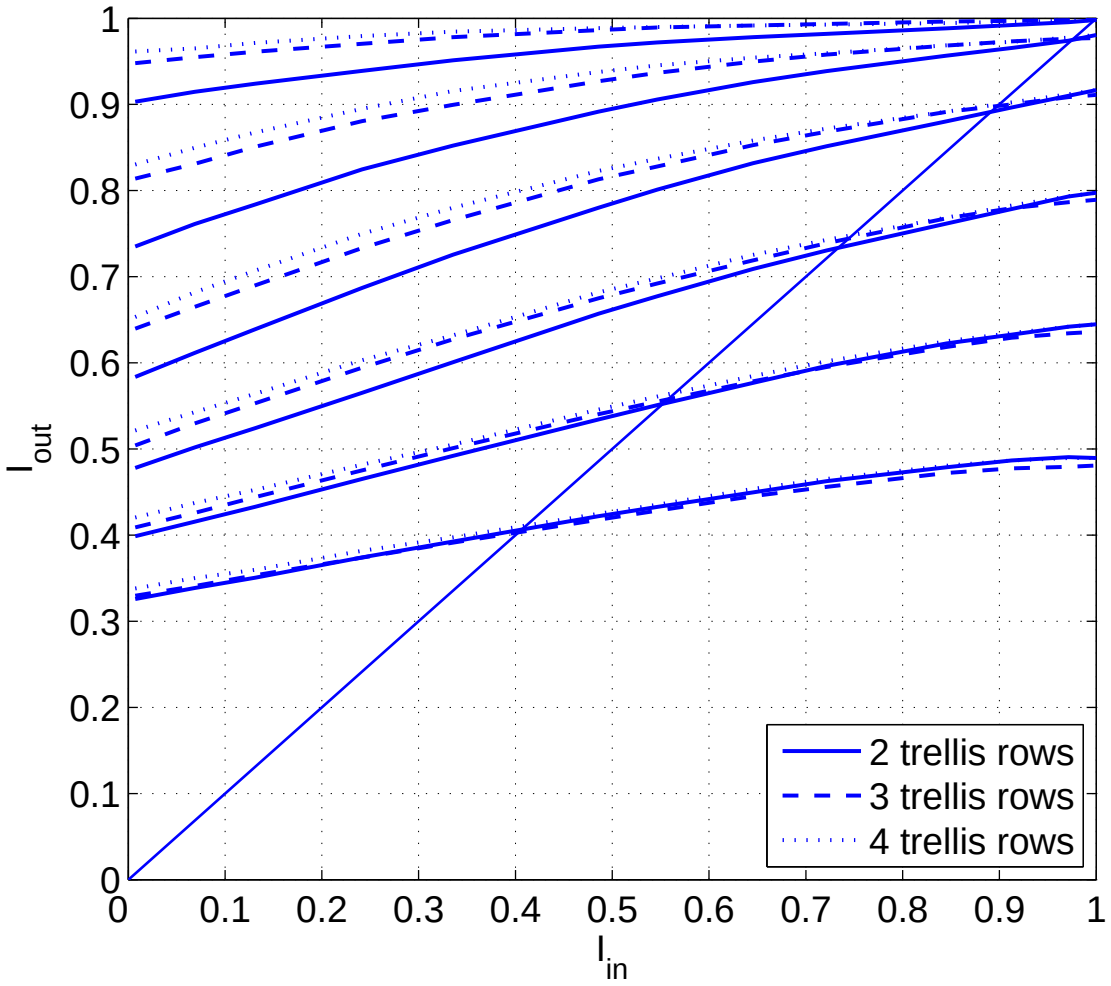

Figure 3.1: EXIT chart for 2×2 mask unit detector in IRCSDFA.

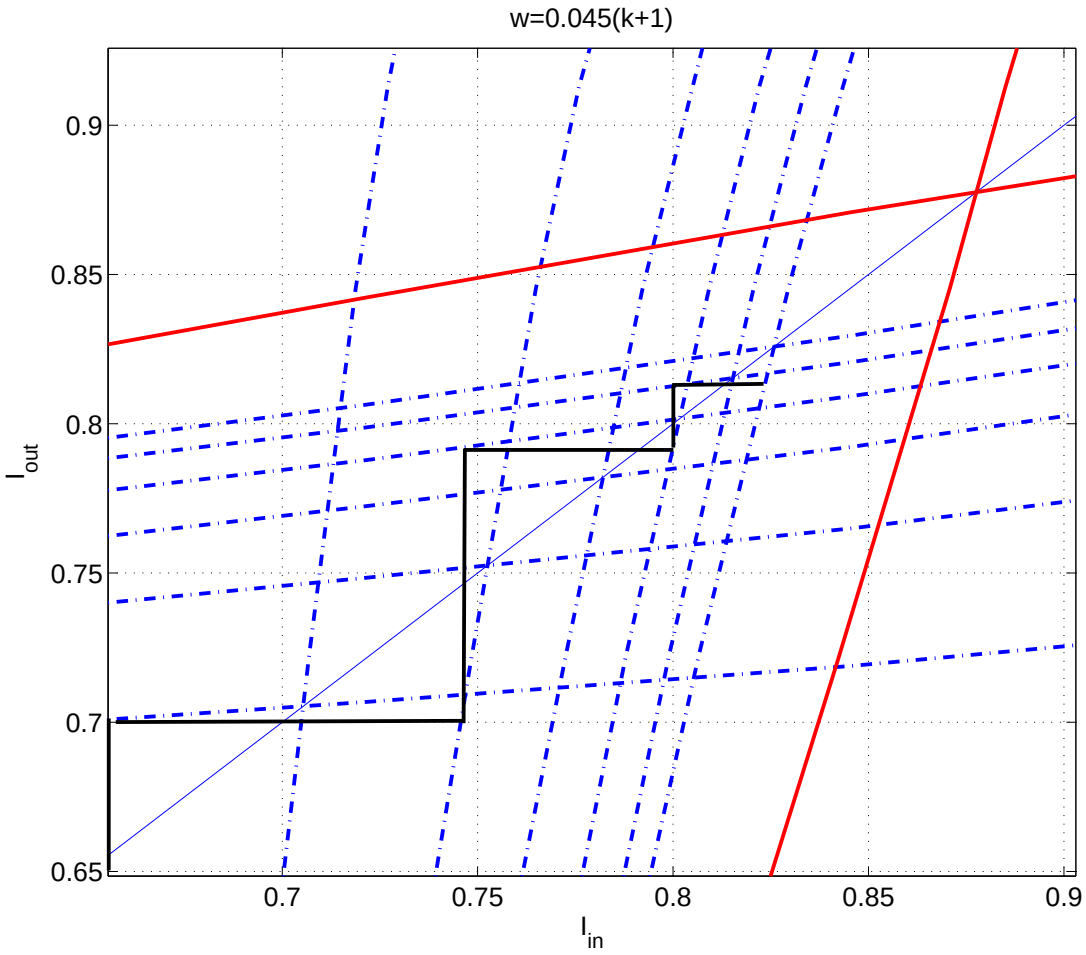

Figure 3.2: EXIT chart for weights schedule w_1.

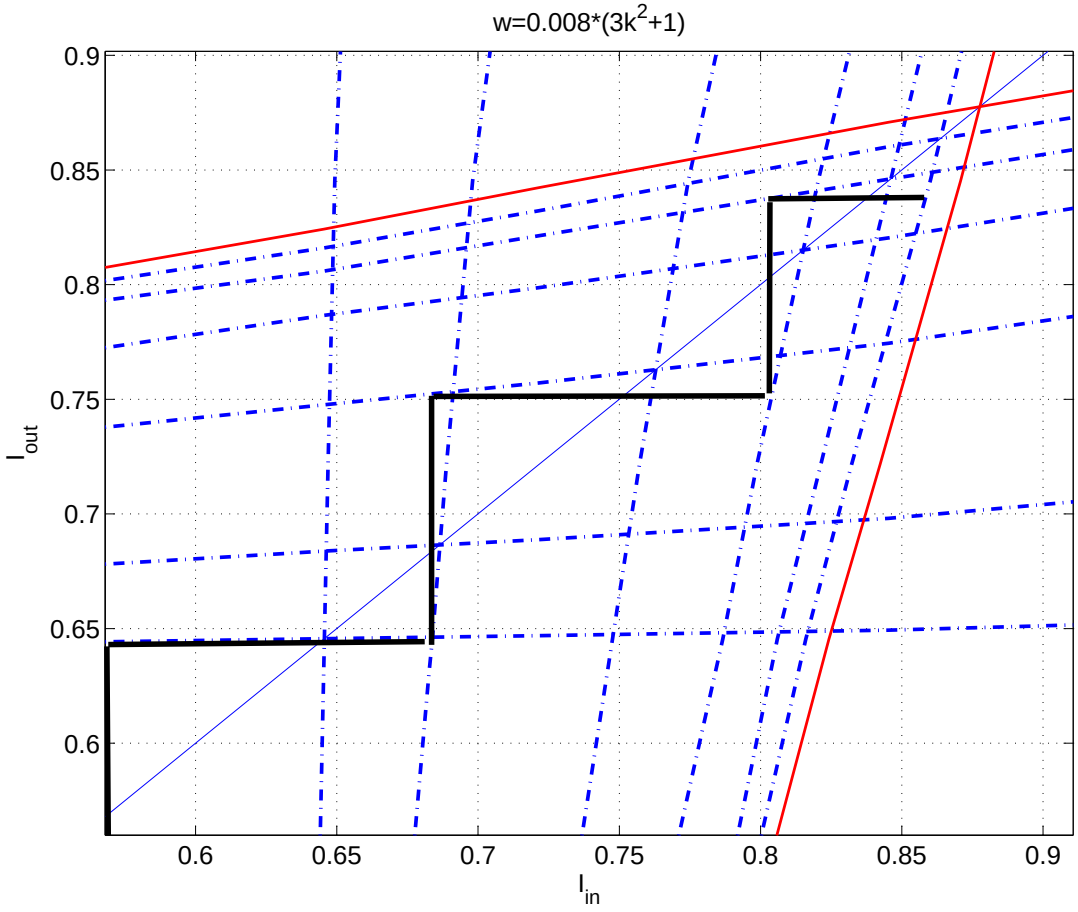

Figure 3.3: EXIT chart for weights schedule w_2.

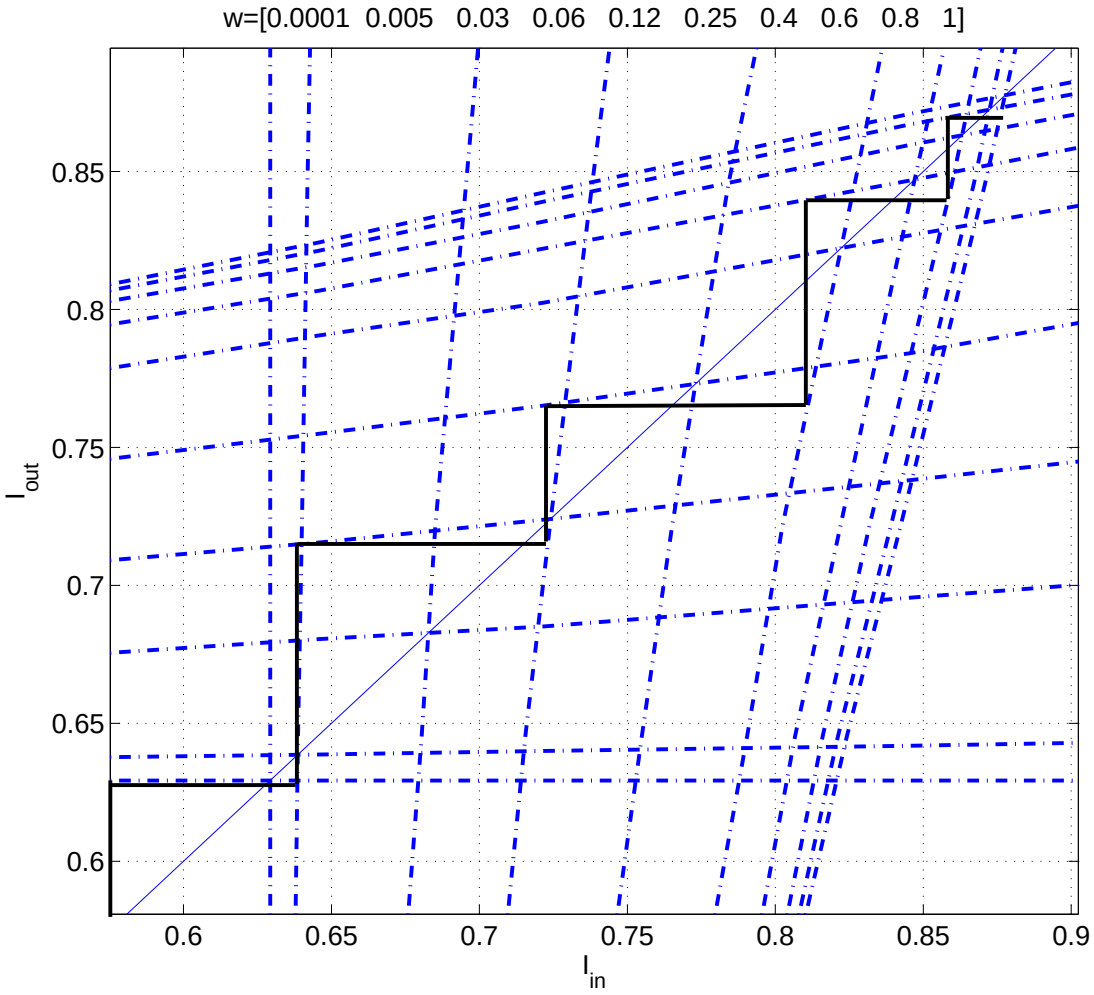

Figure 3.4: EXIT chart for weights schedule w_3.

Chapter 4

Serial Concatenated IRA Codes Design

4.1 Preliminaries of Irregular Repeat-Accumulate (IRA) Codes

IRA codes, introduced in [25] by Jin, Khandekar, and McEliece, are a generalization of the repeat-accumulate code in [29] that combines many of the favorable attributes of turbo codes and LDPC codes. They can be encoded in linear time, like turbo codes. They are amenable to an exact Richardson-Urbanke style analysis. The simulation results show the performance of IRA codes is slightly superior to turbo codes of comparable complexity, and just as good as the best known irregular LDPC codes.

IRA codes could be described by Tanner graph as shown in Fig. 4.1, where $f_i \geq 0,$ is the fraction of nodes with degree i, $\sum_i f_i = 1$; a is a positive integer. The Tanner graph is a bipartite graph with two kinds of nodes: variable nodes (open circles) and check nodes (filled circles). There are k variable nodes on the left, called 'information nodes'; there are r check nodes and r variable nodes on the right, called parity nodes. The number r will be

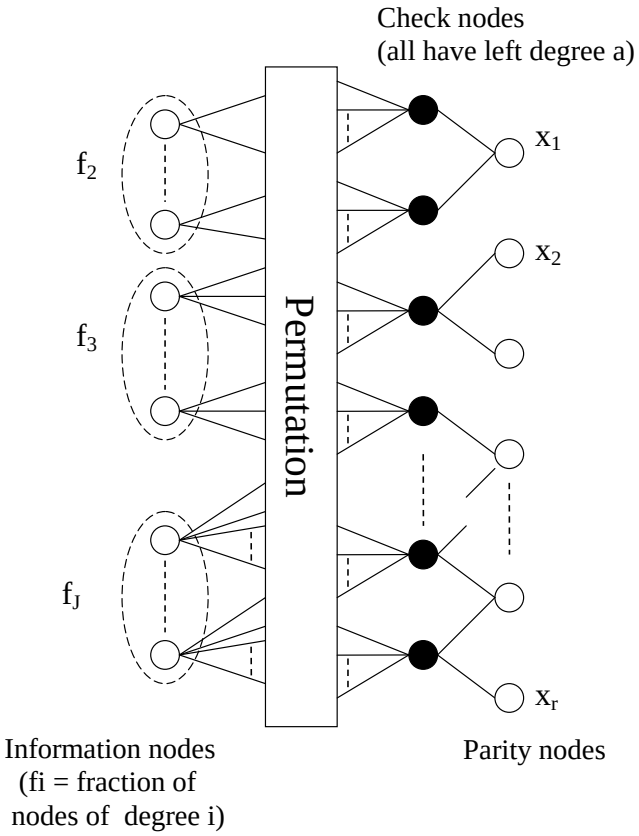

Figure 4.1: Tanner graph for IRA code with parameters ($f_2, f_3, \cdots, f_J; a$).

determined by the following equation:

$$r = \frac{k \sum_i i f_i}{a}.$$

Each information node is connected to a number of check nodes: the fraction of information nodes connected to exactly i check nodes is f_i. Each check node is connected to exactly a information nodes. These connections can made in arbitrary permutation. The check

nodes are connected to the parity nodes in the simple zigzag pattern shown in the figure. After deciding the permutation pattern between information nodes and check nodes, the codeword described by the Tanner graph is also determined: each of the information bits is associated with one of the information nodes, and each of the parity bits is associated with one of the parity nodes. On each check node, the mod-2 sum of all connected nodes should be zero; therefore, the value of a parity bit is determined uniquely. If we mark the ra edges coming out of the permutation box as $(v_1, v_2, \cdots, v_{ra})$, we can use the following recursive formula to determine the parity bits values:

$$x_j = x_{j-1} + \sum_{i=1}^{a} v_{(j-1)a+i}, \tag{4.1}$$

where $j = 1, 2, \cdots, r$ and $x_0 = 0$. Thus, the IRA codes can be encoded in linear time. For the systematic IRA codes, the information bits will be transmitted as a part of the codeword, and the code rate is

$$R_{sys} = \frac{a}{a + \sum_i i f_i}. \tag{4.2}$$

Like most LDPC codes, IRA codes can be decoded by using sum-product algorithm as shown in [38].

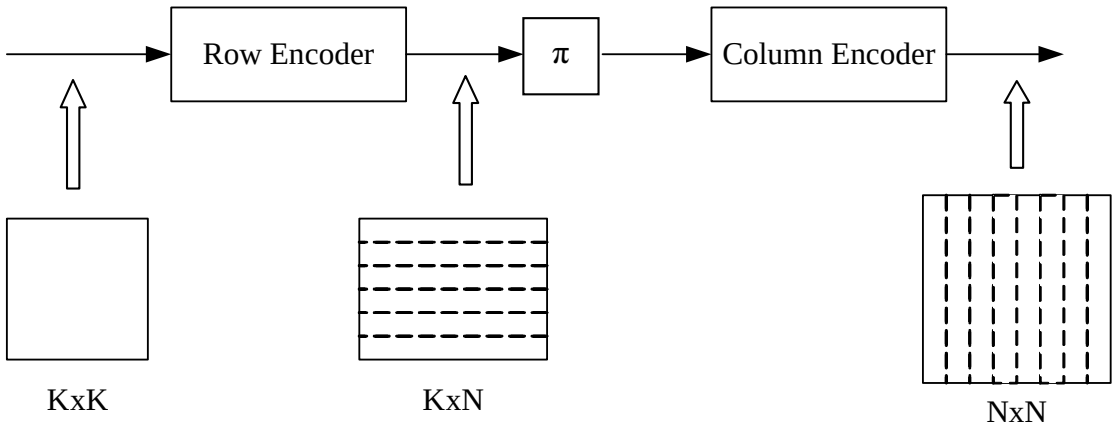

Figure 4.2: Block diagram of the concatenated encoder with systematic IRA component codes connected by interleaver (denoted by π).

4.2 Concatenated IRA Encoder and Decoder

A block diagram of the concatenated encoder is shown in Fig. 4.2. It consists of two systematic IRA component codes connected by an interleaver (denoted by π). In the following discussion, we visualize the concatenated system as a product code, with the two encoders operating on rows and columns. The source data is arranged in a two-dimensional block of size $K \times K$. The rows of the source block are first encoded with the outer $[N, K]$ systematic IRA code, yielding a $K \times N$ coded block in which the first K elements of each row are systematic bits. Then the $K \times N$ coded block is passed through the interleaver. The purpose of the interleaver is to minimize the intersection between the stopping set error

events of the row and column component codes. After the interleaver, each K-bit column is encoded with the inner $[N, K]$ systematic IRA code, producing an $N \times N$ codeword block. The overall code rate is $R = K^2/N^2$. The identical variable-node degree distributions of the two component codes are chosen to optimize their performances in the waterfall region according to the design algorithm given in [25], subject to the constraint that all weight-2 columns appear in the H_2 section of the parity check matrix; the constraint helps lower the error floors of the component codes. All example codes designed in this paper used a fixed check node degree of 10. The variable-to-check node connections in the component codes are optimized using the ACE algorithm of [35], in order to further lower the error floors. In our examples, the variable-to-check node connections in the component codes are different, so that the codes have different stopping sets; however, the interleaver design described in section 4.3 also works if the component codes are identical.

The decoder for the concatenated system employs iterative message passing between the decoders for the two component codes. The decoder block diagram is shown in Fig. 4.3. It consists of column and row decoders connected by the interleaver and de-interleaver. The received channel data is decoded column by column by a standard $[N, K]$ IRA decoder employing the sum-product algorithm (SPA, [38]) on the code's Tanner graph; the column decoder uses the extrinsic information from the row decoder as *a priori* information. The column decoder outputs a $K \times N$ block of extrinsic information LLRs. The column decoder's output extrinsic information is then passed through the interleaver and used as prior information by the row decoder. The row decoder makes use of the de-interleaved chan-

nel information and the prior information to decode the data row by row, and outputs a $K \times N$ block of LLRs to be used for final decoding decisions, along with a $K \times N$ block of extrinsic LLRs for the column decoder to use during the next iteration.

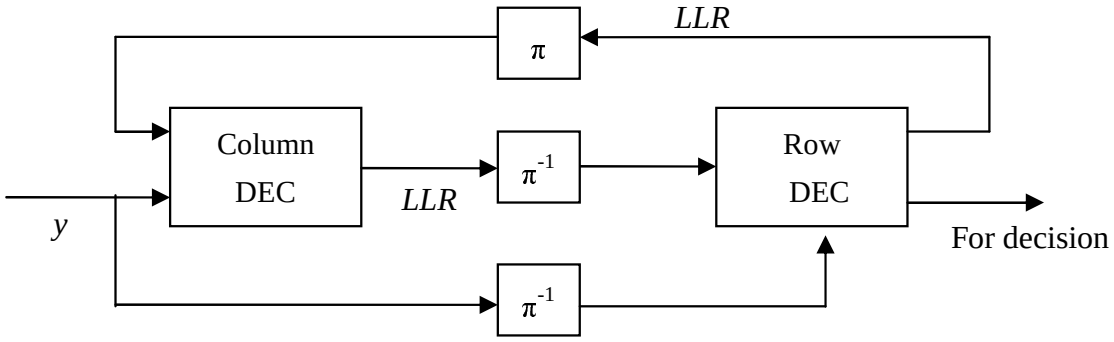

Figure 4.3: Block diagram of the concatenated decoder.

4.3 Interleaver Design

The reasons to encode/decode using the structure described above rather than using a single $[N^2, K^2]$ IRA code are as follows. The performance of an LDPC code at high SNR (i.e., in the error floor region) is not determined by the code's minimum distance, but rather by sets of interconnected short cycles (called stopping sets) that prevent the decoder from converging to a valid codeword. If we can design the interleaver to prevent the mapping of stopping set error events from one of the component codes into stopping set nodes of

the other code, then the concatenated structure will help improve the performance at high SNR.

The definition of a stopping set used in this paper is as follows. A variable-node set is called a stopping set if all its neighbors are connected to this set at least twice [35]. In LDPC codes at high SNR, error events occur on the smallest stopping sets with higher probability than on larger stopping sets or non-stopping sets. To simplify, if a variable node is a part of a stopping set, we call it a sensitive node.

Here is an example of how an error event from one IRA component code could propagate into the other one. Suppose variable nodes $(6, 9, 25)$ are sensitive nodes of the column component code and that errors occur on these positions. Since each column uses the same component code, errors will occur on these positions on most columns, i.e., at the end of column decoding, most positions of rows $(6, 9, 25)$ are errors. If we do nothing but directly input these rows to the row decoder, the outputs will have a large number of errors (perhaps even larger than the number of input errors) due of the bad prior information. If we pass the output extrinsic information from the column decoder through an interleaver before it is fed to the row decoder, the errors will not be concentrated on rows $(6, 9, 25)$ and hence can be corrected more easily. Therefore, we postulate two interleaver design rules for the concatenated system:

1. Spread concentrated errors all over the data block.

2. Avoid mapping the sensitive nodes of the row (column) component code into the

sensitive nodes of the column (row) component code.

The sensitive positions of a component code can be determined experimentally, or by employing the stopping set detection algorithm of [36]. For a short length block LDPC code, sensitive positions can be determined experimentally by Monte Carlo simulation, as in the following example. The row component code used for one of the examples in section 4.4 is $[181, 128]$ systematic IRA code. We let the row decoder receive its channel LLRs as usual, but we artificially set its input extrinsic LLRs to have a fixed amplitude and correct sign, except for a small number of error positions where the sign is incorrect. By changing the position of the errors, running a AWGN-channel Monte Carlo simulation at a fixed SNR for each error position, and then counting the number of decoder output error bits, we find which positions are sensitive: the more output errors, the more sensitive the corresponding positions. Fig. 4.4 shows the sensitivity of single positions chosen from all 181 variable nodes. Each artificially introduced single-position error undergoes a 50-block Monte-Carlo simulation, where each block is decoded with 10 SPA iterations. From the figure it is clear that six of the positions are sensitive.

The sensitive positions of a component code can also be determined by employing the stopping set detection algorithm of [36]. For a given starting variable node, the algorithm in [36] finds a stopping set containing that node, but does not guarantee that the detected set is minimal; thus, some relatively less-sensitive nodes may be included in the set. To find the most sensitive nodes, we repeatedly run the detection algorithm by starting from every variable node in the code, and count the accumulated times each node appears in a

Node Position	Monte Carlo Error Bits (max= 39)	Number of Times in Stopping Set (max= 110)
18	39	66
52	20	75
97	23	78
109	37	76
113	29	78
132	27	76

Table 4.1: Comparison of sensitivity detection methods.

stopping set; the higher the count, the more sensitive the node. Fig. 4.5 shows the results of running the algorithm of [36] over the $[181, 128]$ row component IRA code by starting from each variable node. In Fig. 4.5, the maximum sensitivity count is 181. It is clear from the figure that some nodes are highly sensitive, and that most of the parity bits (bits 129-181) have high sensitivity counts.

To compare these two methods for detecting sensitive nodes, we list the top 6 sensitive nodes detected by the experimental method and the corresponding sensitivity determined by the method of [36] in the table shown in Table. 4.1. In this table, the first column is the node positions, the second column is the accumulated output errors shown in Fig. 4.4, and the third column is the sensitivity counts shown in Fig. 4.5. It is evident that sensitive nodes detected by the experimental method also have high sensitivity counts in the algorithm of [36].

Based on the above design rules, we design the interleaver by starting with a random

interleaver and imposing additional constraints. First, a relatively good $K \times N$ random interleaver is found by simulation. Then the stopping sets of the row and column component codes are detected using the method of [36]. For given sensitive nodes i and j of the row/column component codes $i \in \{I_0, I_1, \cdots, I_n\}$ and $j \in \{J_0, J_1, \cdots, J_m\}$, where $\{I_0, I_1, \cdots, I_n\}$ and $\{J_0, J_1, \cdots, J_m\}$ are the sensitive nodes of the row and column component codes respectively, we modify the random interleaver so that no element in the jth row before passing through the interleaver is located in the ith column after passing through the interleaver. If the random interleaver maps any element in row j to column i (the "bad mapping" condition), then that element is re-mapped to a random position in the output block, and the element formerly at that random position is mapped into the position of the element in row j; this re-mapping continues until either no bad mappings are found or all the possible positions in the interleaver have been checked, in which case no interleaver solution is possible. Since the stopping set detection algorithm yields a large set, we select only the most sensitive nodes (i.e., the nodes with highest sensitivity counts in a histogram like that of Fig. 4.5) to design the interleaver at the beginning. Then we increase the number of selected sensitive nodes step by step until we cannot find a solution for the interleaver.

4.4 Simulation Results

The Monte Carlo simulation results for the proposed concatenated IRA code structure on the binary-input AWGN channel are shown in Fig. 4.6. In the figure, the right-most curve

(marked by '+' symbols) is the performance of a single IRA component code with source block length $K = 128$ bits and code rate 0.707. The second rightmost curve (marked 'x') is the proposed concatenated code with block size $K^2 = 16384$, rate 0.5, and a random interleaver; the random interleaver was found by (non-exhaustive) search over a large number of randomly generated interleavers. The solid line with circle markers is the same code structure as the second curve, but uses an interleaver based on the design rules proposed in section 4.3; this designed interleaver used the random interleaver of the 'x' curve as the design's starting point. The dashed line with star markers is the BER of a rate 1/2, block length $K^2 = 65536$ concatenated IRA code with optimized interleaver. For comparison, we also simulated single long block length IRA codes with rate 0.5; the solid line is with source block length 16384 and the dashed line is with source block length 65536.

All the single IRA code simulations were run until either a valid codeword was decoded, or 100 iterations were performed. For both the 16384-bit concatenated curves the decoder was run for a total of 10 outer iterations between the component codes, and the component codes were each iterated 10 times per outer iteration. Component decoding (on a given row or column) was terminated before 10 iterations if a valid codeword was decoded. The concatenated iteration schedule was determined experimentally, and therefore may not be optimal. (Further optimization of the iteration schedule using, e.g., EXIT charts [39], will be the focus of future work.) The complexity of the 16384-bit concatenated decoder is thus approximately twice that of the 16384-bit single IRA code, although at higher SNR the complexity of the concatenated system is relatively higher because termination events

for the concatenated code eliminate only single rows or columns from the iteration, not the entire codeword. The 65535-bit concatenated decoder was run for a total of 10 outer iterations with 20 inner iterations per outer iteration, so its decoding complexity is about four times that of the single 65535-bit IRA code.

From the figure it is clear that, although the concatenated 16384-bit IRA code has an SNR penalty in the waterfall region (about 2.1 dB SNR at BER 10^{-5}) compared to the single 16384-bit IRA code of equivalent rate, it has a much lower error floor. There is a crossover point between these two codes' BER curves at a BER of about 10^{-7}, and the BER of the concatenated IRA code decreases much faster than that of the single IRA code at high SNR. By comparing the 16384-bit concatenated codes' performance with different interleavers, we see that the proposed interleaver design can achieve significant gains (about 0.7 dB at 10^{-5} and 0.3 dB at 10^{-7}) over the random interleaver used as the design starting point, which means the idea of separating the component codes' stopping sets works.

The $K = 128$ example component codes are quite short. We conjecture that when the block length is increased the penalty in the waterfall region will decrease, since the component IRA codes will asymptotically approach capacity as the block length increases. This conjecture is partly supported by the smaller SNR penalty (about 1.7 dB at BER 10^{-5}) of the 65536-bit rate-1/2 concatenated code compared to the equivalent-rate 65536-bit IRA code, although part of the improvement over the 16384-bit codes may be due to the increased decoder iterations allocated to the 65536-bit concatenated system.

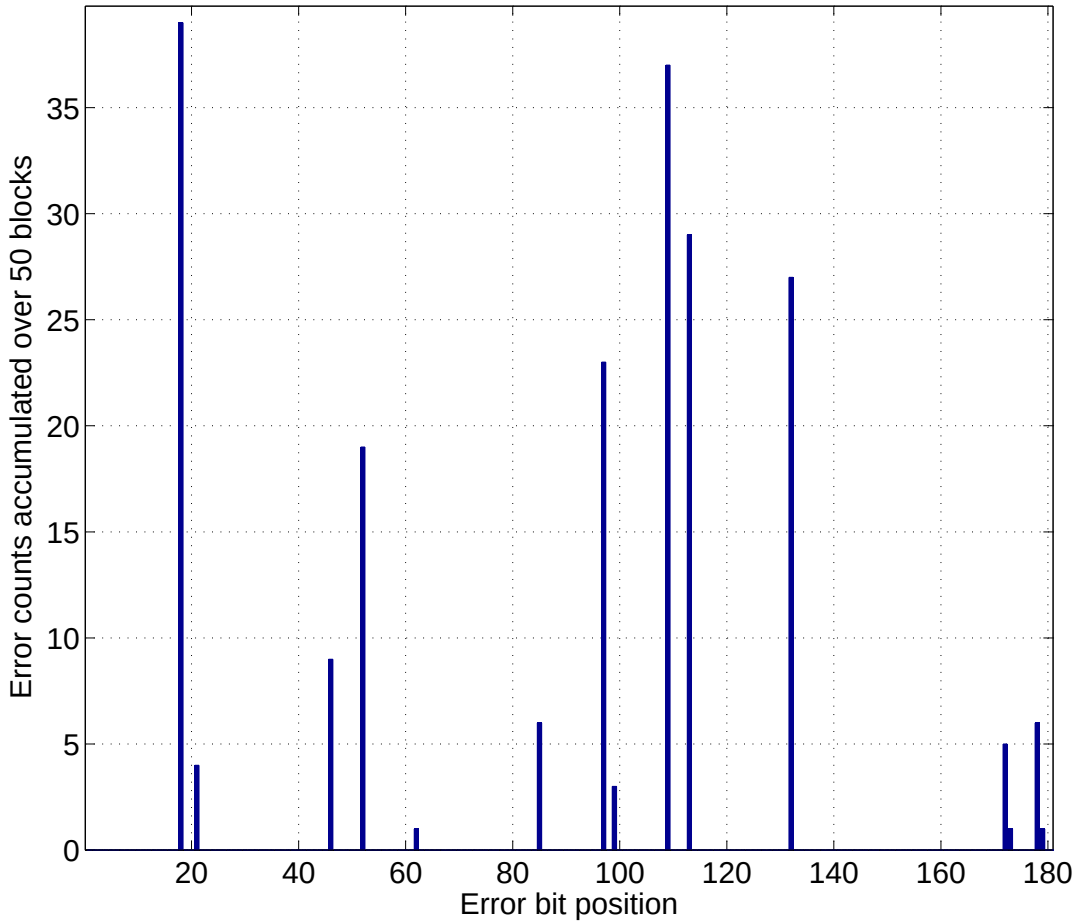

Figure 4.4: Sensitive positions of a $[181, 128]$ systematic IRA code, determined by Monte-Carlo simulation of 50 blocks with artificially introduced single-bit extrinsic LLR sign errors at each possible bit position.

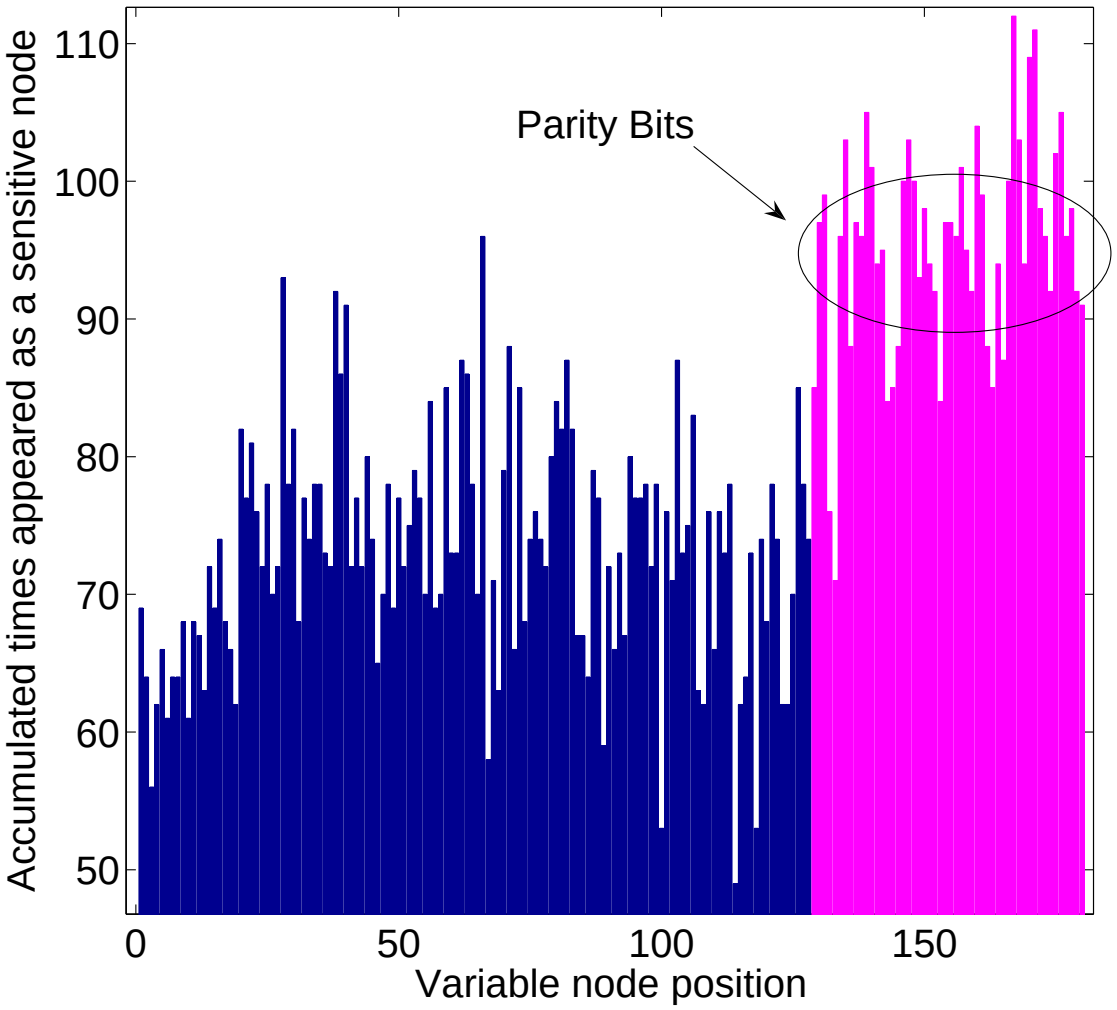

Figure 4.5: Sensitivity measurement via stopping set detection. The sensitivity counts on the vertical axis are accumulated by running the algorithm of [36] on every possible starting variable node, and then counting the number of times any given node appears in the detected stopping sets.

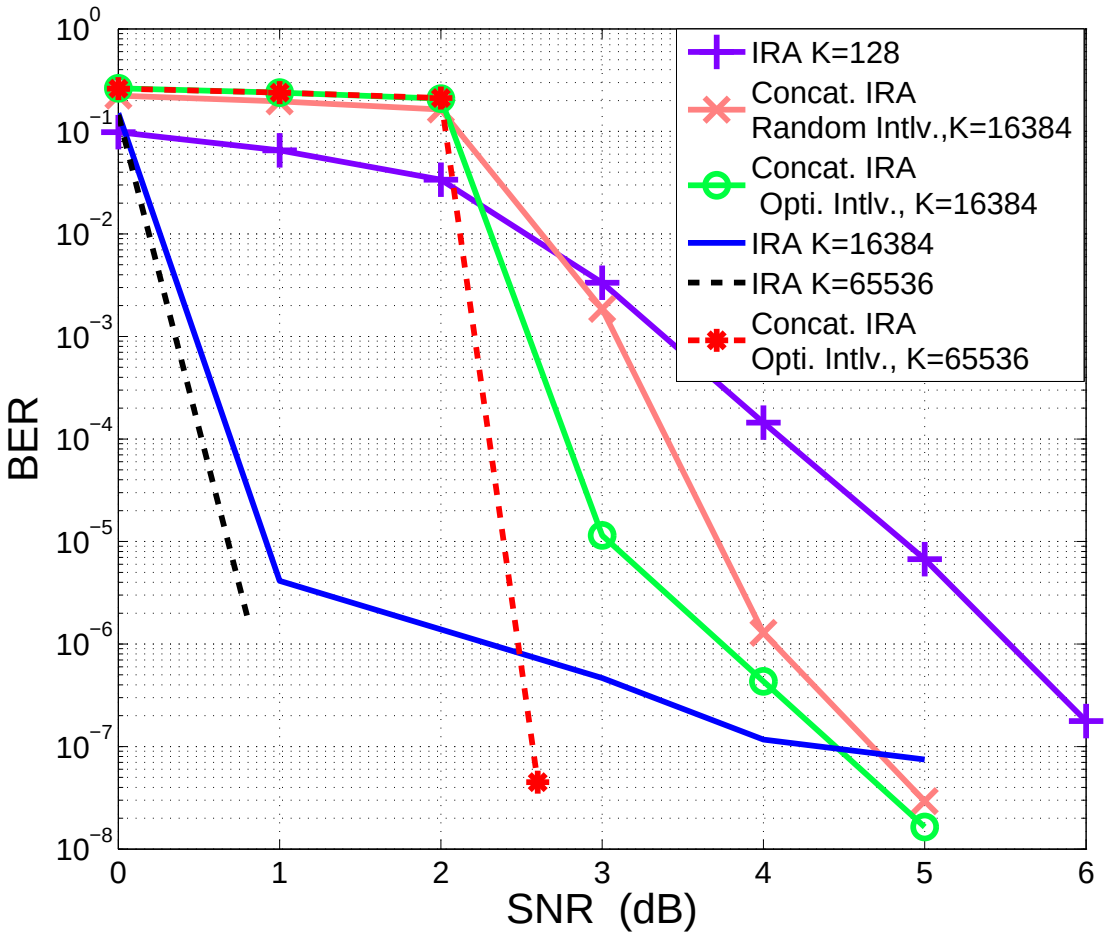

Figure 4.6: Simulation results. All codes are rate 1/2, except for the $K = 128$ IRA code, which is rate 0.707.

Chapter 5

Joint Iterative Detection and Decoding for 2D ISI channels

5.1 System Structure

To achieve improved SNR vs. BER performance on the 2D ISI channel, we propose to investigate coding the source 2D data with a systematic error correction code (ECC) before it passes through the 2D ISI plus AWGN channel, and estimate it with an iterative joint detection/decoding scheme. The whole system structure is shown in Figure 5.1.

For large source data block-sizes, it is well known that LDPC codes outperform turbo convolutional codes. Hence, the encoder uses IRA codes [25], a type of LDPC code which offers low-complexity encoding and decoding, and performs well over a wide range of code rates.

Pioneering work on systems similar to Fig. 5.1 has been reported in [8, 9], for the special case of separable masks. These papers consider three different detection-decoding schemes: (i) message-passing on the combined ISI-channel/LDPC factor-graph; (ii) Wiener filtering, and (iii) iterative MAP detection by message passing between equalizer and LDPC

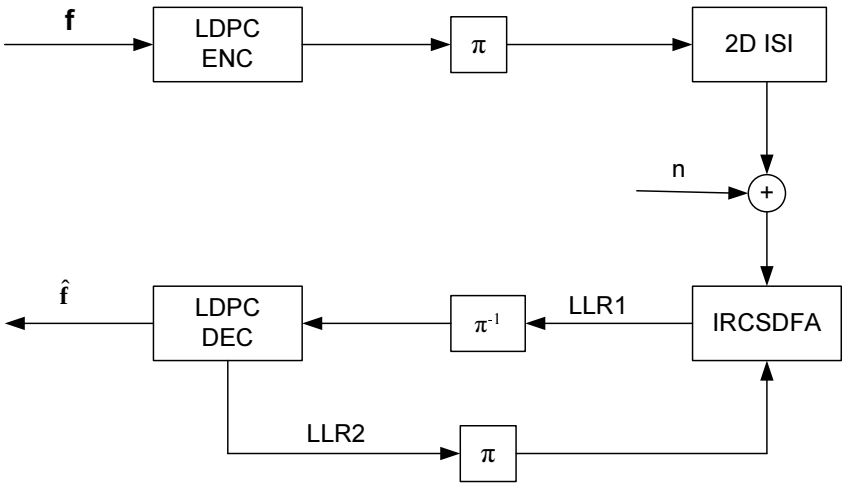

Figure 5.1: System structure of joint iterative detection/decoding for 2D ISI.

SISOs as in Fig. 5.1. Method (i) does not perform as well as either (ii) and (iii), and has relatively high computational complexity, while Weiner filtering has previously been shown to be inferior to row-by-row DFVA [12]. System (iii) performs within 1 dB of the coded system without ISI for a relatively simple 2×2 mask, which is very impressive. The scheme we proposed in this chapter performs better and is not limited to separable masks.

The joint decoding/detection structure shown in Fig. 5.1 actually just gives a general idea for this kind of application. We have shown in chapter 2 that the IRCSDFA consists of 2 SISO-SDF detectors, one each for rows and columns, and that each detector can accept soft *a priori* information from outside itself to improve the performance. As the IRCSDF algorithm is based on decision feedback, the accuracy of the feedback information is critical to the whole system; any errors in the feedback can propagate to the next

set of rows/columns to be processed, and thereby degrade the performance. Therefore,

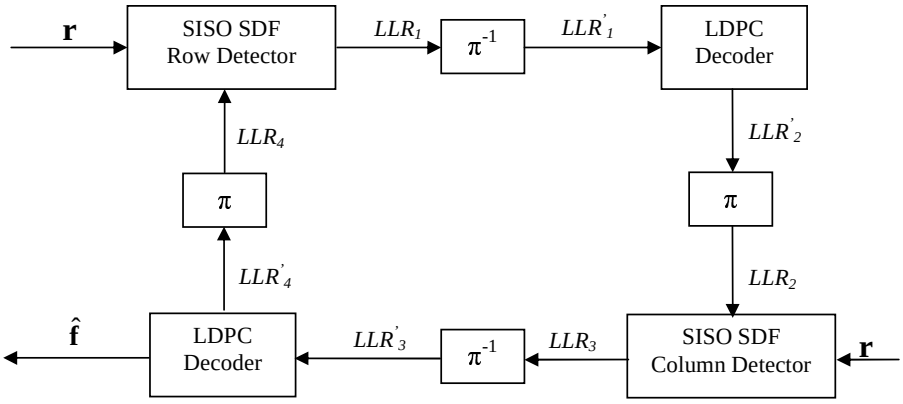

Figure 5.2: Join iterative detector/decoder structure.

we construct the joint iterative detector/decoder as in Figure 5.2. Here there are still two SISO-SDF equalizers (row/column), both of which directly process the received information *r* from channel. Each equalizer passes its extrinsic information (represented by LLR) to the de-interleaver (π^{-1}) and then to the input to the LDPC decoders. There are two identical LDPC decoders; both of them take soft-input extrinsic information from one of the equalizers, and then feed their soft outputs back to the other equalizer as *a priori* information. After several iterations, the second LDPC decoder outputs the final detection results.

5.2 Simulation Results for the Joint Detection/Decoding System

The Monte Carlo simulation results for the iterative joint detection/decoding algorithm on the random binary image $f(m,n)$ with pixel alphabet $\{-1,+1\}$ are presented in this section. The plots in this section show the bit error rate (BER) of the estimated binary input image, versus signal noise ratio (SNR). The SNR is defined as in [9]:

$$\text{SNR} = 10\log_{10}\frac{\text{var}\,[f*h]}{2R\sigma_w^2}, \tag{5.1}$$

where $*$ denotes 2D convolution, σ_w^2 is the variance of the Gaussian r.v.s $w(m,n)$ in (2.1) and R is the rate of the LDPC code. Similar to the simulation of IRCSDFA, we assume a boundary of -1 pixels around the original image $f(m,n)$ which is known by the receiver.

In order to ensure a fair comparison with the system in [9], we use the exact same code as [9]: the rate 0.5, original block length 10000 regular (3, 6) LDPC code proposed by MacKay in [28]. The paper [9] only presented results for the separable mask $[10.5; 0.50.25]$. In chapter 2, we have mentioned that the IRCSDFA achieves the same equalizer performance as the separable algorithm of [9] on this relatively easy mask. After adding the LDPC code, our joint system also has the same performance as [9] does, for the mask $[1 \quad 0.5; \quad 0.5 \quad 0.25]$. Since the IRCSDFA performs better than the separable equalization algorithm on the 2×2 averaging mask channel, we implemented the joint separable algorithm and LDPC decoder ourselves and then compared its performance with combined IRCSDFA and LDPC decoding, on the 2×2 averaging mask channel. Our implementation

of the combined separable equalizer and LDPC decoder matches all the presented results of [9] under same simulation conditions, so we believe that our implementation is equivalent to that in [9].

Figure 5.3 shows the simulation results of IRCSDFA+LDPC and the separable algorithm+LDPC on the 2×2 averaging mask channel. The joint IRCSDFA+LDPC algorithm uses the structure shown in Fig. 5.1; the total iteration times between the SISO-SDF equalizers and the LDPC decoders (we call this outer iteration) are 80; for each outer iteration, there is only one time iteration inside the row/column detector and 10 times iteration inside the LDPC decoder for the sum-product algorithm; a scaling factor, $w = 0.5$ is put on the extrinsic information from SISO-SDF detectors to LDPC decoders where no weight is put on the extrinsic information from LDPC decoders to SISO-SDF detectors. For the separable algorithm+LDPC code, we use the same iteration schedule as in [9]: 10 times outer iteration, for each outer iteration there are 2 sub-iteration between the column and row detectors and 50 iterations inside the LDPC decoder; since the output extrinsic information of LDPC is 2-ary it will be fed to an additional row detector (just run one time for each outer iteration) to generate the 4-ary extrinsic information to send to the column detector. The performance of the LDPC code itself is also included, to provide a baseline without 2D ISI. It is clear that the proposed approach is better than [9]'s over the entire SNR range. In particular, at BER 1×10^{-5}, the gain is about 0.1 dB. Comparing to the 0.3 dB gain achieved by the equalizer only (see Table 2.1), we lose part of equalizer gain after the equalizer is combined with the LDPC code. The most important difference between

these two approaches is that the IRCSDFA+LDPC works well for any 2×2 ISI channel, whereas the separable algorithm will introduce an error floor for non-separable channels as a result of using the best separable approximation to the non-separable mask.

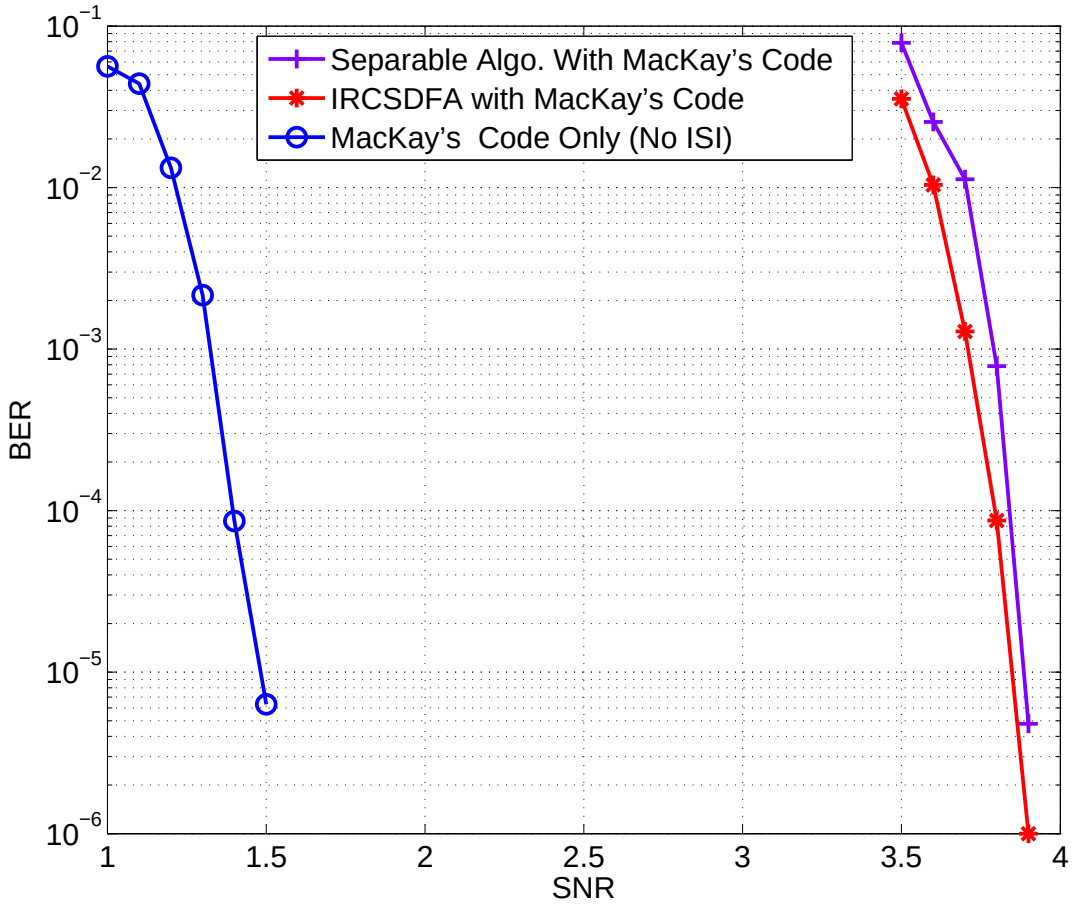

Figure 5.3: Simulation results of joint detection and decoding on the 2×2 averaging mask channel.

Chapter 6

Conclusions

This dissertation has demonstrated a new iterative soft-decision feedback algorithm for reduction or elimination of 2D ISI. The algorithm achieves substantial reductions in required SNR for a given BER, when compared to an earlier soft-input/soft-output iterative algorithm without decision feedback and also when compared to an earlier separable algorithm. A key contribution of our work is that soft-decision feedback from previously estimated rows/columns can substantially improve the performance of iterative 2D-ISI algorithms. In particular, the fact that our algorithm essentially matches the performance of the algorithm in [16], which was designed to avoid decision feedback, demonstrates that decision feedback, if appropriately employed, need not degrade the performance of 2D-ISI detection algorithms.

This dissertation also demonstrates a method, using EXIT charts, to optimize the iteration schedule between its sub-systems and the weights placed on the passed extrinsic information. This optimization approach can give some theoretical guidance for the sys-

tem iteration schedule design, and also can save simulation time compared to optimization done by simulation of the entire iterative equalizer.

A serially concatenated IRA coding structure has been proposed in this dissertation. This dissertation demonstrated that serial concatenation of two IRA codes connected by an appropriately designed interleaver can greatly lower the level and slope of the BER curve in the high SNR region, compared to a single IRA code of equivalent length and rate. We believe that the proposed approach will also work with more general LDPCs as component codes, including, e.g., e-IRA codes or codes optimized with the error-floor lowering algorithm of [36].

An iterative joint detection/decoding system for 2D ISI channel has also been proposed in this dissertation. The simulation results show that the proposed joint detector/decoder performs better than the best previously published joint equalization/decoding system.

Although this dissertation has achieved good 2D equalization performance compared with previous methods, there are still some areas remaining open and need to be investigated in future: for the IRCSDFA, we want to answer the question why the weight schedule works, and find a theoretical justification for it by deriving a relaxation algorithm from first principles; for the serially concatenated IRA system, we need to improve the interleaver design, and further investigate its performance with longer block lengths and other component codes.

Bibliography

[1] W. Coene, “Coding and signal processing for two-dimensional optical storage (TwoDos).” *Powerpoint presentation available at http://cm.bell-labs.com/cm/ms/events/WGIR04/pres/coene.ppt*, Mar.2004.

[2] G. T. Huang, “Holographic memory,” *MIT Technology Review*, vol. 9, Sept. 2005.

[3] J. F. Heanue, K. Gürkan, and L. Hesselink, “Signal detection for page-access optimal memories with intersymbol interference,” *Applied Optics*, vol. 35, pp. 2431–2438, May 1996.

[4] R. Krishnamoorthi, “Two-dimensional Viterbi like algorithms,” Master’s thesis, Univ. Illinois at Urbana Champaign, 1998.

[5] O. Shental, A. Weiss, N. Shental, and Y. Weiss, “Generalized belief propagation receiver for near-optimal detection of two-dimensional chhannel with memory,” *IEEE Information Theory Workshop*, pp.225 - 229, 24 -29 Oct. 2004.

[6] J. S. Yedidia, W. T. Freeman, and Y. Weiss, “Constructing free-energy approximation and generalized belief propagation algorithm,” *IEEE Trans. on Inform. Theory*, vol. 51, pp. 2282 - 2312, July 2005.

[7] G. D. Forney, “The Viterbi algorithm,” *Proceedings of the IEEE*, vol. 61, pp. 268–278, 1973.

[8] N. Singla, J. A. O’Sullivan, R. S. Indect, and Y. Wu, “ Iterative decoding and equalization for 2-D recording channels,” *IEEE Trans. magnetics*, Vol. 38, pp. 2328-2330, Sept. 2002.

[9] Y. Wu, J. A. O’Sullivan, N. Singla, and R. R. Indeck, “Iterative Detection and Decoding for Separable Two-Dimensional Intersymbol Interference,” *IEEE Trans. Magnetics*, Vol. 39, No.4, pp2115-2120, July 2003

[10] K. M. Chugg, “Performance of optimal digital page detection in a two-dimensional ISI/AWGN channel,” *Proc. Asilomar Conf. on Signals, Systems and Comp.*, pp. 958–962, Nov. 1996.

[11] C. L. Miller, B. R. Hunt, M. W. Marcellin, and M. A. Neifeld, “Image restoration using the Viterbi algorithm,” *Journal of the Optical Society of America A*, Vol. 16, No. 2, pp. 265-274, Feb. 2000.

[12] C. L. Miller, B. R. Hunt, M. Neifeld, and M. W. Marcellin, "Binary image reconstruction via 2-D Viterbi search," *IEEE International Conference on Image Processing, (ICIP97)*, vol. 1, pp.181-184, 1997.

[13] M. A. Neifeld, R. Xuan, and M. W. Marcellin, "Communication theoretic image restoration for binary-valued imagery," *Applied Optics*, Vol. 39, No. 2, pp. 269-276, Jan. 2000

[14] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: turbo-codes," *IEEE Trans. Commun.*, vol. 44, pp. 1261 – 1271, Oct. 1996.

[15] X. Chen and K. M. Chugg, "Near-optimal data detection for two-dimensional ISI/AWGN channels using concatenated modeling and iterative algorithms," *Proc. ICC'98*, pp. 952–956, 1998.

[16] M. Marrow and J. K. Wolf, "Iterative detection of 2-dimensional ISI channels," *Proc. Info. Theory Workshop*, (Paris, France), pp. 131–134, Mar./Apr. 2003.

[17] M. Marrow, "Equalization and detection of 2-d ISI channels," available at http://cmrr-wolf08.ucsd.edu/~mmarrow/, May 2003.

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

[19] F. R. Kschischang, B. J. Frey, and H. Loeliger, “Factor graphs and the sum-product algorithm,” *IEEE Trans. Inform. Theory*, vol. 47, NO.2, Feb. 2001.

[20] W. H. Gerstacker, R. R. Müller, and J. B. Huber, “Iterative equalization with adaptive soft feedback,” *IEEE Trans. Commun.*, vol. 48, pp. 1462–1466, Sept. 2000.

[21] Stephan ten Brink, “Convergence behavior of iteratively decoded parallel concatenated codes,” *IEEE Trans. Communications*, vol. 49, NO.10, Oct. 2001.

[22] H. E. Gamal, A. R. Hammons, “ Analyzing the turbo decoder using the Gaussian Approximation,” *IEEE Trans. Inform. Theory*, vol. 47, NO.2, pp. 671–686, Feb. 2001.

[23] R. Koetter, A. C. Singer, and M. Tuechler, “Turbo equalization,” *IEEE Signal Processing Magazine*, vol. 21, NO.1, pp. 67–80, Jan., 2004.

[24] N. Wiberg, “Codes and the decoding on general graphs,” Linkoping Studies in Science and Technology, Linkoping, Sweden, Ph.D. Dissertation No. 440, 1996.

[25] H, Jin, A. Khandekar, and R. McEliece, “Irregula repeat-accumulate codes,” *Proc. 2nd. Int. Symp. Turbo Codes and Related Topics*, Brest, France, Sept. 2000.

[26] M. Yang, W. E. Ryan, and Y. Li, “Design of efficiently encodable moderate-length high-rate irregular LDPC codes,” *IEEE Trans. Communications*, Vol. 52, NO. 4, April 2004.

[27] R. G. Gallager, *Low-Density Parity-Check Codes.* Cambridge, MA: MIT Press, 1963.

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

[29] D. Divsalar, H. Jin, and R. J. McEliece, “Coding theorems for ‘turbo-like’ codes”, *Proc. 36th Allerton Conf. on Communication, Control, and Computing,* pp.201-210, Allerton, Illinois, Sept. 1998.

[30] T. J. Richardson and R. L. Urbanke, “The Capacity of low-density parity-check codes under message-passing decoding,” *IEEE Trans. Inform. Theory*, vol. 47, pp. 599-618, Feb. 2001.

[31] T. Richardson, A. Shokrollahi, and R. Urbanke, “Design of capacity-approaching irregular low-density parity-check codes,” *IEEE Trans. Inform. Theory*, vol. 47, pp. 619-637, Feb. 2001.

[32] M. Yang, W. E. Ryan, and Y. Li, “Design of efficiently encodable moderate-length high-rate irregular LDPC codes,” *IEEE Trans. Commun.*, vol. 52, pp. 564-571, Apr. 2004.

[33] C. Berrou and A. Glavieux, “Near optimum error correcting coding and decoding: turbo-codes,” *IEEE Trans. Commun.*, vol. 44, pp. 1261-1271, Oct. 1996.

[34] C. Di, D. Proietti, E. Telatar, T. Richardson, and R. Urbanke, “Finite length analysis of low-density parity-check codes on the binary erasure channel,” *IEEE Trans. Inform. Theory*, vol. 48, pp. 1570-1579, June 2002.

[35] T. Tian, C. R. Jones, J. D. Villasenor, and R. D. Wesel, "Selective avoidance of cycles in irregular LDPC code construction," *IEEE Trans. Commun.*, vol. 52, pp. 1242-1247, Aug. 2004.

[36] S. H. Lee, K. S. Kim, J. K. Kwon, Y. H. Kim, and J. Y. Ahn, "Design of an LDPC code with low error floor," in *Proc. IEEE Int. Symp. Info. Theory (ISIT 2005),* Adelaide, Australia, Sept. 2005, pp. 990-994.

[37] R. M. Pyndiah, "Near-optimum decoding of product codes: block turbo codes," *IEEE Trans. Commun.*, vol. 46, pp. 1003-1010, Aug. 1998.

[38] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inform. Theory*, vol. 47, pp. 498-519, Feb. 2001.

[39] S. ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Trans. Commun.*, vol. 49, pp. 1727-1737, Oct. 2001.

[40] P. M. Njeim, T.Cheng, B. J. Belzer, K. Sivakumar, and Y. Zhu, "Image detection in 2D intersymbol interference with zig-zag decision-feedback Viterbi algorithm," *Proc. 38th Conference on Information Sciences and Systems (CISS'04),* (CD-ROM), pp. 195-200, Princeton University, March 2004.

[41] T. Cheng, B. J. Belzer, and K. Sivakumar, "Image deblurring with iterative row-column soft-decision feedback algorithm," *Proc. 39th Conf. on Info. Sci. and Systems*

(CISS'05), (CD-ROM), paper 210, Johns-Hopkins University, Baltimore, Maryland, Mar. 2005

[42] P. M. Njeim, T.Cheng, B. J. Belzer, and K. Sivakumar, "Image detection in 2D intersymbol interference with iterative soft-decision feedback zig-zag algorithm," *Proc. 43rd Annual Allerton Conf. on Comm.,Computing, and Control*, U. of Illinois Urbana-Champaign, September 2005.

[43] Y. Zhu, T. Cheng, K. Sivakumar, and B. J. Belzer, "Detection of Markov Random Fields on Two-Dimensional Intersymbol Interference Channels," *Proc. 44th Annual Allerton Conf. on Comm., Computing, and Control*, U. of Illinois Urbana-Champaign, Illinois, Sept. 2006.

[44] T. Cheng, B. J. Belzer, and K. Sivakumar, "An Iterative Row-Column Soft-Decision Feedback Algorithm for Two-Dimensional Intersymbol Interference," *IEEE Signal Processing Letters*, Vol. 14, NO. 7, pp 433-436, July 2007.

[45] T. Cheng, K. Sivakumar,and B. J. Belzer, "Serially Concatenated IRA Codes," *The 45th Annual Allerton Conf. on Comm., Computing, and Control*, U. of Illinois Urbana-Champaign, Illinois, Sept. 2007.

[46] Y. Zhu, T. Cheng, K. Sivakumar, and B. J. Belzer, "Markov random field detection on two-dimensional intersymbol interference channels," accepted by *IEEE Trans. on Sig. Proc*, Nov. 2007.