DISTRIBUTED VIDEO CODING USING NON-BINARY LDPC CODES

A Thesis by

Suma Chilumuru

B.Tech. in Electronics & Communications Engineering

Christu Jyothi Institute of Technology and Science, India, 2004

Submitted to the Department of Electrical and Computer Engineering
and the faculty of the Graduate School of
Wichita State University
in partial fulfillment of
the requirements for the degree of
Master of Science

December 2006

DISTRIBUTED VIDEO CODING USING NON-BINARY LDPC CODES

I have examined the final copy of this thesis for form and content, and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Kamesh Namuduri, Committee Chair

We have read this thesis and
recommend its acceptance:

Hyuck M. Kwon, Committee Member

Brian Driessen, Committee Member

DEDICATION

To God, My Parents, and My Sister

ACKNOWLEDGEMENTS

Primarily, I would like to thank my advisor Dr. Kamesh Namuduri for all his help and able guidance throughout the course of my Master's thesis. He has been a constant support at every point and always lifted my spirits in times of despair. I express my heartfelt gratitude to him. Thank you very much, Dr. Kamesh. I also thank my committee members, Dr. Hyuck M. Kwon and Dr. Brian Driessen, for sparing time to review my thesis work.

Next, I would like to thank my mom, dad, and adorable loving sister for their confidence and belief in me. It's purely their trust and love that acted as the driving force to help me accomplish this task. Sincere thanks to my brother, Pavan, whose philosophies in life have enhanced my optimistic outlook toward everything.

Lastly, I would like to thank my friends, Subhadra, Sudhama, Radhika, and Anil, for their encouragement and support throughout my stay in Wichita. I especially thank my best friend, Vineet for his academic and moral support that helped me to surpass the difficulties in this thesis work and for boosting my morale. Also I would like to thank my NIAR friends and colleagues, Michelle, Saketh, Vincent, and Manish, with whom working had been a pleasure; they always cheered me and relieved my tension. Truly, I found friends for a lifetime in Wichita. I always thank God for all of this.

ABSTRACT

Distributed coding is a new paradigm for video transmission, based on the Wyner-Ziv theorem. In this thesis, a new Wyner-Ziv codec was proposed using non-binary Low-Density Parity-Check (LDPC) codes. Non-binary LDPC codes, developed for use in channel coding, have been extended for source coding to compress correlated non-binary sources, such as video. The approach is based on considering the correlation as a virtual q-ary symmetric channel and applying the syndrome concept. The system considered focused on the compression of a equiprobable memoryless non-binary source with side information at the decoder. Results obtained through simulations demonstrated that for rates 1/2 and 3/4, the non-binary compression scheme performed better than the equivalent binary compression scheme. The non-binary scheme, when extended for distributed video coding, produced the original frame with negligible error.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

Introduction and Preview

1.1 Significance of the Problem

The information-theoretic results produced by Slepian and Wolf [1] and Wyner and Ziv [2] have created a new paradigm for video coding, known as distributed video coding. Conventional video coding schemes have exploited the statistics of the source at the encoder, under the assumption that the video is encoded once and decoded several times. The development of sensor networks, mobile video-camera phones etc., have required low complex encoders, shifting the complexity to the decoder. The Wyner-Ziv video codec, which is used for distributed video coding, performs intra-frame encoding at the transmitter and inter-frame decoding at the receiver.

Low-density parity-check (LDPC) codes, introduced by Gallager in 1962, can be used to compress video [3]. These are a class of linear error-correcting block codes, which were largely forgotten until 1995. The non-binary version of these codes was developed by M. C. Davey [4]. The use of LDPC codes for distributed source coding was suggested by Liveris [5], in which the low-density parity-check codes were used to compress close to the Slepian-Wolf limit for correlated binary sources.

Video frames, being highly correlated, lend themselves well for transmission using

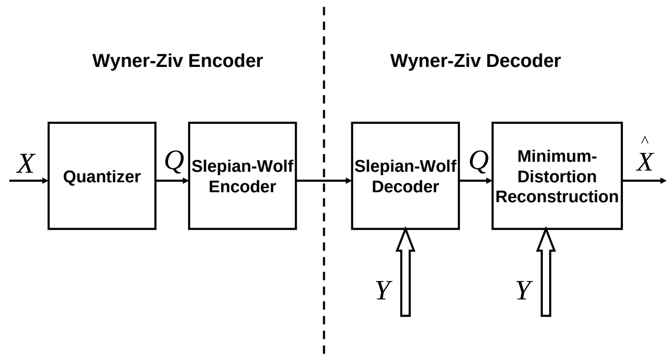

Figure 1.1: Practical Wyner-Ziv coder obtained by cascading a quantizer and a Slepian-Wolf encoder.

distributed source coding. The distributed source coding, suggested by Liveris using LDPC codes is in the binary domain which necessitated the conversion of video frames into their binary equivalent. This conversion caused the correlation between the frames to reduce significantly, thereby rendering the above mentioned scheme unsuitable for video frame transmission.

Hence, this led to the development of modifying the non-binary LDPC scheme for distributed source coding to allow the transmission of video frames without converting them into their equivalent binary sequences. This renders the preservation of correlation.

1.2 Contribution of the Thesis

In this thesis, a Wyner-Ziv codec was developed using non-binary LDPC codes, in which a frame was converted into its non-binary equivalent. This non-binary sequence was compressed and the successive frame was made available at the decoder as the side information. The non-binary LDPC decoder used the correlation information between the two consecutive frames and the side information frame to decode the original frame transmitted. The distributed source coding scheme for non-binary LDPC codes is implemented in probability domain as opposed to the log domain [5]. This was necessitated as the development of distributed source coding for binary sources could not be extended directly to the non-binary sources.

1.3 Thesis Organization

The following chapter ia a literature review that describes Gallager's original low-density parity-check codes. In Chapter 3, the non-binary low-density parity-check codes are introduced. In Chapter 4, distributed source coding using binary LDPC codes are presented. In Chapter 5, a Wyner-Ziv codec is developed using non-binary LDPC codes. Distributed source coding using binary LDPC codes is extended to non-binary LDPC codes. The results are discussed in Chapter 6, followed by conclusions and future work in Chapter 7.

Chapter 2

Literature Review

Source coding is a technique used to compress the source, by removing redundancy and in turn, reduces the bandwidth needed for the transmission through a channel. Source characteristics such as signal redundancy and distortion masking are exploited in the process of compression. The source coding can be classified as lossless source coding and lossy source coding. Compression of a signal, wherein the decompression reproduces the original signal is referred to as lossless source coding. Slepian-Wolf [1] coding is viewed as lossless source coding. The compression of a signal with the removal of redundant information is lossy source coding, the best example of which is Wyner-Ziv coding. Shannon showed that there is a limit to which a source can be compressed without introducing errors at the decoder. Rate distortion theory gives a trade off between compression and quality in lossy source coding. Images, video and audio are often compressed using lossy source coding to achieve better compression techniques, however compressors will have a lossless mode.

Distributed source coding of correlated sources, is the compression of correlated sources that send their output to a common decoder that do not communicate with each other. It has evolved out of the need for removal of redundancy of data in networks involving applications using dense sensor networks. The problem of lossless compression of finite alphabets takes its root from the fundamental paper of Slepian and Wolf [1]. The Slepian-Wolf theorem states that the output of two correlated sources can be compressed to the same extent with or without loss, whether they communicate with each other or not provided the decompression takes place at the joint decoder. Consider two statistically dependent i.i.d finite-alphabet random sequences X and Y. With separate conventional entropy and encoders and decoders, one can achieve $R_X \geq H(X)$ and $R_Y = H(Y)$, where $H(X)$ and $H(Y)$ are the entropies of X and Y respectively. Interestingly, we can do better with joint decoding (but separate encoding), if we are content with a residual error probability for recovering X and Y that can be made arbitrarily small for encoding long sequences. In this case, Slepian-Wolf theorem establishes the rate region as shown in figure 2.1.

$$R_X + R_Y \geq H(X,Y) \tag{2.0.1}$$

$$R_X \geq H(X|Y) \tag{2.0.2}$$

$$R_Y \geq H(Y|X) \tag{2.0.3}$$

Surprisingly, the sum rates $R_X + R_Y$ can achieve the joint entropy $H(X,Y)$, just as for joint encoding of X and Y, despite separate encoders for X and Y.

Wyner-Ziv have extended the Slepian -Wolf theorem to continuous valid Gaussian sources. According to Wyner and Ziv [2] [6] [7], for two correlated sources X and Y, the rate distortion performance obtained for encoding X is the same whether the encoder has an understanding of Y or not, if Y is available at the decoder. The Wyner-Ziv rate distortion function $R^{WZ}_{X|Y}(D)$ gives the minimum rate R necessary to reconstruct the source X with average distortion less than or equal to D. The rate

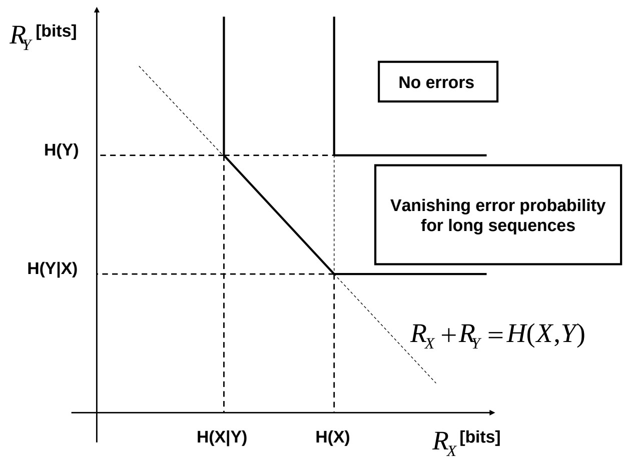

Figure 2.1: Slepian-Wolf theorem: Achievable rate region for distributed compression of two statistically dependent i.i.d. sources X and Y.

required if the side information is available at the encoder as well is given by $R_{X|Y}(D)$. Wyner-Ziv proved that unsurprisingly, a rate loss $R^{WZ}_{X|Y}(D) - R_{X|Y}(D) \geq 0$ is incurred when the encoder does not have access to the side information. However, they also showed that $R^{WZ}_{X|Y}(D) - R_{X|Y}(D) = 0$ in case of Gaussian memoryless sources and mean-squared error distortion [2] [7].

Implementations of current video compression standards, such as ISO MPEG schemes or the ITU-T recommendations H.263 and H.264 [8] [9] require much more computation for the encoder than for the decoder. This asymmetry is well studied for broadcasting or for streaming video-on-demand systems where video is compressed once and decoded many times. However, some applications may require dual applications, i.e., a low-complexity encoders, at the expense of high-complexity decoders. Examples include wireless video sensors for surveillance, wireless PC cameras, mobile camera phones, disposable video cameras, and networked camcorders. In all these examples, the resources at the encoder are scarce.

The Wyner-Ziv theory [2] [10] [11], suggests that an unconventional video coding system, which encodes individual frames independently, but decodes them conditionally, is viable. Such a system might achieve a performance that is closer to conventional interframe coding (MPEG) than to conventional intraframe coding (Motion-JPEG). In contrast to conventional hybrid predictive video coding where motion-compensated previous frames are used as side information, in the proposed system previous frames are used as the side information at the decoder only.

The simplest system is the pixel-domain intraframe encoder and interframe decoder system for video compression [12] [13]. A subset of frames serve as the key frames which are encoded and decoded using a conventional intraframe video codec. The frames between the key frames are "Wyner-Ziv frames" which are inraframe encoded but interframe decoded. Neither motion estimation and prediction, nor DCT and inverse DCT are required at the encoder. Compared to motion-compensated predictive hybrid coding, pixel-domain Wyner-ziv coding is orders of magnitude less complex.

In conventional, nondistributed source coding, orthogonal transforms are widely used to decompose the source vectors into spectral coefficients, which are individually coded with scalar quantizers and entropy coders. The use of transforms in Wyner-Ziv coding of images are considered and analyzed the bit allocation problem for the Gaussian case by Pradhan and Ramchandran . Distributed source coding using KLT (Karhunen-Loive Transform) has been investigated by Gastpar [14] [15], in which they have assumed that the covariance matrix of source vector given the side information doesnot depend on the values of the side information, and the study is not in the context of practical coding scheme. A similar transform-domain Wyner-Ziv video coder has been developed independently by Puri and Ramchandran and presented under the acronym PRISM [16] [17] [18]. Preliminary rate-distortion results show a performance between conventional intraframe transform coding and conventional motion-compensated transform coding, similar to the results produced by Aaron and Girod [19] in their DCT-domain scheme. The DCt-domain scheme has higher encoder complexity than the pixel-domain system. However, it remains much less complex than interframe predictive schemes because motion estimation and compensation are not needed at the encoder.

To achieve high compression efficiency in a Wyner-Ziv video codec, motion gas to be estimated at the decoder. Conventional motion compensated coding benefits from estimating the best motion vector by directly comparing the frame to be encoded with one or more reference frames. The analogous approach for Wyner-Ziv video coding requires joint decoding and motion estimation, using the Wyner-Ziv bits and possibly additional helper information from encoder. An example of this has been implemented in the PRISM system, where the CRC of the quantized symbols aid in determining the motion at the decoder. The CRC of each decoded version is then compared with the transmitted CRC to establish which version should be used [18].

Most of the practical solutions for the implementation of distributed source coding are derived from channel coding concepts. The statistical dependence or correlation between the sources is modeled as a virtual correlation channel equivalent to a binary symmetric channels or additive white Gaussian (AWGN) channel. The design of encoders for the sources are then obtained based on channel coding concepts like block codes, convolutional codes, turbo codes and low-density parity-check codes.

Chapter 3

Low-Density Parity-Check Codes

3.1 Introduction to Error-Correcting Codes

The basic concept of all error-correcting codes is to add redundancy to information in order to correct any errors that occur in the process of transmission. Redundant symbols are appended to the information symbols to obtain a coded sequence or a codeword. The encoding in which the information symbols always appear in the first k positions of a codeword and the remaining $n-k$ symbols provide redundancy, that can be used for error correction/detection purposes, is said to be semantic.

3.2 Introduction to Linear Block Codes

Error correcting codes can be divided into two classes, block and convolutional. For many years, convolutional codes have been preferred over block codes because of the availability of the soft-decision Viterbi decoding algorithm. The recent development in theory and design of soft-decision decoding algorithms for block codes has dispelled this belief.

Block codes process the information on a block-by-block basis, treating each block of information bits independently from others. An (n, k) block code C is said to be linear if the vector sum of two codewords is a codeword. It is also viewed as a mapping of k-dimensional subspace to the n-dimensional vector space V_n by a kXn generator matrix G, where $C = \mathbf{m}G$. Since C is a k-dimensional vector space in V_n, there is a $(n-k)$-dimensional dual space $C^\top$, generated by the rows of a parity-check matrix, H, such that $GH^\top = 0$, where $H^\top$ denotes the transpose of H. For any particular codeword $\bar{v} \in C$,

$$\bar{v}H^\top = 0 \tag{3.2.1}$$

A linear code $C^\top$ generated by H is also a linear code known as dual code of C.

3.3 Introduction to Low-Density Parity-Check codes

Low-density parity-check codes are a class of linear error correcting codes defined by a very sparse parity-check matrix, H. Invented by Gallager [20] in 1963, these codes were largely forgotten until they were rediscovered by Mackay and Neal [21]. The rediscovery of these codes has sparked major research in the coding field because of their near Shannon limit performance and simple description. Each codeword satisfies a number of linear constraints and each symbol of the codeword participates in a small number of constraints.

The binary LDPC codes developed by Gallager were extended to their non-binary versions by Mackay and Davey [4]. Codes having a variable number of non-zero values in the parity-check matrix also were developed, known as irregular LDPC codes. In irregular LDPC codes, the high weight columns help the decoder to identify some errors quickly, making the remaining errors easier to correct. The regular LDPC codes have a fixed number of non-zero values in the H matrix. The non-binary version uses symbols from a finite field with more than two elements, performing encoding and decoding on these symbols. This results in increased complexity in the parity-check matrix but the decoding remains tractable.

A sparse parity-check matrix H is used to define a LDPC code. An (n, p, q) low density code is a code of block length n having a $(n-k) \times n$ parity check matrix, H with each column containing a fixed number, p, of ones and each row containing a fixed number, q, of ones. Gallager proved that, for a fixed p, the error probability of the optimum decoder decreases exponentially for sufficiently low noise and sufficiently long block length.

The H matrix can be represented as a biparitate graph, which is defined as an undirected graph where vertices can be divided into two sets such that no edge connects vertices in the same set.Each bit (column of H) is represented by a variable (left) node and each check (row of H) is represented by a check (right) node. For binary codes, the values in the H matrix are either 1 or 0. A 1 denotes an edge between the corresponding variable node and the check node. If the H matrix has n columns and m rows, the corresponding biparitate graph has n bit nodes and m check nodes. Figure 3.1 shows a (20,3,4) code constructed by Gallager. If all the rows are linearly independent, the code rate is given by $R = 1 - \frac{p}{q}$, otherwise the code rate is $\frac{n-p'}{n}$, where p' is the dimension of the row space of H. An (n, k) encoder accepts k-bit input and produces an n-bit codeword.

3.3.1 Encoding and Decoding of LDPC Codes

The Generator matrix G is needed for encoding the input message. Since H is not in systematic form, Gaussian elimination is performed using row operations and reordering of columns. The resulting parity check matrix has the form $H = [-P|I_m]$, where notation $[A|B]$ indicates the concatenation of matrices A and B; and I_m is

1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0
0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0
0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1
1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0
0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1

Figure 3.1: Example of a parity-check matrix for a (20,3,4) LDPC code.

the $m \times m$ identity matrix. Once H is in systematic form, it is easy to confirm that a valid generator matrix is $G = [I_k|P]$ since $GH^\top = 0$.The generator matrix is not sparse. So the encoding complexity is $O(n \times n)$ per block. The (n, k) block encoder accepts the message of length k and produces a codeword of length n. The decoding is done using the message passing, belief propagation or sum-product algorithm for decoding

$$\mathbf{r}H^\top = \mathbf{S} \; mod2 \tag{3.3.1}$$

where $\mathbf{r}$ is the received vector that is corrupted by the addition of noise, $\mathbf{S}$ is the syndrome vector. Elements corresponding to each row of $\mathbf{S}$ are referred to as checks. Assuming the set of variables and checks as making up a belief network or Bayesian network [22] in which every variable is the parent of checks, and each check is the child of variable. The network of checks and variables form a biparitate graph defined by the H matrix with variables connected only to checks and checks connected only to

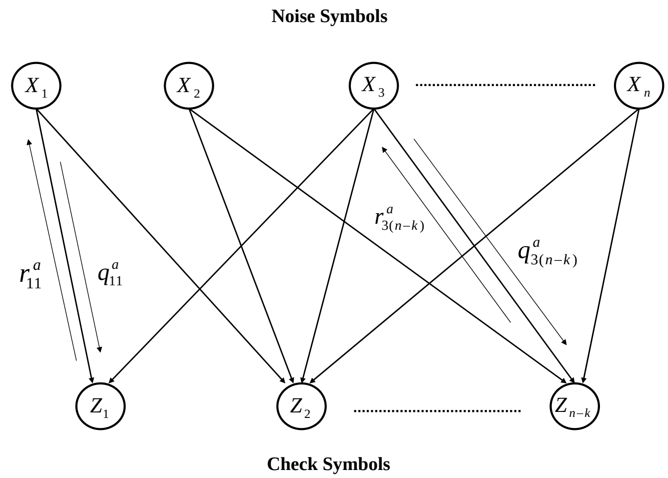

Figure 3.2: Message passing on the biparitate graph representing a parity-check matrix.

variables. The aim of the algorithm is to compute the marginal posterior probabilities from the observed checks. This is done by estimating the posterior probability of the value of each variable node given the received signal and the channel properties. The process can be viewed as message passing algorithm on the biparitate graph with two sets of nodes. Nodes S_i, where $i = 1, 2, 3, ..., m$, and r_j, where $j = 1, 2, 3, ..., n$, are connected if the corresponding entry in the matrix H_{ij} is nonzero. The directed edge shows the casual relationship that the state of the check node is determined by the state of the variable nodes to which it is connected. At each step of the decoding algorithm, each variable node sends messages q_{mn}^x to each child S_i which are supposed to approximate the node's belief that it is in state x, given messages received from all of its other children. Also each check S_i sends messages r_{mn}^x to each parent r_j approximating the probability of check i being satisfied if the parent is assumed to be in state x, taking into account messages received from all its other parents. The messages are observed after each step and a tentative decoding is estimated. The decoding algorithm is then updated iteratively until the tentative decoding satisfies the observed syndrome vector, **S**, upon which the success is declared. If the observed syndrome is not satisfied after a preset maximum number of iterations are reached, a failure is declared. The preset maximum number of iterations may be set to about ten times the typical number to improve the success rate while imposing little overhead on the average decoding time. In practice the algorithm usually converges and all decoding errors are detected. Sometimes the algorithm may fail to converge due to many cycles in the graph. Hence care should be taken to avoid many cycles but in practice the decoding performance is not affected much due to cycles.

Consider the following H matrix with column weight $p = 2$. This matrix represents a set of linear homogeneous modulo 2 equations called parity check equations with the set of codewords as solutions to these equations. The set of digits contained in a parity check equation is known as parity check set. The use of parity check nodes makes coding relatively easy to implement.

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{3.3.2}$$

For any valid codeword c,

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \end{bmatrix} = 0. \tag{3.3.3}$$

This serves as a starting point for the construction of the decoder. The matrix multiplication of the above expression defines the following set of parity check equations

$$p_0 = c_0 \oplus c_3 \oplus c_4 \oplus c_5 \oplus c_6 \tag{3.3.4}$$

$$p_1 = c_1 \oplus c_3 \oplus c_5 \tag{3.3.5}$$

$$p_2 = c_1 \oplus c_2 \tag{3.3.6}$$

$$p_3 = c_0 \oplus c_2 \oplus c_4 \oplus c_6. \tag{3.3.7}$$

Having encoded the source vector, $\mathbf{X}$ of length k using the generator matrix, the codeword $c = \mathbf{X}G$ is transmitted. The received vector is $r = c + N$, where N is the noise introduced by the channel. On multiplying the received vector by the H matrix, we get the syndrome vector,

$$\begin{aligned} \mathbf{S} &= rH^\top && (3.3.8) \\ &= (c+N)H^\top && (3.3.9) \\ &= \mathbf{X}GH^\top + H^\top N && (3.3.10) \\ &= H^\top N. && (3.3.11) \end{aligned}$$

The addition and multiplication is performed over a finite field. The decoding involves finding the most probable vector $\hat{\mathbf{X}}$ according to the channel model such that $\hat{\mathbf{X}}H^\top = \mathbf{S}$.

The set of variable nodes n that participate in check m are denoted by $N(m) = n : H_{mn} = 1$. We also define the set of checks in which variable node n participates, $M(n) = m : H_{mn} = 1$. The set of all variable nodes excluding the particular variable node n is denoted by $N(m) \setminus n$. There are two quantities q_{mn} and r_{mn} associated with each nonzero element in H matrix that are alternatively updated iteratively. The quantity q^a_{mn} is the probability that variable node n of $\mathbf{X}$ has the value a, given information obtained via checks other than check m. The quantity r^a_{mn} is the probability of check m being satisfied if bit n of $\mathbf{X}$ is considered fixed at a with other bits having a separable distribution given by the probabilities $q_{mn'} : n' \in N(m) \setminus n$. The exact posterior probabilities of all bits are produced after a fixed number of iterations if the biparitate graph, defined by the H matrix, has no cycles.

Applying the LDPC codes to a binary symmetric channel (BSC) with crossover probability p. The prior probabilities are given by

$$p(r_j/x_j = 0) = p^{r_j}(1-p)^{1-r_j} \tag{3.3.12}$$

$$p(r_j/x_j = 1) = p^{1-r_j}(1-p)^{r_j} \tag{3.3.13}$$

where $p(r_j/x_j = 0)$ is the prior probability of bit r_j given the bit $x_j = 0$; $p(r_j/x_j = 1)$ is the prior probability of bit r_j given $x_j = 1$.

The horizontal step involves running through the checks m and compute for each $n \in N(m)$, two probabilities r^0_{mn} and r^1_{mn}. The probability r^0_{mn} is the the observed value of S_m when $x_n = 0$ given the other bits $x_{n'} : n' \neq n$ have a separable distribution given by the probabilities $q^0_{mn'}, q^1_{mn'}$. The probability, r^0_{mn}, which is obtained by running through the checks m for each $n \in N(m)$ is given by

$$\sum_{x_{n'}:n' \in N(m)\backslash n} P[S_m/x_n = 0, x_{n'} : n' \in N(m) \setminus n] \times \prod_{n' \in \ N(m)\backslash n} q_{mn'}^{x_{n'}}. \qquad (3.3.14)$$

The probability, r^1_{mn} is the observed value of S_m arising when $x_n = 1$ is defined by

$$\sum_{x_{n'}:n' \in N(m)\backslash n} P[S_m/x_n = 1, x_{n'} : n' \in N(m) \setminus n] \times \prod_{n' \in \ N(m)\backslash n} q_{mn'}^{x_{n'}}. \qquad (3.3.15)$$

The conditional probabilities in the summations are either zero or one, depending on whether the observed z_m matches the hypothesized values of x_n and the $x_{n'}$.

A convenient way to implement these probabilities is using the forward and backward passes with the product of the differences $\delta q_{mn} = q^0_{mn} - q^1_{mn}$ computed.

$$\delta r_{mn} = r^0_{mn} - r^1_{mn} \qquad (3.3.16)$$

$$\delta r_{mn} = (-1)^{S_m} \prod_{n' \in N(m)\backslash n} \delta q_{mn'} \qquad (3.3.17)$$

Also $r^0_{mn} + r^1_{mn} = 1$, and hence $r^0_{mn} = (1 + \delta r_{mn})/2$ and $r^1_{mn} = (1 - \delta r_{mn})/2$.

In the vertical step, the values of r^0_{mn} and r^1_{mn} are used to update the probabilities q^0_{mn} and q^1_{mn}

$$q^0_{mn} = \alpha_{mn} p^0_n \prod_{m' \in M(n)\backslash m} r^0_{m'n} \qquad (3.3.18)$$

$$q^1_{mn} = \alpha_{mn} p^1_n \prod_{m' \in M(n)\backslash m} r^1_{m'n} \qquad (3.3.19)$$

where α_{mn} is chosen such that $q^0_{mn} + q^1_{mn} = 1$. These probabilities can be efficiently computed in a downward pass and an upward pass.

Also the pseudoposterior probabilities q^0_n and q^1_n at this iteration are computed using

$$q_n^0 = \alpha_n p_n^0 \prod_{m \in M(n)\ r_{mn}^0} \tag{3.3.20}$$

$$q_n^1 = \alpha_n p_n^1 \prod_{m \in M(n)\ r_{mn}^1} . \tag{3.3.21}$$

These quantities are used to find $\hat{\mathbf{X}}$, which is a tentative decoded vector that is used to calculate the syndrome using the H matrix. If the syndrome is satisfied, the decoding is halted and a success is declared, otherwise the decoding is repeated for a preset maximum number of iterations. If the syndrome is not satisfied even after a maximum number of iterations, a failure is declared. The difference between this decoding algorithm and the decoding algorithm used in turbo codes is that the decoding algorithm is repeated for a preset number of iterations irrespective of whether a consistent state is reached or not. In the decoding algorithm used in LDPC codes, undetected errors occur only if the $\hat{\mathbf{X}}$ estimated satisfies $\hat{\mathbf{X}}H^{\top} = \mathbf{S}\ mod2$ and is not the same as $\mathbf{X}$. Detected errors occur if the maximum number of iterations is completed without a valid decoding.

Chapter 4

Non-Binary Low-Density Parity-Check Codes

The binary low-density parity-check codes, represented by sparse binary matrices or by corresponding biparitate graphs are generalized by Mackay and Davey [4] by using the same biparitate graphs. The noise nodes are allowed to take values from a finite alphabet and the check nodes impose constraints more complex than binary parity-checks. The codes over finite fields with q elements (denoted $GF(q)$) where $q = 2^b$ for some integer b. The elements of $GF(q)$ are known as symbols. To continue the use of binary channels, powers of two are considered, transmitting one q-ary symbol for every b uses of the channel. A symbol from the $GF(2^b)$ may be represented as a binary string of b bits.

The very sparse random parity-check matrix H is used to define the non-binary LDPC code with a transmitted block length of n and a source block length of k. The nonzero entries in each row of H are chosen to maximize the entropy of the corresponding symbol of the syndrome $\mathbf{S} = \mathbf{X}H^\top$, where $\mathbf{X}$ is a sample from the assumed channel noise model. The parity-check matrix H is constructed as before, but there are $q-1$ choices available for each non-zero entry. The decoding algorithm has the same form described for binary case but the symbol a now ranges over q possible values instead of just two.

The decoder interprets b bits $(X_1, X_2, ..., X_b)$ from the channel as a single q-ary symbol and sets the prior distribution for that symbol by assuming a product of distribution for the values of each constituent bit. That is to say

$$f^a := \prod_{i=1}^{b} f_{X_i}^{a_i} \tag{4.0.1}$$

where $f_{X_i}^{a_i}$ is the likelihood the i^{th} constraint bit X_i is equal to a_i, where $(a_1, ..., a_b)$ is the binary representation of the symbol a.

With each symbol $a \in GF(2^b)$ we can associate a $b \times b$ binary matrix. Multiplication of two symbols is equivalent to matrix multiplication (mod 2) of their corresponding binary strings. By replacing each symbol in the q-ary matrices, G_q and H_q by the associated binary $b \times b$ blocks we obtain binary matrices, G_2 and H_2 that are b times as large as in each direction. To multiply a q-ary vector, X_q by G_q we can form the binary representation of X_q, multiply by G_2, and take the q-ary representation of the binary vector.

4.1 Decoding over $GF(q)$

The decoding algorithm is the generalization of the approximate belief propagation algorithm. The elements of X are variables and the elements of S are checks. $N(m) = n : H_{mn} \neq 0$ denotes the set of variables participating in the check m. $M(n) = m : H_{mn} \neq 0$ denotes the set of checks that depend on variable symbol n. Quantities q_{mn}^a and r_{mn}^a are calculated for each nonzero entry in the H matrix. q_{mn}^a is the probability that the symbol, n of X is a given the information obtained from the check, m. r_{mn}^a is the probability of check, m being satisfied given that symbol n

of X is fixed at a with other variable symbols having a separable distribution given by $q^a_{mn'} : n' \in N(m) \setminus n, a \in GF(q)$.

Applying the codes to binary symmetric channel (BSC) with cross over probability p, we define $g^1_n = p^{1-r_j}(1-p)^{r_j}$ and $g^0_n = 1 - g^1_n$, g^1_n is independent of n. The channel likelihoods are set to $f^a_n = \prod_{i=1}^{b} \, g^{a_i}_{n_i}$ for each noise symbol X_n being equal to a, for each $a \in GF(q)$, a_i is the binary representation of a. Each noise symbol X_n consists of b bits $X_{n_1}, ..., X_{n_b}$ and $g^{a_i}_{n_i}$ is the likelihood of the i^{th} constituent bit X_i being equal to a_i, where $(a_1, ..., a_b)$ is the binary representation of symbol a.

Horizontal step is the computation of the quantity r^a_{mn} which is the probability of check m being satisfied if the symbol n of X is considered fixed at a and other noise symbols have a separable distribution given by $q^a_{mn'} : n' \in N(m) \setminus n, a \in GF(q)$.

Value of r^a_{mn} is computed using

$$r^a_{mn} \;=\; \sum_{\mathbf{X'}:X'_n=a} P[S_m/\mathbf{X'}] \times \prod_{j \;\in\; N(m)\setminus n} q^{X'_j}_{mj'} \tag{4.1.1}$$

$P[S_m/\mathbf{X'}]$ is 0 or 1 depending on whether or not X' satisfies the check m. r^a_{mn} can be calculated efficiently by defining partial sums $\sigma_{mk} := \sum_{j:j\leqslant k} H_{mj}X'_j$ and $\rho_{mk} := \sum_{j:j\geqslant k} H_{mj}X'_j$ and calculating the $P[\sigma_{mk} = a]$ for each $a \in GF(q)$ and each $k \in N(m)$ according to the probabilities given by q^a_{mn}. If i, j are successive indices in *N(m)* and $j > i$ then

$$\Pr[\sigma_{mj} = a] = \sum_{\{s,t:H_{mj}t+s=a\}} \Pr[\sigma_{mj} = s]q^t_{mj} \tag{4.1.2}$$

Along the same lines we can also calculate the distribution of each ρ_{mk}. Using σ_{mk}

and ρ_{mk} the quantity r^a_{mn} can be updated using

$$r^a_{mn} = \Pr[(\sigma_{m(n-1)} + \rho_{m(n+1)}) = S_m - H_{mn}a] \tag{4.1.3}$$

$$\tag{4.1.4}$$

$$= \sum_{s,t:s+t=S_m-H_{mn}a} \Pr[\sigma_{m(n-1)} = s] \times \Pr[\rho_{m(n+1)} = t]. \tag{4.1.5}$$

Vertical step involves updating the q^a_{mn} which is the probability that the symbol n of X is a, given information obtained from checks other than check m.

Value of q^a_{mn} is updated for each m and n using

$$q^a_{mn} = \alpha_{mn} f^a_n \prod_{j\in M(n)\backslash m} r^a_{jn} \alpha_{mn} \tag{4.1.6}$$

where α_{mn} is chosen such that $\sum_{a=1}^{q} q^a_{mn} = 1$

Tentative decoding $\hat{\mathbf{X}}$ is performed using

$$\hat{X}_n = argmax_a \; f^a_n \prod_{j\in M(n)} r^a_{jn} \tag{4.1.7}$$

If a valid decoding of the syndrome $\hat{X}H = S$ is identified then the algorithm halts, else the algorithm is repeated. Valid decoding of the syndrome is an all zero syndrome. Failure is declared if a valid decoding is not obtained even after some maximum number of iterations.

MacKay [23] proved that, given an optimal decoder, LDPC codes can approach arbitrarily close to the Shannon limit if we sufficiently choose high (fixed) column weight of H and then choose a sufficiently large block length. However, as the column weight is increased, the performance of the iterative decoding algorithm decreases as the number of cycles in the associated biparitate graph increases drastically.

By using the generalized LDPC codes, the mean column weight t of the equivalent binary matrix H_2 can be increased, while retaining the same biparitate graph on

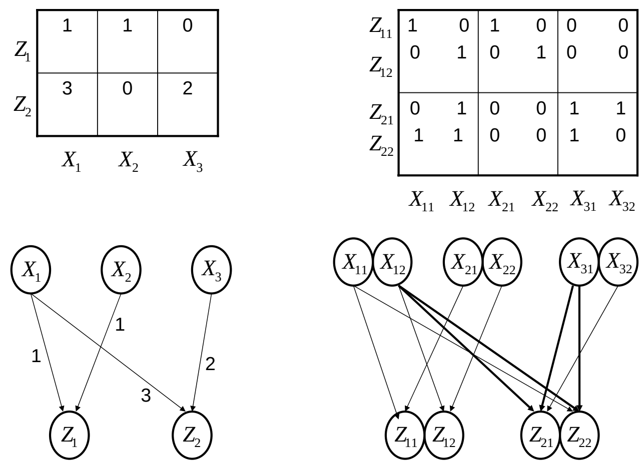

Figure 4.1: Upper panels: fragments of equivalent parity-check matrices over GF(2) and GF(4). Lower panels: comparison of corresponding graph structure. Note the presence of a short cycle in the graph for the binary code.

which the decoding is performed. In the figure 4.1, the graphs of two equivalent matrix fragments are compared. It is seen that the q-ary code contains no cycles, whereas the binary code has a length 4 (highlighted). Hence the decoding behavior is dependent on the choice of field $GF(q)$ even though the transmitted (binary) messages are identical.

Another difference between binary and q-ary codes is that the state space of each node is increased in the decoding graph by decoding over $GF(q)$. This allows us to track correlations in the true posterior distribution that are not detectable by binary algorithm. Also increasing the field order q for LDPC codes is comparable to increasing the memory of convolutional codes. The only drawback of moving to $GF(q)$ is the increase in decoding complexity.

Chapter 5

Distributed Source Coding using Low-Density Parity-Check Codes

Low-density parity-check codes can be used for applications involving compression of two correlated sources using the syndrome concept. The compressed sequence of the source output bits is the syndrome, which is determined using the parity-check matrix H. It has been shown [5] that LDPC codes can be employed when viewing a problem using an equivalent channel and applying the syndrome concept for the case where one of the two correlated sources is available losslessly at the decoder. This can be viewed as the application of LDPC codes to a compression problem with side information. It is based on modifying the conventional message passing LDPC decoder to take into account the syndrome information. Also all LDPC code design techniques can be applied to distributed source coding producing simulation results better than any turbo coding scheme.

The system considered consists of two non binary information sources $\mathbf{X} = [X_1, X_2, \ldots, X_n]$ and $\mathbf{Y} = [Y_1, Y_2, \ldots, Y_n]$, where X_i and Y_i are the elements of the Galois field and n is the length of the sequence. X_i and Y_i are correlated with probability

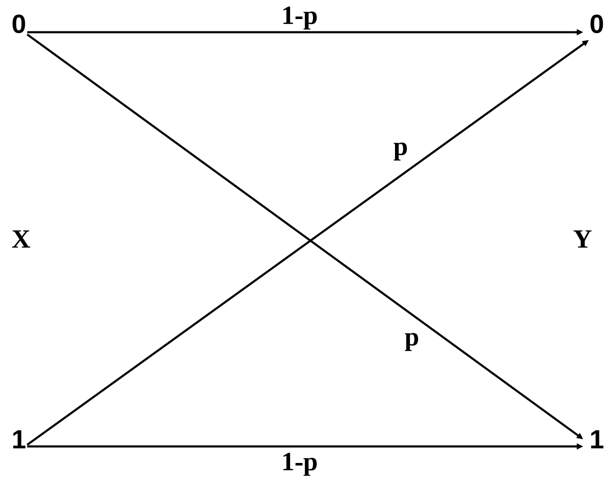

Figure 5.1: Binary Symmetric Channel

$P[X_i \neq Y_i] = p < 0.5$. The correlation between $\mathbf{X}$ and $\mathbf{Y}$ is modeled using a binary symmetric channel with crossover probability p, as shown in figure 5.1. The input to the channel is $\mathbf{X}$ and $\mathbf{Y}$ is the output, where the compressed version of $\mathbf{X}$ is a codeword to the channel. The source $\mathbf{Y}$ is available losslessly at the joint decoder.

The rate used for $\mathbf{Y}$ is its entropy $nR_2 = nH(Y_i) = n\, bits$, therefore the theoretical limit for lossless compression of $\mathbf{X}$ is [1]

$$nR_1 \geqslant nH(X_i/Y_i) = nH(p) \tag{5.0.1}$$

$$= n\left((-p)\log_2 \frac{p}{q-1} - (1-p)\log_2(1-p)\right) \tag{5.0.2}$$

The compression results in mapping a sequence of n input bits into $(n-k)$ syndrome bits resulting in a compression ratio of $n : (n-k)$, known as the "Wyner's scheme" [24]. The all zero syndrome of the linear block code is considered to be the original code for distributed coding.

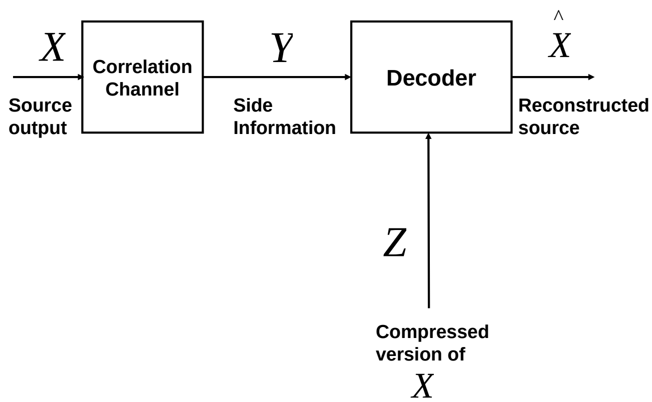

Figure 5.2: Correlation channel used for distributed source coding

5.1 Encoding and Decoding with Binary LDPC Codes

Encoding is done by multiplying the source X, of length n, by the parity-check matrix H, resulting in the compressed X which is the syndrome S, of length $n-k$.

$$S = XH^{\top} \tag{5.1.1}$$

In the biparitate graph, encoding can be viewed as the binary addition of all the variable nodes connected to the same check node. Consider an example of H matrix shown below

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{5.1.2}$$

$$\begin{bmatrix} X_0 & X_1 & X_2 & X_3 & X_4 & X_5 & X_6 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} S_0 & S_1 & S_2 & S_3 \end{bmatrix} \tag{5.1.3}$$

$$X_0 \oplus X_3 \oplus X_4 \oplus X_5 \oplus X_6 = S_0 \tag{5.1.4}$$

$$X_1 \oplus X_3 \oplus X_5 = S_1 \tag{5.1.5}$$

$$X_1 \oplus X_2 = S_2 \tag{5.1.6}$$

$$X_0 \oplus X_2 \oplus X_4 \oplus X_6 = S_3 \tag{5.1.7}$$

Here the vector $\mathbf{S} = \begin{bmatrix} S_0 & S_1 & S_2 & S_3 \end{bmatrix}$ is the compressed version of the source sequence $\mathbf{X}$. As the vector $\mathbf{S}$ is binary, it consists of only 0's and 1's.

Decoding involves the estimation of the n-length sequence $\mathbf{X}$ from the $(n-k)$ length syndrome S and the n length sequence $\mathbf{Y}$. The decoding algorithm is similar to LDPC decoding for channel coding except for the inclusion of the syndrome bits in the horizontal step of the algorithm.

The set of noise bits n that participate in check m are denoted by $N(m) = n : H_{mn} = 1$. We also define the set of checks in which variable node n participates, $M(n) = m : H_{mn} = 1$. The set of all variable nodes excluding the particular variable node n is denoted by $N(m)\backslash n$. There are two quantities q_{mn} and r_{mn} associated with each nonzero element in H matrix that are alternatively updated iteratively. q^a_{mn} is the probability that variable node n of $\mathbf{X}$ has the value a, given information obtained via checks other than check m. r^a_{mn} is the probability of check m being satisfied if bit

n of $\mathbf{X}$ is considered fixed at a with other bits having a separable distribution given by the probabilities $q_{mn'} : n' \in N(m) \setminus n$.

Consider a binary symmetric channel (BSC) with crossover probability p. We model the correlation between the two sources using the binary symmetric channel with X_i as the input to the channel and Y_i as the output. The compressed version of X is the syndrome S which is made to look like a codeword of the channel.The prior probabilities are given by

$$p(Y_j/X_j = 0) = p^{Y_j}(1-p)^{1-Y_j} \tag{5.1.8}$$

$$p(Y_j/X_j = 1) = p^{1-Y_j}(1-p)^{Y_j} \tag{5.1.9}$$

where $p(Y_j/X_j = 0)$ is the prior probability of bit Y_j given the bit $X_j = 0$ and $p(Y_j/X_j = 1)$ is the prior probability of bit Y_j given $X_j = 1$.

The horizontal step involves the computation of two probabilities r^0_{mn} and r^1_{mn} are the probabilities of the observed values of S_m when $X_n = 0$ and $X_n = 1$ respectively, given the other bits $X_{n'} : n' \neq n$ have a separable distribution given by the probabilities $q^0_{mn'}$ and $q^1_{mn'}$, which are obtained by running through the checks m for each $n \in N(m)$. The syndrome information is included in this step to modify the message passing LDPC decoder for distributed source coding. r^0_{mn} is given by

$$\sum_{x_{n'}:n' \in N(m)\setminus n} P[S_m/X_n = 0, X_{n'} : n' \in N(m) \setminus n] \times (1-2S_m) \times \prod_{n' \in N(m)\setminus n} q^{X_{n'}}_{mn'} \tag{5.1.10}$$

r^1_{mn} is the probability of the observed value of S_m arising when $X_n = 0$, defined by

$$\sum_{X_{n'}:n' \in N(m)\setminus n} P[S_m/X_n = 1, X_{n'} : n' \in N(m) \setminus n] \times (1-2S_m) \times \prod_{n' \in N(m)\setminus n} q^{X_{n'}}_{mn'} \tag{5.1.11}$$

The conditional probabilities in the summations are either zero or one, depending on whether the observed z_m matches the hypothesized values of X_n and the $X_{n'}$.

A convenient way to implement these probabilities is using the forward and backward passes with the product of the differences $\delta q_{mn} = q^0_{mn} - q^1_{mn}$ computed.

$$\begin{aligned} \delta r_{mn} &= r^0_{mn} - r^1_{mn} && (5.1.12) \\ \delta r_{mn} &= (-1)^S_m \prod_{n' \in N(m)\backslash n} \delta q_{mn'} && (5.1.13) \end{aligned}$$

Also $r^0_{mn} + r^1_{mn} = 1$, and hence $r^0_{mn} = (1 + (1 - 2S_m)\delta r_{mn})/2$ and $r^1_{mn} = (1 - (1 - 2S_m)\delta r_{mn})/2$. The syndrome is included in the horizontal step in the calculation of $r^0_{mn} = (1 + (1 - 2S_m)\delta r_{mn})/2$ and $r^1_{mn} = (1 - (1 - 2S_m)\delta r_{mn})/2$.

In the vertical step, the values of r^0_{mn} and r^1_{mn} are used to update the probabilities q^0_{mn} and q^1_{mn}

$$\begin{aligned} q^0_{mn} &= \alpha_{mn} p^0_n \prod_{m' \in M(n)\backslash m} r^0_{m'n} && (5.1.14) \\ q^1_{mn} &= \alpha_{mn} p^1_n \prod_{m' \in M(n)\backslash m} r^1_{m'n} && (5.1.15) \end{aligned}$$

where α_{mn} is chosen such that $q^0_{mn} + q^1_{mn} = 1$. These probabilities can be efficiently computed in a downward pass and an upward pass.

Also the pseudoposterior probabilities q^0_n and q^1_n at this iteration are computed using

$$q^0_n = \alpha_n p^0_n \prod_{m \in M(n)\ r^0_{mn}} \qquad (5.1.16)$$

$$q^1_n = \alpha_n p^1_n \prod_{m \in M(n)\ r^1_{mn}} \qquad (5.1.17)$$

These quantities are used to find $\hat{\mathbf{X}}$, which is a tentative decoded vector that is used to calculate the syndrome using the H matrix.

The main difference in the decoding of LDPC codes for distributed source coding and channel coding is that in case of channel coding, the message passing of the probabilities between the horizontal and the vertical step is stopped on the occurrence of a zero syndrome vector. The syndrome in case of channel coding is the decoded message multiplied with a parity-check matrix. In distributed source coding, iterations are fixed to a given value, after which the message vector $\hat{\mathbf{X}}$ is estimated. The decoding algorithm has been implemented in the probability domain.

Chapter 6

Distributed Video Coding using Non-Binary Low-Density Parity-Check Codes

6.1 Distributed Source Coding using Non-Binary LDPC Codes

Generalized non-binary low-density parity-check codes can be used for applications involving compression of two correlated non-binary sources using syndrome concept. The compressed sequence of the source output bits is the syndrome, which is determined using the parity-check matrix H. The non-binary LDPC algorithm has been modified to taken into account the syndrome information during decoding.

The system considered consists of two non-binary information sources $\mathbf{X} = [X_1, X_2, \ldots, X_n]$ and $\mathbf{Y} = [Y_1, Y_2, \ldots, Y_n]$, where X_i and Y_i are the elements of the Galois field and n is the length of the sequence. The correlation between $\mathbf{X}$ and $\mathbf{Y}$ is modeled using a q-ary symmetric channel. The q-ary symmetric channel is a generalization of the binary symmetric channel. The input and the output alphabets both have q elements, and the probability that a symbol is received correctly is given by

$$P(Y = X) = (1 - p), \tag{6.1.1}$$

where $p < \frac{(q-1)}{q}$. If a symbol is corrupted, the $(q - 1)$ alternative symbols are all equally probable making

$$P(Y = a | a \neq X) = \frac{p}{q-1}. \tag{6.1.2}$$

When $q = 2$, we obtain a binary symmetric channel. The correlation coefficient, ρ is defined by

$$\rho = \frac{cov(\mathbf{X}, \mathbf{Y})}{\sigma_X \sigma_Y} \tag{6.1.3}$$

where

$$\begin{aligned} cov(\mathbf{X}, \mathbf{Y}) &= E[(\mathbf{X} - E(\mathbf{X}))(\mathbf{Y} - E(\mathbf{Y}))] && (6.1.4) \\ \sigma_{\mathbf{X}} &= \sqrt{E(\mathbf{X}^2) - E(\mathbf{X})^2} && (6.1.5) \\ \sigma_{\mathbf{Y}} &= \sqrt{E(\mathbf{Y}^2) - E(\mathbf{Y})^2}. && (6.1.6) \end{aligned}$$

The compressed version of $\mathbf{X}$ is the input to the channel. The source $\mathbf{Y}$ is assumed to be available at the decoder as “side information”. The source $\mathbf{Y}$ is available losslessly at the decoder. The rate used for $\mathbf{Y}$ is its entropy $R_2 = H(\mathbf{Y})$, therefore the theoretical limit for lossless compression of $\mathbf{X}$ is given by [1]

$$nR_1 \geqslant nH(X_i/Y_i) = nH(p) \tag{6.1.7}$$

$$= n \left((-p) \log_2 \frac{p}{q-1} - (1-p) \log_2 (1-p) \right). \tag{6.1.8}$$

This compression results in mapping a sequence of n input symbols into $(n - k)$ syndrome symbols resulting in a compression ratio of $n : (n - k)$, known as the “Wyner’s scheme” [24].

6.1.1 Encoding and Decoding with Non-Binary LDPC Codes

The source X can be encoded by multiplying with the parity-check matrix H, resulting in a compressed X which is the syndrome S, of length $(n-k)$, i.e.,

$$\mathbf{S} = \mathbf{X}H^{\top} \tag{6.1.9}$$

In the biparitate graph, encoding can be viewed as the addition of all variable nodes connected to the same check node in the generalized Galois field. Consider an example of the following H matrix, the symbols of which are from GF(4) field.

$$\begin{bmatrix} X_0 & X_1 & X_2 & X_3 & X_4 & X_5 & X_6 \end{bmatrix} \begin{bmatrix} 3 & 0 & 0 & 2 \\ 0 & 2 & 1 & 0 \\ 0 & 0 & 3 & 1 \\ 2 & 2 & 0 & 0 \\ 1 & 0 & 0 & 3 \\ 1 & 3 & 0 & 0 \\ 3 & 0 & 0 & 3 \end{bmatrix} = \begin{bmatrix} S_0 & S_1 & S_2 & S_3 \end{bmatrix} \tag{6.1.10}$$

$$3X_0 + X_3 + X_4 + X_5 + X_6 = S_0 \tag{6.1.11}$$

$$X_1 + X_3 + X_5 = S_1 \tag{6.1.12}$$

$$X_1 + X_2 = S_2 \tag{6.1.13}$$

$$X_0 + X_2 + X_4 + X_6 = S_3 \tag{6.1.14}$$

The application of the channel codes for distributed source coding requires a modification in the decoding algorithm in that the syndrome, which represents the compressed information, is used along with side information by the decoder to decode the original source, X. Therefore, the horizontal step of the decoding algorithm has been modified to take into account the syndrome information.

The decoding algorithm for source coding is similar to that of the channel coding. The elements of X are variable symbols and the elements of S are checks. The set of the indices that denote the variable symbols is given by $N(m) = n : H_{mn} \neq 0$. The set of the indices of the checks that depend on the variable symbol n is given by $M(n) = m : H_{mn} \neq 0$. Probabilities q^a_{mn} and r^a_{mn} are calculated for each nonzero entry in the H matrix. The probability that the symbol n of X is a, given the information obtained from checks other than check m is given by q^a_{mn}. The probability of check m being satisfied given that the symbol n of X is fixed at a, with the other variable symbols having a separable distribution given by q^a_{mn}, where $n \in N(m)\backslash n, a \in GF(q)$ is given by r^a_{mn}. The probabilities r^a_{mn} are updated using the syndrome, which is the compressed information. The inclusion of the syndrome in the horizontal step of the decoding algorithm makes the LDPC codes suitable for the implementation of distributed source coding.

The initialization is similar to that of the channel coding, the only difference being that the side information is initialized based on the likelihoods of the channel probabilities. Applying the codes to a binary symmetric channel (BSC) with a crossover probability p, we define $g^1_n = p^{1-r_j}(1-p)^{r_j}$ and $g^0_n = 1 - g^1_n$, where g^1_n is independent of n.

The channel probabilities are set to $f^a_n = \prod_{i=1}^{b} \ g^{a_i}_{n_i}$, for each variable symbol X_n being equal to a, for each $a \in GF(q)$, a_i is the binary representation of a. Each variable symbol X_n consists of b bits $X_{n_1}, ..., X_{n_b}$ and $g_{n_i}a_i$ is the likelihood of the i^th constituent bit (X_i) being equal to a, where $(a_1, ...a_b)$ is the binary representation of a.

The initialization is done on a q-ary symmetric channel wherein, the side information is initialized based on channel probabilities. The codes are applied to a q-ary symmetric channel (QSC) which has the channel probabilities

$$f_n^a = \frac{(1-p)}{p/(q-1)} \tag{6.1.15}$$

$$f_n^b = \frac{p/(q-1)}{(1-p)} \tag{6.1.16}$$

where $a \neq b$.

Horizontal step is the computation of the quantity r_{mn}^a given by

$$r_{mn}^a = \sum_{\mathbf{Y'}:Y_n'=a} P[S_m/\mathbf{Y'}] \times \prod_{j \,\epsilon\, N(m)\backslash n} q_{mj'}^{Y_j'}. \tag{6.1.17}$$

$P[S_m/\mathbf{Y'}]$ is 0 or 1 depending on whether or not $\mathbf{Y'}$ satisfies the check m. The quantity r_{mn}^a can be calculated efficiently by defining partial sums

$$\sigma_{mk} := \sum_{j:j \leqslant k} H_{mj} Y_j' \tag{6.1.18}$$

$$\rho_{mk} := \sum_{j:j \geqslant k} H_{mj} Y_j'. \tag{6.1.19}$$

$P[\sigma_{mk} = a]$ is computed for each $a \in GF(q)$ and for each $k \in N(m)$ using the probabilities q_{mn}^a [**?**]. If i, j are successive indices in *N(m)* and $j > i$ then

$$P[\sigma_{mj} = a] = \sum_{\{s,t:H_{mj}t+s=a\}} P[\sigma_{mj} = s] q_{mj}^t. \tag{6.1.20}$$

Along the same lines we can also calculate the distribution of each ρ_{mk}. Using σ_{mk} and ρ_{mk}, the quantity r_{mn}^a can be updated using

$$r_{mn}^a = P[(\sigma_{m(n-1)} + \rho_{m(n+1)}) = S_m - H_{mn}a] \tag{6.1.21}$$

$$\tag{6.1.22}$$

$$= \sum_{s,t:s+t=S_m-H_{mn}a} \Pr[\sigma_{m(n-1)} = s] \times \Pr[\rho_{m(n+1)} = t]. \tag{6.1.23}$$

In the non-binary LDPC channel decoding $s, t : s + t = S_m - H_{mn}a$ is solved such that the above equation is satisfied when $S_m = 0$. However, in distributed source coding S_m represents the m^{th} bit of the compressed information, S. Therefore $s, t : s + t = S_m - H_{mn}a$ is solved for the given value of S_m.

For channel coding

$$s, t : s + t + H_{mn}a = 0 \tag{6.1.24}$$

as $S_m = 0$, but for distributed source coding

$$s, t : s + t + H_{mn}a = S_m \tag{6.1.25}$$

where $S_m \neq 0$.

Vertical step involves updating the q^a_{mn} for each m and n using

$$q^a_{mn} = \alpha_{mn} f^a_n \prod_{j \epsilon M(n) \backslash m} r^a_{jn} \tag{6.1.26}$$

where α_{mn} is chosen such that $\sum_{a=1}^{q} q^a_{mn} = 1$.

Tentative decoding $\hat{\mathbf{X}}$ is performed using

$$\hat{X}_n = argmax_a f^a_n \prod_{j \epsilon M(n)} r^a_{jn}. \tag{6.1.27}$$

The decoding is repeated for a fixed number of preset iterations after which $\hat{\mathbf{X}}$ is estimated. If

$$H \times \hat{\mathbf{X}} = \mathbf{S} \tag{6.1.28}$$

the decoding halts, declaring $\hat{\mathbf{X}}$ as the original input sequence that has been compressed, otherwise the decoding repeats. A failure is declared if the decoded sequence is not valid even after a maximum number of iterations.

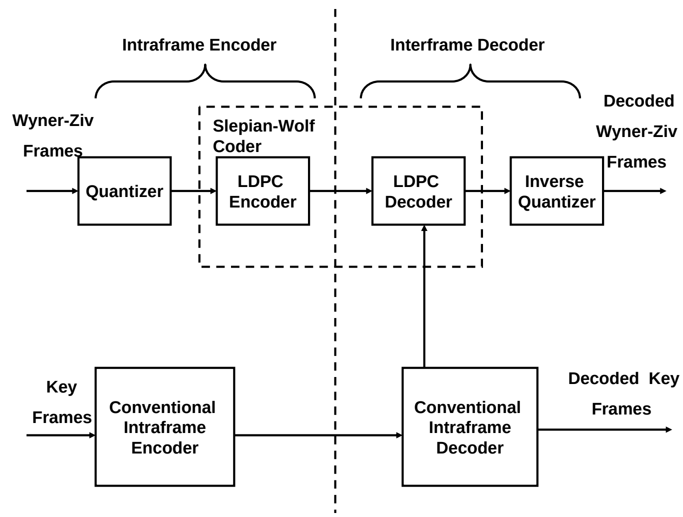

Figure 6.1: Wyner-Ziv video codec using non-binary LDPC codes.

6.2 Distributed Video Coding using Non-Binary LDPC Codes

The conventional video compression systems allow the encoder to perform predictive coding by exploiting the statistics of successive frames. The Wyner-Ziv Theorem on source coding with side information available only at the decoder suggests that an asymmetric video codec, where individual frames are encoded separately, but decoded conditionally (given temporally adjacent frames) could achieve similar efficiency.

The non-binary scheme developed is proposed as a Wyner-Ziv coding technique for video. Video frames, which are highly correlated lend themselves well for transmission using distributed source coding. On conversion of video frames into binary sequences, it was observed that the correlation between the video frames decreased drastically thereby rending the distributed source coding using binary LDPC codes unsuitable for video transmission. The Wyner-Ziv video codec using non-binary LDPC codes is shown in figure 6.1 . The video sequence is first partitioned into even and odd frames. The odd frames are transmitted using conventional intraframe coding, and the even frames are Wyner-Ziv coded. The odd frames are used as side-information at the decoder to decode the Wyner-Ziv encoded frames.

6.2.1 Quantizer

A uniform quantizer with a step size of 16 is used. Each quantized pixel value of the frame is represented by two non-binary symbols.

6.2.2 Conventional Encoder and Decoder

The odd frames serving as key frames are encoded and decoded using any conventional video codec such as an $8X8$ discrete cosine transform (DCT) codec.

6.2.3 Slepian-Wolf Coder

The non-binary LDPC encoder and decoder with the virtual correlation channel constitutes the Slepian-Wolf coder. The encoder compresses the quantized input vector and generates the syndrome. The syndrome and the correlation between the two frames, after their conversion into non-binary symbols, are transmitted through a lossless channel. The decoder uses the side information frame, which is decoded using a conventional intraframe decoder, the channel likelihoods and the syndrome information to decode the original quantized vector that has been compressed.

6.2.4 Inverse Quantizer

The output of the Slepian-Wolf coder which is a sequence of non-binary symbols are inverse quantized to reconstruct the frame.

Since the channel is lossless, distortion is caused only by the quantization error. The quantization error is represented as the mean square error.

Chapter 7

Experimental Results

7.1 Distributed Source Coding using Non-Binary LDPC Codes

Non-binary LDPC codes for distributed source coding were simulated over GF(4) field. Two correlated sources, each consisting of 1000 symbols each were used. One of the sources was compressed to half of its original size. The correlation between the two sources was modeled using a virtual q-ary symmetric channel. Simulations were carried out for different correlation coefficients between the two sources. Figure 7.1 and Figure 7.2 show the simulation results over GF(4) with rates 1/2 and 3/4 respectively. Bit error rate becomes negligible for a correlation coefficient greater than 0.8 for rate 1/2 and greater than or equal to 0.7 for rate 3/4.

The comparison results of the performance of LDPC codes over fields GF(2) and GF(4) with compression rates 1/2 and 3/4 are shown in Figure 7.3 and Figure 7.4 respectively. The two sources, each consists of 1000 symbols for non-binary LDPC codes and equivalent 2000 bits for binary LDPC codes. Binary LDPC codes are applied over a binary symmetric channel. It is clearly seen that the non-binary scheme outperforms the binary scheme.

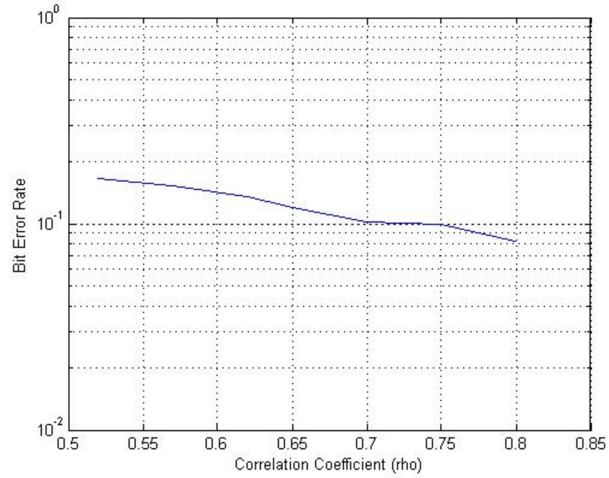

Figure 7.1: Simulation result for distributed source coding using non-binary LDPC codes with rate 1/2 over GF(4). The bit error rate becomes negligible for a correlation coefficient greater than or equal to 0.80.

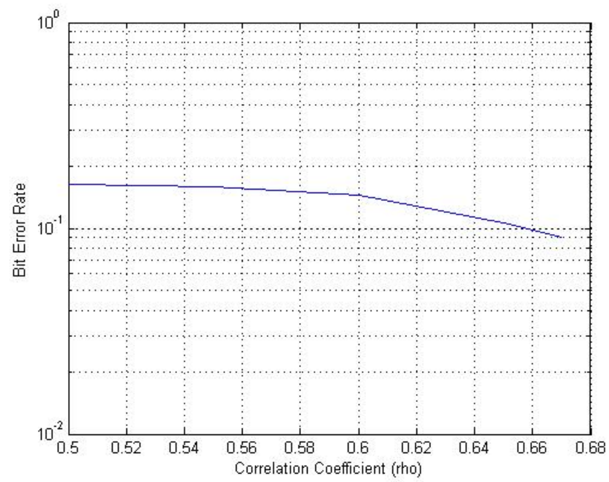

Figure 7.2: Simulation result for distributed source coding using non-binary LDPC codes over GF (4) with rate 3/4. The bit error rate becomes negligible for a correlation coefficient greater than or equal to 0.65.

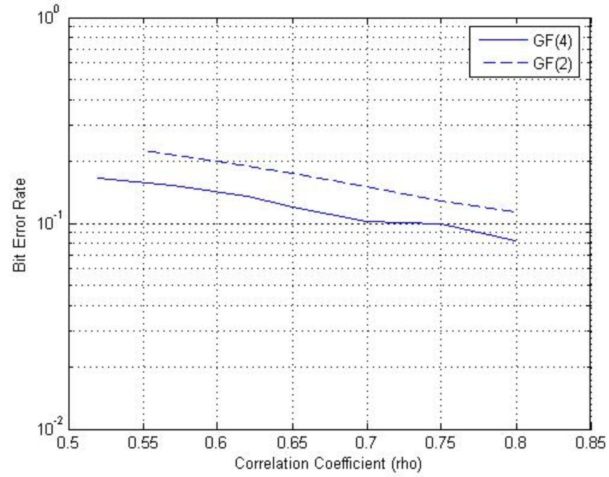

Figure 7.3: Comparison of performance of distributed source coding using LDPC codes over GF(2) (dashed curve) and GF(4) (solid curve) fields with rate 1/2.

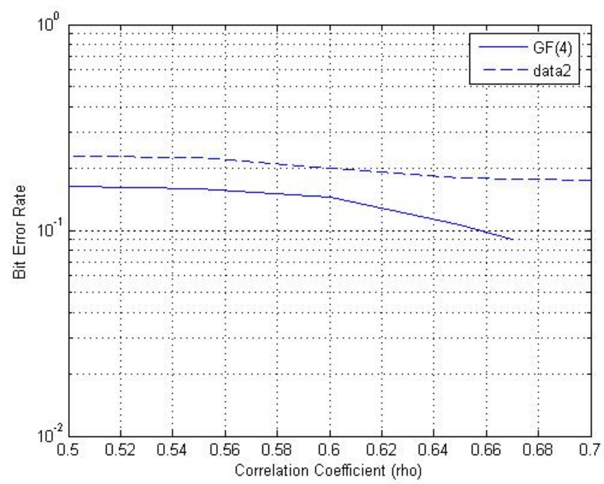

Figure 7.4: Comparison of performance of distributed source coding using LDPC codes over GF(2) (dashed curve) and GF(4) (solid curve) fields with rate 3/4.

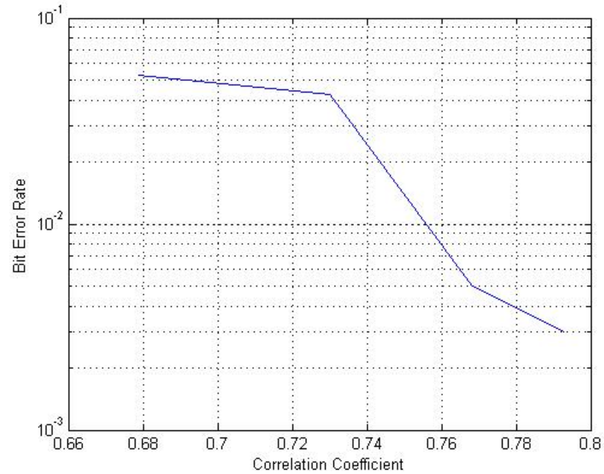

Figure 7.5: Simulation result for distributed video coding using non-binary LDPC codes with rate 1/2. The bit error rate becomes negligible for correlation coefficients greater than 0.90.

7.2 Distributed Video Coding using Non-Binary LDPC Codes

The non-binary distributed source coding scheme, when applied to highly correlated video, produced the original frame with a negligible probability of error. The plots of distributed video coding for different correlation coefficients and their corresponding BERs, with rates 1/2 and 3/4 are shown in figure 7.5 and figure 7.6 respectively.

From the simulation results it is clearly seen that highly correlated video frames can be recovered using the non-binary LDPC codes developed. BER becomes negligible for correlation coefficients greater than 0.9 and 0.7 for rates 1/2 and 3/4 respectively.

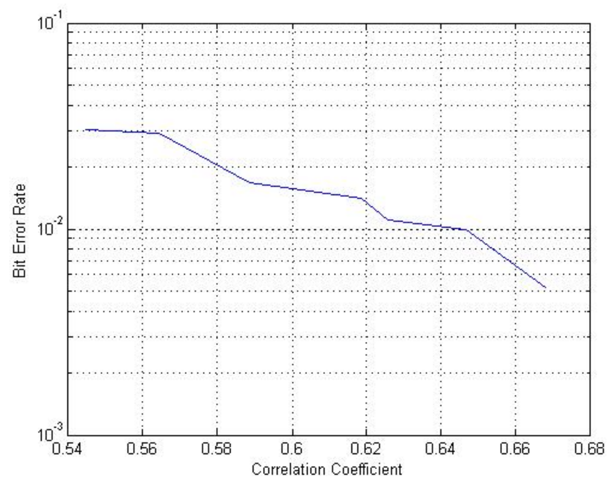

Figure 7.6: Simulation result for distributed video coding using non-binary LDPC codes with rate 3/4. The bit error rate becomes negligible for correlation coefficients greater than 0.70.

Chapter 8

Conclusions

In this thesis, it is shown that the non-binary low-density parity-check (LDPC) codes can be used to compress correlated non-binary sources. The correlation between the two sources is modeled using a virtual q-ary symmetric channel. The scheme is implemented using correlation as a channel and applying the syndrome concept. The theoretical model for distributed source coding was presented and implemented in MATLAB.

It is observed from the results that binary LDPC in the probability domain can be used to implement distributed source coding. Also, for the compression of non-binary sources, non-binary LDPC codes can be used, which avoids the reduction in correlation when converted its equivalent binary form. The results demonstrate that the non-binary scheme outperforms the binary version. From the plots, it can be observed that the compressed source can be recovered without any loss, for correlation coefficients greater than 0.9.

A new Wyner-Ziv codec using non-binary LDPC codes is proposed. This scheme can be extended for the compression of video frames as they are highly correlated. It is observed from the simulations that video frames can be reconstructed with negligible error when the correlation coefficient is greater than 0.9. This can be used for practical implementation of distributed source coding in video communication and sensor networks. Since the computational complexity of non-binary scheme increases exponentially, reduction techniques need to be investigated.

REFERENCES

REFERENCES

[1] D. Slepian and J. Wolf, “Noiseless coding of correlated information sources,” *IEEE Trans. Inform. Theory*, vol. 19, pp. 471–480, July 1973.

[2] A. Wyner and J. Ziv, “The rate distortion function for source coding with side information at the decoder,” *IEEE Transactions on Information Theory*, vol. 22, pp. 1–10, January 1976.

[3] Bernd Girod, Anne Margot Aaron, Shantanu Rane, David Rebollo-Monedero, “Distributed video coding,” *Proceedings of the IEEE*, vol. 93, pp. 71–83, January 2005.

[4] M. C. Davey and D. Mackay, “Low density parity check codes using GF(q),” *IEEE Commum. Lett.*, vol. 2, pp. 165–167, 1998.

[5] Z. A. D. Liveris and C. N. Georghiades, “Compression of binary sources with side information at the decoder using LDPC codes,” *IEEE Commum. Lett*., vol. 6, pp. 440–442, 1999.

[6] A. Wyner, “On source coding with side-information at the decoder,” *IEEE Ttransactions on Information Theory*, vol. 21, pp. 294–300, May 1975.

[7] “The rate distortion function for source coding with side information at the decoder ii: general sources,” *Inf. Control*, vol. 38, pp. 60–80, July 1978.

[8] J. Y. Wang and Y. Zhang, “Distributed video coding,” *Video Processing and communications*, 2002.

[9] G. B. T. Wiegand, G. Sullivan and A. Luthra, “Overview of the h.264/avc Video coding standard,” *IEEE Transactions on Video on Circuit Systems and Video Technology*, vol. 13, pp. 560–576, July 2003.

[10] J. C. S. Pradhan and K. Ramchandran, “Duality between source coding and channel coding and its extension to the sidde information case,” *IEEE Transactions on Information Theory*, vol. 49, pp. 1181–1203, May 2003.

[11] R. Zamir, “The rate-loss in the wyner-ziv problem,” *IEEE Transactions on Information Theory*, vol. 42, pp. 2073–2084, November 1996.

[12] A. Aaron, R. Zhang, and B. Girod, “Wyner-ziv coding of motion video,” *Asilomar Conf. Signals, Systems and Computers*, 2002.

[13] E. S. A. Aaron and B. Girod, “Toward practical wyner-ziv coding of video,” *IEEE Int. Conf. Image Processing*, 2003.

[14] M. Gastpar, P. L. Dragotti, and M. Vetterli, "The distributed karhunen-loeve transform," *IEEE Int. Workshop Multimedia Signal Processing*, pp. 57–60, 2002.

[15] M. Gastpar, P. L. Dragotti, and M. Vetterli "Karhunen-Loeve transforms, The distributed, partial and conditional," *IEEE Data Compression Conference*, pp. 283–292, 2003.

[16] R. Puri and K. Ramchandran, "Prism: a new robust video coding architecture based on distributed compression principles," *Allerton Conf. Communication, Control and Computing*, 2002.

[17] "Prism: an uplink-friendly multimedia coding paradigm," *International Conf. Acoustics, Speech, and Signal Processing*, 2002.

[18] "Prism: A reversed multimedia coding paradigm," *IEEE Int. Conf. Image Processing*, 2003.

[19] Bernd Girod, Anne Margot Aaron, David Rebollo-Monedero, "Transforms for high-rate distributed source coding," *Asilomar Conf. Signals, Systems and Computers*, 2003.

[20] R. Gallager, "Low density parity check codes," Ph.D. dissertation, MIT, Cambridge, Mass., September 1960.

[21] D. Mackay and R. Neal, "Near shannon limit performance of low density parity check codes," in *Electron. Lett.*, 1997.

[22] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of plausible Inference. Morgan Kaufmann, 1988.

[23] D. Mackay, "Good error correcting codes based on very sparse matrices," *IEEE Transactions on Information Theory*, vol. 45, no. 2, pp. 399–431.

[24] S. R. Zamir and U. Erez, "Nested linear/lattice codes for structured multiterminal binning," *IEEE Transactions on Information Theory*, vol. IT-48, 2002.