Slepian-Wolf source coding using LDPC codes for free viewpoint television

Fangping Ye

UNIVERSITE BRETAGNE LOIRE / MATHSTIC

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

THÈSE DE DOCTORAT DE

L'ÉCOLE NATIONALE SUPÉRIEURE MINES-TÉLÉCOM ATLANTIQUE BRETAGNE PAYS DE LA LOIRE - IMT ATLANTIQUE

COMUE UNIVERSITÉ BRETAGNE LOIRE

ECOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies de l'Information et de la Communication
Spécialité : Télécommunications

Par

Fangping YE

Slepian-Wolf source coding using LDPC codes for Free Viewpoint Television

Thèse présentée et soutenue à Brest, le 2 Décembre 2019
Unité de recherche : UMR CNRS 6285 Lab-STICC
Thèse N° : 2019IMTA0162

Rapporteurs avant soutenance :

Jérôme Lacan	Professeur, ISAE
Anissa Mokraoui	Professeur, Université Paris 13

Composition du Jury :

Présidente :	Aline Roumy	Chargée de recherche HDR, INRIA
Examinateurs :	Miodrag Mihaljević	Professeur, Mathematical Institute SANU, Serbia
	Cyrille Siclet	Maître de conférences, GIPSA-lab
	Jérôme Lacan	Professeur, ISAE
	Anissa Mokraoui	Professeur, Université Paris 13
Dir. de thèse :	Karine Amis	Professeur, IMT Atlantique
Encadrante :	Elsa Dupraz	Maître de conférences, IMT Atlantique

Acknowledgments

First of all, I would like to express my sincere gratitude to my supervisor Karine AMIS and Elsa DUPRAZ for their careful guidance and consistent support during my thesis. Their rigorous and precise scientific attitude taught me a lot, working with them improved my academic writing skills and also enhanced my comprehension of my research field.

Elsa is a talented, patient, motivated and productive professor, also she is a really kind and warm person. The discussion we held two times each week helped me resolve a lot of problems and progress my research. She can always point me out several possible directions to resolve the problem and correct my misunderstanding with her deep and large research comprehension. Her encouragement and support helped me step out from the confusion and failure. And finally we got good results and papers are written. Karine shows me strict academic writing skills and serious research attitude. Her suggestions and guidance are important to me. She is really patient and her comments in the report helped me correct all the mistakes. Our discussion together helped us better organize my research and also generate some new ideas.

Secondly, I would like to thank our project partners in Inria Rennes: Professor Aline Roumy and Professor Thomas Maugey, Post-doctoral Mai Quyen PHAM and PHD Navid Mahmoudian Bidgoli. The conferences which are organized in Rennes and in Brest helped us exchange our understanding of the project and make a lot of progress. Specially I would like to thank Navid, for the discussion we had and he always respond to my questions patiently and precisely. Also I want to thank Michel Kieffer, the invited professor of our project. He is also my professor during my M1. The brainstroming he held with other professors helped us to better apply our research results. With the effort of all the project members, we finally realized the total video codec.

Also, I really appreciate the time I passed in Serbia. I presented my results in Mathematical Institute SANU in Serbia with the help of Professor Velimir Ilic and Professor Miodrag Mihaljevic. The professors there gave me several advises and great suggestions. Velimir is a kind and humorous people, he helped me a lot when I was in Serbia.

I also want to thank all the group members in our department Signal & Communication, in IMT-Atlantique. Thanks to Professor Samir SAOUDI, Professor Christophe LAOT, Professor Francois-Xavier SOCHELEAU, Professor Abdeldjalil AISSA EL BEY,etc. I learned a lot when talking with them. I want to say a big thanks to all the PHD students in our department: Nicolas GROLLIER, Zahran HAJJI, Guillaume, Nacerredine, Alex, Mohamed,etc. It is with them our laboratory is always filled with fun and pleasure, also when I am trapped in those technique

and un-technique questions, they can always respond to me.

The weekend in Brest is never boring with my friends like Zhang Di, Tai Yupeng, Sun Miao, Li Huimin, Wang chen, Zhao yuan, Hu Denghui, Ke Weixin, etc. I really enjoy the meal with these big chefs, the activities we played together creat a lot of fun.

Lastly, I want to say thanks to my family members: YE Xiaohong, Huang Yuanxia, YE Fangyan. They give me a lot of support and care even they are in China. They always tell me that I still have them supporting me even if I loss. I appreciate for having this warm family.

List of Figures

List of Tables

Acronyms

BSC Binary Symmetric Channel

BER Bit Error Rate

CN Check Node

FTV Free Viewpoint Television

LDPC Low-Density Parity-Check

LDPCA Low-Density Parity-Check Accumulated

LLR Log-likelihood Ratio

LSB Least Significant Bit

MSB Most Significant Bit

PEG Progressive Edge-Growth

QC Quasi-Cyclic

SNR Signal-to-Noise Ratio

SW Slipian-Wolf

VN Variable Node

List of Publications

Journal Papers

Fangping Ye, Elsa Dupraz, Zeina Mheich, Karine Amis, Optimized Rate-Adaptive Protograph-Based LDPC Codes for Source Coding with Side Information, in IEEE Transactions on Communications, vol. 67, no. 6, pp. 3879-3889, June 2019

Conference Papers

Fangping Ye, Elsa Dupraz, Zeina Mheich, Karine Amis, Construction de Codes LDPC Compatibles en Rendement pour le Codage de Sources avec Information Adjacent(In French), Colloque GRETSI'19, Lille, France, April 2019

Fangping Ye, Zeina Mheich, Elsa Dupraz, Karine Amis, Optimized Short-Length Rate-Adaptive LDPC Codes for Slepian-Wolf Source Coding, International Conference on Telecommunication (ICT), Saint-Malo, France, June 2018

Fangping Ye, Elsa Dupraz, Karine Amis, Rate-adaptive LDPC code construction for Free Viewpoint Television, National Conference on Information Theory and Complex Systems (TINKOS), Belgrade, Serbia, June 2018

Contents

Chapter 1

Introduction

Contents

1.1 Source coding for Free Viewpoint Television

1.1.1 InterCom project

My thesis entitled "Slepian-Wolf source coding using LDPC codes for Free Viewpoint Television" was realized in the framework of a project Cominlabs called "InterCom" which held from November 2016 to October 2019. The objective of this project was to design compression solutions for Free Viewpoint Television (FTV). The different partners of the InterCom project were: Sirocco team (INRIA, Rennes), i4s team (INRIA Rennes), L2s (CentraleSupelec/Univ Paris-Sud, Paris), and Lab-STICC (IMT Atlantique, Brest).

FTV [2, 3, 4] is a system for watching videos in which the user can choose its viewpoint freely and change it instantaneously at any time. For example, when watching a football game, the user can select the viewpoint he wants, as if he could change his watching position, although the images remain 2D. This example is depicted in Figure 1.1, left. As shown in Figure 1.1, all the views of the video are stored on a server, and the users send requests to the server in order to get their desired views. When building a storage/transmission system for FTV, we want to consider a large dataset of videos, and a large number of users. In this context, data compression can help to reduce the amount of data to be stored on the server and transmitted to the users.

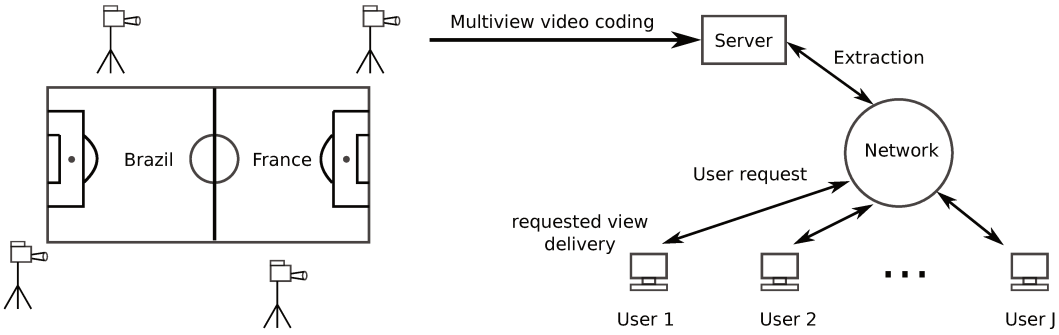

Figure 1.1: Multi-view video coding scheme

1.1.2 Source coding for FTV

We now describe the FTV system more formally. We assume that each user can request a random subset of the views. Previous views are still available in the user's memory when the current view is requested by the user. This can be represented as a problem of source coding with side information available at the decoder, where the current requested view is the source X and the previous requested views are represented by the side informations Y, see Figure 1.3. This scheme is known as Slepian-Wolf coding scheme [5, 6, 7, 8] for the lossless case, and as Wyner-Ziv source coding for the lossy case [9, 10, 11, 12, 13, 14]. Random requests means that the statistical correlation between the source and the side information varies depending on the previous user requests. Therefore, the coding rate must be adapted on the fly depending on the previous requests.

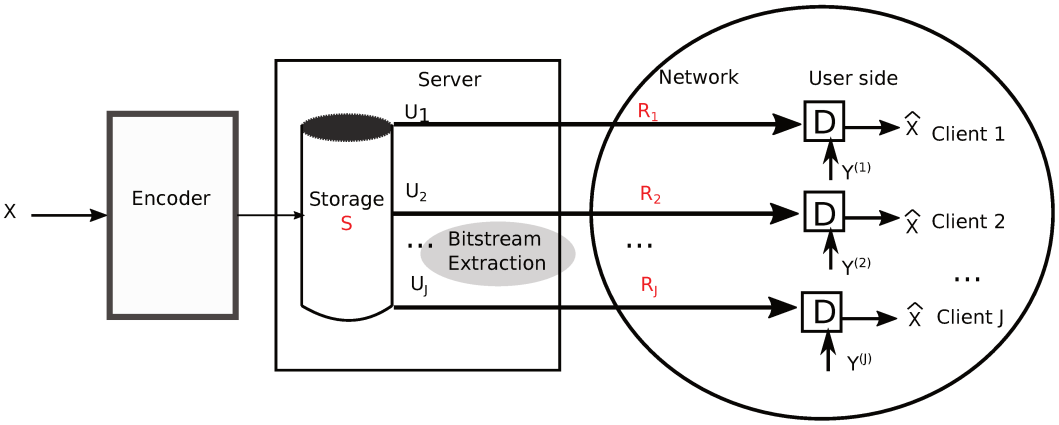

Figure 1.2: Source coding model for FTV

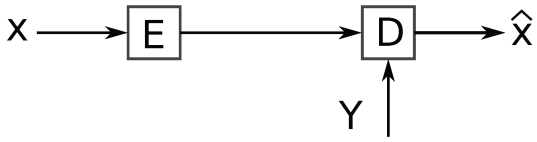

Figure 1.3: Slepian Wolf source coding

1.2 Limitations of standard compression systems to tackle FTV

In FTV, the views of the videos will first be encoded once together and then stored at the server. The users can request their desired views through the network, and the server will send them the corresponding encoded bits. In order to minimize the storage rate on the server, all the views should be compressed together. But since there is a large number of users, we do not want the server to decode the whole database before extracting the current requested views. This is why, in the InterCom project, it was proposed to use a rate-adaptive coding scheme [15]. Rate-adaptive schemes allow to extract only a part of the compressed data. The amount of data to extract will depend on the statistical relation between the previous received views and the current requested view. The problem is that classical compression methods (Huffman, Lempel Ziv, H264 [16], HEVC [17], etc) do not allow extraction of data without decompression of the whole dataset, and proposed solutions for FTV are largely based on these methods [18, 6, 19, 20].
On the opposite, it was shown that channel codes such as Low-Density Parity-Check (LDPC) codes allow to construct rate-adaptive schemes that permit data extraction. This is why the objective of this thesis is to use LDPC codes to design rate-adaptive coding schemes for FTV. We focus on the lossless coding part, since the lossy part (quantization, prediction, etc.) has been addressed by other members of the InterCom project.

1.3 Slepian-Wolf source coding with LDPC codes

Low-density parity-check (LDPC) codes were first proposed by Gallager [21]. LDPC codes were first used for channel coding [22, 23, 24, 25, 26, 27, 28]. These linear codes use a sparse binary parity-check matrix to decode the information received from the channel. It was proposed in [29, 30, 31, 32] to use them for source coding with side information, and in this thesis we aim to use them for FTV.
To construct good LDPC codes for source coding, we can combine different methods. The parity-check matrix of an LDPC code can be equivalently represented by a Tanner graph that connects variable nodes: the source bits, and check nodes: the parity check equations. A protograph [33] is a small Tanner Graph that represents connections between different types of variable nodes and check nodes. An LDPC matrix can be generated from a protograph by repeating the protograph structure, and by interleaving the connections between the variable nodes and the check nodes

of the corresponding types. The performance of an LDPC code depends on its underlying protograph, and density evolution [26, 34, 35, 36] enables to evaluate the decoder error probability depending on the protograph. Further, short cycles in the Tanner graph of the LDPC matrix can degrade the code performance. In order to limit the amount of short cycles, the progressive edge-growth (PEG) algorithm was proposed in [37] to construct the matrix from a given protograph.

In FTV, the statistical relation between source and side information varies depending on the previously requested views. So we should adapt the coding rate. Two traditional rate-adaptive code construction methods can be applied: using Low-Density Parity-Check Accumulated (LDPCA) codes [38] to decrease the rate from a high initial rate, or using Rateless codes [39] to increase the rate from a low initial rate. In this thesis, we aim to improve these existing rate-adaptive methods. It is hard to construct good performance LDPC matrices for very low rates and this is why Rateless method only allows to consider a limited range of rates. To solve this issue, we consider the solution of [40] that proposes to use LDPCA codes and rateless codes together from a middle initial rate. Unfortunately, this mixed solution is still penalized by the weakness of LDPCA, which shows a very bad performance if there exists too many short circles in the corresponding LDPC matrix. So we look for new code constructions which allows to optimize the code performance at all the rates and in particular to reduce the amount of short cycles in the LDPC matrices.

1.4 My contributions

In this thesis, as first contribution, we proposed two novel rate-adaptive code constructions for Slepian-Wolf source coding. The LDPCA construction combines several lines of the initial LDPC matrix in order to construct lower rate codes. LDPCA code construction does not leave the choice of line combinations (accumulated structure) and bad combinations can generate a lot of short cycles. As short cycles may highly degrade the code performance, we proposed a new method that limits the number of short cycles. In this method, we choose line combinations that add the least number of cycles. In this way, we generate a sequence of rate-adaptive codes that perform better than LDPCA. After this, we proposed a second method that relies on the optimization of the protographs at all rates. Since the protograph can help us to choose better lines combinations, we proposed a method that can optimize the protograph for all the considered rates. The two proposed methods show a better performance than LDPCA, especially for short codes (less than 1000 bits) that are particularly sensitive to short cycles.

As a second contribution, we worked on the integration of these two rate-adaptive code constructions into the FTV compression system. InterCom project members of INRIA Rennes worked on the compression pipeline for FTV, but they did not optimize the lossless part. So they provided us with files containing examples of source and side information realizations, and we applied our code constructions and decoding algorithms to these data. For this, we first had to determine and estimate a

statistical model between the source and side information. Second, the data we were provided with was on the form of symbols, and we had to adapt our solutions which work on binary vectors. To finish, we applied our coding solutions and evaluated the rate performance. The simulation results show that we can achieve transmission and storage rates close to the theoretical limits.

1.5 Organization of the manuscript

We now present the organization of this manuscript. In Chapter 2, we formally describe the FTV problem as a source coding problem and we provide the information theory results that were already obtained for this problem. The information theory results provide the limiting compression performance of FTV systems, and suggest design guideline for the practical schemes. Chapter 3 presents the preliminaries of standard and rate-adaptive LDPC codes. It shows the definitions, decoding methods and construction methods of LDPC codes. In Chapter 4, we describe the two rate-adaptive code constructions we propose for lossless source coding with side information. In Chapter 5, we apply our methods to real FTV data, and show the obtained simulation results. Chapter 6 presents the conclusions and perspectives.

Chapter 2

Information theory results

Contents

In this section, we describe information theory results for FTV. FTV can be presented as a source coding problem with side information available at the decoder, and the side information is the already received past views. In this thesis, we are mainly interested in lossless source coding. Figure 2.1 illustrates the source coding scheme that can be considered for FTV.

In this figure, X is the source which generates sequences $X^n = [X_1, \ldots, X_n]$ of n

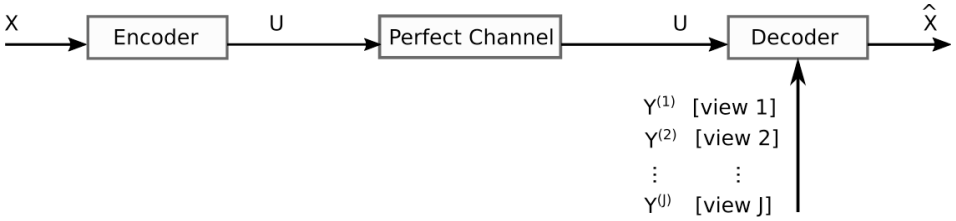

Figure 2.1: Lossless Source Coding for FTV

symbols. Each source symbol X_k takes its value in an alphabet $\{0, 1, \ldots, I-1\}$ and follows a probability distribution $P(X_k = i) = p_i, i \in \{0, 1, \ldots, I-1\}$. In source coding, the source X^n is compressed into a codeword U^m of length m, where $m \leq n$. The codeword U^m is then transmitted to the decoder. At the decoder, the side information Y generates sequences Y^n of n bits. The side information symbol Y_i takes also its value in the alphabet $\{0, 1, \ldots, I-1\}$ but follows a probability distribution which is different from the one of the source symbol X_i. The decoder reconstructs the source sequence $\hat{X}^n$ from the side information sequence Y^n and from the received codeword U^m. In this thesis we always suppose that the codeword U^m is perfectly received at the decoder. The source rate R is given by $R = \frac{m}{n}$.

In our case, there exist several available side informations $Y^{(j)}, j \in \{1, \ldots, J\}$ potentially available at the decoder, J is the number of previous received views. The statistical relations between X and each $Y^{(j)}$ are defined by the joint probability distributions $P(X, Y^{(j)})$. Only one of these potentially available side informations will be used for decoding, and the choice of the available side information depends on the previous views requested by the user.

2.1 Entropy definitions

Entropy [41, page13-18] is a measure of the information contained in a source. We present the definitions of entropy, conditional entropy, and the Binary Symmetric Channel (BSC) in this part. For a discrete source X, its entropy is denoted by $H(X)$ and defined as

$$H(X) = H(p) = -\mathbb{E}(\log_2 P(X)) = -\sum_{i=0}^{I-1} p_i \log_2(p_i) \tag{2.1}$$

For example, for a Bernoulli source, X takes values in $\{0, 1\}$. If $P(X = 0) = p$, Then

$$H(X) = -p \cdot \log_2(p) - (1-p) \cdot \log_2(1-p) \tag{2.2}$$

Specifically, if $P(X = 0) = 0.5$, then $H(X) = 1$ bit/source symbol.
The conditional entropy $H(X \mid Y)$ is defined as

$$\begin{aligned} H(X \mid Y) &= \sum_{y \in 0,\ldots,I-1} P(y) H(X \mid Y = y) \\ &= -\sum_{y \in 0,\ldots,I-1} P(y) \sum_{x \in 0,\ldots,I-1} P(x \mid y) \log_2 P(x \mid y) \end{aligned} \tag{2.3}$$

By the properties of conditional entropy [41, page 29] we have $H(X \mid Y) \leq H(X)$. We get $H(X \mid Y) = H(X)$ if and only if X and Y are independent.

Figure 2.2 shows us a binary symmetric channel, where X takes values in $\{0, 1\}$ and $P(Y = 1 \mid X = 0) = P(Y = 0 \mid X = 1) = p$.

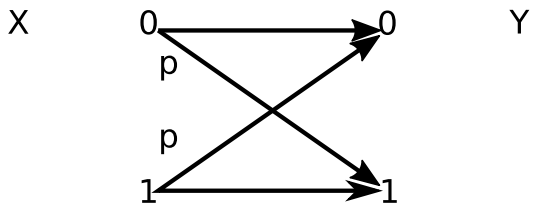

Figure 2.2: Binary Symmetric Channel

Specially, if $P(X = 0) = \frac{1}{2}$ and $P(Y = 1 \mid X = 0) = P(Y = 0 \mid X = 1) = p$, where p is the crossover probability, then $P(Y = 0) = \frac{1}{2}$. By applying these

expressions in (2.3), we have

$$H(X \mid Y) = -p \cdot \log_2(p) - (1-p) \cdot \log_2(1-p) \tag{2.4}$$

2.2 Theoretical results

In this section, we provide the minimum achievable rates for lossless source coding without side information, lossless source coding with side information, and source coding for FTV.

2.2.1 Lossless source coding without side information

We first consider the case where no side information is available at the decoder. In FTV, this corresponds to the case where we do not use the previously received views as side information at the decoder.

Figure 2.3 shows us the lossless source coding scheme without side information. In

Figure 2.3: Lossless Source Coding

this scheme, the error probability is defined as

$$P_e^n = P(X^n \neq \hat{X}^n) \tag{2.5}$$

It represents whether the source X^n can be perfectly reconstructed at the decoder or not. If one can construct a coding scheme such that $\lim_{n \to +\infty} P(X^n \neq \hat{X}^n) = 0$, then lossless decoding can be achieved.

The source coding theorem [41, page 112] tells us that a rate R is achievable if and only if

$$R \geq H(X) \tag{2.6}$$

where $H(X)$ is the entropy of the source X.

Example 2.2.1 For a binary source X with $P(X = 0) = 0.4$, we have

$$\begin{aligned} H(X) &= -0.4 \cdot \log_2(0.4) - (1-0.4) \cdot \log_2(1-0.4) \\ &= 0.971 \end{aligned} \tag{2.7}$$

So the minimum lossless source coding rate $R \geq 0.971$ bits/symbol.

Figure 2.4: Lossless Source Coding with side information

2.2.2 Lossless source coding with side information

In FTV, the already received views can be used as side information at the decoder. When there is only one possible side information Y, this can be represented as the Slepian-Wolf [42] source coding problem.
Figure 2.4 represents Slepian-Wolf source coding. The Slepian-Wolf source coding theorem [42] shows that a rate R is achievable if and only if:

$$R \geq H(X \mid Y) \tag{2.8}$$

Since $H(X \mid Y) \leq H(X)$, the side information at the decoder can help to decrease the source coding rate compared to source coding without side information. When the side information Y is available at the decoder, the minimum rate R will depend on the statistical relation between X and Y.

Example 2.2.2 In BSC for a source $X \in \{0,1\}$ with $P(X = 0) = 0.4$, and crossover probability $p = 0.1$ we have

$$H(X \mid Y) = 0.4585 \tag{2.9}$$

So the minimum lossless source coding rate is $R \geq 0.4585$ bits/symbol. Compared to the example 2.2.1, $0.4585 < 0.971$, it shows that the source coding with side information can help to reduce the minimum source rate.

In FTV, several different side informations can be available at the decoder.

2.2.3 Lossless source coding for FTV

For FTV, as described in Figure 2.1, several views $Y^{(j)}, j \in \{1, \ldots, J\}$ can be potentially available at the decoder. Only one of these potentially available side information will be used for decoding, and the available side information depends on the previous views requested by the user. For instance, $Y^{(1)}$ can be the prediction of X based on the previous received views received by user 1, and $Y^{(1)}$ will serve as side information for the transmission of X to user 1. We suppose that the transmissions of X to different viewpoints are independent. Figure 2.5 presents the transmission of sending a single source X, with several clients request for this view X using different SI $Y^{(1)}, \ldots, Y^{(J)}$ by network. S defines the storage rate on the server. R_j defines the rate needed to transmit source X to user j with side information $Y^{(j)}$.
For FTV, we are interested in the achievable rate-storage region which is a set of

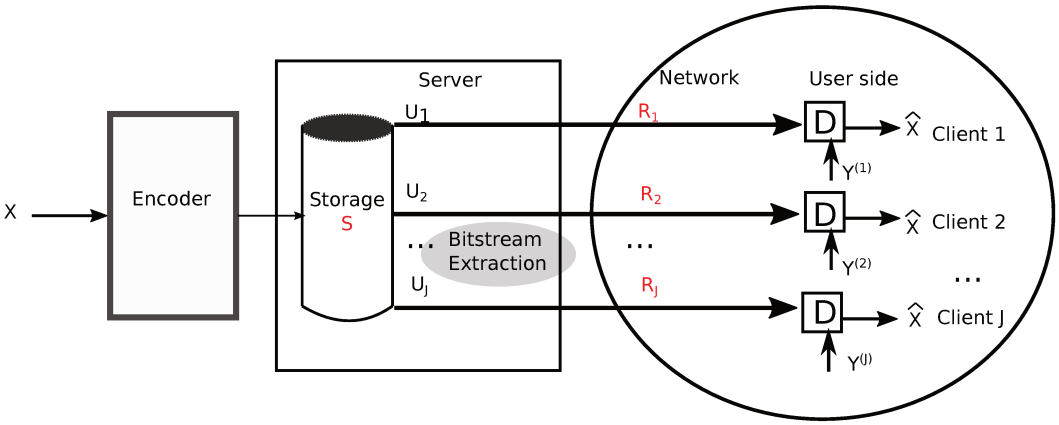

Figure 2.5: Storage and Transmission in FTV

achievable rate-storage pairs $(R_1, R_2, \ldots, R_J, S)$. According to (2.8) we would like to achieve transmission rates

$$R_j \geq H(X \mid Y^{(j)}), j \in \{1, 2, \ldots, J\} \tag{2.10}$$

Two methods to achieve these rates can be applied [15, 43].

Method 1 [6] **(Exhaustive storage)** We store one different codeword for each possible side information. Then the storage rate is a sum of all the transmission rates,

$$S = \sum_{j=1}^{J} R_j \geq \sum_{j=1}^{J} H(X \mid Y^{(j)}) \tag{2.11}$$

Method 2 [15, 43] **(Incremental storage)** We construct one unique incremental codeword. For user j, the serve extracts a subpart of rate $H(X \mid Y^{(j)})$ of the codeword. So we have

$$S \geq \max_{j=1,\ldots,J} H(X \mid Y^{(j)}) \tag{2.12}$$

It is easy to show that

$$\sum_{j=1}^{J} H(X \mid Y^{(j)}) \geq \max_{j=1,\ldots,J} H(X \mid Y^{(j)}) \tag{2.13}$$

As a result, method 2 allows to reduce the storage rate.
For example, consider a Binary Symmetric Channel with $P(X = 0) = \frac{1}{2}$ and $P(Y^{(j)} = 1 \mid X = 0) = P(Y^{(j)} = 0 \mid X = 1) = p_j, j = 1, 2, \ldots, J$. By applying (2.4), we can calculate the storage rate of method 2 as follows

$$\begin{aligned} S &\geq \max_{j=1,\ldots,J} H(X \mid Y^{(j)}) \\ &\geq \max_{j=1,\ldots,J} H(p_j) \end{aligned} \tag{2.14}$$

Assuming that in binary case $p_j \leq 0.5$, with the properties of the entropy function, we have:

$$S \geq H(\max_{j=1,\ldots,J} p_j) \tag{2.15}$$

Example 2.2.3 In BSC for a source $X \in \{0,1\}$ with $P(X=0) = 0.4$, three different side informations $Y^{(j)}, j = 1,2,3$ which follow the crossover probability $P(Y^{(j)} = 1 \mid X = 0) = P(Y^{(j)} = 0 \mid X = 1) = p_j, p_j = 0.05, 0.1, 0.2$ are available at the decoder. Table 2.1 provides the minimum rates for each possible Y_j. The minimum storage rate is: $S \geq 0.7033$ bits/symbol.

	$Y^{(1)}$	$Y^{(2)}$	$Y^{(3)}$
p_j	0.05	0.1	0.2
R_j	0.2808	0.4585	0.7033

Table 2.1: The minimum transmission rate with different side informations available at the decoder

2.3 Conclusion

In this section, we presented information theory results for FTV. These information theory results suggest that we use an incremental coding scheme in order to build a practical scheme for FTV. We know that LDPC codes can be used to construct Slepian-Wolf source coding schemes, including rate-adaptive ones. This is why we consider them in the following.

Chapter 3

Low Density Parity Check codes

Contents

As described in Chapter 2, the lossless source coding part of FTV can be seen as a Slepian-Wolf [42] source coding problem, and LDPC codes are often used as practical codes for this problem [29] [44] [45] [31]. LDPC [21] codes were invented by Gallager and published in his thesis in 1960. LDPC codes are a class of linear block

codes and they were initially introduced for channel coding. They show very good performances in channel coding and they are known to approach the channel capacity. LDPC codes can also be applied to Slepian-Wolf source coding, as described in this Chapter.

In this section, we will first introduce the definition of LDPC codes, and consider protograph [33, 46]-based LDPC code construction. Three decoding methods (Bit-flipping decoder [47], Gallager-A/B decoder [26], Sum-product decoder [48]) are described as the most used decoding methods for LDPC codes. We then introduce Density evolution [34] to evaluate the asympotic performance of LDPC codes. We also present the Progressive Edge Growth (PEG) [37] algorithm that allows to construct good finite-length codes.

3.1 LDPC codes

3.1.1 Definition of LDPC codes

LDPC codes can be represented by a parity check matrix H of dimension $m \times n$, where $m < n$. In this thesis we mainly consider binary LDPC codes, which means that the components of H are either 0 or 1. In LDPC codes, the matrix H is sparse in the sense that it contains only a few $1's$ which will allow to reduce the decoding complexity. In Slepian-Wolf source coding, a codeword U^m is constructed from the source vector X^n and from the parity check matrix as

$$U^m = X^n \cdot H^T \tag{3.1}$$

The source coding rate is $R = \frac{m}{n}$.
The LDPC parity check matrix H can be represented by a Tanner Graph [49]. The n columns of this matrix are represented by variable nodes (VN), and the m rows are represented by check nodes (CN). If $H_{j,i} = 1$, there is an edge between the j-th CN and the i-th VN in the Tanner Graph. The number of connections of a given VN x_i is called the VN degree and denoted by d_{v_i}. The set of CNs connected to VN x_i is denoted by $\mathcal{C}_i$. The number of connections of a given CN u_j is called the CN degree and denoted by d_{c_j}. The set of VNs connected to CN u_j is denoted by $\mathcal{V}_j$.
For example, for a code with $n = 8$, $m = 4$, a possible matrix H is given by:

$$H = \begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \end{array} \begin{array}{c} \begin{array}{cccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{pmatrix} \end{array} \tag{3.2}$$

and the corresponding Tanner Graph is shown in Figure 3.1. In this example, each VN $x_i, i \in \{1, \cdots, 8\}$ is connected to 2 CN $u_j, j \in \{1, \cdots, 4\}$, and $d_{v_i} = 2$. Each CN is connected to 4 VN, which gives $d_{c_j} = 4$.
An LDPC code is regular if the variable node degree d_{v_i} is the same for each VN

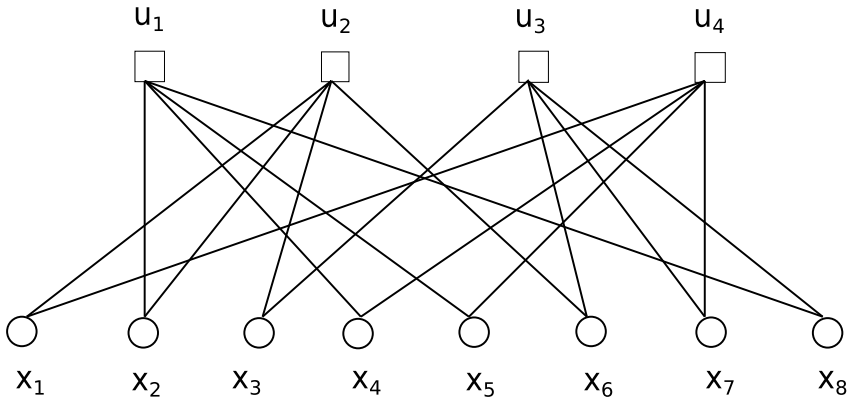

Figure 3.1: Tanner Graph of H with $n = 8$, $m = 4$

and the check node degree d_{c_j} is the same for each CN. For example, the matrix H in (3.2) represents a regular LDPC codes with degrees $d_v = 2, d_c = 4$. For irregular LDPC codes, d_v and d_c can vary from one node to another, and they can be described with a degree distribution pair $(\lambda(x), \rho(x))$:

$$\lambda(x) = \sum_{k=2}^{d_{v_{\max}}} \lambda_k x^{k-1} \qquad \rho(x) = \sum_{k=2}^{d_{c_{\max}}} \rho_k x^{k-1} \tag{3.3}$$

where $\sum_{k=2}^{d_{v_{\max}}} \lambda_k = 1$, $\sum_{k=2}^{d_{c_{\max}}} \rho_k = 1$, $d_{v_{\max}}$ is the maximum VN degree, $d_{c_{\max}}$ is the maximum CN degree. In these expressions, the fraction of edges belonging to VNs with $d_v = k$ is noted as λ_k, and the fraction of edges belonging to CNs with $d_c = k$ is noted as ρ_k. The rate R depends on the degree distributions $\lambda(x)$ and $\rho(x)$. It can be computed as follows:

$$R(\lambda, \rho) = \frac{n-m}{n} = 1 - \frac{m}{n} = 1 - \frac{\int_0^1 \rho(x)\,dx}{\int_0^1 \lambda(x)\,dx} \tag{3.4}$$

For example, for the regular code described in 3.1, with $d_v = 2$, $d_c = 4$, we have

$$\lambda(x) = x \qquad \rho(x) = x^3 \tag{3.5}$$

yielding to

$$R(\lambda, \rho) = 1 - \frac{\int_0^1 x^3 dx}{\int_0^1 x dx} = \frac{1}{2} \tag{3.6}$$

For an irregular code, a VN degree distribution given by

$$\begin{aligned} \lambda(x) = {} & 0.071428x + 0.230118x^2 + 0.079596x^9 + 0.147043x^{10} \\ & + 0.073821x^{48} + 0.397994x^{49} \end{aligned} \tag{3.7}$$

means that the fraction of an edge belonging to VNs with $d_v = 2$ is $\lambda_2 = 0.071428$, the fraction of an edge belonging to VNs with $d_v = 3$ is $\lambda_3 = 0.230118$, the fraction of an edge belonging to VNs with $d_v = 10$ is $\lambda_{10} = 0.079596$, etc.

If the CN degree distribution is given by

$$\rho(x) = x^{27} \tag{3.8}$$

then

$$R(\lambda, \rho) = 1 - \frac{\int_0^1 \lambda(x)dx}{\int_0^1 \rho(x)dx} = \frac{3}{4} \tag{3.9}$$

3.1.2 LDPC code construction from protographs

An alternative way to represent irregular LDPC codes is the use of protographs [33, 46]. Protographs allow for a precise control of the connections in the parity check matrix of the code, and lead to efficient Quasi-Cyclic (QC) parallel hardware implementations [50].
A protograph $\mathcal{S}$ is a small Tanner Graph of size $S_m \times S_n$ with $S_m/S_n = m/n = R$. Each row (respectively column) of $\mathcal{S}$ represents a type of CN (respectively of VN). The protograph $\mathcal{S}$ thus describes the number of connections between S_n different types of VNs and S_m different types of CNs. A parity check matrix H can be generated from a protograph $\mathcal{S}$ by repeating the protograph structure Z times such that $n = ZS_n$, and by interleaving the connections between the VNs and the CNs. The interleaving can be done by a PEG algorithm [37] that not only permits to satisfy the protograph constraints, but also to lower the number of short cycles that could severely degrade the decoding performance of the matrix H.

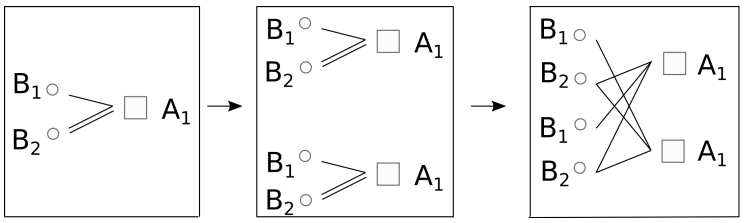

Figure 3.2: Construction of a parity check matrix H of size 2×4 (right picture) from a protograph $\mathcal{S}$ of size 1×2 (left picture)

Figure 3.2 gives an example of construction of a parity check matrix H from the protograph S defined by

$$\mathcal{S} = \begin{bmatrix} 1 & 2 \end{bmatrix} \tag{3.10}$$

Components of S represent the connections between one CN of type A_1 and two VNs of types B_1 and B_2. A parity check matrix H can be constructed from this protograph S, where

$$H = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix}. \tag{3.11}$$

The Tanner graph of protograph S is represented in Figure 3.2 (left part). In order to construct a parity check matrix H, the protograph is first duplicated $Z = 2$ times (middle part of Figure 3.2), and the edges are then interleaved (right part of Figure 3.2). In the final Tanner graph, one can verify that each VN of type B_1 is connected to one CN of type A_1, and each VN of type B_2 is connected to two CNs of type A_1.

The performance of a given parity check matrix H highly depends on its underlying protograph $\mathcal{S}$. The protograph optimization will be presented later in this section.

3.2 Decoding algorithms for LDPC codes

There exist several decoding methods for LDPC codes. These methods were invented in the context of channel coding, but they can also be applied to source coding with side information, as described in this section. Bit-flipping decoder, Gallager-A/B decoder and Sum-product decoder are among the most common decoders [49]. In this section we describe these three decoders.

3.2.1 Bit-flipping decoder

The rows of the LDPC matrix H represent parity check equations. The idea of the Bit-flipping [47] algorithm is to correct one by one the bits that are involved in the largest number of unsatisfied parity check equations.

Some definitions are given first:

- We denote by e_i the number of unsatisfied parity check equations associated to VN x_i.
- $\hat{x}_i \in \{-1, 1\}$ is the polar representation of x_i, $\hat{x}_i = -1$ corresponds to $x_i = 1$, $\hat{x}_i = 1$ corresponds to $x_i = 0$.
- The number of iterations is denoted by ℓ.
- The message from CN u_j to VN x_i ($i \in \mathcal{V}_j$) is denoted by $\Psi_c(j \to i)$.
- The message from VN x_i to CN u_j ($j \in \mathcal{C}_i$) is denoted by $\Psi_v(i \to j)$.
- The function that takes the decision on the value of x_i is denoted by Q_i.

The steps of the Bit-flipping are as follows:

1. Initialization: The value $\hat{x}_i$ of VNs $x_i, i \in \{1, \cdots, n\}$ are initialized with the side information bits y_i, that is $\hat{x}_i = 1 - 2y_i$. The counters e_i are set to zero. The messages from VNs to CNs are initialized as

$$\Psi_v^{(0)}(i \to j) = \hat{x}_i, \forall j \in \mathcal{C}_i \tag{3.12}$$

2. CN update: The messages $\Psi_c^{(\ell)}(j \to i)$ are calculated as:

$$\Psi_c^{(\ell)}(j \to i) = (1 - 2u_j) \prod_{i \in \mathcal{V}_j} \Psi_v^{(\ell)}(i \to j) \tag{3.13}$$

3. VN update: With (3.13), all the m parity equations are checked. Once a parity equation is not satisfied, the counters e_i of the associated VN are increased by one. It means that,

$$e_i^{(\ell)} = \sum_{j \in \mathcal{C}_i} \mathbb{1}\{\Psi_c^{(\ell)}(j \to i) = -1\}, \forall i \in \mathcal{V}_j \tag{3.14}$$

The message $\Psi_v^{(\ell)}(i \to j)$ of the VN with highest $e_i^{(\ell)}$ will then be corrected, $\forall j \in \mathcal{C}_i$,

$$\Psi_v^{(\ell+1)}(i \to j) = \begin{cases} -\hat{x}_i, & \text{if } e_i^{(\ell)} = \max\limits_{k \in \{1,\dots,n\}} e_k^{(\ell)} \\ \hat{x}_i, & \text{else} \end{cases} \tag{3.15}$$

4. Detection: The final value $\hat{x}_i$ is set to:

$$\hat{x}_i = \Psi_v^{(\ell+1)}(i \to j) \tag{3.16}$$

5. If all parity equations are satisfied (which means for $j = 1, \cdots, m$, $\Psi_c^{(\ell)}(j \to i) = 1$) or if the maximum number of iteration is reached, the decoding stops. If not, it goes back to step 2.

3.2.2 Gallager-A/B decoder

The idea of the Gallager-A/B [26] algorithm is also to correct the codeword bits depending on unsatisfied parity equations. The difference is that here we apply a majority voting operation at VNs, and consider only extrinsic messages at both VNs and CNs.

We use the same notation as in Section 3.2.1. The steps of the Gallager-A decoder are as follows.

1. Initialization: The message of VNs $x_i, i = 1, \dots, n$ are initialized with the side information y_i, $\hat{x}_i = 1 - 2y_i$. The messages are initialized as

$$\Psi_v^{(0)}(i \to j) = \hat{x}_i, \forall j \in \mathcal{C}_i \tag{3.17}$$

2. CN update: The parity check messages are given by

$$\Psi_c^{(\ell)}(j \to i) = (1 - 2u_j) \prod_{k \in \mathcal{V}_j, k \neq i} \Psi_v^{(\ell)}(k \to j) \tag{3.18}$$

3. VN update (majority voting): $\Psi_v^{(\ell+1)}(i \to j)$ is calculated from the initial message $\hat{x}_i$ and from the CN messages $\Psi_c^{(\ell)}(j \to i)$, as

$$\Psi_v^{(\ell+1)}(i \to j) = \begin{cases} -\hat{x}_i, & \text{if } \Psi_c^{(\ell)}(k \to i) = -\hat{x}_i, \forall k \in \mathcal{C}_i, k \neq j \\ \hat{x}_i, & \text{otherwise} \end{cases} \tag{3.19}$$

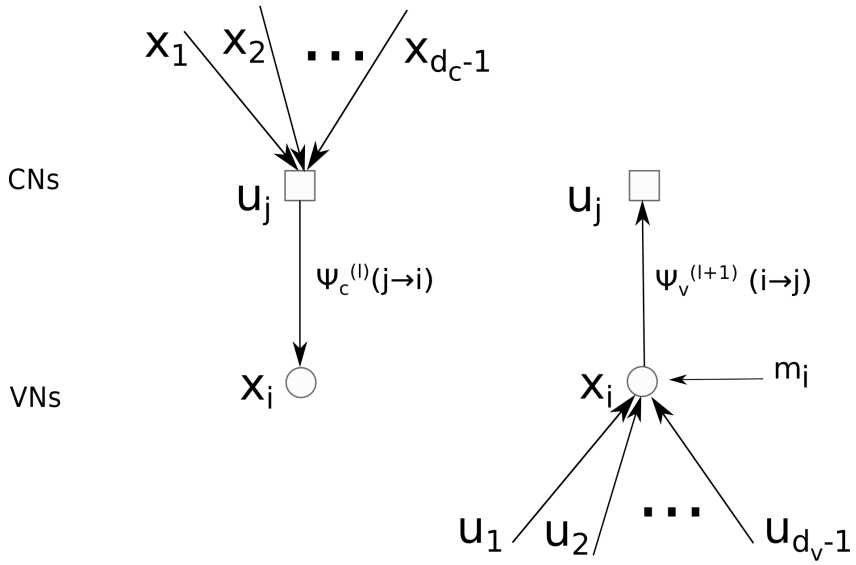

Figure 3.3: Message passing in Gallager-A/B and soft decision. Left figure shows parity check computation, right figure shows majority voting.

4. Detection: The final value of $\hat{x}_i$ is equal to $Q_i^{(\ell+1)}$ and

$$Q_i^{(\ell+1)} = \begin{cases} -\hat{x}_i, & \text{if } \Psi_c^{(\ell)}(j \to i) = -\hat{x}_i \ \ \forall j \in \mathcal{C}_i \\ \hat{x}_i, & \text{otherwise} \end{cases} \tag{3.20}$$

5. If all the parity equations are satisfied or if the maximum number of iterations is reached, the decoding stops. Otherwise, it goes back to step 2.

Gallager B is different from Gallager A at step 3 and step 4. At step 3, the Gallager B decoder sets the value of $\Psi_v^{(\ell+1)}\{i \to j\}$ to $-\hat{x}_i$ if and only if at least b incoming check messages are equal to $-\hat{x}_i$. The same is applied at step 4.

3.2.3 Sum-Product decoder

Sum-product [48] is also called belief-propagation or message-passing algorithm. In this algorithm, the exchanged messages are no longer -1 or 1, but Log-Likelihood Ratios (LLR). Therefore, before presenting the algorithm, we need to introduce additional notations. The messages are initialized as $m_i = \log \frac{P(x_i=0|y_i)}{P(x_i=1|y_i)}$. For instance for a BSC,

$$m_i = \log \frac{P\left(y_i \mid x_i = 0\right)}{P\left(y_i \mid x_i = 1\right)} = (1 - 2y_i) \cdot \log \frac{1-p}{p} \tag{3.21}$$

where p is the crossover probability and $p = P(y_i \neq x_i)$.

The Sum-Product decoding algorithm is then described by the following steps.

1. Initialization: We compute initial messages as $m_i = \log \frac{P(x_i=0|y_i)}{P(x_i=1|y_i)}$, and

$$\Psi_v^{(0)}(i \to j) = m_i, \forall j \in \mathcal{C}_i \tag{3.22}$$

2. CN update: The parity check messages are given by

$$\Psi_c^{(\ell)}(j \to i) = \log \frac{1 + (1-2u_j) \cdot \prod\limits_{k \in V_j \setminus i} \left(\tanh(\frac{\Psi_v^{(\ell)}(k \to j)}{2} \right)}{1 - (1-2u_j) \cdot \prod\limits_{k \in V_j \setminus i} \left(\tanh(\frac{\Psi_v^{(\ell)}(k \to j)}{2} \right)} \tag{3.23}$$

3. Majority voting: The updated VN messages are given by

$$\Psi_v^{(\ell+1)}(i \to j) = m_i + \sum_{k \in \mathcal{C}_i \setminus j} \Psi_c^{(\ell)}(k \to i) \tag{3.24}$$

4. The value of x_i is calculated from

$$Q_i^{(\ell+1)} = m_i + \sum_{k \in \mathcal{C}_i} \Psi_c^{(\ell)}(k \to i) \tag{3.25}$$

$$\hat{x}_i = \begin{cases} 1, & \text{if } Q_i^{(\ell+1)} \geq 0 \\ -1, & \text{if } Q_i^{(\ell+1)} < 0 \end{cases} \tag{3.26}$$

5. If all the parity equations are satisfied or if the maximum number of iterations is reached, the decoding stops. If not, it goes back to step 2.

3.3 Density evolution

Density evolution [34] is a method to evaluate the performance of LDPC codes. It is then used to compute a channel threshold from which the decoder can decode without error. It consists in computing the statistical distribution of messages exchanged during the iterative decoding process. More formally, if we denote by $P_e^{(\ell)}$ the codeword error probability after ℓ iterations, then for a BSC with crossover probability p, the threshold ε satisfies

$$\lim_{\ell \to +\infty} P_e^{(\ell)} = \lim_{\ell \to +\infty} P(\hat{X}^{(\ell)} \neq X) = 0, \qquad \forall p \leq \varepsilon \tag{3.27}$$

Let us describe steps of the density evolution algorithm as follows.

- In the following, we assume that the side information Y is generated from a BSC with crossover probability p. For the source X, we consider the all-zero assumption [26], that is $x_i = 0, \forall i$. The codeword U is such that $u_j = 0, \forall j$.
- For the considered decoding algorithm (Gallager-A/B, Sum-product) we estimate the statistical distribution of the messages $\Psi_c^{(\ell)}, \Psi_v^{(\ell)}$ for L decoder iterations.
- We estimate the codeword error probabilities $P_e^{(\ell)}$, and then compute the threshold from (3.27).

3.3.1 Density evolution for the Gallager-A decoder

We now describe into details the density evolution equations for the Gallager A decoder and for the sum-product decoder. For Gallager-A decoder, we calculated the messages $\Psi_c^{(\ell)}(j \to i)$ by equation (3.18), and $\Psi_v^{(\ell)}(i \to j)$ by equation (3.19). The Density evolution evaluates the performance of an ensemble of codes with the same CN and VN degree distributions. Here, we consider regular LDPC code with CN degree d_c and VN degree d_v. In order to evaluate the probability distributions of $\Psi_c^{(\ell)}$, $\Psi_v^{(\ell)}$, we define

- Probability distribution of $\Psi_c^{(\ell)}$: $q_\alpha^{(\ell)} = P(\Psi_c^{(\ell)}(j \to i) = \alpha)$, where $\alpha \in \{-1, 1\}$.
- Probability distribution of $\Psi_v^{(\ell)}$: $p_\alpha^{(\ell)} = P(\Psi_v^{(\ell)}(i \to j) = \alpha)$, where $\alpha \in \{-1, 1\}$.

Then, the probability distribution of $\Psi_c^{(\ell)}$ after applying equation (3.18) is [26].

$$q_{-1}^{(\ell)} = \frac{1}{2}\left(1 - \left(1 - 2p_{-1}^{(\ell-1)}\right)^{(d_c-1)}\right) \qquad q_1^{(\ell)} = \frac{1}{2}\left(1 + \left(1 - 2p_{-1}^{(\ell-1)}\right)^{(d_c-1)}\right) \tag{3.28}$$

Then with the majority voting operation in equation (3.19), we have

$$p_{-1}^{(\ell)} = p_1^{(0)}\left(q_{-1}^{(\ell)}\right)^{d_v-1} + p_{-1}^{(0)}\left(1 - q_1^{(\ell)}\right)^{d_v-1} \tag{3.29}$$

$$p_1^{(\ell)} = p_{-1}^{(0)}\left(q_1^{(\ell)}\right)^{d_v-1} + p_{-1}^{(0)}\left(1 - q_1^{(\ell)}\right)^{d_v-1} \tag{3.30}$$

The error probability is evaluated as $P_e^{(\ell)} = p_{-1}^{(\ell)}$, then perfect decoding is achieved if $\lim_{\ell \to +\infty} P_e^{(\ell)} = 0$. Equations (3.28) (3.29) (3.30) allow to compute $p_{-1}^{(\ell)}$ recursively. As $p_{-1}^{(0)} = p$, the error probability $P_e^{(\ell)}$ can be calculated as

$$\begin{aligned} P_e^{(\ell)} = p - p &\left[\frac{1 + \left(1 - 2P_e^{(\ell-1)}\right)^{d_c-1}}{2}\right]^{d_v-1} \\ &+ (1-p)\left[\frac{1 - \left(1 - 2P_e^{(\ell-1)}\right)^{d_c-1}}{2}\right]^{d_v-1} \end{aligned} \tag{3.31}$$

Since P_e is a increasing function of p, the highest p which still satisfies $\lim_{\ell \to +\infty} P_e^{(\ell)} = 0$ will be defined as the threshold ε.

3.3.2 Density evolution for the Sum-product decoder

For the sum-product decoder, we will also evaluate the probability distribution of $\Psi_c^{(\ell)}$ and $\Psi_v^{(\ell)}$, we define

- Probability distribution of $\Psi_c^{(\ell)}$: $q_\alpha^{(\ell)} = P(\Psi_c^{(\ell)}(j \to i) = \alpha)$, where $\alpha \in \{-\infty, \infty\}$.
- Probability distribution of $\Psi_v^{(\ell)}$: $p_\alpha^{(\ell)} = P(\Psi_v^{(\ell)}(i \to j) = \alpha)$, where $\alpha \in \{-\infty, \infty\}$.

For the Gallager-A decoder, the probability distributions of $\Psi_c^{(\ell)}$ and $\Psi_v^{(\ell)}$ are discrete. But for sum-product decoder, the probability distributions of $\Psi_c^{(\ell)}$ and $\Psi_v^{(\ell)}$ are continuous. Therefore, it is not possible to calculate analytically the probability distribution for each single value of α. This is why the Monte-Carlo [51] method is applied here.

The Monte-Carlo method consists in generating K CN and VN messages, in order to estimate their probability distribution. $M_{vn}[K]$ and $M_{cn}[K]$ will store the generated values of $\Psi_c^{(\ell)}$ and $\Psi_v^{(\ell)}$, and the K elements in these two vectors follow the probability distributions of $\Psi_c^{(\ell)}$ and $\Psi_v^{(\ell)}$. As before, we consider the all-zero codeword assumption for X, and a BSC for Y. Then the initial LLR is equal to $m_i = \log \frac{P(x_i=0|y_i)}{P(x_i=1|y_i)} = (1-2y_i) \log \frac{1-p}{p}$. The Monte Carlo density evolution algorithm steps are as follows.

- The value of $M_{vn}[K]$ are initialized with LLR information. $M_{vn}[k] = \pm m_k, k \in \{1, \cdots, K\}$ and $P(M_{vn}[k] = m_k) = p$, $P(M_{vn}[k] = -m_k) = 1 - p$.
- CN update: We apply the equation (3.23) to update the values of all the elements in $M_{cn}[K]$. For each element, we choose randomly $d_c - 1$ elements from $M_{vn}[K]$ and apply (3.23) over these elements.
- VN update: We apply the equation (3.24) to update the values of all the elements in $M_{vn}[K]$. For each element, we choose randomly $d_v - 1$ elements from $M_{cn}[K]$ and apply (3.24) over these elements.
- Repeat the CN update and VN update for L iterations. The final error probability is evaluated as

$$P_e^{(\ell)} = \frac{\{k : M_{vn}[k] < 0\}}{K} \tag{3.32}$$

The highest crossover probability p which satisfies $\lim_{\ell \to +\infty} P_e^{(\ell)} = 0$ will be defined as the threshold ε.

3.4 Protograph optimization

LDPC codes performance depend on the underlying protograph, and this is why it is necessary to optimize the protograph for a given rate. Here, we apply Differential Evolution [52] for protograph optimization.

3.4.1 Differential Evolution

Differential Evolution is a genetic optimization algorithm, which we now describe. We consider an optimization problem with D parameters represented by a vector x_D and a cost function $f(x_D)$ to minimize. To apply Differential Evolution, we first randomly generate an initial population of NP vectors $x_{i,j}^{(0)}, i \in \{1, \cdots, NP\}, j \in \{1, \cdots, D\}$. We usually set $5D \leq NP \leq 10D$. Then we apply some recombination operations among vectors $x_{i,j}^{(0)}$ in order to obtain NP trial vectors $u_{i,j}^{(1)}, i \in \{1, \cdots, NP\}, j \in \{1, \cdots, D\}$. A new population $x_{i,j}^{(1)}$ is then generated, where

$$x_{i,j}^{(1)} = \begin{cases} u_{i,j}^{(1)}, & \text{if } f(u_i^{(1)}) < f(x_i^{(0)}) \\ x_{i,j}^{(0)}, & \text{otherwise} \end{cases} \tag{3.33}$$

The recombination and selection operations are repeated for several iterations. The number of iterations is defined as G. The vector $x_{i,j}^{(G)}$ with best cost function value in the last iteration is chosen as the optimized solution.

The steps of Differential Evolution are precisely detailed as follows:

Mutation A vector of the population is defined as $x_{i,j}^{(g)}$, where $i \in \{1, 2, \ldots, NP\}$, $j \in \{1, 2, \ldots, D\}$ and g represents the current iteration. At iteration $g+1$, the NP mutant vectors are generated as

$$v_{i,j}^{(g+1)} = x_{i,j}^{(g)} + K \cdot (x_{r_1,j}^{(g)} - x_{i,j}^{(g)}) + F \cdot (x_{r_2,j}^{(g)} - x_{r_3,j}^{(g)}) \tag{3.34}$$

where $i = 1, 2, \ldots, NP$, $j = 1, 2, \ldots, D$. The indices $r_1, r_2, r_3 \in \{1, 2, \ldots, NP\}$ are chosen randomly. For a given i, r_1, r_2, r_3 must be different from each other. K is called the combination factor, and it is often simplified as $K = 1$. F is the scaling factor and it takes its value in $[0, 2]$.

Crossover The population of iterations g is mixed with the newly generated mutant vectors to generate the trial vectors $u_{i,j}^{(g+1)}$

$$u_{i,j}^{(g+1)} = \begin{cases} v_{i,j}^{(g+1)} & \text{if } (\alpha_j \leq CR) \text{ or } j = r_j \\ x_{i,j}^{(g)} & \text{if } (\alpha_j > CR) \text{ and } j \neq r_j \end{cases} \tag{3.35}$$

where $i = 1, 2, \ldots, NP$, $j = 1, 2, \ldots, D$; $r_j \in \{1, 2, \ldots, D\}$ is chosen randomly for each sample i in the population; CR is a parameter of the algorithm called the crossover constant and it takes its value in $[0, 1]$; $\alpha_j \in [0, 1]$ is chosen randomly.

Selection We now select the best vector between trial vector $u_{i,j}^{(g+1)}$ and current iteration vector $x_{i,j}^{(g)}$ by using the cost function f as

$$x_{i,j}^{(g+1)} = \begin{cases} u_{i,j}^{(g+1)} & \text{if} f(u_i^{(g+1)}) \leq f(x_i^{(g)}), \quad i = 1, 2, \ldots, NP \\ x_{i,j}^{(g)} & \text{otherwise} \end{cases} \tag{3.36}$$

In this way, we generate a new population which reduces the value of cost function compared to the population of the previous iteration.

3.4.2 Optimization of protographs using Differential Evolution

In this section, we propose a modification of the Differential Evolution algorithm in order to optimize protographs. In this case, the D-dimensional parameter vector x_D is now a matrix of dimension $S_m \times S_n$. We denote by τ the maximum value of elements in protograph. When we generate randomly the initial population of NP elements, the values $x^{(g)}_{i,j_m,j_n} \in \{0, 1, \ldots, \tau\}$ are integers, with $i \in \{1, 2, \ldots, NP\}$, $j_m \in \{1, \ldots, S_m\}$, $j_n \in \{1, \ldots, S_n\}$. Then we apply the three operators of Differential Evolution (Mutation, Crossover, Selection) in a slightly different way:

As our protograph contain only integers and non negative elements, several adaptations are made during mutation. We set combination factor $K = 1$ and

$$v^{(g+1)}_{i,j_m,j_n} = \text{abs}\left(\text{round} \quad \left(x^{(g)}_{r_1,j_m,j_n} + F \cdot \left(x^{(g)}_{r_2,j_m,j_n} - x^{(g)}_{r_3,j_m,j_n}\right)\right)\right) \tag{3.37}$$

where "abs" is the operation to take absolute value and "round" is the rounding off operation. The cost function is set as the threshold of the protograph calculated by using density evolution (see section 3.3). After G iterations, the protograph with the best threshold will be selected as the result of optimization.

3.4.3 Optimization results

Here we apply the differential evolution method to optimize the protograph. For rate $R = 1/2$, $R = \frac{S_m}{S_n} = 1/2$, the maximum threshold we can get is $p_{opt} = 0.11$ [26]. The parameters used in differential evolution are as follows:

- The number of vectors in the population: $NP = 30$.
- $CR = 0.1$, and $F = 1$.
- The iteration number of differential evolution operations: $G = 30$.

For the cost function, the parameters used in density evolution are:

- The crossover probability: $p \in [0.1, 0.2]$, with a pace of 0.005.
- Length of Monte-Carlo simulation vector: $W = \text{round} \quad (\frac{1000}{S_m \times S_n})$.
- The decoding iteration number: $L = 100$.

With different numbers of S_m, S_n, τ, we get the optimization results in Table 3.1.

In our optimizations, the best protographs we found have dimensions $S_m = 2, S_n = 4$. The three values $\tau = 3, 6, 9$ give the same threshold but since too many connections in the LDPC matrix may generate more short cycles and reduce the code performance, we finally choose $\tau = 3$. At the end the retained protograph is

$$s = \begin{bmatrix} 1 & 2 & 1 & 3 \\ 1 & 1 & 1 & 6 \end{bmatrix} \tag{3.38}$$

This protograph will later be used to generate the LDPC matrix.

S_m	S_n	τ		Threshold	Optimized protograph
1	2	3	$CR = 0.1; D = 1 \times 2; NP = 15$	0.09	$\begin{bmatrix} 5 & 2 \end{bmatrix}$
1	2	6		0.09	$\begin{bmatrix} 4 & 2 \end{bmatrix}$
1	2	9		0.09	$\begin{bmatrix} 4 & 2 \end{bmatrix}$
2	4	3	$CR = 0.1; D = 2 \times 4; NP = 60$	0.1	$\begin{bmatrix} 1 & 2 & 1 & 3 \\ 1 & 1 & 1 & 6 \end{bmatrix}$
2	4	6		0.1	$\begin{bmatrix} 1 & 1 & 5 & 1 \\ 1 & 1 & 2 & 3 \end{bmatrix}$
2	4	9		0.1	$\begin{bmatrix} 1 & 1 & 1 & 6 \\ 2 & 1 & 1 & 2 \end{bmatrix}$
3	6	4	$CR = 0.1; D = 3 \times 6; NP = 135$	0.07	$\begin{bmatrix} 3 & 4 & 2 & 2 & 1 & 1 \\ 3 & 1 & 2 & 2 & 2 & 2 \\ 1 & 2 & 1 & 3 & 1 & 1 \end{bmatrix}$
4	8	4	$CR = 0.1; D = 4 \times 8; NP = 240$	0.05	$\begin{bmatrix} 6 & 2 & 4 & 3 & 2 & 1 & 1 & 3 \\ 4 & 4 & 3 & 4 & 2 & 4 & 1 & 2 \\ 3 & 2 & 1 & 4 & 2 & 3 & 4 & 1 \\ 2 & 5 & 1 & 4 & 1 & 1 & 2 & 5 \end{bmatrix}$

Table 3.1: Construction of protograph based on differential evolution

3.5 LDPC codes construction

3.5.1 Progressive Edge-Growth (PEG) construction

In a parity check matrix H, the connections between VNs and CNs generate cycles, and the length of the shortest cycle is called girth. The performance of a given H depends on the girth and on the number of short cycles.

PEG [37] is an algorithm to construct H with girth as large as possible, and with reduced number of short cycles. When it wants to add a new edge to a given VN, the algorithm finds the most distant CN and then it places the new edge connecting the VN and this most distant CN.

Here we first describe the PEG algorithm for regular LDPC codes, for which the CN degree is denoted by d_c, and the VN degree is denoted by d_v. Within the algorithm, the current VN degree of VN x_i is denoted by $\delta_v[i], i \in \{1, \ldots, n\}$, the current CN degree of CN u_j is denoted by $\delta_c[j], j \in \{1, \ldots, m\}$, $\delta_v[i]$ and $\delta_c[j]$ are initialized to zero. The path length from a CN u_j to a VN x_i is denoted by $d(u_j, x_i)$. Finally, given m, n, d_c, d_v, the steps of the PEG algorithm are as follows:
Variable nodes are processed one by one from x_1 to x_n. Given the i-th VN x_i, connections are progressively added until its VN degree reaches the value d_v: $\delta_v[i] = d_v$. We then go to next VN x_{i+1}. When adding one connection, two cases may occur.

Case 1 If the current VN x_i is not connected to any CNs or there exist some CNs which are not reachable and their CN degree smaller than d_c, then choose randomly one among these CNs with lowest CN degree $\delta_c[j]$. New connection will be made

between x_i and this selected CN, and the CN degree information will be incremented by 1: $\delta_c[j] = \delta_c[j] + 1$.

Case 2 If all the CNs are reachable for current VN x_i, choose the CN u_j with largest distance $d(u_j, x_i)$ and their CN degree should be smaller than d_c. If there exist a lot of CNs having largest girth, choose randomly one CN u_j with least CN degree $\delta_c[j]$. When this new connection is added, we also update $\delta_c[j] = \delta_c[j] + 1$.

We continue to add new connections until all the VN degree d_v are satisfied: $\delta_v[i] = d_v, \forall i \in \{1, \cdots, n\}$.

The irregular LDPC codes construction follows the same steps, the difference is that the CN degree d_c and VN degree d_v are now vectors $d_c[m]$ and $d_v[n]$.

3.5.2 LDPC codes construction using protograph based on PEG algorithm

A PEG algorithm can also be used in order to construct the parity check matrix H from a protograph [33]. The steps are almost the same, but the difference is that now the VN and CN degree distributions should follow the values in protograph. So the VN and CN types need to be declared, and they should be updated when adding a new edge.

3.6 Performance comparison of the three decoders

We now compare the decoding performance of the three decoding algorithms (Bit-flipping, Gallager-A, Sum-product). With a Wimax code [1] of length $n = 504$, by using 10000 tests, 50 iterations for decoding, we get the results in Figure 3.4. We also generate a LDPC matrix of length $n = 1024$ from a protograph s, where

$$s = \begin{bmatrix} 2 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 2 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 2 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 2 & 1 & 1 & 1 & 0 \end{bmatrix} \tag{3.39}$$

Applying also 10000 tests and 50 iterations, we get the results in Figure 3.5.

In both cases, we observe that the Sum-product decoder has a significantly better performance than Gallager-A decoder and Bit-flipping decoder. This is why it will be considered in the following. Nevertheless, Bit-flipping and Gallager-A decoder are less complex than Sum-product decoder, and they can be used in the case of limited computational resources.

For the LDPC code construction in Section 3.5, the coding rate R is fixed once for all. But if the available side information $Y^{(t)}$ comes from a set of possible side informations $\{Y^{(1)}, \cdots, Y^{(T)}\}$, sending the data at fixed rate R will cause either a rate loss or a decoding failure. This is why we now describe rate-adaptive LDPC code constructions that allow to adapt the coding rate depending on the side information $Y^{(t)}$ available at the decoder. In channel coding, we refer to "rate-compatible LDPC

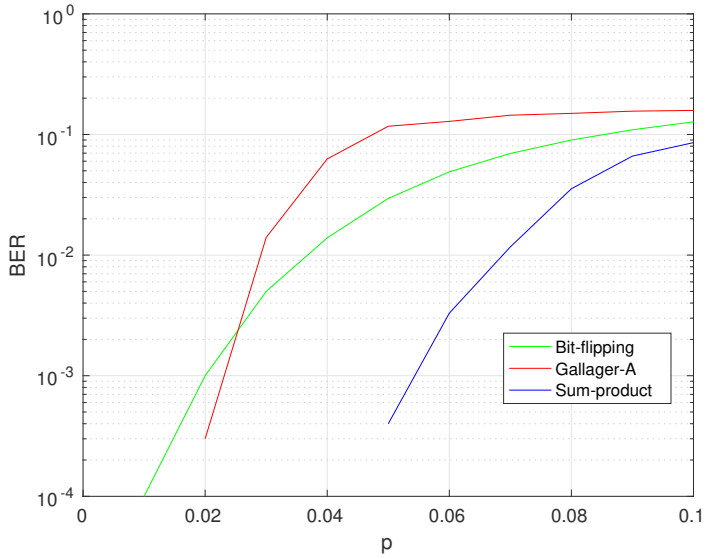

Figure 3.4: LDPC decoder with n=504 Tests=10000 iterations=50 using Wimax code [1]

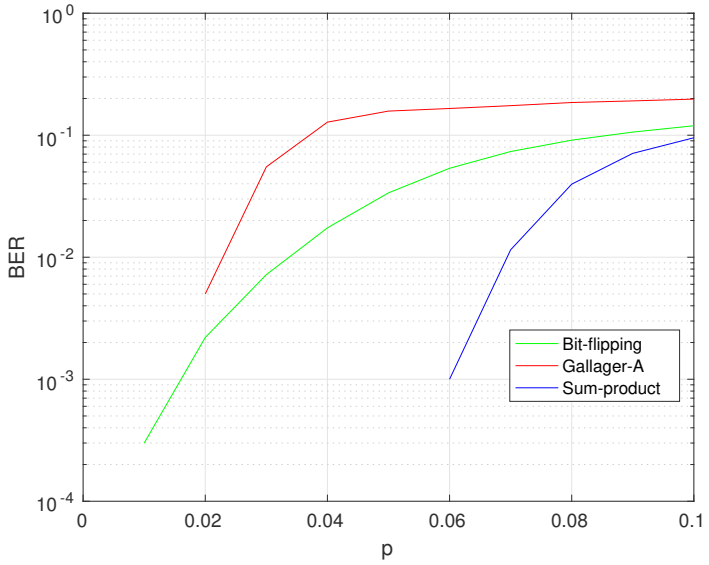

Figure 3.5: LDPC decoder with n=1024 Tests=10000 iterations=50 using protograph given by (3.39)

codes", while in source coding, we refer to "rate-adaptive LDPC codes". We first describe existing rate-compatible code constructions in channel coding, and then present methods developed for source coding with side information.

3.7 Rate-compatible LDPC codes for channel coding

In this section, we present two standard methods for the construction of rate-compatible LDPC codes for channel coding. These two methods are Puncturing [53] and Parity Check Matrix extension [54]. We also explain the limitations of these methods when applied to source coding.

3.7.1 Puncturing

Puncturing [53] is a rate compatible method that starts from a high rate and achieves a lower rate by not sending several punctured codeword bits. Recall that X^n is the codeword obtained with a channel coding rate R. After puncturing, only a part of the bits of X^n will be transmitted over the channel. The conserved codeword bits after puncturing are defined by $r^{n(1-\zeta)}$ as shown in Figure 3.6, where ζ is the proportion of punctured bits.

So as to compute the channel coding rate after puncturing, we have to introduce new useful notations. Let us define $\tilde{\lambda}_k$ (respectively $\tilde{\rho}_k$) as the fraction of VNs (respectively CNs) having k edges. The relation between $\tilde{\lambda}_k$ ($\tilde{\rho}_k$) and λ_k (ρ_k) in (3.3) is

$$\tilde{\lambda}_k = \frac{\lambda_k/k}{\sum_{k'=2}^{d_{v\max}} \lambda_{k'}/k'} \qquad \tilde{\rho}_k = \frac{\rho_k/k}{\sum_{k'=2}^{d_{c\max}} \rho_{k'}/k'} \tag{3.40}$$

The puncturing proportion $\pi^{(0)}(x)$ is defined as

$$\pi^{(0)}(x) = \sum_{k=2}^{d_{v\max}} \pi_k^{(0)} x^{k-1} \tag{3.41}$$

where $0 \leq \pi_k^{(0)} \leq 1$, $\pi_k^{(0)}$ represents the puncturing fraction for variable nodes of degree k. Then the puncturing fraction ζ that represents the total puncturing proportion on the n-length source code can be expressed as

$$\zeta = \frac{\sum_{k=2}^{d_{v\max}} \pi_k^{(0)} \cdot n_k}{n} = \frac{\sum_{k=2}^{d_{v\max}} \pi_k^{(0)} \cdot \lambda_k/k}{\sum_{k=2}^{d_{v\max}} \lambda_k/k} = \sum_{k=2}^{d_{v\max}} \tilde{\lambda}_k \cdot \pi_k^{(0)} \tag{3.42}$$

The rate of the punctured LDPC code is then [53]

$$R\left(\lambda, \rho, \pi^{(0)}\right) = \frac{R(\lambda, \rho)}{1-\zeta} \tag{3.43}$$

In order to design a good rate-compatible code, one can design puncturing proportions $\pi_k^{(0)}$ for all k, that minimize the SNR threshold for a given puncturing fraction ζ. Alternatively, we can fix a target SNR threshold and maximize the puncturing fraction ζ and puncturing proportions $\pi_k^{(0)}$ while satisfying the threshold.

3.7.1.1 Puncturing in source coding

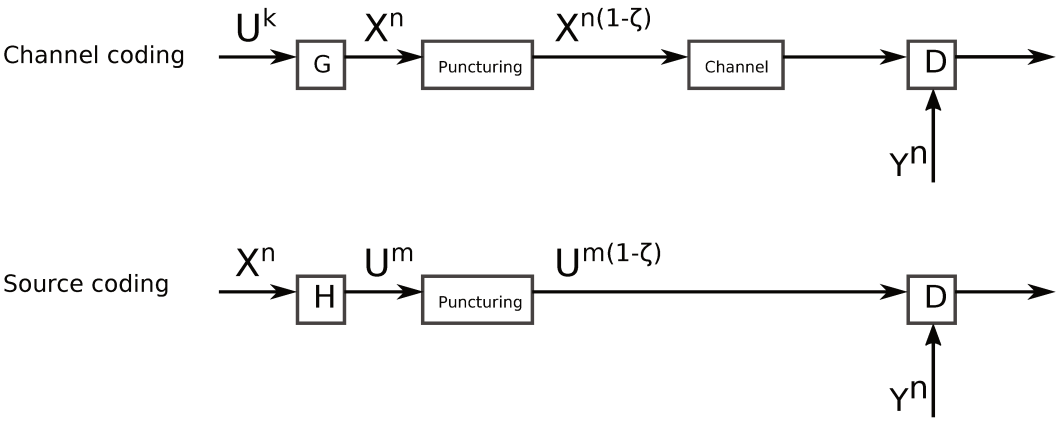

Figure 3.6: Puncturing in channel coding and in source coding. D is the decoder. In channel coding, G is the generator matrix, and in source coding, H is the parity check matrix.

As shown in Figure 3.6, puncturing can also be applied in source coding. After puncturing, the codeword U^m is shortened to $U^{m(1-\zeta)}$, and the puncturing proportions may also be optimized. Unfortunately, it is shown in [55] that puncturing on the codeword U fastly destroys the code structure and leads to very poor performance. As a result, puncturing is not a good solution for our source coding problem.

3.7.2 Parity Check Matrix extension

Parity check matrix extension [54] is a channel coding rate-compatible construction that starts from an initial low rate LDPC code and then constructs the higher rates. Figure 3.7 shows an example of extension of a PEG-constructed LDPC code. It starts from an initial LDPC code H_0 of rate $R_0 = 8/13$ with dimensions $n = 1664, k = 640$. In order to encode the information sequence of length $k = n - m$ into a longer codeword ($n = 1664 + 768$), the original parity check matrix is concatenated with three other matrices. This method fills the left-down matrix H_{uni} with zero and uniform matrix and fills the right-up matrix with zeros in order to add a stronger dependency between the columns of the initial parity-check matrix and the newly added columns. The right-down matrix H_{ext} which permits to generate new columns is constructed with a PEG algorithm with zeros and with a given fixed matrix h_{ext} in the diagonal. Using PEG for both the initial matrix H_0 and h_{ext} improves the girth distribution of the Tanner Graph.
Unfortunately, the Extension method cannot be applied to source coding. Indeed, in source coding, the dimension n is fixed and given by the source sequence to be compressed.
Since most rate-compatible methods used in channel coding are based either on

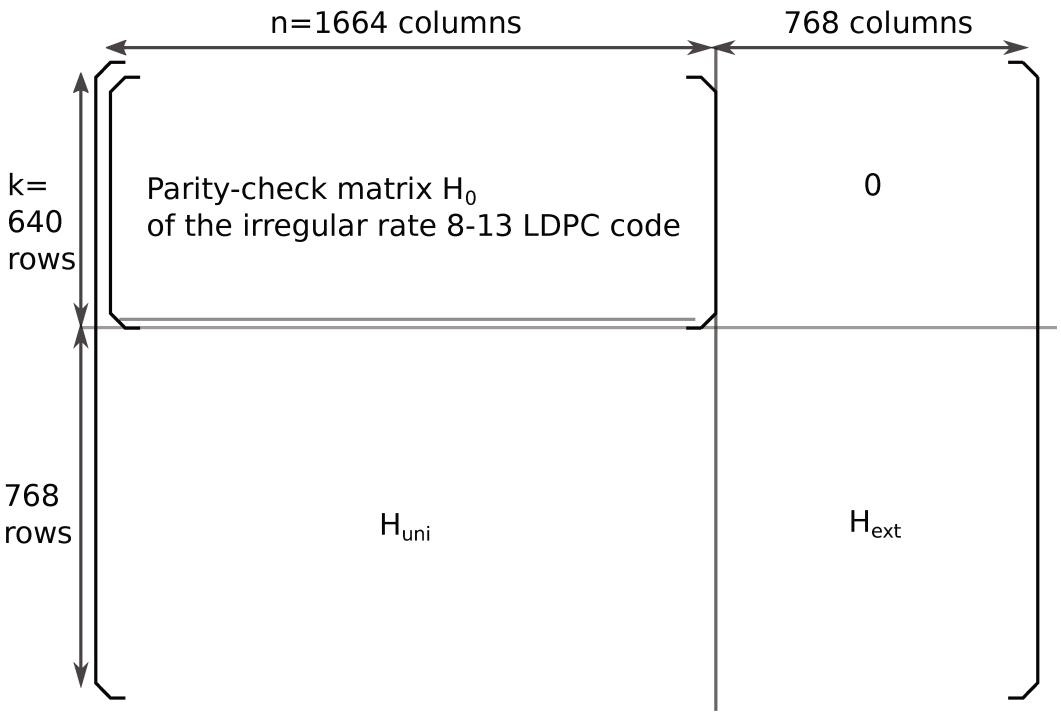

Figure 3.7: Extension of rate-8/13 PEG-constructed LDPC code with $n = 1024$

puncturing or on extension, several alternative methods were specifically designed for source coding. We now describe these methods.

3.8 Rate-adaptive LDPC codes for source coding

We now describe two rate-adaptive code constructions for source coding: Rateless [39, 56] and LDPCA [55].

3.8.1 Rateless

The rate-adaptive Rateless scheme [39, 56] starts by constructing an initial low-rate LDPC code. If a higher rate is needed, a part of the source bits $\underline{x}^n$ will be sent in addition to the syndrome $\underline{u}^m$. In order to select the additional transmitted source bit x^n, we choose the most unreliable bits of x^n after applying an LDPC decoder. For this, we choose the bits x_i with the lowest absolute value $|Q_i|$, where Q_i is defined in (3.25). However, the major drawback of the Rateless scheme is that it difficult to construct good low rate LDPC codes [57, 38][1], and a bad initial low rate code will cause poor performance at any considered higher rate. Therefore, it is not desirable to apply the Rateless construction from very low rates.

[1]Low rate LDPC codes for source coding correspond to high rate LDPC codes for channel coding

3.8.2 LDPCA

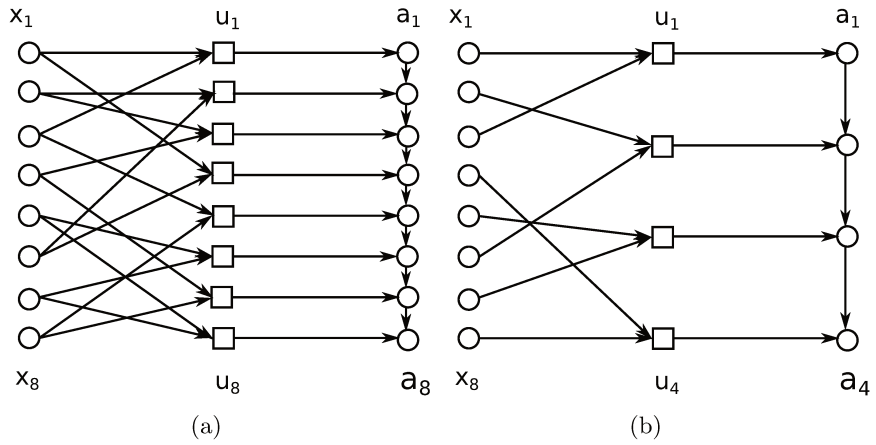

Figure 3.8: (a) The LDPCA encoder for rate $R = 1$, (b) The LDPCA encoder for rate $R = 1/2$

On the opposite, the LDPCA scheme [55] starts from a high-rate LDPC code. It then computes new accumulated symbols $\underline{a}^m = [a_1, a_2, \cdots, a_m]^T$ from the syndrome $\underline{u}^m$ (3.1) as

$$\begin{aligned} a_1 &= u_1, \\ a_i &= a_{i-1} + u_i, \quad \forall i = \{2, \cdots, m\}, \end{aligned} \tag{3.44}$$

where the binary sum in (3.44) corresponds to XOR operations. If a lower rate is demanded, only a part of the symbols $(a_1, a_2, \cdots, a_m)$ will be sent. For instance, if the original rate is R and a rate $R/2$ is demanded, only the even symbols $a_2, a_4, a_6, \cdots$ will be transmitted. The decoder will then compute all the differences $a_i - a_{i-2} = u_i + u_{i-1}$, before applying a BP decoder in order to estimate the source vector $\underline{x}^n$ from all the obtained XOR sums $u_i + u_{i-1}$.
In this construction, puncturing the source symbols a_i rather than the syndrome bits u_i was shown to better preserve the code structure and to greatly improve the decoding performance [55]. However, in the LDPCA construction, the accumulator structure (3.44) is fixed and does not allow for an optimization of the combinations of syndrome symbols u_i that are used by the decoder. The accumulator structure may in particular induce short cycles in the lowest rates and eliminate some source bits from the CN constraints. In [58], the LDPCA structure is improved by considering a non-regular accumulator. The non-regular accumulator is designed for any rate of interest by optimizing its polynomial degree distribution under asymptotic conditions. Unfortunately, [58] does not propose any finite-length code construction that could solve the short cycles and VN elimination issues.

3.8.3 Method starting from rate 1/2

Due to the drawbacks of Rateless and LDPCA schemes, an intermediate solution was proposed in [59]. It first constructs an initial code of rate $R = 1/2$. It then applies either the LDPCA method to obtain rates lower than $1/2$ or the Rateless method for rates higher than $1/2$. In this way, the shortage of the Rateless construction can be avoided, but the drawbacks of LDPCA remain.

In this thesis, we thus propose a novel rate-adaptive construction that replaces the LDPCA part in the solution of [59]. The construction we propose replaces the regular LDPCA accumulator described in (3.44) by a non-regular structure. In addition, in our construction, this non-regular structure is also described by a protograph. We show that the daughter code protographs can be calculated explicitly from the non-regular accumulator protograph and from the mother code protograph. This allows us to optimize the asymptotic code performance by carefully selecting the code protograph at all rates. We also propose a finite length code construction method that permits to reduce the amount of short cycles in all the considered codes. The proposed method is thus well adapted to the construction of short length LDPC codes.

3.9 Conclusion

In this Chapter, we presented existing methods for the construction and decoding of LDPC source codes. We also introduced existing rate-adaptive source coding solutions based on LDPC codes. Since existing methods show some limitations and drawbacks, we now propose two novel rate-adaptive constructions for source coding.

Chapter 4

Proposed methods for the construction of rate-adaptive LDPC codes

Contents

In this Chapter, we introduce our novel rate-adaptive code constructions for source coding. The first Section 4.1 is common to both methods.

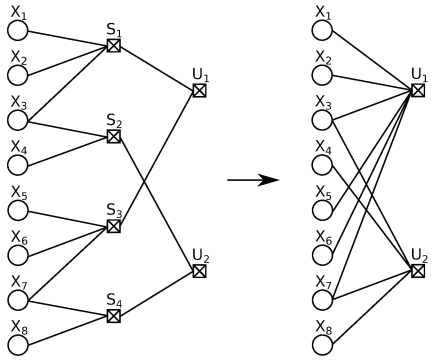

Figure 4.1: The left part of the figure shows the combination of $\mathcal{T}_1$ with $\mathcal{T}_{1\to 2}$. The right part of the figure shows the resulting $\mathcal{T}_2$. Here, the matrix $H_{1\to 2}$ is full rank, and one may choose between $S' = \{s_1, s_2\}$, $S' = \{s_3, s_4\}$, $S' = \{s_1, s_4\}$, or $S' = \{s_2, s_3\}$.

4.1 Rate-adaptive code construction

The rate-adaptive code design method we propose in this thesis is based on a rate-adaptive code structure initially proposed in [40, 60]. For the sake of clarity, this section describes the rate-adaptive code structure of [40, 60]. This construction starts from a mother code of the highest rate and then builds a sequence of daughter codes of lower rates. This section only describes the construction of one code of rate R_2 from a code of rate $R_1 > R_2$. This construction is generalized to more rates later in the thesis.

4.1.1 Rate-adaptive code construction

In the rate-adaptive construction of [40, 60], the mother code is described by a parity check matrix H_1 of size $m_1 \times n$ with coding rate $R_1 = m_1/n$. The Tanner graph $\mathcal{T}_1$ connects the n VNs $\mathcal{X} = \{x_1, \cdots, x_n\}$ to m_1 CNs $S = \{s_1, \cdots, s_{m_1}\}$. The matrix H_1 is constructed from a protograph $\mathcal{S}_1$ according to the code design method described in Section 3.1.2. From the mother matrix H_1, we want to construct a daughter matrix H_2 of size $m_2 \times n$, with $m_2 < m_1$, and rate $R_2 = m_2/n < R_1$. The Tanner graph $\mathcal{T}_2$ will connect the n VNs $\mathcal{X}$ to m_2 CNs $\mathcal{U} = \{u_1, \cdots, u_{m_2}\}$.

In the considered construction, the daughter matrix H_2 and the mother matrix H_1 are linked by an intermediate matrix $H_{1\to 2}$ of size $m_2 \times m_1$ such that

$$H_2 = H_{1\to 2} H_1. \tag{4.1}$$

The Tanner graph $\mathcal{T}_{1\to 2}$ of $H_{1\to 2}$ connects the m_1 CNs S of $\mathcal{T}_1$ to the m_2 CNs $\mathcal{U}$ of $\mathcal{T}_2$. Figure 4.1 shows an example of the construction of $\mathcal{T}_2$ from $\mathcal{T}_1$ and $\mathcal{T}_{1\to 2}$. Note that LDPCA codes can be seen as a particular case of this construction. The intermediate matrix $H_{1\to 2}$ should be chosen not only to give a good decoding performance for H_2, but also to allow H_1 and H_2 to be rate-adaptive in a sense we now describe.

4.1.2 Rate adaptive condition

In the construction of [40, 60], the following transmission rules are set in order to allow H_1 and H_2 related by (4.1) to be rate-adaptive. In order to get a rate R_2, we simply transmit all the syndrome values $\underline{u}^{m_2}$, which corresponds to m_2 equations defined by the set $\mathcal{U}$. The decoding is then realized with the matrix H_2. In order to get a rate R_1, we transmit all syndrome values in $\underline{u}^{m_2}$ but also a subset $S' \subseteq S$ of size $m_1 - m_2$ of the values in $\underline{s}^{m_1}$. This guarantees that the code construction is incremental and that the storage rate is given by $R_1 = \max(R_1, R_2) < R_1 + R_2$. However, in order to use the matrix H_1 for decoding, the receiver must be able to recover the full syndrome $\underline{s}^{m_1}$ from $\underline{u}^{m_2}$ and S'. The code that results from the choice of (H_1, $H_{1\to 2}$, S') is thus said to be rate-adaptive if is satisfies the following condition.

Definition 1 ([40, 60]) *The sets $\mathcal{U}$ and S' define a system of m_1 equations with m_1 unknown variables S. If this system has a unique solution, then the triplet (H_1, $H_{1\to 2}$, S') is said to be a rate-adaptive code.*

The following proposition gives a simple condition that permits to verify whether a given intermediate matrix $H_{1\to 2}$ gives a rate-adaptive code.

Proposition 1 ([40, 60]) *If the matrix $H_{1\to 2}$ is full rank, then there exists a set $S' \subseteq S$ of size $m_1 - m_2$ such that (H_1, $H_{1\to 2}$, S') is a rate-adaptive code.*

The above proposition shows that if $H_{1\to 2}$ is full rank, it is always possible to find a set S' that ensures that H_1 and H_2 are rate-adaptive. The decoding performance of H_1 does not depend on the choice of the set S', since at rate R_1, the decoder uses H_1 and at rate R_2, the decoder uses H_2. On the opposite, according to (4.1), the decoding performance of the matrix H_2 heavily depends on the matrix $H_{1\to 2}$. In [40], the matrix $H_{1\to 2}$ is constructed from an exhaustive search, which is hardly feasible when the codeword length increases (from 100 bits). In [60], a more efficient method is proposed to construct the intermediate matrix $H_{1\to 2}$ so as to avoid short cycles in H_2. However, the method of [60] does not optimize the theoretical threshold of the degree distribution of H_2, which also influences the code performance. In this thesis, we propose a novel method based on protographs for the design of the intermediate matrix $H_{1\to 2}$. This novel method not only allows to optimize the threshold of the protograph of H_2, but also to reduce the amount of short cycles in H_2.

4.2 Intermediate matrix construction without protograph

This section describes our first novel method for the construction of the intermediate matrix $H_{1\to 2}$ introduced in Section 4.1. The proposed construction seeks to minimize the threshold of new constructed parity check matrix H_2 at rate R_2 by reducing the amount of short cycles.

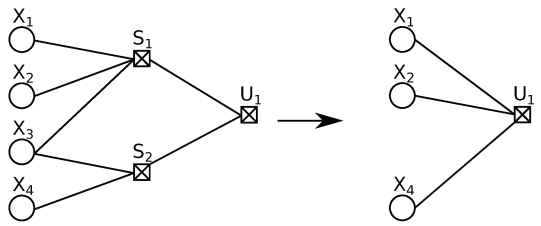

Figure 4.2: Since the VN x_3 appears in both s_1 and s_2, combining these CNs into u_1 makes x_3 disappear in H_2.

As the decoding performance of the matrix H_2 heavily depends on the matrix $H_{1\to 2}$, the careful choice of the connections in $\mathcal{T}_{1\to 2}$ is very important. Constructing $H_{1\to 2}$ can be seen as combining the CNs S of H_1 in order to create the CNs $\mathcal{U}$ of the new parity check matrix H_2. Combining the CNs S can however cause three issues that could degrade the decoding performance of H_2.

First, combining some of the CNs of H_1 could degrade the connectivity of some VNs in H_2, see Figure 4.2 for an example. In the worst case, some VNs may not be connected anymore to any CNs in H_2. Second, combining H_1 and $H_{1\to 2}$ may introduce short cycles in H_2 which could severely degrade the decoding performance. As a third issue, the matrix $H_{1\to 2}$ has to be full rank in order to satisfy the rate-adaptive condition described in Section 4.1.2. The construction method we now propose for $H_{1\to 2}$ addresses these three issues through the choice of the degree distribution and of the connections in $H_{1\to 2}$.

4.2.1 Degree distribution for $H_{1\to 2}$

In order to be able to construct the intermediate matrix $H_{1\to 2}$ in a systematic way, we first need to choose a degree distribution for $H_{1\to 2}$. As a first constraint, we impose that each CN $s_i \in S$ is connected to exactly one CN of $\mathcal{U}$. We also impose that each $u_j \in \mathcal{U}$ is connected to one or more CN of S. These two conditions ensure that $H_{1\to 2}$ will be full rank. In addition, the CNs in S are all of degree 1, and we only need to describe the degree distribution of the $u_j \in \mathcal{U}$.

We denote the degree distribution of the CNs in $\mathcal{U}$ in $H_{1\to 2}$ as $(\underline{\alpha}, \underline{d})$, where $\underline{\alpha} = [\alpha_1, \cdots, \alpha_K]$, $\underline{d} = [d_1, \cdots, d_K]$, and K represents the number of possible degrees. The value α_k denotes the proportion of CNs of $\mathcal{U}$ connected to exactly d_k symbols of $\mathcal{S}$. The degree distribution $(\underline{\alpha}, \underline{d})$ satisfies

$$\frac{m_1}{m_2} = \sum_{k=1}^{K} \alpha_k d_k. \tag{4.2}$$

Note that the degree distribution $(\underline{\alpha}, \underline{d})$ of the CNs in $\mathcal{U}$ in $H_{1\to 2}$ is not the same as the degree distribution of the CNs $\mathcal{U}$ in H_2. We could think of optimizing the degree distribution $(\underline{\alpha}, \underline{d})$ in $H_{1\to 2}$ by applying density evolution on the resulting degree distribution in H_2. However, here, in order to focus on the finite length code construction, we do not consider optimization from density evolution and we

simply choose the degrees d_k as small as possible. For example, if $R_2 = 3/8$, we set $\underline{d} = [1, 2]$ and the proportions α_1 and α_2 are set to $\alpha_1 = 1/2$, $\alpha_2 = 1/2$. Setting low degrees in $H_{1\to 2}$ increases the chances of avoiding short cycles in the resulting H_2.

4.2.2 Connections in $H_{1\to 2}$

We now explain how to choose the connections between S and $\mathcal{U}$ according to the degree distribution $(\underline{\alpha}, \underline{d})$. In our method, the degree of each CN $u_j \in \mathcal{U}$ is selected at random according to the degree distribution $(\underline{\alpha}, \underline{d})$. Then, whatever the degree d_k of a given u_j, we impose the following two conditions in order to choose the CNs of S that will be connected to u_j:

1. We choose d_k CNs in S that are not connected to any common VN. This permits to avoid eliminating VN connections in the resulting H_2.
2. We choose the d_k CNs in S in order to minimize the number of resulting cycles in H_2.

Condition 1) is very easy to verify while condition 2) requires to count the number of cycles in H_2. There exists several methods to calculate the number of shorts cycles in the parity check matrix of an LDPC codes. Here, since we are mainly concerned with short cycles, we choose the method proposed in [61] which is very efficient for the counting of short cycles of length 4, 6, and 8.

Then, in order to construct u_j, we need to select d_k CNs of S. The first CN s_i is selected at random from the set of CNs that have not yet been used in any already constructed u'_j. The next $d_k - 1$ CNs s_i are chosen so as to minimize the number of length-4, length-6, and length-8 cycles introduced in H_2 by the newly created u_j. In order to select the best $d_k - 1$ CNs s_i, we try T possible combinations of $d_k - 1$ CNs selected at random from the set of remaining CNs. As an example, Algorithm 1 shows the algorithm that is used in a particular case $(\underline{\alpha}, \underline{d}) = (1, 2)$ when we only want to minimize the number of length-4 cycles.

4.2.3 Construction of the set S'

The degree distribution defined in Section 4.2.1 as well as the code construction proposed in Section 4.3.3 ensure that the matrix $H_{1\to 2}$ is full rank. This guarantees that the rate-adaptive condition presented in Section 4.1.1 is satisfied. In order to completely define the rate-adaptive code $(H_1, H_{1\to 2}, S')$, we need to define a set S' of symbols of S that will be sent together with the set $\mathcal{U}$ in order to obtain the rate R_1.

The set S' will serve to solve a system of m_2 equations $\mathcal{U}$ with m_2 unknowns $S \setminus S'$. For each equation $u_i \in \mathcal{U}$ of degree d_k, we hence decide to put $d_k - 1$ of the d_k CNs connected to u_i into S'. For example, if $u_1 = s_1 \oplus s_2 \oplus s_3$, s_1 and s_2 can be placed into S'. This strategy gives that the set S' is, as expected, composed by

$$\sum_{k=1}^{K} \alpha_k (d_k - 1) m_2 = m_1 - m_2 \tag{4.3}$$

Algorithm 1 Construction of the intermediate matrix $H_{1\to 2}$ in the particular case $(\underline{\alpha}, \underline{d}) = (1, 2)$

Fix T, $S = \{1, 2 \cdots, m_1\}$
for $j = 1$ to m_2 **do**
  Fix MinCycle $= \infty$
  Select a value p at random among the set S
  Remove p from the set S
  for $t = 1$ to T **do**
    Select a value q at random among the set S
    if s_p and s_q are connected to a common VN, **then**
      break
    else
      Count the number $NbCycles$ of length-4 cycles in H_2 with j-th line $H_{2,j} = H_{1,p} \oplus H_{1,q}$
      if NbCycles $<$ MinCycle **then**
        MinCycle $=$ NbCycles
        Set $qchoosen = q$
  Set $H_{2,j} = H_{1,p} \oplus H_{1,qchoosen}$

different CNs of S. It also guarantees that it is always possible to reconstruct the set S from $\mathcal{U}$ and S'. In the above example, it indeed suffices to recover s_3 as $s_3 = u_1 \oplus s_1 \oplus s_2$.

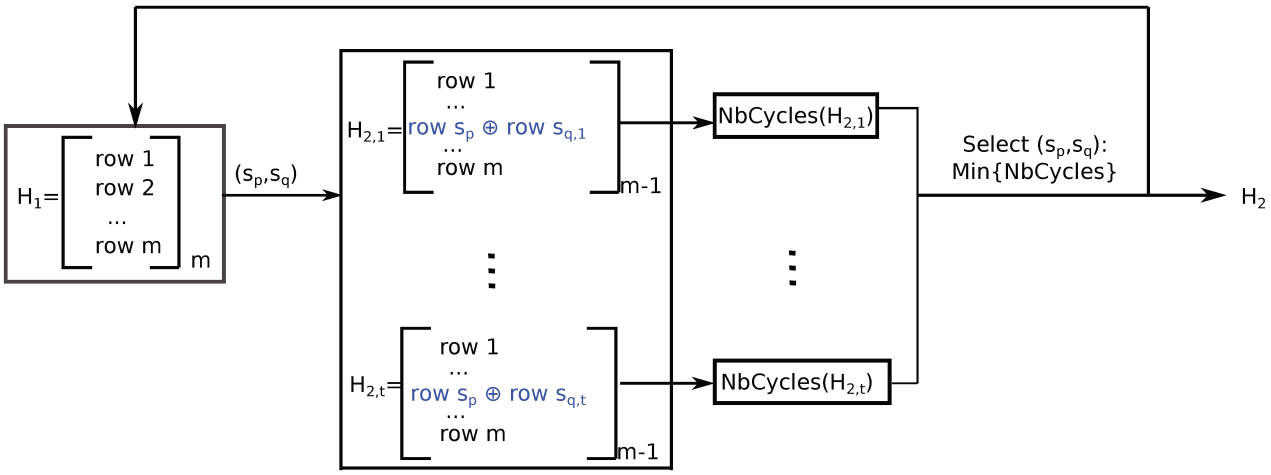

Figure 4.3: Schema of Algo 1

4.3 Intermediate matrix construction with protograph

This section describes our second novel method for the construction of the intermediate matrix $H_{1\to 2}$ introduced in Section 4.1. This method is based on the

optimization of the code protographs. The proposed construction seeks to minimize the protograph threshold at rate R_2, and also to reduce the amount of short cycles in the parity check matrix H_2.

4.3.1 Protograph $\mathcal{S}_2$ of parity check matrix H_2

In order to construct a good parity check matrix H_2 from the initial matrix H_1, we first want to select a protograph $\mathcal{S}_2$ with a good theoretical threshold. In this section, we consider the following notation. Generally speaking, consider the protograph $\mathcal{S}_g$ of size $S_{m_g} \times S_{n_g}$ associated with the matrix H_g, where $g \in \{1, 2, 1 \to 2\}$. As a particular case, note that $S_{m_{1\to 2}} = S_{m_2}$ and $S_{n_{1\to 2}} = S_{m_1}$. For all $(i, j) \in \{1, \cdots, S_{m_g}\} \times \{1, \cdots, S_{n_g}\}$, denote by $s_{i,j}^{(g)}$ the coefficient at the i-th row, j-th column of $\mathcal{S}_g$. In the protograph $\mathcal{S}_g$, the CN types are denoted $A_1^{(g)}, \cdots, A_{S_{m_g}}^{(g)}$ and the VN types are denoted $B_1^{(g)}, \cdots, B_{S_{n_g}}^{(g)}$. In the parity check matrix H_g, the set of CNs of type $A_i^{(g)}$ is denoted $\mathcal{A}_i^{(g)}$ and the set of VNs of type $B_j^{(g)}$ is denoted $\mathcal{B}_j^{(g)}$. Finally, denote by $\underline{h}_k^{(g)}$ the k-th row of H_g, and denote by $h_{\ell,k}^{(g)}$ the coefficient at the ℓ-th row, k-th column of H_g.

Based on the above notation, the following proposition gives the relation between the three protographs $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_{1\to 2}$.

Proposition 2 *Consider a matrix H_1 with protograph $\mathcal{S}_1$ of size $S_{m_1} \times S_n$, a matrix $H_{1\to 2}$ with protograph $\mathcal{S}_{1\to 2}$ of size $S_{m_2} \times S_{m_1}$, and a matrix $H_2 = H_{1\to 2}H_1$. Also consider the following two assumptions:*

1. *Type structure: for all $j \in \{1, \cdots, S_{m_1}\}$, $\mathcal{B}_j^{(1\to 2)} = \mathcal{A}_j^{(1)}$.*
2. *No VN elimination: For all $\ell \in \{1, \cdots, m_2\}$, denote by $\mathcal{N}_\ell^{(1\to 2)}$ the positions of the non-zero components in $\underline{h}_\ell^{(1\to 2)}$. Then, $\forall k_1, k_2 \in \mathcal{N}_\ell^{(1\to 2)}$ such that $k_1 \neq k_2$, and $\forall i \in \{1, \cdots, n\}$, $h_{k_1,i}^{(1)} \neq h_{k_2,i}^{(1)}$.*

If these two assumptions are fulfilled, then the matrix

$$\mathcal{S}_2 = \mathcal{S}_{1\to 2}\mathcal{S}_1 \tag{4.4}$$

is of size $S_{m_2} \times S_n$ and it is a protograph of the matrix H_2. The operation in (4.4) corresponds to standard matrix multiplication over the field of real numbers.

In this proof, for clarity, we denote by $\bigoplus$ the modulo two sums and by $\sum$ the standard sums over the field of real numbers. With the above notation, relation (4.1) can be restated row-wise as

$$\underline{h}_\ell^{(2)} = \bigoplus_{k=1}^{m_1} h_{\ell,k}^{(1\to 2)} \underline{h}_k^{(1)} = \bigoplus_{j=1}^{S_{m_1}} \bigoplus_{\substack{k \in \mathcal{B}_j^{(1\to 2)} \\ \text{s.t. } h_{\ell,k}^{(1\to 2)} \neq 0}} \underline{h}_k^{(1)}. \tag{4.5}$$

Relation (4.5) depends on index ℓ only through $h_{\ell,k}^{(1\to2)}$. This implies that, in (4.5) the type combination is the same for every $\ell \in \mathcal{A}_i^{(1\to2)}$. As a result, for all $i \in \{1,\cdots,S_{m_2}\}$, $\mathcal{A}_i^{(2)} = \mathcal{A}_i^{(1\to2)}$. In the same way, deriving relation (4.1) column-wise permits to show that $\forall j \in \{1,\cdots,S_n\}$, $\mathcal{B}_j^{(2)} = \mathcal{B}_j^{(1)}$.

Now consider $i \in \{1,\cdots,S_{m_2}\}$, $v \in \{1,\cdots,S_n\}$, and $\ell \in \mathcal{A}_i^{(2)}$. Then, from (4.5),

$$s_{i,v}^{(2)} = \sum_{u\in\mathcal{B}_v^{(2)}} h_{l,u}^{(2)} = \sum_{u\in\mathcal{B}_v^{(2)}} \left(\bigoplus_{j=1}^{S_{m_1}} \bigoplus_{\substack{k\in\mathcal{B}_j^{(1\to2)} \\ \text{s.t. } h_{\ell,k}^{(1\to2)}\neq 0}} h_{k,u}^{(1)} \right). \tag{4.6}$$

In the vector $\underline{h}_k^{(1)}$ with $k \in \mathcal{B}_j^{(1\to2)}$, there are $s_{j,v}^{(1)}$ non-zero values over the components $h_{k,u}$ such that $u \in \mathcal{B}_v^{(2)}$. In addition, for $k \in \mathcal{B}_j^{(1\to2)}$, there are $s_{i,j}^{(1\to2)}$ non-zero values over the components $h_{\ell,k}^{(1\to2)}$. As a result, and since there is not VN elimination,

$$s_{i,v}^{(2)} = \sum_{j=1}^{S_{m_1}} s_{i,j}^{(1\to2)} s_{j,v}^{(2)}, \tag{4.7}$$

which implies (4.4).

In Proposition 2, assumption 1) is required because various interleaving structures may be used to construct *e.g.* a matrix H_1 from a given protograph $\mathcal{S}_1$. This assumption guarantees that the same interleaving structure is used for the CNs of $\mathcal{S}_1$ and the VNs of $\mathcal{S}_{1\to2}$. Further, assumption 2 guarantees that relation (4.1) does not eliminate any VN from the parity check equations in H_2. This permits to preserve the code structure that will be characterized by protograph $\mathcal{S}_2$. Then, by comparing (4.1) and (4.4), we observe that there is the same relation between the protographs $\mathcal{S}_1$, $\mathcal{S}_2$, and between the parity check matrices H_1, H_2. Further, according to (4.4), the problem of finding a good protograph $\mathcal{S}_2$ for H_2 can be reduced to finding the intermediate protograph $\mathcal{S}_{1\to2}$ that maximizes the threshold of $\mathcal{S}_2$.

4.3.2 Optimization of the intermediate protograph $\mathcal{S}_{1\to2}$

The protograph $\mathcal{S}_{1\to2}$ of size $S_{m_2} \times S_{m_1}$ must be full rank in order to satisfy the rate-adaptive condition defined in Section 4.1.2. However, even if S_{m_1} and S_{m_2} are small, there is still a lot of possible protographs $\mathcal{S}_{1\to2}$. This is why, here, we impose that each row of $\mathcal{S}_{1\to2}$ has either 1 or 2 non-zero components, that each column has exactly 1 non-zero component, and that all the non-zero components are equal to 1. These constraints are equivalent to considering that each row of $\mathcal{S}_2$ is either equal to a row of $\mathcal{S}_1$ or equal to the sum of two rows of $\mathcal{S}_1$. They limit the number of possible $\mathcal{S}_{1\to2}$ without being too restrictive. They will also make the intermediate matrix $H_{1\to2}$ quite sparse, which will help limiting the amount of short cycles in

the matrix H_2. Finally, we observe that these constraints provide satisfactory rate-adaptive code constructions in our simulations. The design algorithms described in the remaining of the thesis can also be easily generalized to other constraints on the intermediate protograph.

For the optimization, we then generate all the possible intermediate protographs $\mathcal{S}_{1\to 2}$ that satisfy the above two conditions ($\mathcal{S}_{1\to 2}$ is full rank and each of its rows has either 1 or 2 non-zero components), and select the intermediate protograph that maximizes the threshold of the protograph $\mathcal{S}_2$ calculated from (4.4).

The intermediate protograph $\mathcal{S}_{1\to 2}$ defines the degree distribution of the intermediate matrix $H_{1\to 2}$. It also indicates the rows of H_1 that can be combined in order to construct the daughter matrix H_2. We would like those rows to be combined in the best possible way in order to produce H_2. In particular, we would like to avoid both short circles and VN elimination during the construction of H_2. In the following, we propose an algorithm that constructs H_2 from these conditions.

4.3.3 Algorithm Proto-Circle: connections in $H_{1\to 2}$

Algorithm 2 Proto-Circle: construction of the low-rate matrix H_2

Inputs: H_1, $\mathcal{S}_1$, $\mathcal{S}_{1\to 2}$, K, $H_2 = \{\phi\}$
for $i = 1$ to S_{m_2} **do**
  if i-th row of $\mathcal{S}_{1\to 2}$ has two non-zero components $s_{i,j_1}^{(1\to 2)}$, $s_{i,j_2}^{(1\to 2)}$ **then**
    for $\ell = 1$ to m_1/S_{m_1} **do**
      Pick u at random in $\mathcal{A}_{j_1}^{(1)}$ and $v_1, \cdots, v_K$ at random in $\mathcal{A}_{j_2}^{(1)}$ such that $\forall k \in \{1, \cdots, K\}$, $\forall w \in \{1, \cdots, m_1\}$, $h_{v_1,w}^{(1)}.h_{v_2,w}^{(1)} = 0$
      For all $k \in \{1, \cdots, K\}$, count the number $N_{4,k}$ of length-4 cycles in $H_2 \cup \{\underline{h}_u^{(1)} + \underline{h}_{v_k}^{(1)}\}$
      For the index $k^\star$ that minimizes $N_{4,k}$, do $H_2 \leftarrow H_2 \cup \{\underline{h}_u^{(1)} + \underline{h}_{v_{k^\star}}^{(1)}\}$
      Remove u from $\mathcal{A}_{j_1}^{(1)}$ and $v_{k^\star}$ from $\mathcal{A}_{j_2}^{(1)}$
  else
    for $\ell = 1$ to m_1/S_{m_1} **do**
      Pick u at random in $\mathcal{A}_{j_1}^{(1)}$ ($s_{i,j_1}^{(1\to 2)} \neq 0$) and do $H_2 \leftarrow H_2 \cup \{\underline{h}_u^{(1)}\}$, remove u from $\mathcal{A}_{j_1}^{(1)}$
outputs: H_2, N_4 (number of length-4 cycles in H_2)

In Section 4.3.2, we selected the intermediate protograph $\mathcal{S}_{1\to 2}$ that gives the protograph $\mathcal{S}_2$ with highest threshold. We now explain how to construct $H_{1\to 2}$ in order to follow the degree distribution defined by protograph $\mathcal{S}_{1\to 2}$, but also to limit the amount of short cycles in H_2 and to avoid VN elimination. Applying the PEG algorithm directly on $H_{1\to 2}$ would reduce the amount of short cycles on $H_{1\to 2}$ but would not guarantee that the number of cycles in H_2 is reduced as well. As an alternative, the algorithm Proto-Circle we propose is described in Algorithm 2. It constructs one row of H_2 at a time by combining rows of H_1, which can be regarded

as defining the coefficients of the intermediate matrix $H_{1\to2}$. For each new row of H_2, we want to limit the number of short cycles that are added to the parity check matrix H_2.

According to section 4.3.2, each row of the protograph $\mathcal{S}_{1\to2}$ has either 1 or 2 non-zero components. The rows of $\mathcal{S}_{1\to2}$ that have 2 non-zero components indicate that two rows of H_1 of some given types should be combined in order to obtain one row of H_2. More formally, assume that the i-th row of $\mathcal{S}_{1\to2}$ is such that $s_{i,j_1}^{(1\to2)} = 1$ and $s_{i,j_2}^{(1\to2)} = 1$ ($j_1 \neq j_2$). This means that two rows of H_1 of types $A_{j_1}^{(1)}$ and $A_{j_2}^{(1)}$ should be combined in order to obtain one row of H_2 of type $A_i^{(2)}$. For this, we select at random one row $\underline{h}_u^{(1)}$ of H_1 of type $A_{j_1}^{(1)}$ and K rows $\underline{h}_{v_1}^{(1)}, \cdots \underline{h}_{v_K}^{(1)}$ of type $A_{j_2}^{(1)}$ such that $\forall k \in \{1, \cdots, K\}$, $\forall w \in \{1, \cdots, m_1\}$, $h_{v_1,w}^{(1)}.h_{v_2,w}^{(1)} = 0$ (binary AND operation). This condition avoids VN elimination. The algorithm counts the number $N_{4,k}$ of length-4 cycles that would be added if a new row $\underline{h}_u^{(1)} + \underline{h}_{v_k}^{(1)}$ was added to H_2. The number of length-4 cycles in H_2 is computed with the algorithm proposed in [61]. Note that the algorithm can be easily modified to also consider larger cycles. The algorithm then chooses the row combination that adds least cycles in H_2.

Once all the lines of types $A_{j_1}^{(1)}$ and $A_{j_2}^{(1)}$ have been combined, the algorithm passes to the next row of $\mathcal{S}_{1\to2}$ with two non-zero components and repeats the same process. It then processes the rows of $\mathcal{S}_{1\to2}$ with one non-zero component. For instance, assume that row i' of $\mathcal{S}_{1\to2}$ has one non-zero component $s_{i',j'}^{(1\to2)} = 1$. Then, all the lines of H_1 of type $A_{j'}^{(1)}$ are placed into H_2. The placement order does not have any influence on the amount of cycles in the matrix H_2.

After constructing all the rows of H_2, the algorithm counts the total number of length-4 cycles in the newly created H_2. At the end, repeating the algorithm Proto-Circle several times allows us to choose the matrix H_2 with least short cycles.

4.3.4 Construction of the set S'

The intermediate matrix $H_{1\to2}$ follows the structure of the protograph $\mathcal{S}_{1\to2}$. As a result, according to Section 4.3.2, each of its lines has either 1 or 2 non-zero components. Further, the algorithm Proto-Circle introduced in Section 4.3.3 imposes that each row of H_1 participates to exactly one combination for the constructions of the rows of H_2. These two conditions guarantee that $H_{1\to2}$ is full-rank so that the rate-adaptive condition presented in Section 4.1.2 is satisfied. However, in order to completely define the rate-adaptive code $(H_1, H_{1\to2}, S')$, we need to define a set S' of symbols of S that will be sent together with the set $\mathcal{U}$ in order to obtain the rate R_1.

The set S' will serve to solve a system of m_1 equations $\mathcal{U}$ with m_1 unknowns $S \setminus S'$. For each syndrome symbol $u_i \in \mathcal{U}$ of degree d_k in $H_{1\to2}$, we hence decide to put $d_k - 1$ of the d_k CNs connected to u_i into S'. For example, if $u_1 = s_1 \oplus s_2 \oplus s_3$, s_1 and s_2 may be placed into S'. This strategy guarantees that it is always possible to reconstruct the set S from $\mathcal{U}$ and S'. In the above example, it indeed suffices to recover s_3 as $s_3 = u_1 \oplus s_1 \oplus s_2$.

We now count the number of symbols s_i that are placed into S' with this strategy. Since each line of $H_{1\to 2}$ has either 1 or 2 non-zero components, we have $d_k = 1$ or $d_k = 2$. Denote by α the proportion of values u_k of degree 1. We have the following relation between m_1, m_2 and α:

$$m_1 = \alpha m_2 + 2(1-\alpha)m_2.$$

This gives that $\alpha = 2 - \frac{m_1}{m_2}$. Further, according to the code construction proposed in Section 4.3.3, each s_i participates to exactly one equation u_j. As a result, in the above strategy, the set S' is composed by $(1-\alpha)m_2 = m_1 - m_2$ different values s_i, which is exactly what is required by the rate-adaptive construction.

4.4 Generalization to several rates

The above methods construct the matrix H_2 of rate $R_2 < R_1$ from the matrix H_1. In order to obtain lower rates $R_T < R_{T-1} < \cdots < R_2 < R_1$, we need to construct the successive matrices H_t, $t \in \{2, \cdots, T\}$. As initially proposed in [40], the matrices H_t can be constructed recursively from intermediate matrices $H_{t-1\to t}$ such that $H_t = H_{t-1\to t}H_{t-1}$. The intermediate matrices $H_{t-1\to t}$ are constructed by from the method described in Section 4.3.

However, with the method of Section 4.3, the rate values $R_2, \cdots, R_T$ are constrained by the size of the initial protograph $\mathcal{S}_1$. For a protograph $\mathcal{S}_1$ of size $S_{m_1} \times S_n$, the rate granularity is given by

$$r_g = \frac{R_1}{S_{m_1}}. \tag{4.8}$$

For instance, if $R_1 = 1/2$ and $\mathcal{S}_1$ is of size 4×8, only rates $R_2 = 3/8$, $R_3 = 1/4$, $R_4 = 1/8$ can be achieved. This is why, in this section, we propose two alternatives methods that allow to decrease the rate granularity r_g.

4.4.1 Protograph extension

The first method called "protograph extension" consists of lifting the mother protograph $\mathcal{S}_1$ by a factor Z_e, in the same way as for producing a parity check matrix from a given protograph (see Section 3.1.2). This extension produces a protograph $\mathcal{S}'_1$ of size $Z_e S_{m_1} \times Z_e S_n$. For instance, the protograph

$$\mathcal{S}_1 = \begin{bmatrix} 1 & 2 & 1 & 3 \\ 1 & 0 & 2 & 5 \end{bmatrix} \tag{4.9}$$

can be extended as

$$\mathcal{S}'_1 = \begin{bmatrix} 1 & 1 & 1 & 2 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 1 & 2 \\ 1 & 0 & 1 & 4 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 & 4 \end{bmatrix}. \tag{4.10}$$

The protograph $\mathcal{S}_1$ permits to generate an ensemble $\mathcal{H}_1$ of parity check matrices with asymptotic codeword length. According to [26, Theorem 2], all the asymptotic parity check matrices in $\mathcal{H}_1$ have the same decoding performance given by the threshold of $\mathcal{S}_1$. The extended protograph $\mathcal{S}'_1$ generates a code ensemble $\mathcal{H}'_1 \subseteq \mathcal{H}_1$. As a result, the asymptotic matrices in $\mathcal{H}'_1$ have the same decoding performance as the matrices in $\mathcal{H}_1$, and $\mathcal{S}_1$ and $\mathcal{S}'_1$ have the same theoretical threshold.

The above protograph extension allows to consider more rates, since the rate granularity r'_g of $\mathcal{S}'_1$ is given by $r'_g = r_g/Z_e \leq r_g$. However, it is not desirable neither to end up with an extended protograph $\mathcal{S}'_1$ of large size, *e.g.* in the order of magnitude of m_1. Indeed, in this case, the number of possibilities for intermediate protographs $\mathcal{S}_{t-1\to t}$ would also become very large. In addition, it becomes computationally difficult to compute the theoretical thresholds for large protographs. As a result, if the size of $\mathcal{S}'_1$ is large, it will be very difficult to optimize the successive protographs $\mathcal{S}_t$ according to the method described in Section 4.3.2. This is why we now propose a second method that allows to push further the rate granularity improvement.

4.4.2 Anchor rates

In this second method, consider a protograph $\mathcal{S}_1$ of size $S_{m_1} \times S_n$. As a first step, we do the protograph optimization of Section 4.3.2 for all the possible rates

$$R_t = R_1 - \frac{(t-1)R_1}{S_{m_1}}, \tag{4.11}$$

where $t \in \{1, \cdots, S_{m_1}\}$, and $R_{t-1} - R_t = R_1/S_{m_1}$. This produces a sequence of protographs $\mathcal{S}_t$, and the rates R_t are called the anchor rates. We now want to construct all the possible intermediate rates between any R_{t-1} and R_t, with a rate granularity $r_g = R_1/m_1$.

According to Section 4.3.2, the rows of the intermediate protographs $\mathcal{S}_{t-1\to t}$ have either one or two non-zero components. In addition, in order to obtain all the rates R_t defined in (4.11), exactly one row of $\mathcal{S}_{t-1\to t}$ has two non-zero components. This is why, in order to obtain a rate $R_{t-1} - \frac{1}{m_1}$, we propose to combine two rows of the corresponding type in H_{t-1}. The resulting matrix contains the considered row combination, as well as all the non-combined rows of H_{t-1}. As in the algorithm Proto-Circle described in Section 4.3.3, we choose the row combination that minimizes the amount of short cycles that will be added in the resulting matrix. Applying this process recursively allows to obtain all rates $R_{t-1} - kR_1/m_1$, with $k \in \{1, \cdots, m_1/S_{m_1}\}$, and $m_1/S_{m_1} = Z_1$, where Z_1 is the lifting factor. This approach also guarantees that at rate R_t, the resulting matrix follows the structure of protograph $\mathcal{S}_t$.

The anchor rates method allows to obtain a rate granularity $r_g = R_1/m_1$. In the simulation section, the performances of two code construction methods in Section 4.2 and Section 4.3 are compared to LDPCA. And we combine both approaches (protograph extension and anchor rates) in order to obtain an incremental code con-

struction that permits to handle a wide range of statistical relations between the source and the side information.

4.5 Simulation results

4.5.1 Simulation results without protograph

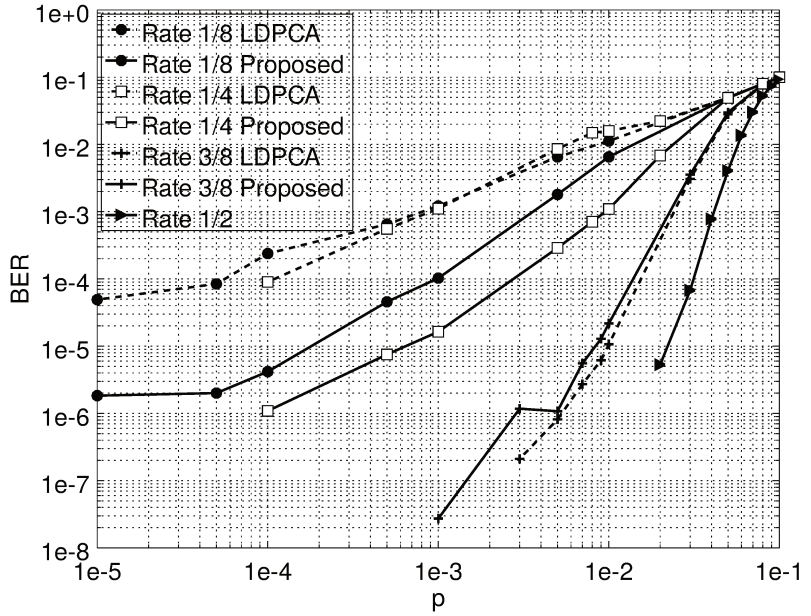

Figure 4.4: BER performance on a BSC channel for code C_1 of size 128×256

	$N4(C_1)$	$N6(C_1)$	$N_4(C_2)$	$N_6(C_2)$
$R = 1/2$	0	1856	0	584
$R = 3/8$ LDPCA	256	5232	204	2256
$R = 3/8$ Proposed	184	6823	83	2232
$R = 1/4$ LDPCA	928	15328	568	5600
$R = 1/4$ Proposed	465	19073	200	6130
$R = 1/8$ LDPCA	2632	67384	2336	42620
$R = 1/8$ Proposed	2425	166227	1193	53101

Table 4.1: Number of length-4 (N4) and length-6 (N6) cycles for the two considered codes

In this section, we evaluate the performance of the first code construction method in Section 4.2 (without protograph) compared to LDPCA. The two codes which we choose to evaluate were obtained from [62]. The first code called C_1 code is of size 128×256. The second code called C_2 code is generated from a WiMax protograph. It is of size 96×192. These two codes have rate $R = 1/2$. For each of the two codes,

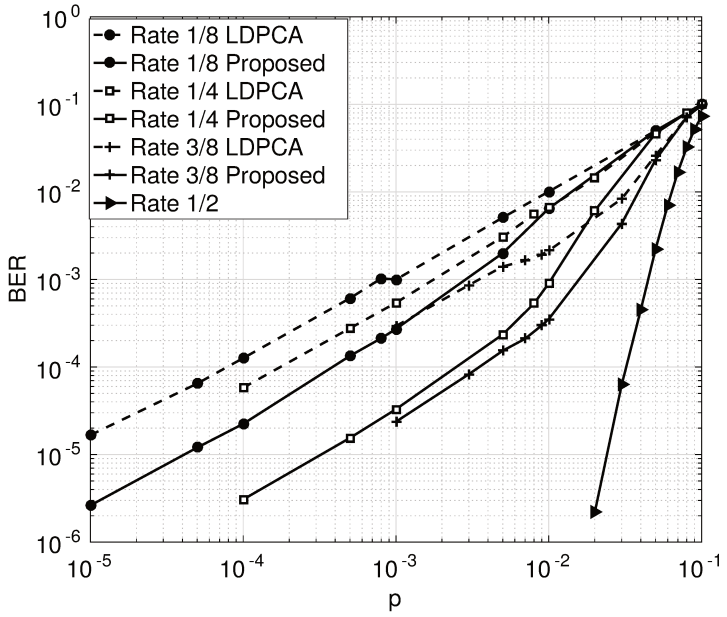

Figure 4.5: BER performance on a BSC channel for code C_2 of size 96×192

we generated three codes of rates 3/8, 1/4, 1/8, both with the LDPCA method and with our construction. In our algorithm, only short cycles of length 4 were considered.

In our simulations, we assumed a binary symmetric channel of parameter p and we evaluated the Bit Error Rate (BER) performance of both LDPCA and our construction for the two considered codes. The results are shown in Figure 4.4 for the code C_1 and in Figure 4.5 for code C_2. In both cases and for almost all rates, we observe that our rate-adaptive construction gives a better performance than the LDPCA. It even outperforms LDPCA by almost one order of magnitude. The only particular case is the rate 3/8 for code C_1. In this case, LDPCA shows slightly better performance than our method. After a cycle analysis given in Table 4.1 from the method of [61], we observe that at all rates, our code construction contains less length-4 cycles than LDPCA, which explains its improved performance. On the opposite, our code construction contains more length-6 cycles than LDPCA. This probably explains why LDPCA works slightly better than our method for the rate 3/8 for C_1. This can probably be improved in the future. We can however conclude that our method works better than LDPCA in almost all the considered cases.

4.5.2 Simulation results with protograph

This section evaluates from Monte Carlo simulations the performance of the proposed rate-adaptive protograph-based construction in Section 4.3. We assume a BSC of parameter p and we consider three binary LDPC codes $\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_3$ constructed from protographs. These codes are set as mother codes for the initial rate

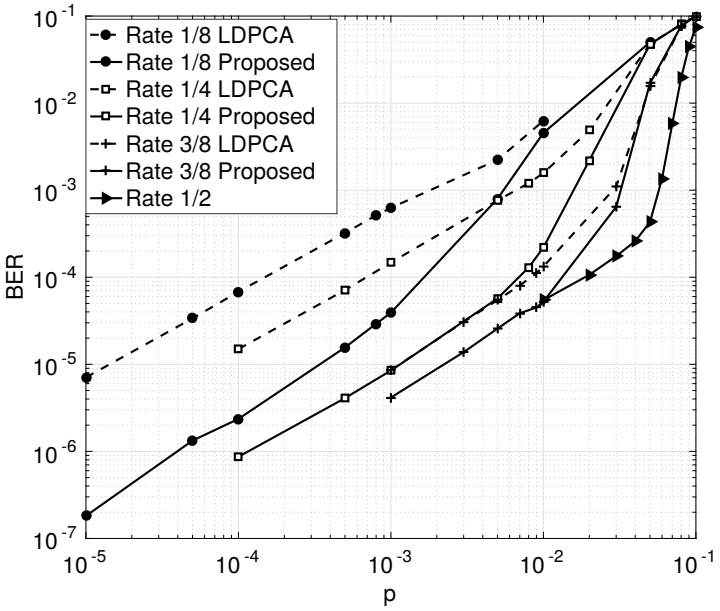

Figure 4.6: BER performance of code $\mathcal{C}_1$ with dimension 248×496 using proposed construction compared with LDPCA

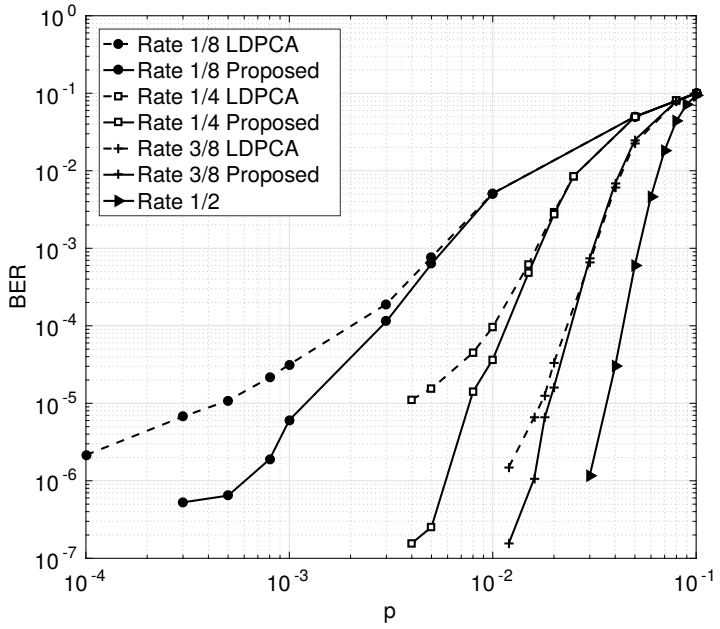

Figure 4.7: BER performance of code $\mathcal{C}_2$ with dimension 256×512 using proposed construction compared with LDPCA

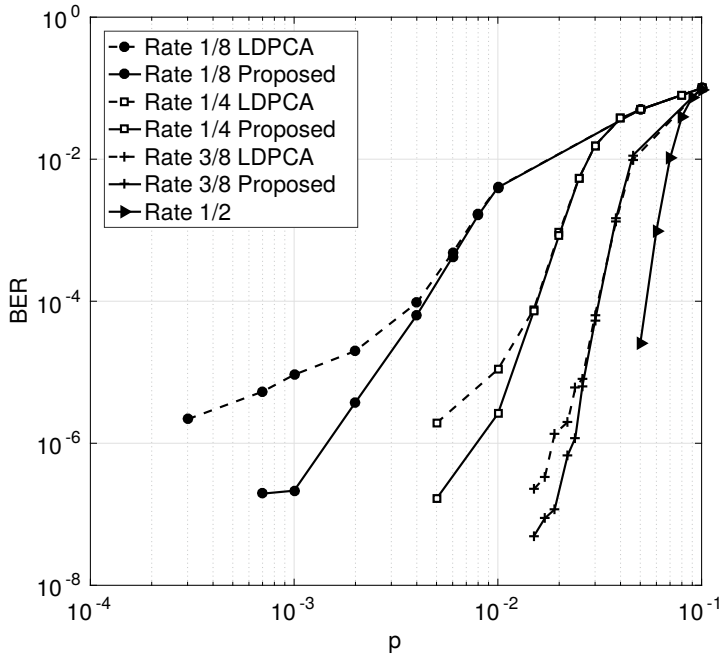

Figure 4.8: BER performance of code $\mathcal{C}_3$ with dimension 512×1024 using proposed construction compared with LDPCA

Rate	LDPCA	Our method
$R = 3/8$	453	455
$R = 1/4$	1216	737
$R = 1/8$	5361	3477

Table 4.2: Number of length-4 cycles for code $\mathcal{C}_1$

$R = 1/2$. The algorithm introduced in Section 4.3.3 then produces the corresponding daughter codes for lower rates 3/8, 1/4, 1/8. In the following, we compare the performance of the obtained rate-adaptive codes with LDPCA.

The first code $\mathcal{C}_1$ is of size 248x496. In order to construct $\mathcal{C}_1$, we first obtained the protograph $\mathcal{S}_1$ of size 2×4 in (4.9) from the Differential Evolution optimization method described in Section 3.4.1. Differential Evolution was applied by considering $V = 60$ elements in the population. This follows [63] which suggests to choose $5D < V < 10D$, where in our case, $D = S_n S_m = 8$. In addition, the number of iterations was set as $L = 100$, and the maximum degree was set as $d_{\max} = 10$. The theoretical threshold of $\mathcal{S}_1$ is equal to $p = 0.094$, which is very close to the maximum value $p = 0.11$ that can be considered at rate 1/2. Protograph $\mathcal{S}_1$ was then extended to the protograph $\mathcal{S}_1'$ of size 4×8 in (4.10) according to the method described in Section 4.4.1. This extension allows ton consider more anchor rates 3/8, 1/4 and

1/8, rather than 1/4 only with $\mathcal{S}_1$.

The parity check matrix of $\mathcal{C}_1$ was constructed from the protograph $\mathcal{S}_1'$ by the PEG algorithm [37]. We then applied our construction method introduced in Section 4.3 in order to obtain lower rates 3/8, 1/4 and 1/8. For this, we first needed to decide which rows of the protograph $\mathcal{S}_{opt1}$ should be combined (see Section 4.3.2) by checking the thresholds of all the possible combinations using Density Evolution. From Density Evolution, we chose row combinations $A_1^{(1)} + A_3^{(1)}$ for rate 3/8 and $A_1^{(1)} + A_3^{(1)}, A_2^{(1)} + A_4^{(1)}$ for rate 1/4, where the $A_i^{(1)}, i = 1, 2, \cdots, S_m$, denotes the rows of $\mathcal{S}_1'$. From the selected row combinations, we then constructed the corresponding matrices of rate 3/8, 1/4, 1/8 from the algorithm Proto-Circle described in Section 4.3.3. This algorithm was applied with $K = 20$ and repeated 10 times in order to choose the low-rate matrices with the least short cycles.

Figure 4.6 shows the Bit Error Rate (BER) performance with respect to the BSC parameter p for the four considered rates for $\mathcal{C}_1$. We observe that our code construction performs better than LDPCA at all the considered rates. Table 4.2 indeed shows that there are less length-4 cycles at rates 1/4 and 1/8 in our construction than in the LDPCA matrices.

The second code $\mathcal{C}_2$ is of size 256×512 and it was generated from another protograph

$$\mathcal{S}_{opt2} = \begin{bmatrix} 2 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 2 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 2 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 2 & 1 & 1 & 1 & 0 \end{bmatrix} \tag{4.12}$$

obtained from Differential Evolution and protograph extension. The codes of lower rates 3/8, 1/4, and 1/8 were constructed by following the same steps as for $\mathcal{C}_1$, according to the construction of Section 4.3. The BER performance of these codes are shown in Figure 4.7 and compared to LDPCA. For this case as well, our construction shows better performance than LDPCA. Finally, the code $\mathcal{C}_3$ is of size 512×1024 and it was generated from the same protograph $\mathcal{S}_{opt2}$ as $\mathcal{C}_2$. Figure 4.8 shows that for $\mathcal{C}_3$ as well, our algorithm perform better than LDPCA at all the considered rates, with a larger code size.

The curves of Figures 4.6, 4.7, 4.8, considered the code performance for the anchor rates given in Section 4.4.2. We then applied the method described in Section 4.4.2 to codes $\mathcal{C}_2$ and $\mathcal{C}_3$ in order to obtain rate granularities of $R_1/m_1 = 9.8 \times 10^{-4}$ for $\mathcal{C}_2$ and $R_1/m_1 = 4.9 \times 10^{-4}$ for $\mathcal{C}_3$, rather than $R_1/S_{m_1} = 0.125$. For this, we considered different values of p, and for every considered value, we generated 1000 couples $(\underline{x}^n, \underline{y}^n)$ from a BSC or parameter p. For every generated couple, we found the minimum rate that permits to decode $\underline{x}^n$ from $\underline{y}^n$ without any error. The same kind of analysis was performed in [55] and [58], with different criterion to measure the rate needed for a given couple $(\underline{x}^n, \underline{y}^n)$. In [55], this rate was determined as the minimum rate such that the decoded codeword $\underline{\hat{x}}^n$ verifies $H^T \underline{\hat{x}}^n = \underline{c}^m$, see (3.1). However, this criterion does not necessarily means that the codeword was correctly decoded ($\underline{\hat{x}}^n$ can be different from $\underline{x}^n$), and this is why we

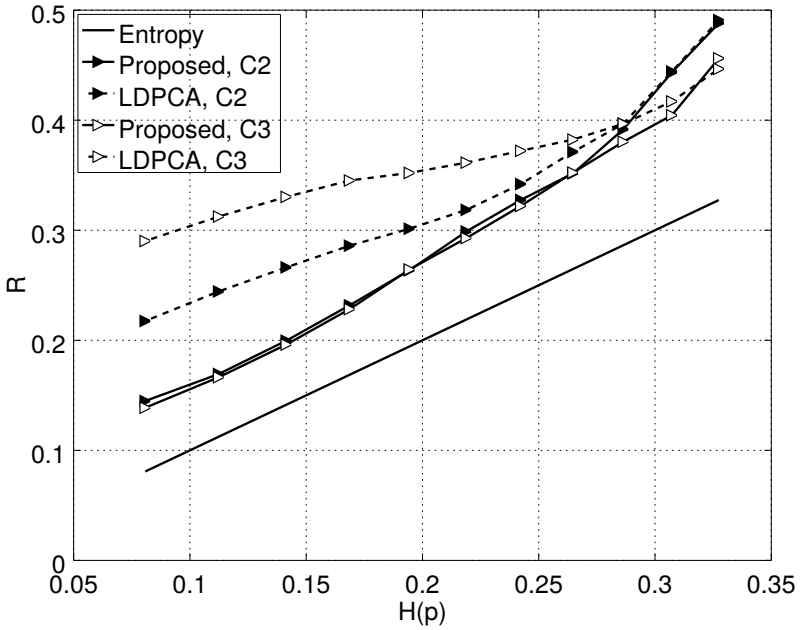

Figure 4.9: Required rate R with respect to $H(p)$ for LDPCA and for our method, for codes $\mathcal{C}_2$ and $\mathcal{C}_3$

do not consider it here. In [58], the required rate was determined as the minimum rate that gives a BER lower than 10^{-6}. This is equivalent to our criterion, since one uncorrectly decoded bit gives a BER of 2.0×10^{-3} for $\mathcal{C}_2$, and of 1.0×10^{-3} for $\mathcal{C}_3$.

At the end, Figure 4.11 represents the average rates needed for the considered values of p with respect to $H(p)$. We first observe that our method shows a loss compared to the optimal rate $H(p)$. This rate loss is expected since we consider relatively short codeword length 512 for $\mathcal{C}_2$ and 1024 for $\mathcal{C}_3$. In addition, for the same codes $\mathcal{C}_2$ and $\mathcal{C}_3$, LDPCA shows a much more significant rate loss compared to our method, which was also expected from the results of Figures 4.7 and 4.8. This shows that our construction combined with the anchor rates method is valid and outperforms LDPCA at all the considered values of p.

Name	Protograph	Theoretical threshold
Code 1	$A_1^{(2)} + A_2^{(2)}, A_3^{(2)}, A_4^{(2)}$	0.051
Code 2	$A_1^{(2)} + A_2^{(2)}, A_3^{(2)} + A_4^{(2)}, A_4^{(2)}$	0.049
Code 3	$A_1^{(2)} + A_2^{(2)}, A_3^{(2)} + A_4^{(2)}, A_2^{(2)}$	0.050
Code 4	$2A_1^{(2)} + A_2^{(2)}, A_3^{(2)}, A_4^{(2)}$	0.049
Code 5	$A_1^{(2)} + A_2^{(2)}, 2A_3^{(2)}, A_4^{(2)}$	0.049

Table 4.3: Protographs with different constraints for rate 3/8. The protographs are described in the form of combination of the lines $A_i^{(2)}$ of the original protograph $\mathcal{S}_{opt2}$.

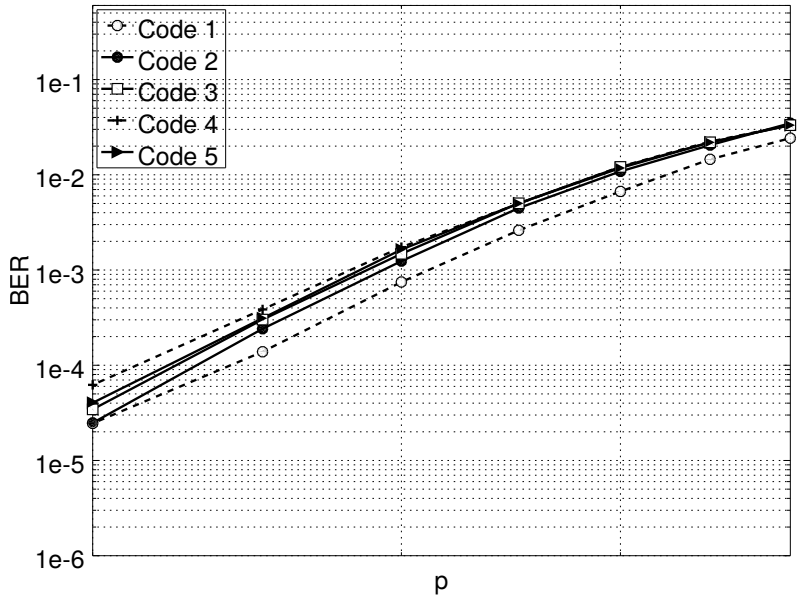

Figure 4.10: Performance of five code constructions with different protographs at rate 3/8

To finish, we discuss the influence of the constraints expressed in Section 4.3.2 for the code construction. These constraints were that in the intermediate protograph $\mathcal{S}_{1\to 2}$, each column has only 1 non-zero components, and that all the non-zero components are equal to 1. In order to discuss the influence of these constraints, we again consider the extended protograph $\mathcal{S}_{opt2}$ of rate 1/2 given in (4.12) and we consider the construction of a daughter code of rate 3/8. If the above constraints are not satisfied, there are many possibilities to construct such daughter codes. We consider 5 different protographs that can be obtained from $\mathcal{S}_{opt2}$ for the daughter codes. These protographs are given in Table 4.3, with their theoretical thresholds. Code 1 corresponds to a protograph that satisfies the constraints of Section 4.3.2. We see that the other protographs have either more than 1 non-zero component per column (Codes 2 and 3), or a non-zero component equal to 2 (Codes 4 and 5). Note that we selected give protographs with close theoretical thresholds, in order to compare the finite-length performance of the considered codes.
The performance of the five considered codes are compared in Figure 4.10. We see that despite the fact that the five codes have very close theoretical thresholds, their BER performance vary from one code to another. As expected, Code 1 shows the best performance. This can be explained by the fact that its parity check matrix is more sparse than the other ones. We also note that Codes 4 and 5 for which the protographs contain non-zero components equal to 2 show the worst performance. This shows that, in the considered case, the above constraints are reasonable.

4.5.3 Simulation results at full rate range

In Section 3.8.3, we construct an initial code of rate $R = 1/2$ and apply either the LDPCA method to obtain rates lower than $1/2$ or the Rateless method for rates higher than $1/2$ to avoid the drawbacks of Rateless and LDPCA. Now we evaluate this construction method and compare it with LDPCA. In this section, we suppose that the source X is uniformly distributed. The source X is obtained from a BSC with crossover probability p. We first construct two mother codes of rate $R = 1/2$. The first mother code is denoted by $O1$ with length $n = 512$ and constructed from a protograph

$$S = \begin{bmatrix} 0 & 2 & 3 & 1 \\ 2 & 0 & 3 & 2 \end{bmatrix} \tag{4.13}$$

by applying the methods with two steps described in [50] which permit to construct quasi-cyclic codes. The second mother code is a WIMAXLike code of length $n = 192$ obtained from [1]. For each mother code, we construct two family of compatible codes with a rate variation of $1/n$. The first code family is obtained by applying Rateless for rates higher than $1/2$ and LDPCA for rates smaller than $1/2$. The second code family is obtained by using the proposed method in this article which takes place of LDPCA. During the application of algorithm Circle, we use $K = 50$.
In order to evaluate the performances of different code families, we will apply the following method which is initially proposed in [55]. For different values of p, we generate 1000 vectors x^n and y^n. For each set of vectors (x^n, y^n), we search the minimum rate which permit to decode x^n without error. Then we calculate the average value of these rates.

The results are presented in Figure 4.11. For the rates smaller than $1/2$, we find out a significant gain of our method compared to LDPCA. We can also find out that the performance of the method Rateless for rates higher than $1/2$. We find a disconnecting of the curve for higher rates, which implies that the method Rateless could also be optimized.

4.6 Conclusion

In this Chapter, we presented our two novel rate-adaptive solutions. The two proposed solutions show a better performance than LDPCA, especially for short codes (less than 1000 bits) that are particularly sensitive to short cycles. Now we will apply these solutions in the real FTV system.

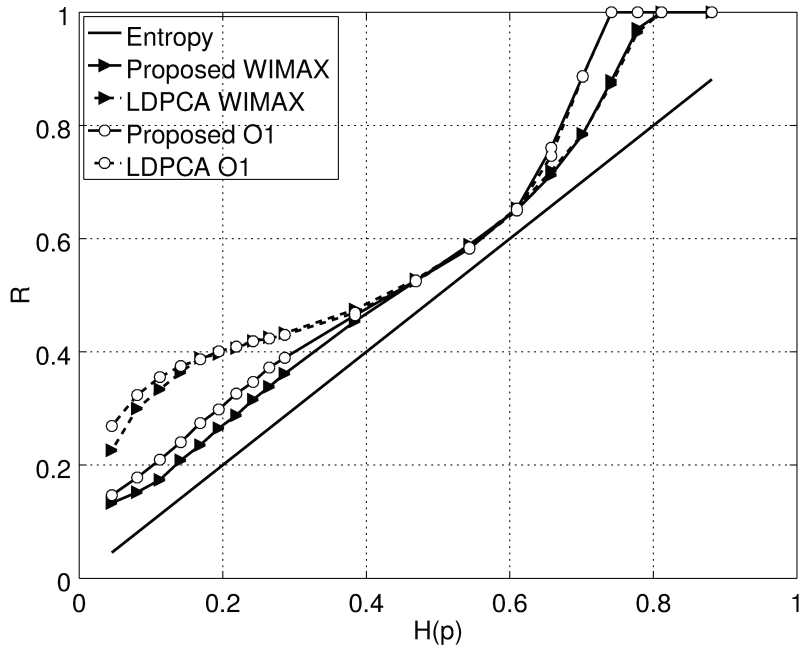

Figure 4.11: Obtained rates based on entropy, for two mother codes: $O1$ and WIMAX.

Chapter 5

Application to FTV

Contents

Equipped with rate-adaptive code constructions and decoders for lossless source coding, we now apply our solutions to the problem of Free Viewpoint Television (FTV). Members of the InterCom project provided us with files containing source and side information vectors. These vectors were generated by a video encoder they implemented for Free Viewpoint Television. They however did not implement the lossless part, and this is our objective here.

5.1 Generation of video files

In our project, since we deal with applications like FTV or the 360 degree video [64], the original video images are spherical. In order to compress these videos, the project members proposed a solution that is described in [64]. In this solution described in Figure 5.1, the spherical video images are first projected onto 2D images. Then several operations are applied such as: Discrete Cosine Transformation (DCT), Quantization, Prediction. These operations allow us to obtain a source vector $\underline{X}^n$ to be transmitted losslessly to the decoder. They also provide prediction $Y^{(1)}, \ldots, Y^{(J)}$ of X that can serve as side informations at the decoder. These predictions can be

built from previous viewpoints received by the decoder, and they are also known by the encoder. The obtained sequence $\underline{X}^n$ will then be encoded with an LDPC matrix H, and the syndrome $\underline{U}^m$ will be stored in the server. When users send their requests for desired views, a rate-adaptive extraction is realized and the adapted length syndrome $\underline{U}^{m_j}$ will be sent, depending on the side information $Y^{(j)}$ available at the decoder.

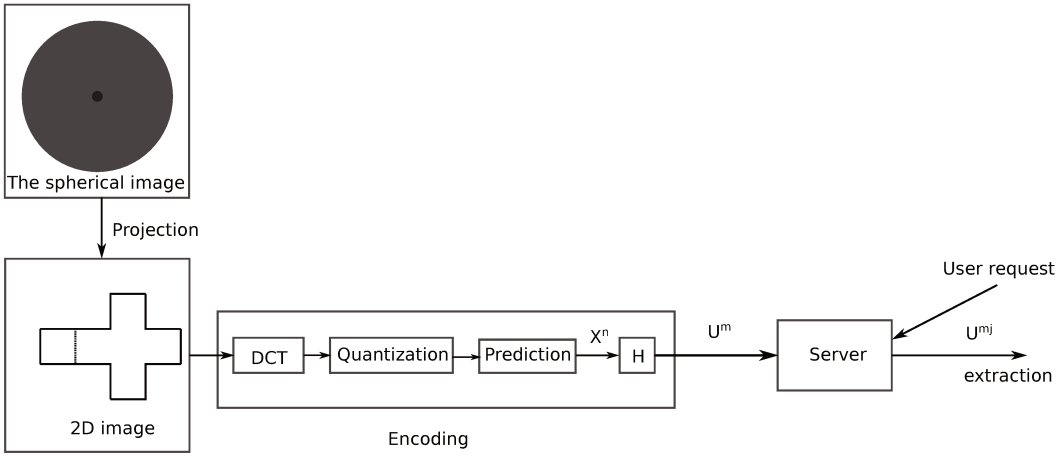

Figure 5.1: FTV compression scheme

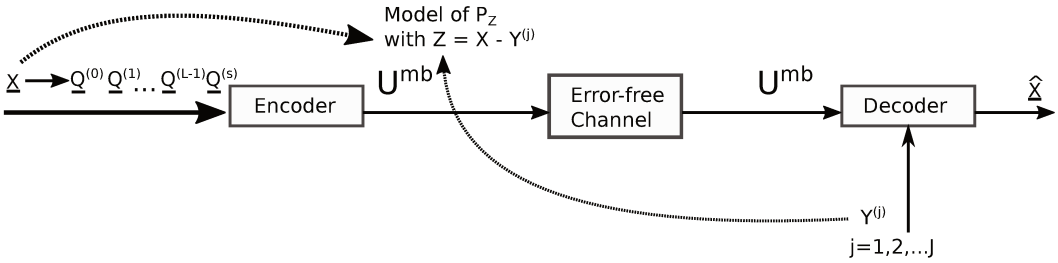

Figure 5.2: FTV Application

The files we were provided by the project members give realization of source vector $\underline{X}^n$ and corresponding side informations $Y^{(1)}, \ldots, Y^{(J)}$. The source and side information vectors are sequences of n symbols which take integer values. But our codes and decoder treat bits. Therefore, some adaptions need to be made in order to apply our solutions to the FTV problem.
In this chapter, we denote one symbol-based source vector by $\underline{X}^n$ with

$$\underline{X}^n = [X_1 X_2 \ldots X_k \ldots X_n] \tag{5.1}$$

We denote the binary representation of k-th symbol X_k by $Q_k^{(0)} Q_k^{(1)} \ldots Q_k^{(b)} \ldots Q_k^{(L-1)}$, where $b \in [0, L-1]$, with $b = 0$ is the Least Significant Bit (LSB) and $b = L-1$ is the Most Significant Bit (MSB), and L is the bit length of X_k. The value of each

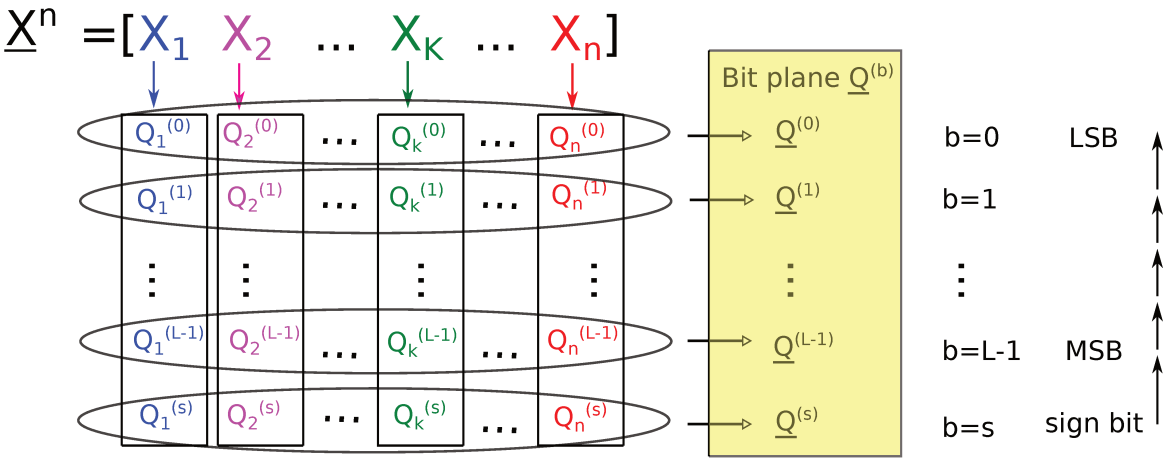

Figure 5.3: X^n: from symbol-based to bit planes

bit is given by

$$Q_k^{(b)} = (X_k/2^b)\%2 \tag{5.2}$$

where "%2" denotes the modulo 2 operation.
One symbol-based side information vector is denoted by $\underline{Y}^n$ where $\underline{Y}^n = [Y_1, Y_2, \ldots, Y_n]$. In practice, there are several possible side information $Y^{(j)}$. But in the following, for simplicity, we describe our solution for one side information $\underline{Y}^n$.
The source bits with the same significant power are called bit planes. For example for bit position b, the bit plane $\underline{Q}^{(b)}$ is:

$$\underline{Q}^{(b)} = [Q_1^{(b)}, \ldots Q_n^{(b)}]^T \tag{5.3}$$

In next section, we show how to transform the symbol-based model into a bit-based model. We also show that passing from the symbol-based model to the bit-based model theoretically does not degrade the coding performance. Then we describe the joint statistical model we consider between X and Y.

5.2 Symbol-based and bit-based models

In [65], the symbol-based model and the bit-based model are compared. It is shown that the minimal achievable rates for symbol-based coding schemes and bit-based coding schemes are identical. The conditional entropies of the two models can indeed be related by:

$$\begin{aligned} H(\underline{X}^n|\underline{Y}^n) &= H(Q^{(0)}, \ldots, Q^{(L-1)}|\underline{Y}^n) \\ &= \sum_{b=0}^{L-2} H(Q^{(b)}|\underline{Y}^n, Q^{(b+1)}, \ldots, Q^{(L-1)}) + H(Q^{(L-1)}|\underline{Y}^n) \end{aligned} \tag{5.4}$$

This suggests that, in the bit-based decoder, bit plane $\underline{Q}^{(b)}$ is decoded with conditional probability $P(Q^{(b)} \mid \underline{Y}^n, Q^{(b+1)}, \ldots, Q^{(L-1)})$. In practice, the bit planes $\underline{Q}^{(b)}$

will be decoded one after each other, from $\underline{Q}^{(L-1)}$ to $\underline{Q}^{(0)}$. Previously decoded bit planes will be used to decode current bit plane $\underline{Q}^{(b)}$. This bit plane strategy was also considered in [66, 67].
In the following, we first propose a symbol-based statistical model between $\underline{X}^n$ and $\underline{Y}^n$. We then show how to obtain a bit-based model from this symbol-based model.

5.3 Joint statistical model between $\underline{X}^n$ and $\underline{Y}^n$

In FTV, as in standard video compression, the statistical relation between the source $\underline{X}^n$ and the side information $\underline{Y}^n$ varies a lot from frame to frame and from video to video [68, 69, 70, 14, 71]. A well-chosen statistical model is important for the LDPC decoder as described in the previous chapters. In the following, for simplicity, we suppose that X_k are all i.i.d. and Y_k are i.i.d. too. An additive model $Z = X - Y$ is supposed here just as people often do in the practice. A Laplacian model and a Q-ary symmetric model are considered here for Z.

Before all the views of video are stored at the server, the server has access to all the X and Y while encoding them. So it can know the model parameters of them at this moment, these parameters will then be stored and transmitted with the codeword together. Surely this requires additional bits for the model parameters, but that costs just a little if we consider a model with little model parameters, just as we do here.

5.3.1 Laplacian Model

The Laplacian model is often considered in video coding to model the statistical relation between the source and side information. The Laplacian density probability function of distribution $\mathcal{L}(\mu, \delta^2)$ is given by

$$f_Z(z) = \frac{1}{\sqrt{2\delta^2}} \exp - \frac{\sqrt{2}|z-\mu|}{\delta^2} \tag{5.5}$$

where μ is the mean of Z, and δ^2 is the variance. The expressions of μ and δ^2 are given by

$$\mu = \mathbf{E}(Z) \tag{5.6}$$

$$\delta^2 = \mathbf{E}(Z-\mu)^2 \tag{5.7}$$

For a side information Y, the density of $P(X_k|Y_k) = P(X_k - Y_k|Y_k) = P(Z_k \mid Y_k)$. We suppose that Z_k and Y_k are independent, which gives $P(Z_k \mid Y_k) = P(Z_k)$ and simplifies our problem. Then we model $P(Z_k)$ by using a Laplacian distribution $\mathcal{L}(\mu, \delta^2)$. The values of μ and δ^2 can be estimated from vectors $\underline{Z}^n$ as

$$\hat{\mu} = \mathbf{Median}(\underline{Z}^n) \tag{5.8}$$

$$\hat{\delta}^2 = \frac{1}{n}\sum_{k=1}^{n}(Z_k - \hat{\mu})^2 \tag{5.9}$$

The Laplacian model however has two issues. First it is a continuous model while our data are discrete. Second, when the variance δ^2 is small, the density (5.5) applied to values of Z_k is numerically equal to 0, which poses problems in our decoder. This is why we also consider a second model described in section 5.3.2.

5.3.2 Q-ary symmetric Model

The probability mass function of a Q-ary symmetric [72] model $\mathcal{Q}(q, Z_{\max}, Z_{\min})$ is given by

$$f_Z(z) = \begin{cases} q & \text{if } z = 0 \\ \frac{1-q}{Z_{\max}-Z_{\min}} & \text{otherwise} \end{cases} \tag{5.10}$$

where $q \in [0, 1]$ is a constant, $Z_{\max}$ is the maximum value of Z, $Z_{\min}$ is the minimum value of Z. The value of q can be estimated as

$$\hat{q} = \frac{N_{Z0}}{n} \tag{5.11}$$

where N_{Z0} is the number of symbol $Z_k = X_k - Y_k = 0$, and n is the total length of sequence $\underline{Z}^n$.

5.4 Probability calculation from symbol-based model to bit-based model

In [65], it is shown how to obtain the bit-based conditional probability $P(Q_k^{(b)} \mid Y_k = y, Q_k^{(b+1)}, \ldots, Q_k^{(L-1)})$ from the symbol-based conditional entropy $P(X_j \mid Y_k)$. Let us denote by P_Z the probability distribution of Z. It corresponds to f_Z either given by (5.5) or (5.10). The bit-based probability can be obtained as follows:

$$\begin{aligned} P_0 &= P\left(Q_k^{(b)} = 0 \mid Y_k = y, Q_k^{(b+1)}, \ldots, Q_k^{(L-1)}\right) \\ &= \frac{P\left(Q_k^{(b)} = 0, Q_k^{(b+1)}, \ldots, Q_k^{(L-1)} \mid Y_k = y\right)}{P\left(Q_k^{(b+1)}, \ldots, Q_k^{(L-1)} \mid Y_k = y\right)} \\ &= \frac{P\left(Q_k^{(b)} = 0, Q_k^{(b+1)}, \ldots, Q_k^{(L-1)} \mid Y_k = y\right)}{P\left(Q_k^{(b)} = 0, Q_k^{(b+1)}, \ldots, Q_k^{(L-1)} \mid Y_k = y\right) + P\left(Q_k^{(b)} = 1, Q_k^{(b+1)}, \ldots, Q_k^{(L-1)} \mid Y_k = y\right)} \end{aligned} \tag{5.12}$$

and

$$P_1 = P\left(Q_k^{(b)} = 1 \mid Y_k = y, Q_k^{(b+1)}, \ldots, Q_k^{(L-1)}\right)$$
$$= \frac{P\left((Q_k^{(b)} = 1, Q_k^{(b+1)}, \ldots, Q_k^{(L-1)} \mid Y_k = y\right)}{P\left(Q_k^{(b+1)}, \ldots, Q_k^{(L-1)} \mid Y_k = y\right)}$$
$$= \frac{P\left(Q_k^{(b)} = 1, Q_k^{(b+1)}, \ldots, Q_k^{(L-1)} \mid Y_k = y\right)}{P\left(Q_k^{(b)} = 0, Q_k^{(b+1)}, \ldots, Q_k^{(L-1)} \mid Y_k = y\right) + P\left(Q_k^{(b)} = 1, Q_k^{(b+1)}, \ldots, Q_k^{(L-1)} \mid Y_k = y\right)} \tag{5.13}$$

where,

$$P\left(Q_k^{(b)} = 0, Q_k^{(b+1)}, \ldots, Q_k^{(L-1)} \mid Y_k = y\right) = \sum_{i=0}^{2^b-1} P\left(\sum_{k=b+1}^{L-1} Q_k^{(k)} \cdot 2^k + i \mid Y_k = y\right)$$
$$= \sum_{i=0}^{2^b-1} P_Z\left(\sum_{k=b+1}^{L-1} Q_k^{(k)} \cdot 2^k + i - y\right) \tag{5.14}$$

$$P\left(Q_k^{(b)} = 1, Q_k^{(b+1)}, \ldots, Q_k^{(L-1)} \mid Y_k = y\right) = \sum_{i=2^b}^{2^{b+1}-1} P\left(\sum_{k=b+1}^{L-1} Q_k^{(k)} \cdot 2^k + i \mid Y_k = y\right)$$
$$= \sum_{i=2^b}^{2^{b+1}-1} P_Z\left(\sum_{k=b+1}^{L-1} Q_k^{(k)} \cdot 2^k + i - y\right) \tag{5.15}$$

As a result, we just need to calculate $P(Q_k^{(b)} = 0, Q_k^{(b+1)}, \ldots, Q_k^{(L-1)} \mid Y_k = y)$ and $P(Q_k^{(b)} = 1, Q_k^{(b+1)}, \ldots, Q_k^{(L-1)} \mid Y_k = y)$ by using the probability distribution $P_Z(\cdot)$, and then normalize them to get P_0 and P_1.

5.5 Decoding scheme with bit-based source and symbol-based side information

We now present the lossless coding scheme we consider for FTV. In FTV, the value of $\underline{X}^n$ can be negative, so we need to add a sign bit. If X_k is converted into $L+1$ bits, we have

$$X_k \leftrightarrow Q_k^{(0)} Q_k^{(1)} \ldots Q_k^{(b)} \ldots Q_k^{(L-1)} Q_k^{(s)} \tag{5.16}$$

$Q_k^{(s)}$ is the sign bit of X_k. The value of $Q_k^{(s)}$ is given by

$$Q_k^{(s)} = \begin{cases} 0 & \text{if } X_k \geq 0 \\ 1 & \text{if } X_k < 0 \end{cases} \tag{5.17}$$

$\underline{X}^n$ is encoded and decoded from bit plane to bit plane from $b = s$, $b = L-1$ to $b = 0$. For the encoding of b-th bit plane $\underline{Q}^{(b)}$, we use a parity check matrix H_b with dimension $m_b \times n$. We compute a syndrome $\underline{U}^{(b)}$ as $\forall b, \underline{U}^{(b)} = \underline{Q}^{(b)} \cdot H_b^T$, where the syndrome $\underline{U}^{(b)}$ is of length m_b.

The bit plane $\underline{Q}^{(s)}$ is encoded and decoded first. The received syndrome $\underline{U}^{(s)}$, $\underline{Y}^n$ and the estimated model parameters will be used for the decoding. In the decoding, we use the bit probability of $Q_k^{(s)}$ in $\underline{Q}^{(s)}$, calculated as

$$P(Q_k^{(s)} = 0 \mid Y_k = y) = \sum_{i=0}^{2^L-1} P_Z(i-y) \tag{5.18}$$

$$P(Q_k^{(s)} = 1 \mid Y_k = y) = \sum_{i=-2^L+1}^{-1} P_Z(i-y) \tag{5.19}$$

Then the received syndrome $\underline{U}^{(b)}$, $\underline{Y}^n$, the estimated model parameters ($\hat{\mu}$ and $\hat{\delta}^2$ for Laplacian model, q, $Z_{\max}$ and $Z_{\max}$ for Q-ary symmetric model) and the previous decoded bit planes $\underline{\hat{Q}}^{(b+1)} \dots \underline{\hat{Q}}^{(L-1)} \underline{\hat{Q}}^{(s)}$ will be used to decode the current bit plane $\underline{\hat{Q}}^{(b)}$. By applying (5.12) and (5.13), the bit probability of $Q_k^{(b)}$ in $\underline{Q}^{(b)}$ is calculated as

$$P\left(Q_k^{(b)} = 0 \mid Y_k = y, \hat{Q}_j^{(b+1)}, \dots, \hat{Q}_j^{(L-1)}, \hat{Q}_j^{(s)}\right) = \sum_{i=0}^{2^b-1} P_Z\left(\left(1 - 2\hat{Q}_j^{(s)}\right) \cdot \left(\sum_{k=b+1}^{L-1} \hat{Q}_j^{(k)} \cdot 2^k + i\right) - y\right) \tag{5.20}$$

$$P\left(Q_k^{(b)} = 1 \mid Y_k = y, \hat{Q}_j^{(b+1)}, \dots, \hat{Q}_j^{(L-1)}, \hat{Q}_j^{(s)}\right) = \sum_{i=2^b}^{2^{b+1}-1} P_Z\left(\left(1 - 2\hat{Q}_j^{(s)}\right) \cdot \left(\sum_{k=b+1}^{L-1} \hat{Q}_j^{(k)} \cdot 2^k + i\right) - y\right) \tag{5.21}$$

When the bit planes $\underline{\hat{Q}}^{(0)}$, $\underline{\hat{Q}}^{(1)}$, $\dots$, $\underline{\hat{Q}}^{(L-1)}$, $\underline{\hat{Q}}^{(s)}$ are all decided, $\underline{X}^n$ can be reconstructed with

$$\underline{\hat{X}}^n = \hat{Q}_1^{(0)} \hat{Q}_1^{(1)} \dots \hat{Q}_1^{(L-1)} \hat{Q}_1^{(s)} \quad , \dots , \quad \hat{Q}_n^{(0)} \hat{Q}_n^{(1)} \dots \hat{Q}_n^{(L-1)} \hat{Q}_n^{(s)} \tag{5.22}$$

5.6 Rate-adaptive construction

To encode each bit plane, we construct H_b by using our rate-adaptive method described in Chapter 4. Starting with the same mother code H, we construct H_b for all the bit planes so as to decode $\underline{\hat{Q}}^{(b)}$ without error. This can be done since the

encoder knows the possible side information $Y^{(j)}$ and can choose the transmission rate accordingly.
This rate-adaptive construction strategy is applied on all the possible side informations $Y^{(j)}, j \in \{1, \ldots, J\}$. We evaluate the performance of our rate-adaptive scheme as follows.
The transmission rate of b-th bit plane by using j-th side information $Y^{(j)}$ is :

$$R_b^{(j)} = \frac{m_b^{(j)}}{n} \tag{5.23}$$

The total transmission rate by using j-th side information $Y^{(j)}$ is:

$$R_{tot}^{(j)} = \sum_{b=0}^{L-1} R_b^{(j)} + R_s^{(j)} \tag{5.24}$$

For $\forall b \in 0, \ldots, L-1$, the storage rate of b-th bit plane is:

$$S_b = \max_{j \in \{1,\ldots,J\}} R_b^{(j)} \tag{5.25}$$

and the storage rate of sign bit plane is:

$$S_s = \max_{j \in \{1,\ldots,J\}} R_s^{(j)} \tag{5.26}$$

The total storage rate is:

$$S_{tot} = \sum_{b=0}^{L-1} S_b + S_s \tag{5.27}$$

As shown in Section 5.2, there is no loss in terms of transmission rate. But on the other hand, there will be some loss on storage rate. There is no loss on storage rate only if we can reach at the maximum value of H_{symbol} as shown in [15, 43]. But in this decoding scheme, only a sum of maximum values of H_{bits} can be achieved. Some loss on the storage rate will surely happen in order to realise this rate-adaptive decoding scheme.

In our simulation, the value of the transmission rates $R_b^{(j)}$ for the most significant bits can be really small. This means that the bit plane $\underline{Q}^{(b)}$ can almost be deduced from previous bit planes and from corresponding symbols in $Y^{(j)}$. So if the LLR values already allow to perfectly detect $\underline{\hat{Q}}^{(b)}$ without need for decoding, we define $R_b = 0$.

5.7 Simulation results

For FTV, for a source X, several side informations $Y^{(j)}$ are available. The number of available side informations $Y^{(j)}$ is denoted by J. Each side information $Y^{(j)}$ gives

a different empirical probability distribution on $P(X - Y^{(j)})$ with different parameters for Laplacian model or a Q-ary symmetric model.
In our simulations, we observed that the Laplacian model would generate too many 0 for LLR information, so we applied the Q-ary symmetric model to avoid this disadvantage. Since the amount of dependency between X and $Y^{(j)}$ can give different results, we tested three types of video files with low dependency, middle dependency and high dependency.

5.7.1 X and $Y^{(j)}$ with high dependency

In the first file we used, we have $J = 8$. The estimated parameter values of the Q-ary symmetric model for each side information are shown in Table 5.1.

j	$\hat{q}$	$Z_{\max}$	$Z_{\min}$
1	0.989	1	-2
2	0.989	1	-2
3	0.990	1	-2
4	0.989	1	-2
5	0.990	1	-2
6	0.989	1	-2
7	0.989	1	-2
8	0.989	4	-2

Table 5.1: Q-ary symmetric model for X and $Y^{(j)}$ with high dependency

We observe that the value of $\hat{q}$ does not vary much with the side information.

After testing $\underline{X}^n$, We observe that $L + 1 = 8$ bits are sufficient to completely represent the source X. The obtained transmission rates $R_b^{(j)}$ for all the bit planes b ($b \in [0, \ldots, L-1, L]$) are shown in Table 5.2. We find out that the rates for the most significant bits are smaller.

Given the Q-ary symmetric model of $P(X|Y^{(j)})$, we can calculate the model entropy

$$H(X|Y^{(j)})_{\text{model}} = -q\log_2(q) - (1-q)\log_2\left(\frac{1-q}{Z_{\max} - Z_{\min}}\right) \tag{5.28}$$

For a given sequence $\underline{Z}^n$, we can also evaluate an empirical entropy as

$$H(\underline{Z}^n) = \sum_{i=0}^{N_Z} -p(i) \times \log_2\left(p(i)\right) \tag{5.29}$$

where N_Z is the number of different values that can take z_k in $\underline{Z}^n$, and $p(i)$ is the corresponding empirical frequency of element i.
As seen in Chapter 2, Slepian-Wolf theorem tells us that $R \geq H(X \mid Y)$. In our problem, the conditional entropy $H(X \mid Y)$ can be simplified as $H(Z)$. In addition,

j	R_0	R_1	R_2	R_3	R_4	R_5	R_6	$R_7(R_s)$
1	0.090820	0.019531	0	0	0	0	0	0
2	0.108398	0.015625	0	0	0	0	0	0
3	0.091796	0.015625	0	0	0	0	0	0
4	0.108398	0.015625	0	0	0	0	0	0
5	0.091796	0.015625	0	0	0	0	0	0
6	0.092773	0.015625	0.015625	0	0	0	0	0
7	0.090820	0.019531	0	0	0	0	0	0
8	0.091796	0.017578	0	0.015625	0	0	0	0
S_b	0.108398	0.019531	0.015625	0.015625	0	0	0	0
S_{tot}	0.159179							

Table 5.2: The obtained rates (bit/symbol) for each bit plane and each side information $Y^{(j)}$

the value of $H(Z)$ can be estimated by $H(\underline{Z}^n)$. It means that,

$$R_{tot} \geq H(\underline{Z}^n) \tag{5.30}$$

The Q-ary symmetric model is an approximate model of sequence $\underline{Z}^n$ with three model parameters $q, Z_{\max}, Z_{\min}$. The entropy of Q-ary model $H(X|Y^{(j)})_{\text{model}}$ is larger than $H(\underline{Z}^n)$ as it uses less parameters than considering the empirical frequency and thus describes less precisely the statistics of $\underline{Z}^n$. The Q-ary symmetric model is used at the decoder to calculate P_0, P_1, and the corresponding LLR. Therefore, using the Q-ary symmetric model rather than the empirical frequency will result in different values P_0 and P_1. But we know that using mismatched values of P_0 and P_1 does not degrade much the decoder performance. Perhaps a model which follows exactly the empirical probability distribution can achieve a slightly smaller R_{tot} , but it also means that more model parameters should be sent and this will however increase the transmission rate.

The simulation results of total transmission rates R_{tot}, model entropy $H(X|Y^{(j)})_{\text{model}}$ and empirical entropies $H(\underline{Z}^n)$ are shown in Table 5.3.

We can find that for all the side information

$$H(X|Y^{(j)})_{\text{model}} \geq H(\underline{Z}^n) \tag{5.31}$$

$$R_{tot} \geq H(\underline{Z}^n) \tag{5.32}$$

This is expected from the theoretical results. We can see that R_{tot} is close to $H(X|Y^{(j)})_{\text{model}}$ and $H(\underline{Z}^n)$. Sending no information (rate 0 in Table 5.2) when LLR is sufficient to retrieve the bit planes clearly helps to achieve this good result. The storage rate S_{tot} is just a little larger than the maximum value of R_{tot}, which means our decoder can help to realise the rate-adaptive decoding for all the possible side informations $Y^{(j)}$ with a small extra rate cost.

j	R_{tot}	$H(X\|Y^{(j)})_{\text{model}}$	$H(\underline{Z}^n)$
1	0.1103	0.1027	0.0997
2	0.1240	0.1027	0.0998
3	0.1074	0.0947	0.0925
4	0.1240	0.1027	0.0998
5	0.1074	0.0947	0.0925
6	0.1240	0.1027	0.1001
7	0.1103	0.1027	0.0997
8	0.1250	0.1134	0.1036
max	0.1250	0.1134	0.1036

Table 5.3: Total rate compared with entropy in case of high dependency

5.7.2 X and $Y^{(j)}$ with middle dependency

In this file we have $J = 8$. The informations of the Q-ary symmetric model for each side information is shown in Table 5.4.

j	$\hat{q}$	$Z_{\max}$	$Z_{\min}$
1	0.950	10	-16
2	0.950	7	-5
3	0.948	24	-24
4	0.950	18	-18
5	0.950	7	-11
6	0.951	12	-12
7	0.950	10	-13
8	0.950	9	-7

Table 5.4: Q-array model for X and $Y^{(j)}$ with middle dependency

A bit length of $L + 1 = 7$ is sufficient to completely represent the source X. The obtained transmission rates R_b for all the bit planes b ($b \in [0, \ldots, L-1, L]$) are shown in Table 5.5.

The simulation results of total transmission rates R_{tot}, model entropy $H(X|Y^{(j)})_{\text{model}}$ and empirical entropies $H(\underline{Z}^n)$ are shown in Table 5.6.

We can find that for all the side information

$$H(X|Y^{(j)})_{\text{model}} \geq H(\underline{Z}^n) \tag{5.33}$$

$$R_{tot} \geq H(\underline{Z}^n) \tag{5.34}$$

It follows the theoretical results too. And the total transmission rate R_{tot} is still close to the theoretical limit $H(\underline{Z}^n)$.

j	R_0	R_1	R_2	R_3	R_4	R_5	$R_6(R_s)$
1	0.219726	0.194335	0.083984	0.0341796	0.015625	0	0
2	0.193359	0.189453	0.116210	0.0283203	0.015625	0	0
3	0.191406	0.222656	0.130859	0.0371093	0.022460	0.015625	0
4	0.222656	0.195312	0.0869140	0.046875	0.020507	0	0
5	0.189453	0.183593	0.1435546	0.016601	0	0	0
6	0.182617	0.194335	0.0839843	0.037109	0.015625	0	0
7	0.208984	0.200195	0.1025390	0.034179	0.015625	0	0
8	0.177734	0.200195	0.1191406	0.036132	0.015625	0	0
S_b	0.222656	0.222656	0.1435546	0.046875	0.022460	0.015625	0
S_{tot}	0.6738266						

Table 5.5: The obtained rates (bit/symbol) for each bit plane and for each side information $Y^{(j)}$

j	R_{tot}	$H(X\|Y^{(j)})_{\text{model}}$	$H(\underline{Z}^n)$
1	0.5478	0.5197	0.4462
2	0.5429	0.4641	0.4386
3	0.6201	0.5829	0.4630
4	0.5722	0.5431	0.4473
5	0.5332	0.4932	0.4495
6	0.5136	0.5053	0.4526
7	0.5615	0.5109	0.4435
8	0.5488	0.4848	0.4459
max	0.6201	0.5829	0.4630

Table 5.6: Total rate compared with entropy in case of middle dependency

The storage rate S_{tot} is just a little larger than the maximum value of R_{tot}, which means our decoder can help to realise the rate-adaptive decoding for all the possible side informations $Y^{(j)}$ with a small extra rate cost.

5.7.3 X and $Y^{(j)}$ with low dependency

In this file we have $J = 8$. The informations of the Q-ary symmetric model for each side information is shown in Table 5.7.

A bit length of $L + 1 = 9$ is sufficient to completely represent the source X. The obtained transmission rates R_b for all the bit planes b ($b \in [0, \ldots, L-1, L]$) are shown in Table 5.8.

The simulation results of total transmission rates R_{tot}, model entropy $H(X|Y^{(j)})_{\text{model}}$ and empirical entropies $H(\underline{Z}^n)$ are shown in Table 5.9.

We can find that for all the side information

$$H(X|Y^{(j)})_{\text{model}} \geq H(\underline{Z}^n) \tag{5.35}$$

j	$\hat{q}$	$Z_{\max}$	$Z_{\min}$
1	0.669	29	-22
2	0.668	26	-21
3	0.668	26	-21
4	0.667	29	-20
5	0.668	26	-61
6	0.669	50	-21
7	0.668	26	-21
8	0.668	26	-25

Table 5.7: Q-ary symmetric model for X and $Y^{(j)}$ with low dependency

j	R_0	R_1	R_2	R_3	R_4	R_5	R_6	R_7	$R_8(R_s)$
1	1	0.575195	0.278320	0.183593	0.209960	0.015625	0	0	0
2	1	0.580078	0.279296	0.191406	0.175781	0	0	0	0
3	1	0.556640	0.279296	0.185546	0.175781	0.015625	0	0	0
4	1	0.551757	0.275390	0.209960	0.219726	0	0	0	0
5	1	0.593750	0.279296	0.152343	0.126953	0	0	0	0
6	1	0.576171	0.279296	0.149414	0.173828	0.163085	0.015625	0	0
7	1	0.580078	0.279296	0.188476	0.175781	0.015625	0	0	0
8	1	0.575195	0.279296	0.181640	0.177734	0	0	0	0
S_b	1	0.593750	0.279296	0.209960	0.219726	0.163085	0.015625	0	0
S_{tot}	2.481442								

Table 5.8: The obtained rates (bit/symbol) for each bit plane and each side information $Y^{(j)}$

j	R_{tot}	$H(X\|Y^{(j)})_{\text{model}}$	$H(\underline{Z}^n)$
1	2.2626	2.7873	1.9017
2	2.2265	2.7549	1.9113
3	2.2128	2.7549	1.9133
4	2.2568	2.7812	1.8988
5	2.1523	3.0490	1.9133
6	2.3574	2.9449	1.9050
7	2.2392	2.7549	1.9106
8	2.2138	2.7939	1.9133
max	2.3574	3.0490	1.9133

Table 5.9: Total rate compared with entropy in case of low dependency

$$R_{tot} \geq H(\underline{Z}^n) \tag{5.36}$$

It is also in accordance with the theoretical results. And our methods work well, it

generates a total rate R_{tot} quite close to theoretical limit $H(\underline{Z}^n)$.
The storage rate S_{tot} is just a little larger than the maximum value of R_{tot}, which means our decoder can help to realise the rate-adaptive decoding for all the possible side informations $Y^{(j)}$ with a small extra rate cost.

5.8 Conclusion

In this Chapter, we applied our rate-adaptive solutions to the lossless part of FTV. We transformed a symbol-based model to a bit-based model in order to be able to apply our coding solution. The simulation results show that the required transmission and storage rates are just a little larger than the theoretical performance. It means that we successfully incorporate the proposed lossless code construction into a complete lossy source coding scheme that was developped for FTV in the framework of the InterCom project.

The on-going work is testing this solution with a large database of video files and comparing the performance of transmission rate and entropy for different dependency relations between the source and the side information. Also, as we know our Q-ary model is just a simplified model of sequence $\underline{Z}^n$, we should look for other models with both limited defining parameters and a closer probability distribution to the empirical frequency of $\underline{Z}^n$, in order to achieve better performance in terms of transmission rate.

Chapter 6

Conclusions & Perspectives

Conclusions

Many multimedia applications such as Free Viewpoint Television (FTV) use a distant service provider that offers customized services depending on the user request. The main challenge is the efficient storage of a huge amount of data and the real-time extraction of a small fraction of these data upon request. In some applications such as FTV, the requests previously addressed by the user can help to optimize both the storage and the extraction. The problem can thus be seen as a source coding problem with side information at the user side. This PhD thesis fits into this context. It is part of the CominLabs project InterCom that focuses on solutions for massive random access to subsets of correlated data.

In Chapter 3, we investigate practical lossless source coding schemes with side information based on Low Density Parity Check (LDPC) codes. We first analysed the performance of LDPC codes with density evolution, and we constructed efficient finite-length LDPC codes with the PEG algorithm. We also analysed the limitations of rate-compatible channel coding methods and rate-adaptive source coding methods like Rateless and LDPCA. Rateless codes perform poorly at low coding rates while LDPCA is not adapted to high-rates.In this thesis, we combine both methods to construct rate-adaptive LDPC codes offering a wide range of rates. However LDPCA does not allow to optimize the code degree distribution, nor to control the amount of short cycles at all rates. This is why we propose two novel rate-adaptive LDPC code constructions to replace the LDPCA part.

In Chapter 4, these two novel rate-adaptive LDPC code constructions are presented. The first construction replaces the LDPCA accumulator by intermediate graphs that combine the syndrome bits in order to obtain lower rate codes. This method allows to reduce the amount of short cycles in the codes and it shows a great performance improvement compared to standard solutions. However it only considers unstructured finite-length code constructions, that is without design of the degree distributions of the lower rate codes. The second construction introduces a novel design method that allows to select the photographs of the intermediate graphs so as to optimize the decoding performance of all the codes constructed at all rates of interest. We also propose a new algorithm called Proto-Circle that constructs the intermediate graphs according to their protographs, while minimizing the amount

of short cycles in the codes at all the considered rates. Simulation results show improved performance compared to LDPCA.

In Chapter 5, we applied our rate-adaptive solutions to the lossless part of FTV. We transformed a symbol-based model to a bit-based model in order to be able to apply our coding solution. The simulation results show that the required transmission and storage rates are just a little larger than the theoretical performance. It means that we successfully incorporate the proposed lossless code construction into a complete lossy source coding scheme that was developped for FTV in the framework of the InterCom project.

Perspectives

In the future, several problems related to this thesis may be considered.

1. Comparison to other multiview video coding standards
   As we know, there exists a lot of video coding standards like H264, H265, or HEVC. Some multiview video coding schemes have been proposed before. Our solution and these standards may be compared in terms of achieved transmission and storage rates.

2. Other families of channel codes
   LDPC codes, Turbo codes and Polar codes are among the most efficient channel coding methods. They all have good decoding performances and show different properties in terms of decoding latency, depending on the code length, etc. It could be interesting to investigate the application of Turbo codes and Polar into the FTV problem, and then, to compare the three methods.

3. Latency
   Since FTV may also be used in a real-time video transmission system, the problem of latency need to be considered. The main latency comes at the encoding step that stores the videos and generates the model parameters at the server. Large quantity of views and large number of users may increase this latency. Doing less complex encoding operations may help us to improve this latency. In addition, transmission of views over a real network (packet loss, delay in transmission, etc.) may also be taken into account.

4. Machine Learning
   Machine learning has known an increasing success over the last years. The choice of line combinations in the rate-adaptive LDPC code constructions may be done by relying on machine learning. It may help us to better choose the combinations in order to improve the code performance at all rates.

Bibliography

[1] "https://www.uni-kl.de/channel-codes/channel-codes-database/more-ldpc-codes/." (Cited on pages iii, 30, 31 and 56.)

[2] M. Tanimoto, "FTV: Free-viewpoint television," vol. 27, no. 6, pp. 555–570, Jul. 2012. (Cited on page 5.)

[3] D. Tian, P. Lai, P. Lopez, and C. Gomila, "View synthesis techniques for 3D videos," *Proceedings of the International Society for Optical Engineering*, vol. 7443, 2009. (Cited on page 5.)

[4] L. Toni, T. Maugey, and P. Frossard, "Correlation-aware packet scheduling in multi-camera networks," vol. 16, no. 2, pp. 496–509, 2014. (Cited on page 5.)

[5] D. Slepian and J. Wolf, "Noiseless coding of correlated information sources," *IEEE Transactions on Information Theory*, vol. 19, no. 4, pp. 471–480, July 1973. (Cited on page 6.)

[6] S. C. Draper and E. Martinian, "Compound conditional source coding, Slepian-Wolf list decoding, and applications to media coding," in *IEEE International Symposium on Information Theory*, 2007. (Cited on pages 6, 7 and 15.)

[7] A. Eckford and W. Yu, "Rateless Slepian-Wolf Codes," in *Conference Record of the Thirty-Sixth Asilomar Conference on Signals, Systems and Computers*, 2005, pp. 1757 – 1761. (Cited on page 6.)

[8] V. Stankovic, A.D.Liveris, Z. Xiong, and C. Georghiades, "On code design for the Slepian-Wolf problem and lossless multiterminal networks," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1495 –1507, april 2006. (Cited on page 6.)

[9] S. Jalali, S. Verdú, and T. Weissman, "A universal scheme for Wyner-Ziv coding of discrete sources," *IEEE Transactions on Information Theory*, vol. 56, no. 4, pp. 1737–1750, 2010. (Cited on page 6.)

[10] K. Iwata, "An information-spectrum approach to rate-distortion function with side information," in *IEEE International Symposium on Information Theory, Proceedings.*, 2002, p. 156. (Cited on page 6.)

[11] C. Heegard and T. Berger, "Rate distortion when side information may be absent," *IEEE Transactions on Information Theory*, vol. 31, no. 6, pp. 727–734, Nov 1985. (Cited on page 6.)

[12] R. Zamir, "The rate loss in the wyner-ziv problem," *IEEE Transactions on Information Theory*, vol. 42, no. 6, pp. 2073–2084, 1996. (Cited on page 6.)

[13] 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, no. 1, pp. 1–10, 1976. (Cited on page 6.)

[14] F. Bassi, A. Fraysse, E. Dupraz, and M. Kieffer, "Rate-distortion bounds for wyner–ziv coding with gaussian scale mixture correlation noise," *IEEE Transactions on Information Theory*, vol. 60, no. 12, pp. 7540–7546, 2014. (Cited on pages 6 and 62.)

[15] E. Dupraz, A. Roumy, T. Maugey, and M. Kieffer, "Rate-storage regions for extractable source coding with side information," *Physical Communication*, p. 100845, 2019. (Cited on pages 7, 15 and 66.)

[16] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h. 264/avc video coding standard," *IEEE Transactions on circuits and systems for video technology*, vol. 13, no. 7, pp. 560–576, 2003. (Cited on page 7.)

[17] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (hevc) standard," *IEEE Transactions on circuits and systems for video technology*, vol. 22, no. 12, pp. 1649–1668, 2012. (Cited on page 7.)

[18] A. Sgarro, "Source coding with side information at several decoders," *IEEE Transactions on Information Theory*, vol. 23, no. 2, pp. 179–182, 1977. (Cited on page 7.)

[19] G. Cheung, A. Ortega, and N. Cheung, "Interactive streaming of stored multiview video using redundant frame structures," *IEEE Transactions on Image Processing*, vol. 3, no. 3, pp. 744–761, Mar. 2011. (Cited on page 7.)

[20] S. C. Draper, "Universal incremental slepian-wolf coding," in *Allerton Conference on Communication, control and computing*, 2004, pp. 1757 – 1761. (Cited on page 7.)

[21] R. Gallager, "Low-density parity-check codes," *IRE Transactions on information theory*, vol. 8, no. 1, pp. 21–28, 1962. (Cited on pages 7 and 17.)

[22] T. Richardson and R. Urbanke, *Modern coding theory*. Cambridge Univ Press, 2008. (Cited on page 7.)

[23] M. Davey and D. MacKay, "Low Density Parity Check codes over GF(q)," in *Information Theory Workshop, Proceedings.*, 1998, pp. 70–71. (Cited on page 7.)

[24] A. Bennatan and D. Burshtein, "Design and analysis of nonbinary LDPC codes for arbitrary discrete-memoryless channels," *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 549–583, 2006. (Cited on page 7.)

[25] Ardakani, M. and Kschischang, F.R., "A more accurate one-dimensional analysis and design of irregular LDPC codes," *IEEE Transactions on Communications*, vol. 52, no. 12, pp. 2106–2114, 2004. (Cited on page 7.)

[26] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE transactions on information theory*, vol. 47, no. 2, pp. 619–637, 2001. (Cited on pages 7, 8, 18, 22, 24, 25, 28 and 48.)

[27] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Transactions on information theory*, vol. 47, no. 2, pp. 599–618, 2001. (Cited on page 7.)

[28] S.-Y. Chung, T. J. Richardson, and R. L. Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a gaussian approximation," *IEEE Transactions on Information theory*, vol. 47, no. 2, pp. 657–670, 2001. (Cited on page 7.)

[29] A. D. Liveris, Z. Xiong, and C. N. Georghiades, "Compression of binary sources with side information at the decoder using ldpc codes," *IEEE communications letters*, vol. 6, no. 10, pp. 440–442, 2002. (Cited on pages 7 and 17.)

[30] R. Bhattar, K. Ramakrishnan, and K. Dasgupta, "Density Evolution Technique for LDPC Codes in Slepian-Wolf Coding of Nonuniform Sources," *International Journal of Computer Applications IJCA*, vol. 7, no. 8, pp. 1–7, 2010. (Cited on page 7.)

[31] E. Dupraz, A. Roumy, and M. Kieffer, "Source coding with side information at the decoder and uncertain knowledge of the correlation," *IEEE Transactions on Communications*, vol. 62, no. 1, pp. 269–279, 2013. (Cited on pages 7 and 17.)

[32] Z. Xiong, A. D. Liveris, and S. Cheng, "Distributed source coding for sensor networks," *IEEE signal processing magazine*, vol. 21, no. 5, pp. 80–94, 2004. (Cited on page 7.)

[33] J. Thorpe, "Low-Density Parity-Check (LDPC) codes constructed from protographs," *IPN progress report*, vol. 42, no. 154, pp. 42–154, 2003. (Cited on pages 7, 18, 20 and 30.)

[34] S.-Y. Chung, G. D. Forney, T. J. Richardson, R. Urbanke *et al.*, "On the design of low-density parity-check codes within 0.0045 db of the shannon limit," *IEEE Communications letters*, vol. 5, no. 2, pp. 58–60, 2001. (Cited on pages 8, 18 and 24.)

[35] J. Chen and M. P. Fossorier, "Density evolution for two improved bp-based decoding algorithms of ldpc codes," *IEEE communications letters*, vol. 6, no. 5, pp. 208–210, 2002. (Cited on page 8.)

[36] E. Dupraz, V. Savin, and M. Kieffer, "Density evolution for the design of non-binary low density parity check codes for slepian-wolf coding," *IEEE Transactions on Communications*, vol. 63, no. 1, pp. 25–36, 2014. (Cited on page 8.)

[37] X.-Y. Hu, E. Eleftheriou, and D. M. Arnold, "Regular and irregular progressive edge-growth tanner graphs," *IEEE Transactions on Information Theory*, vol. 51, no. 1, pp. 386–398, Jan 2005. (Cited on pages 8, 18, 20, 29 and 53.)

[38] S.-Y. Shin, M. Jang, S.-H. Kim, and B. Jeon, "Design of binary LDPCA codes with source revealing rate-adaptation," *IEEE Communications Letters*, vol. 16, no. 11, pp. 1836–1839, 2012. (Cited on pages 8 and 34.)

[39] A. W. Eckford and W. Yu, "Rateless Slepian-Wolf codes," in *in Asilomar conference on signals, systems and computers*, 2005, pp. 1757–1761. (Cited on pages 8 and 34.)

[40] Z. Mheich and E. Dupraz, "Short length non-binary rate-Adaptive LDPC codes for Slepian-Wolf source coding," in *IEEE Wireless Communications and Networking Conference*, 2018. (Cited on pages 8, 38, 39 and 47.)

[41] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley & Sons, 2012. (Cited on pages 12 and 13.)

[42] D. Slepian and J. Wolf, "Noiseless coding of correlated information sources," *IEEE Transactions on Information Theory*, vol. 19, no. 4, pp. 471–480, July 1973. (Cited on pages 14 and 17.)

[43] A. Roumy and T. Maugey, "Universal lossless coding with random user access: the cost of interactivity," in *2015 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2015, pp. 1870–1874. (Cited on pages 15 and 66.)

[44] J. Chen, D.-k. He, and A. Jagmohan, "On the duality between slepian–wolf coding and channel coding under mismatched decoding," *IEEE Transactions on Information Theory*, vol. 55, no. 9, pp. 4006–4018, 2009. (Cited on page 17.)

[45] Y. Polyanskiy, H. V. Poor, and S. Verdú, "Channel coding rate in the finite blocklength regime," *IEEE Transactions on Information Theory*, vol. 56, no. 5, p. 2307, 2010. (Cited on page 17.)

[46] D. G. Mitchell, R. Smarandache, and D. J. Costello, "Constructing good qc-ldpc codes by pre-lifting protographs," in *Information Theory Workshop (ITW), 2012 IEEE*. IEEE, 2012, pp. 202–206. (Cited on pages 18 and 20.)

[47] J. Zhang and M. P. Fossorier, "A modified weighted bit-flipping decoding of low-density parity-check codes," *IEEE Communications Letters*, vol. 8, no. 3, pp. 165–167, 2004. (Cited on pages 18 and 21.)

[48] T. Richardson and R. Urbanke, "The renaissance of gallager's low-density parity-check codes," *IEEE Communications Magazine*, vol. 41, no. 8, pp. 126–131, Aug 2003. (Cited on pages 18 and 23.)

[49] B. M J Liner, "Ldpc codes-a brief tutorial," 01 2005. (Cited on pages 18 and 21.)

[50] D. G. Mitchell, R. Smarandache, and D. J. Costello, "Quasi-cyclic LDPC codes based on pre-lifted protographs," *IEEE Transactions on Information Theory*, vol. 60, no. 10, pp. 5856–5874, 2014. (Cited on pages 20 and 56.)

[51] M. Gorgoglione, V. Savin, and D. Declercq, "Optimized puncturing distributions for irregular non-binary ldpc codes," in *2010 International Symposium On Information Theory & Its Applications*. IEEE, 2010, pp. 400–405. (Cited on page 26.)

[52] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of global optimization*, vol. 11, no. 4, pp. 341–359, 1997. (Cited on page 26.)

[53] J. Ha, J. Kim, and S. W. McLaughlin, "Rate-compatible puncturing of low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 50, no. 11, pp. 2824–2836, 2004. (Cited on page 32.)

[54] M. Yazdani and A. H. Banihashemi, "On construction of rate-compatible low-density parity-check codes," in *2004 IEEE International Conference on Communications (IEEE Cat. No. 04CH37577)*, vol. 1. IEEE, 2004, pp. 430–434. (Cited on pages 32 and 33.)

[55] D. Varodayan, A. Aaron, and B. Girod, "Rate-adaptive codes for distributed source coding," *Signal Processing*, vol. 86, no. 11, pp. 3123–3130, 2006. (Cited on pages 33, 34, 35, 53 and 56.)

[56] C. Yu and G. Sharma, "Improved low-density parity check accumulate (LDPCA) codes," *IEEE Transactions on Communications*, vol. 61, no. 9, pp. 3590–3599, 2013. (Cited on page 34.)

[57] M. Yang, W. E. Ryan, and Y. Li, "Design of efficiently encodable moderate-length high-rate irregular LDPC codes," *IEEE Transactions on Communications*, vol. 52, no. 4, pp. 564–571, 2004. (Cited on page 34.)

[58] F. Cen, "Design of degree distributions for LDPCA codes," *IEEE Communications Letters*, vol. 13, no. 7, pp. 525–527, 2009. (Cited on pages 35, 53 and 54.)

[59] K. Kasai, T. Tsujimoto, R. Matsumoto, and K. Sakaniwa, "Rate-compatible Slepian-Wolf coding with short non-binary LDPC codes," in *Data Compression Conference*, 2010, pp. 288–296. (Cited on page 36.)

[60] F. Ye, Z. Mheich, E. Dupraz, and K. Amis, "Optimized short-length rate-adaptive LDPC Codes for Slepian-Wolf source coding," in *International Conference on Telecommunication (ICT)*, 2018. (Cited on pages 38 and 39.)

[61] Y. Mao and A. H. Banihashemi, "A heuristic search for good low-density parity-check codes at short block lengths," in *IEEE International Conference on Communications*, vol. 1, 2001, pp. 41–44. (Cited on pages 41, 46 and 50.)

[62] M. Helmling, S. Scholl, F. Gensheimer, T. Dietz, K. Kraft, S. Ruzika, and N. Wehn, "Database of Channel Codes and ML Simulation Results," www.uni-kl.de/channel-codes, 2017. (Cited on page 49.)

[63] R. Storn and K. Price, "Minimizing the real functions of the icec'96 contest by differential evolution," in *Proceedings of IEEE International Conference on Evolutionary Computation*, 1996, pp. 842–844. (Cited on page 52.)

[64] N. M. Bidgoli, T. Maugey, and A. Roumy, "Evaluation framework for 360-degree visual content compression with user view-dependent transmission," in *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019, pp. 3756–3760. (Cited on page 59.)

[65] R. P. Westerlaken, S. Borchert, R. K. Gunnewiek, and R. L. Lagendijk, "Analyzing symbol and bit plane-based ldpc in distributed video coding," in *2007 IEEE International Conference on Image Processing*, vol. 2, Sep. 2007, pp. II – 17–II – 20. (Cited on pages 61 and 63.)

[66] Y. Yang, V. Stankovic, Z. Xiong, and W. Zhao, "Multiterminal source code design based on slepian-wolf coded quantization," in *Proc. Allerton*, vol. 4, 2004. (Cited on page 62.)

[67] Y. Vatis, S. Klomp, and J. Ostermann, "Inverse bit plane decoding order for turbo code based distributed video coding," in *2007 IEEE International Conference on Image Processing*, vol. 2. IEEE, 2007, pp. II–1. (Cited on page 62.)

[68] C. Brites, J. Ascenso, and F. Pereira, "Modeling correlation noise statistics at decoder for pixel based wyner-ziv video coding," in *in Proc. of PCS 2006*. Citeseer, 2006. (Cited on page 62.)

[69] T. Maugey, J. Gauthier, B. Pesquet-Popescu, and C. Guillemot, "Using an exponential power model forwyner ziv video coding," in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2010, pp. 2338–2341. (Cited on page 62.)

[70] M. Vaezi and F. Labeau, "Improved modeling of the correlation between continuous-valued sources in ldpc-based dsc," in *2012 Conference Record of the Forty Sixth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*. IEEE, 2012, pp. 1659–1663. (Cited on page 62.)

[71] R. P. Westerlaken, S. Borchert, R. K. Gunnewiek, and R. L. Lagendijk, "Dependency channel modeling for a ldpc-based wyner-ziv video compression scheme," in *2006 International Conference on Image Processing*. IEEE, 2006, pp. 277–280. (Cited on page 62.)

[72] C. Weidmann and G. Lechner, "A fresh look at coding for q-ary symmetric channels," *IEEE Transactions on Information Theory*, vol. 58, no. 11, pp. 6959–6967, 2012. (Cited on page 63.)

Chapter 7

Résumé de thèse en français

Résumé de thèse en Français

Titre : Codage de sources de Slepian-Wolf utilisant des codes LDPC et application à la télévision interactive

Mots clés : Codes LDPC, Codage de source, Rendement compatible, Codage, FTV.

1. Introduction

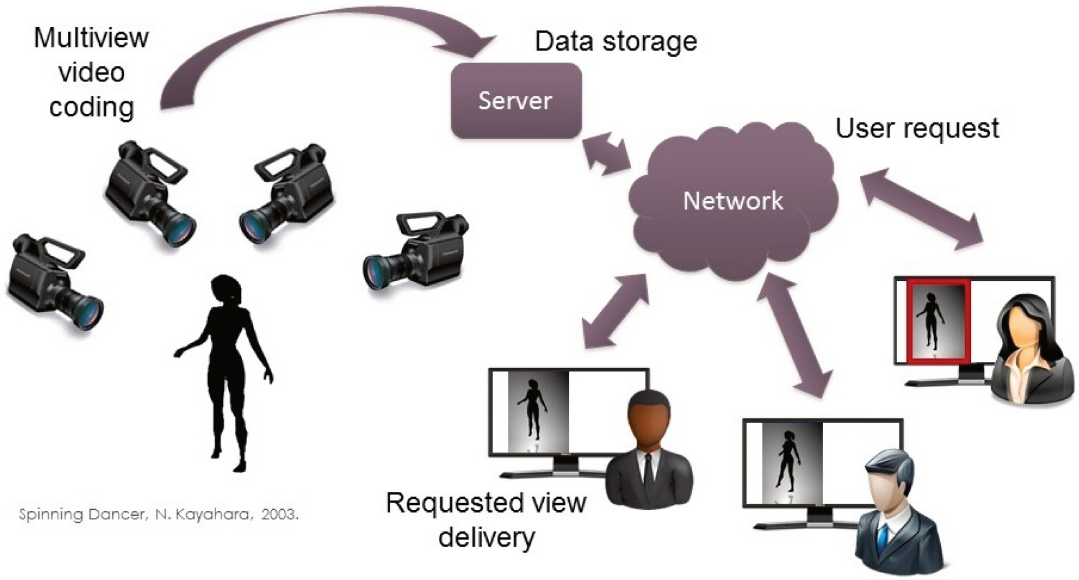

Figure 1: La Télévision Interactive

La Télévision Interactive (FTV) est un service de vidéo à la demande qui permet au client de choisir l'angle de vue de la vidéo. Un des principaux défis est de stocker un énorme volume de données dans un serveur avec un outil de compression efficace et d'en extraire une petite partie à la demande et en temps réel, sans devoir décoder l'intégralité des données. L'utilisateur, de son côté, possède déjà des vues, issues de ses demandes précédentes, et cette connaissance peut être exploitée pour la compression et le décodage. Le projet CominLabs InterCom s'inscrit dans ce contexte. Le principal objectif est de proposer des solutions pour l'accès aléatoire à des sous-ensembles de données massives corrélées. Intégrée à ce projet, cette thèse s'intéresse au problème de codage de sources sans perte avec information adjacente du côté de l'utilisateur, dans le but de développer des schémas pratiques adaptés à l'application visée.

Une modélisation de ce problème de codage de sources est illustrée par la figure 2. Elle suppose que le client mémorise les vues précédentes, notées $Y^{(j)}$ et appelées informations adjacentes. Ces informations sont utilisées pour décoder la source X (qui correspond à la vue demandée). Des méthodes de compression classiques de type Huffman, Lempel-Ziv, H264, HEVC, etc ... n'exploitent pas d'informations adjacentes et ne peuvent donc pas s'appliquer à notre problématique. Il est donc nécessaire de trouver d'autres solutions.

Les objectifs de cette thèse sont

- d'étudier des schémas de codage de source sans perte adaptés à l'application FTV.
- de construire des codes qui prennent en compte différentes informations adjacentes $Y^{(j)}$.
- d'appliquer les solutions développées à l'application Vidéo en 360 degrés.

2. Rappels de théorie de l'information en codage de source

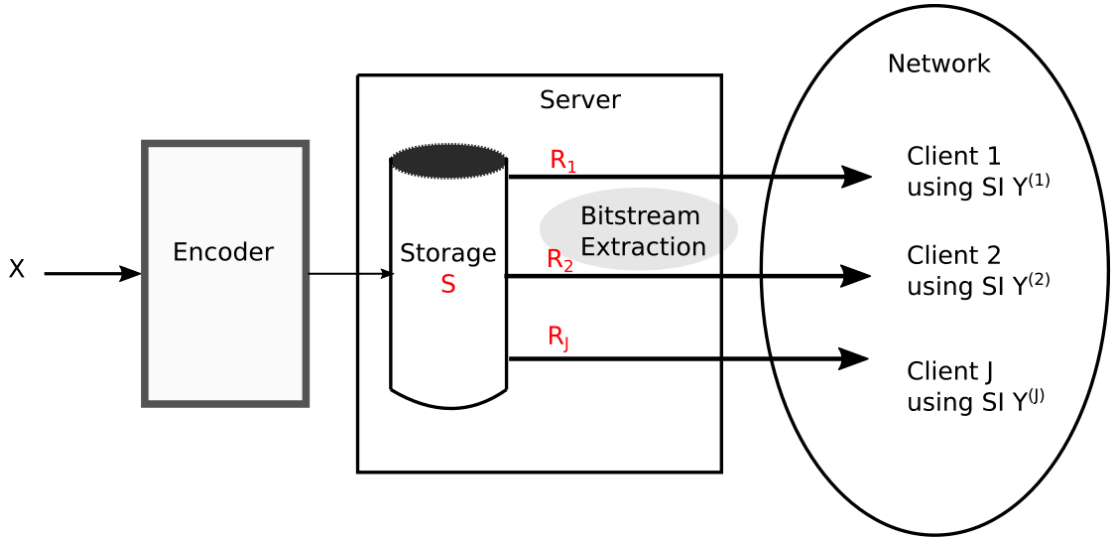

Figure 2: Modélisation du codage de sources pour l'application FTV

Figure 3: Codage de source sans perte

2.1. Codage de source sans perte

La théorie de l'information fournit les performances limites atteignables en fonction des hypothèses. La figure 3 résume le schéma de codage de source sans perte. On suppose que le mot de code U est envoyé sur un canal parfait. Le théorème du codage source indique qu'une compression sans perte de taux R est réalisable si et seulement si:

$$R \geq H(X) \tag{1}$$

où $H(X)$ est l'entropie de la source X.

2.2. Codage de source sans perte avec information adjacente

Figure 4: Codage de source sans perte avec information adjacente

Dans le cas où le récepteur dispose d'une information adjacente Y, le théorème du codage de source sans perte de Slepian-Wolf s'applique. La figure 4 représente le schéma de Slepian-Wolf. Selon le théorème, un taux R est réalisable si et seulement si:

$$R \geq H(X \mid Y) \tag{2}$$

Puisque $H(X|Y) \leq H(X)$, l'information adjacente au niveau du décodeur permet de diminuer le débit de codage de source par rapport au codage de source sans information adjacente. Dans ce cas, le débit minimum R dépend de la corrélation entre X et Y.

En pratique, il a été démontré que les codes LDPC peuvent être utilisés pour construire un schéma de codage de source de type Slepian-Wolf qui soit efficace.

2.3. Codage de source sans perte adapté à l'application FTV

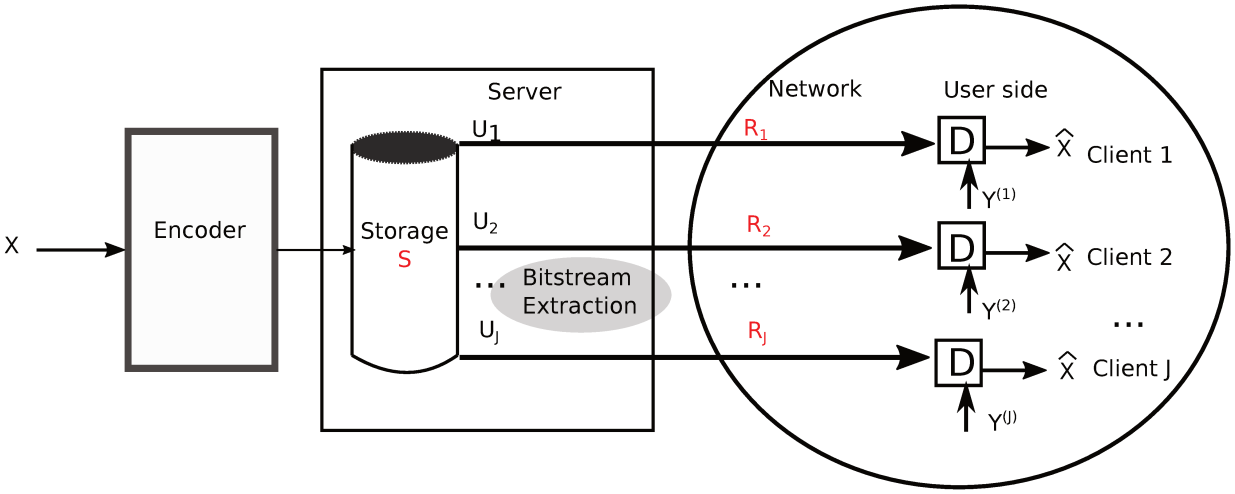

Figure 5: Codage de source sans perte pour FTV

Dans une application de type FTV, différentes informations adjacentes peuvent être disponibles au décodeur en fonction du client.
Notons S le débit de stockage sur le serveur et R_j le débit nécessaire pour transmettre la source X à l'utilisateur j avec les informations adjacentes $Y^{(j)}$. Alors d'après (2), les débits R_j réalisables doivent vérifier la contrainte suivante :

$$R_j \geq H(X \mid Y^{(j)}). \tag{3}$$

Le débit de stockage doit être suffisant pour satisfaire tous les clients. Il s'agit donc d'optimiser conjointement l'uplet $(R_1, R_2, \cdots, R_J, S)$. La meilleure stratégie est la stratégie de "Stockage incrémental" qui permet de minimiser le débit de transmission et le débit de stockage. Elle est basée sur la construction d'un mot de code unique qui couvre tous les débits de compression : la longueur du mot de code transmis sera adaptée au débit de compression demandé. Le débit de stockage satisfait la relation suivante :

$$S \geq \max_j H(X \mid Y^{(j)}) \tag{4}$$

En pratique, les codes LDPC adaptatifs en débit permettent la conception de schémas de codage de source satisfaisant aux contraintes (3) et (4).

3. Des codes LDPC

Les codes LDPC ont été inventés par Gallager et publiés dans sa thèse en 1960. Les codes LDPC sont des codes en blocs linéaires, initialement conçus pour le codage de canal. Très performants, ils approchent la capacité du canal dans ce cas. Les codes LDPC peuvent également être appliqués au codage de source de Slepian-Wolf.

3.1. Définition des codes LDPC

Utilisé en codage de source, un code LDPC de matrice de contrôle de parité H s'applique comme suit. Dans ce cas, il s'agit de calculer le syndrome U^m correspondant au message X^n :

$$U^m = X^n \cdot H^T$$

Figure 6: Codage de source sans perte avec information adjacent

Un exemple de matrice H est le suivant

$$H = \begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \end{array} \begin{array}{c} \begin{array}{cccccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{pmatrix} \end{array} \tag{5}$$

On définit les noeuds de variables (VN) : $x_i, i \in \{1, \cdots, n\}$ et les noeuds de parité (CN) : $u_j, j \in \{1, \cdots, m\}$. $H_{j,i} = 1$ implique une connexion entre x_i et u_j. Le code LDPC peut encore être représenté sous la forme d'un graphe dit de Tanner.

Dans la thèse, nous utilisons l'algorithme "Sum-Product" pour décoder le code LDPC.

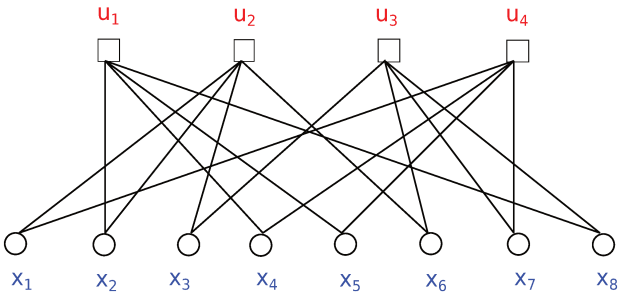

Figure 7: Graphe de Tanner de H donnée par (5)

3.2. Construction des codes LDPC

Une autre façon de représenter les codes LDPC est l'utilisation de protographes. Les protographes permettent un contrôle précis des connexions dans les codes LDPC.

Un protographe $\mathcal{S}$ est un petit graphe de Tanner de taille $S_m \times S_n$ avec $S_m/S_n = m/n = R$. Chaque ligne (respectivement colonne) de S représente un type de CN (respectivement de VN). Le protographe $\mathcal{S}$ décrit ainsi le nombre de connexions entre S_n différents types de VN et S_m différents types de CN. Une matrice H peut être construite à partir d'un protographe $\mathcal{S}$ en répétant la structure du protographe Z fois avec $n = ZS_n$, et en entrelaçant les connexions entre les VN et les CN.

La figure 8 donne un exemple de construction d'une matrice H à partir du protographe $\mathcal{S}$ défini par

$$\mathcal{S} = \begin{array}{c} \\ A_1 \end{array} \begin{array}{c} \begin{array}{cc} B_1 & B_2 \end{array} \\ \begin{pmatrix} 1 & 2 \end{pmatrix} \end{array}$$

Les composantes de $\mathcal{S}$ représentent les connexions entre un CN de type A_1 et deux VN de types B_1 et B_2. Une matrice H peut être construite à partir de ce protographe $\mathcal{S}$ telle que

$$H = \begin{array}{c} \\ A_1 \\ A_1 \end{array} \begin{array}{c} \begin{array}{cccc} B_1 & B_2 & B_1 & B_2 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{pmatrix} \end{array}$$

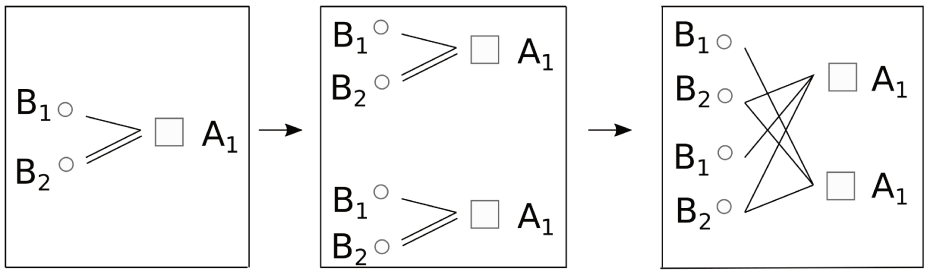

Figure 8: Construction de H à partir d'un protographe

Le graphe de Tanner du protographe $\mathcal{S}$ est représenté sur la figure 8 (partie gauche). Pour construire une matrice H, le protographe est d'abord dupliqué $Z = 2$ fois (partie centrale de la figure 8), puis les bords sont entrelacés (partie droite de la figure 8). Dans le graphe de Tanner final, on peut vérifier que chaque VN de type B_1 est connecté à un CN de type A_1, et chaque VN de type B_2 est connecté à deux CN de type A_1. Les performances d'un code LDPC donné, de matrice H, dépend fortement du protographe $\mathcal{S}$. L'algorithme "Differential evolution" permet d'optimiser le protographe.

Des cycles courts dans le graphe de H dégraderont les performances du code LDPC. La figure 9 illustre ce phénomène. La courbe rouge représente un cycle court de longueur 4. On définit le paramètre "girth"

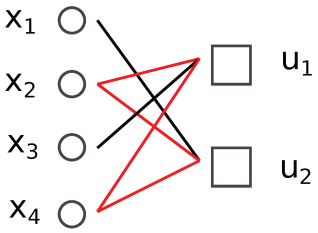

Figure 9: Cycles courts en H

comme la longueur du cycle le plus court. Un critère d'optimisation est la construction d'une matrice H possédant un "girth" maximum, avec un nombre minimum de cycles courts.

L'algorithme Progressive Edge-Growth (PEG) permet de réduire les cycles courts. Son fonctionnement est illustré sur la Figure 10. L'idée est d'ajouter des connexions entre des x_i et des u_j jusqu'à ce que tous

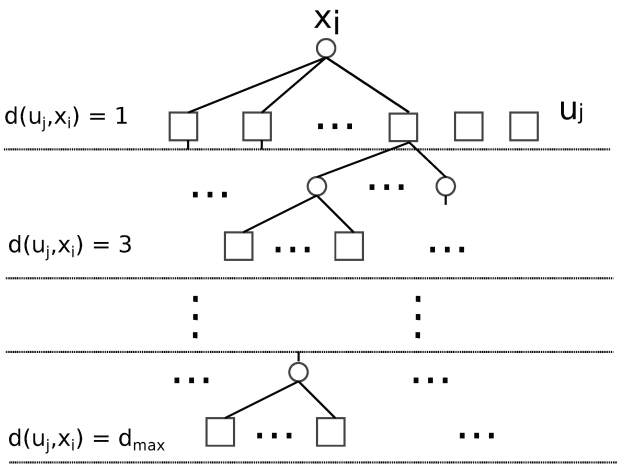

Figure 10: PEG

les VN soient connectés à d_v CN, et que tous les CN soient connectés à d_c VN. Sur le schéma, $\delta_v[i]$ et $\delta_c[i]$

représentent les nombres de connexions à une étape donnée du processus. Différents cas se présentent :

- il n'existe pas de connexion entre u_j et x_i (CN inaccessible pour x_i)
- la distance entre u_j et x_i est $d(u_j, x_i) = 1, 3, \cdots, d_{\max}$

Lorsqu'on ajoute une nouvelle connexion, deux cas peuvent survenir et la stratégie de choix du CN à connecter sera différente.

- cas 1: certains CN sont inaccessibles.
  On choisit parmi les CN inaccessibles, celui qui possède $\delta_c[i]$ minimum.
- cas 2: tous les CN sont accessibles.
  On choisit le CN tel que $d(u_j, x_i) = d_{\max}$ et $\delta_c[i]$ minimum.

Le processus se poursuit jusqu'à ce que les conditions suivantes soient satisfaites :

$$\delta_v[i] = d_v, \forall i \in \{1, \cdots, n\}$$

$$\delta_c[j] = d_c, \forall j \in \{1, \cdots, m\}$$

3.3. Codes LDPC adaptatifs

En codage de canal, les méthodes de construction de codes LDPC adaptatifs utilisent le poinçonnage (Puncturing) ou l'extension de la matrice de parité (Parity check matrix extension). Ces méthodes ne sont pas adaptées au codage de source. En codage de source, la littérature propose deux techniques de construction de codes LDPC adaptatifs, dites "Rateless" et "LDPCA".
La figure 11 représente le principe Rateless qui augmente le débit en partant d'un code LDPC de bas débit.

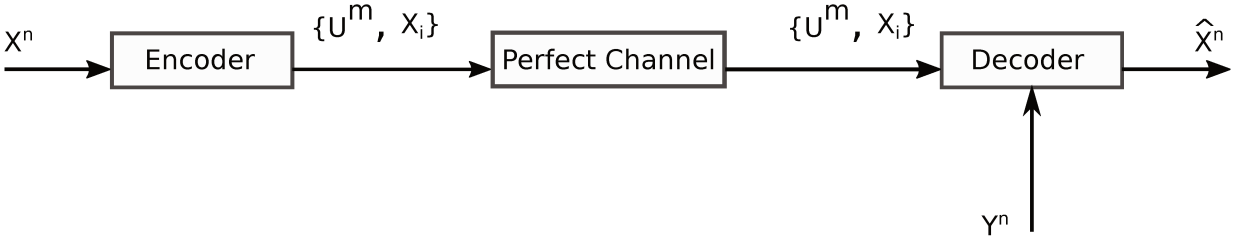

Figure 11: Rateless LDPC

Pour augmenter le débit, une partie des bits source X^n sera envoyée en plus du syndrome U^m. Afin de sélectionner le bit source supplémentaire transmis, nous choisissons les bits les moins fiables de X^n après avoir appliqué un décodeur LDPC.
Cependant, l'inconvénient majeur du Rateless est qu'il est difficile de construire de bons codes LDPC à faible débit. Il n'est donc pas conseillé d'appliquer la construction Rateless à partir d'un débit trop bas.

À l'opposé, la technique LDPCA (Figure 12) part d'un code LDPC à haut débit. Il calcule ensuite de nouveaux symboles par accumulation, notés $a^m = [a_1, a_2, \cdots, a_m]^T$, à partir du syndrome u^m comme suit

$$a_1 = u_1,$$

$$a_i = a_{i-1} + u_i, \quad \forall i = \{2, \cdots, m\}$$

où la somme binaire correspond à XOR. Si un débit inférieur est demandé, seule une partie des symboles $(a_1, a_2, \cdots, a_m)$ sera envoyée.

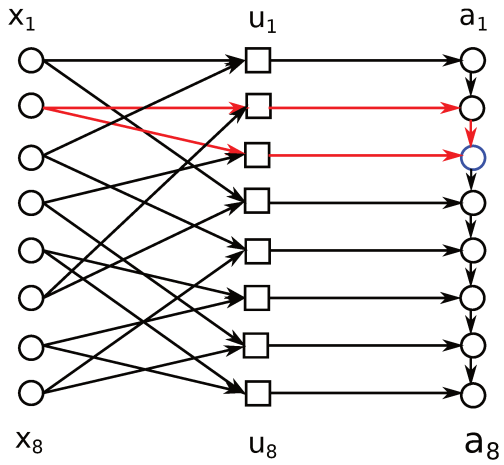

Figure 12: LDPCA

Cependant, dans la construction LDPCA, la structure de l'accumulateur est fixe et ne permet pas une optimisation des combinaisons des symboles de syndrome u_i utilisés par le décodeur. La structure de l'accumulateur peut notamment induire des cycles courts aux débits les plus bas et éliminer certains bits de sources des équations de parité (cf. la courbe rouge sur la figure 12).

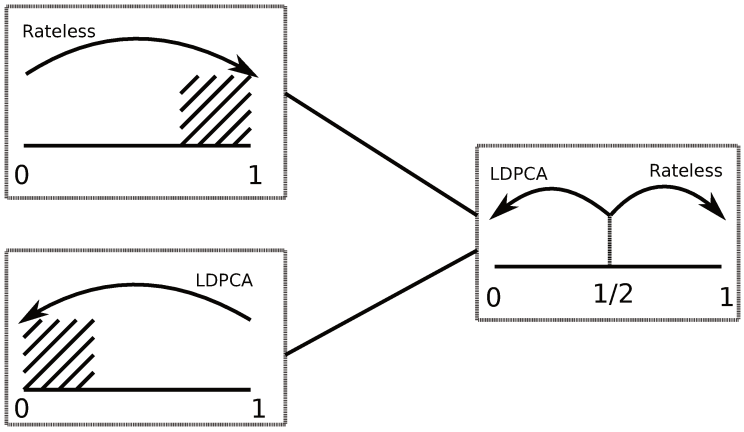

Figure 13: Une solution intermédiaire de construction de codes LDPC adaptatifs

Pour palier les inconvénients des techniques Rateless et LDPCA, une solution intermédiaire a été proposée (figure 13). Elle part d'un code initial de débit R = 1/2. Elle applique ensuite soit la méthode LDPCA pour obtenir des débits inférieurs à 1/2 ou la méthode Rateless pour des débits supérieurs à 1/2. De cette façon, le défaut de la technique Rateless peut être évité, mais les inconvénients du LDPCA demeurent. Notre objectif dans cette thèse est de proposer une méthode de construction qui permet d'éviter les défauts de la technique LDPCA (cycles courts, élimination de noeuds de variable).

4. Proposition de méthodes de construction de codes LDPC adaptatifs

4.1. Principe de construction des codes à débit adaptatif

Notre construction part d'un code mère H_1 ($m_1 \times n$) de débit le plus élevé, puis construit une séquence de codes filles H_2 $m_2 \times n$ de débits inférieurs. Le débit du code mère est noté R_1 et le débit de code de fille est noté R_2, tel que $R_1 > R_2$. Une matrice intermédiaire $H_{1\to 2}$ est introduite pour décrire la relation entre le code mère H_1 et le code fille H_2, comme suit :

$$H_2 = H_{1\to 2} H_1 \tag{6}$$

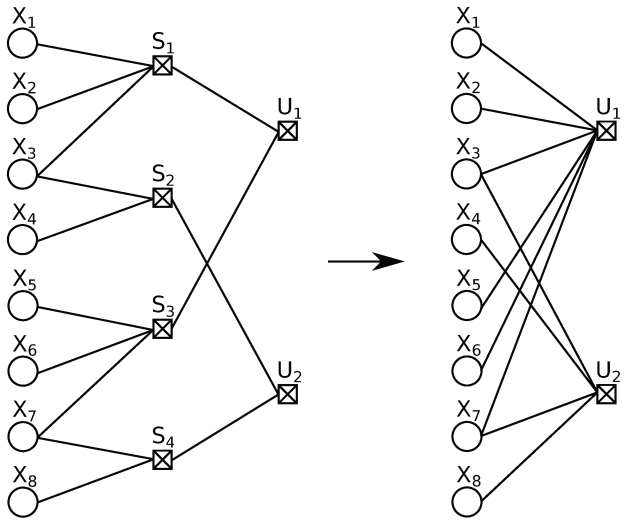

Figure 14: Première construction de code LDPC à débit adaptatif

Le problème revient à construire cette matrice intermédiaire $H_{1\to 2}$, par combinaison des syndromes s_i. Les combinaisons sont optimisées selon trois critères

- éviter l'élimination des VN,
- empêcher les cycles courts,
- assurer que $H_{1\to 2}$ soit de rang plein.

La matrice intermédiaire d'un code LDPCA est de la forme

$$H_{1\to 2} = \begin{bmatrix} 1 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 1 & \cdots & 0 \\ & & & \cdots & & \\ & & & \cdots & & \\ 0 & 0 & \cdots & 0 & 1 & 1 \end{bmatrix}$$

La combinaison des s_i est sous-optimale. D'autres matrices intermédiaires existent. Par exemple :.

$$H_{1\to 2} = \begin{bmatrix} 1 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 1 & \cdots & 0 \\ & & & \cdots & & \\ & & & \cdots & & \\ 0 & 0 & \cdots & 1 & 0 & 1 \end{bmatrix}$$

$$\cdots\cdots$$

$$H_{1\to 2} = \begin{bmatrix} 1 & 0 & 0 & 1 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 1 \\ & & & \cdots & & \\ & & & \cdots & & \\ 0 & 1 & \cdots & 0 & 1 & 0 \end{bmatrix}$$

La performance de H_2 dépend de la structure de $H_{1\to 2}$. Notre construction est basée sur l'optimisation du choix des lignes de H_1 à combiner selon les trois critères énoncés précédemment.

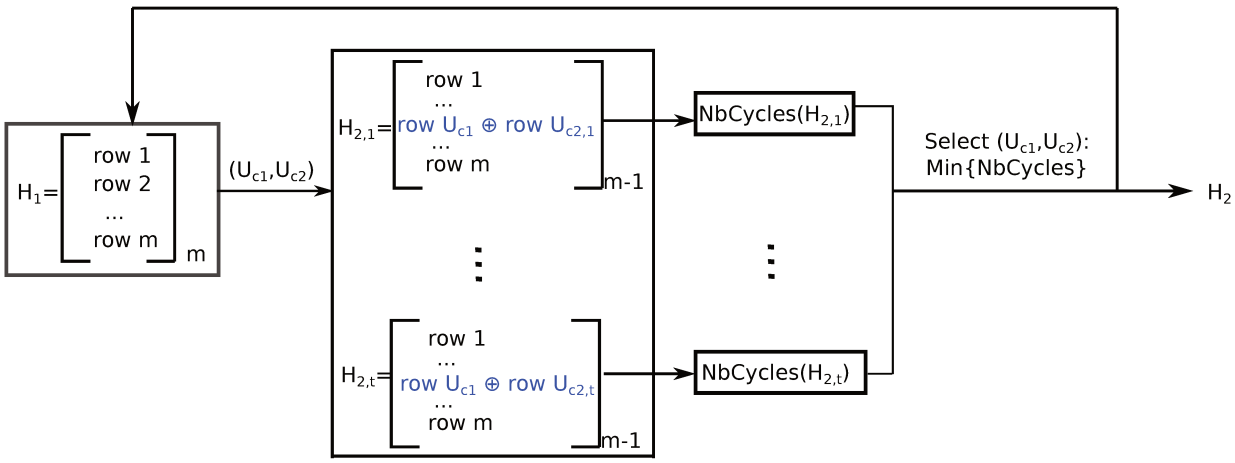

Figure 15: Algorithme 1

4.2. Premier algorithme basé sur l'optimisation directe de $H_{1\to 2}$

La figure 15 représente le schéma de construction des codes LDPC de débit adaptatif selon l'algorithme 1 pour passer de H_1 à H_2. Notons U_{c1} et U_{c2} les composantes du syndrome S qui seront additionnées (correspond à l'addition des lignes de même indice de H_1).

- On sélectionne les paires de composantes U_{c1}, U_{c2} de S telles que les CN d'indices correspondants ne sont connectés à aucun VN commun. Ceci permet d'éviter d'éliminer les noeuds de variable dans H_2.
- Parmi les composantes éligibles, on choisit celles qui minimisent le nombre de cycles d'une longueur donnée.
- Une ligne de H_1 est sélectionnée au plus une fois pour assurer que $H_{1\to 2}$ soit de rang plein.

L'application de notre méthode au code Wimax de longueur $n = 256$ est illustrée sur la figure 16. On a généré trois codes de débits respectifs 3/8, 1/4, 1/8, avec la méthode LDPCA et avec notre algorithme de construction. Dans notre algorithme, seuls les courts cycles de longueur 4 ont été pris en considération. Pour tous les débits, on observe que notre construction à débit variable donne une meilleure performance que la méthode LDPCA.

	$N_4(C_2)$
$R = 1/2$	0
$R = 3/8$ LDPCA	204
$R = 3/8$ Algorithme 1	83
$R = 1/4$ LDPCA	568
$R = 1/4$ Algorithme 1	200
$R = 1/8$ LDPCA	2336
$R = 1/8$ Algorithme 1	1193

Table 1: Nombre de cycles de longueur-4 pour les codes issus de la méthode LDPCA et de l'algorithme 1 proposé (code Wimax n=256)

D'après l'analyse des cycles reportée dans la Table 1, on observe que dans tous les cas, notre construction de code contient moins de cycles de longueur 4 que LDPCA, ce qui explique sa supériorité en termes de taux d'erreur binaire.

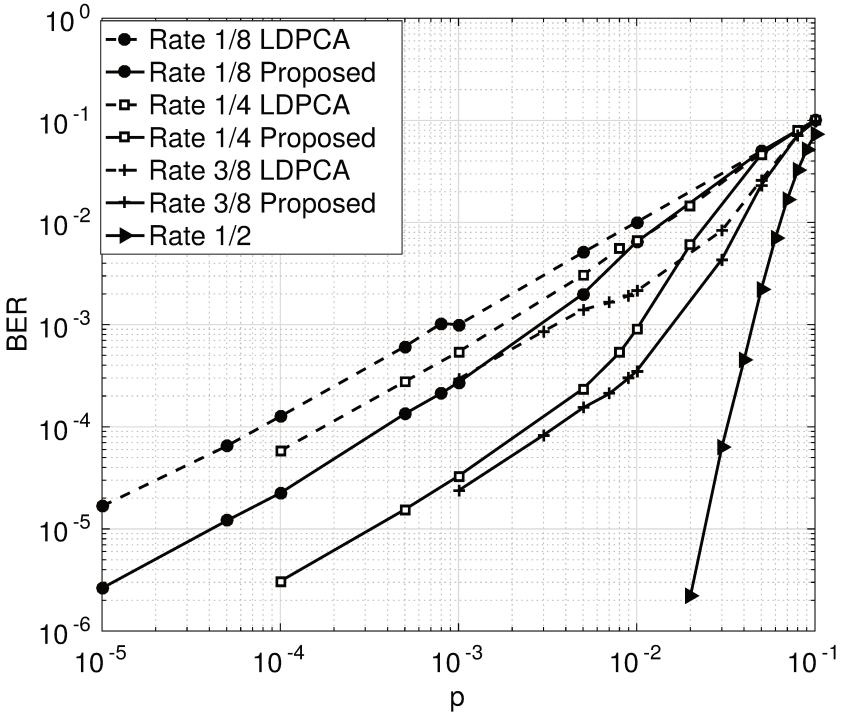

Figure 16: Application de l'algorithme 1 sur le code Wimax n=256

4.3. Algorithme 2 : basé sur l'optimisation préliminaire du protographe

Proposition: Supposons que H_1 soit générée par le protographe $\mathcal{S}_1$, $H_{1\to2}$ par le protographe $\mathcal{S}_{1\to2}$, telles que $H_2 = H_{1\to2}H_1$. Alors il existe toujours un protographe $\mathcal{S}_{1\to2}$ qui génère $H_{1\to2}$ et qui satisfait la relation

$$\mathcal{S}_2 = \mathcal{S}_{1\to2}\mathcal{S}_1 \tag{7}$$

L'algorithme 2 que nous proposons se déroule en deux étapes.

1- Optimiser $\mathcal{S}_{1\to2}$ pour obtenir $\mathcal{S}_2$ avec un bon seuil de décodage.

2- Construire $H_{1\to2}$ selon $\mathcal{S}_{1\to2}$ et telle que le nombre de cycles courts de longueur donnée pour H_2 soit minimal.

Autrement dit les deux étapes sont les suivantes :

1. Première étape : détermination de $\mathcal{S}_{1\to2}$ par recherche exhaustive. Critère : obtention de $\mathcal{S}_2$ avec le meilleur seuil de décodage.

2. Deuxième étape : optimisation des combinaisons de lignes de H_1, (U_{c1}, U_{c2}), pour minimiser le nombre de cycles courts, en appliquant l'algorithme 1 avec la contrainte supplémentaire $U_{c1} \in A_{j_1}, U_{c2} \in A_{j_2}$.

L'application de cet algorithme sur un code de longueur $n = 512$ est illustrée sur la figure 17.

Nous observons la supériorité de notre algorithme 2 sur la méthode LDPCA, justifiée par la réduction du nombre de cycles courts.

En appliquant la méthode intermédiaire basée sur la technique Rateless et notre algorithme 2, on obtient les performances reportées sur la figure 18, qui confirment l'efficacité du schéma de construction proposé.

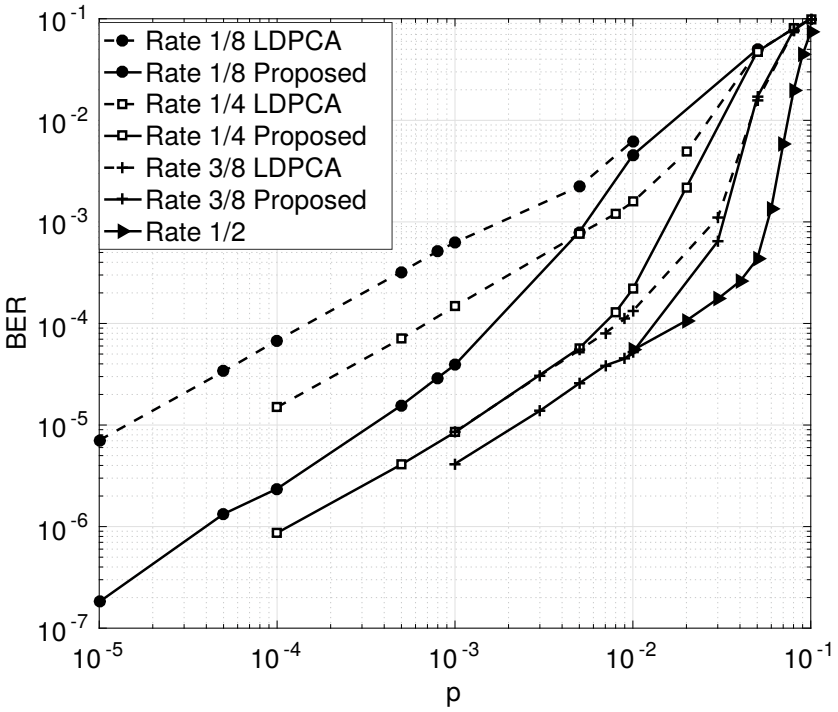

Figure 17: Performances de l'algorithme 2 sur un code de longueur $n = 512$

N_4	LDPCA	Algorithme 2
$R = 3/8$	453	455
$R = 1/4$	1216	737
$R = 1/8$	5361	3477

Table 2: Nombre de cycles de longueur 4 obtenu par application des méthodes LDPCA et de l'algorithme 2 sur un code de longueur $n = 512$

5. Application: vidéo en 360 degrés

Les images en 360 degrés nous sont fournies après traitement, par nos partenaires du projet CominLabs de l'Inria (Rennes). Les images 3D sont d'abord transformées en images 2D, à partir desquelles sont générées des informations de source X et des informations adjacentes $Y^{(j)}$, $j = 1, 2, \cdots, J$. La Figure 19 décrit l'insertion du schéma de codage de source proposé dans la chaîne de traitement des vidéo en 3D. Voici les principales étapes que nous avons suivies.

1. Notre schéma nécessite la connaissance de $\Pr(X|Y)$. Aussi nous introduisons la variable aléatoire $Z = X - Y$. Nous avons étudié deux modèles de distribution pour Z : Laplacienne et Q-aire. Finalement, nous avons retenu le modèle Laplacien. La transmission des paramètres de la distribution de Z doit également être prise en compte dans le bilan.

2. Le schéma de codage utilise un code LDPC binaire. Une transformation préalable des composantes du vecteur source $\underline{X}^n$ en différents plans de bits est nécessaire.

3. Codage séparé des plans de bits de $\underline{Q}^{(s)}$, $\underline{Q}^{(L-1)}$, jusqu'à $\underline{Q}^{(0)}$.

$$\underline{U}^{m,b} = \underline{Q}^{(b)} H^T$$

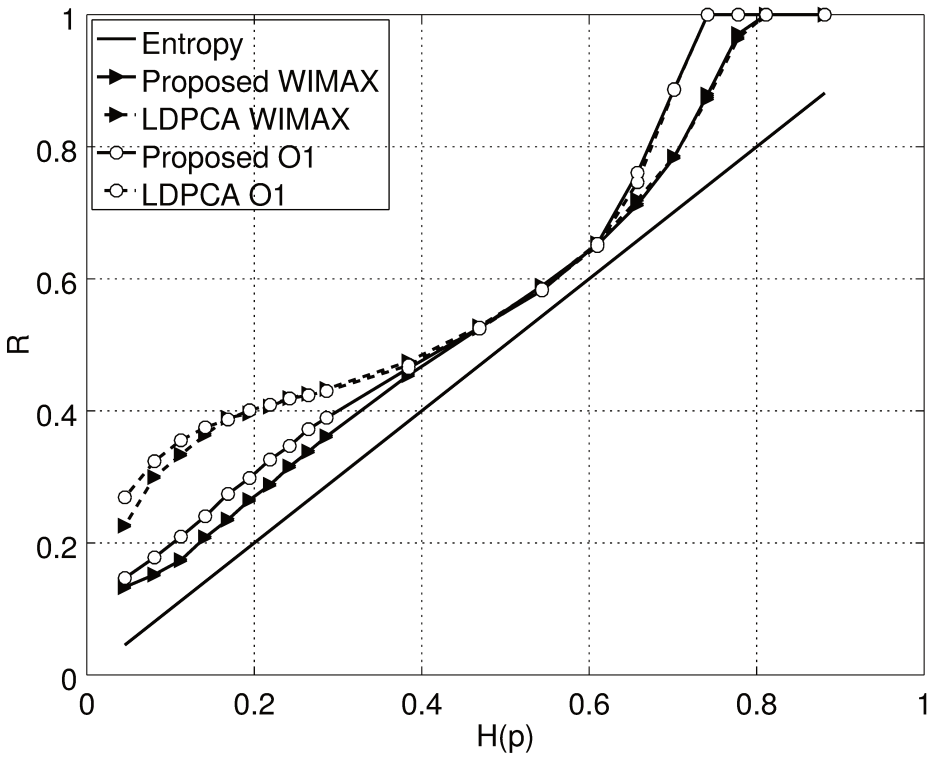

Figure 18: Taux de compression obtenu sur un canal parfait tel que $H(X|Y) = H(p)$

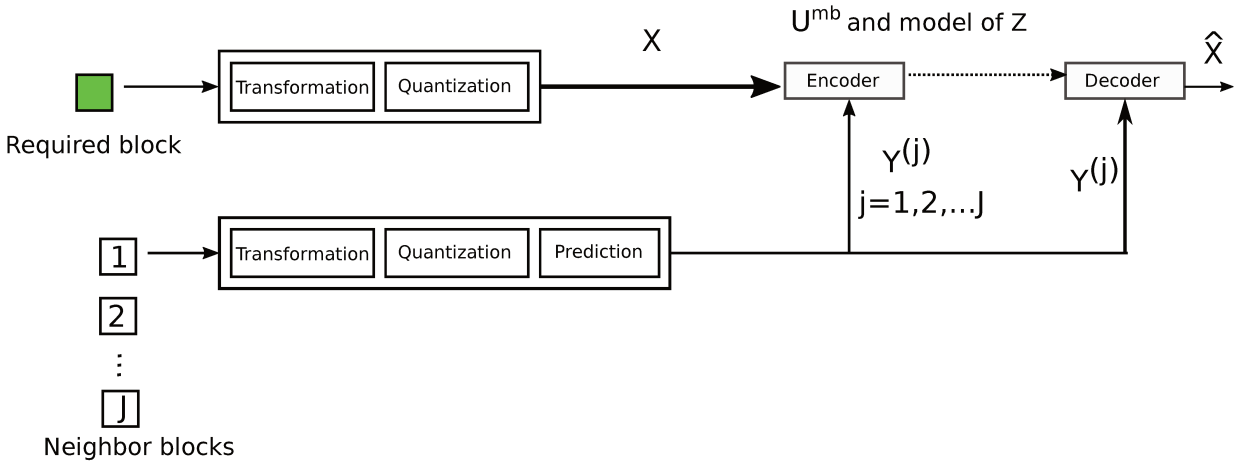

Figure 19: Application du schéma de codage de source proposé à la vidéo en 360 degrés

4. Décodage successif de chaque plan de bits $\underline{Q}^{(b)}$ utilisant $\underline{U}^{m,b}$, l'information adjacente ainsi que $\hat{\underline{Q}}^{(i)}$ avec $i > b$.

Nous avons également introduit une stratégie dite "coding/no coding" pour optimiser le débit et mis en oeuvre une quantification des paramètres de la distribution de Z. Appliqué à des fichiers vidéos en 360 degrés fournis par nos partenaires de l'INRIA, le schéma global atteint des performances en termes de débit de compression proches des limites théoriques, démontrant ainsi son efficacité. .

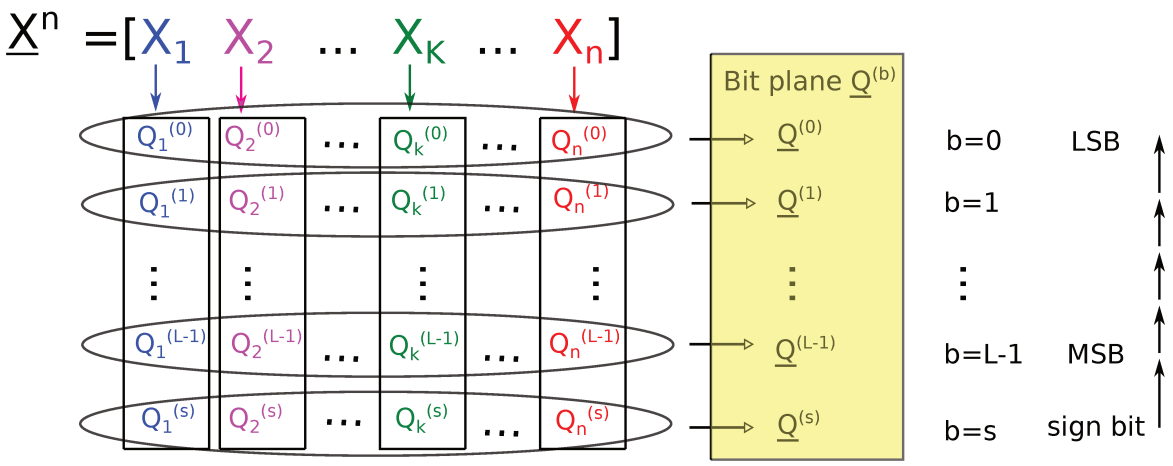

Figure 20: Génération des plans de bits $\underline{Q}$ à partir du vecteur de symboles $\underline{X}$

6. Conclusions

Dans cette thèse, nous nous sommes intéressés à une problématique de codage de source avec différentes informations adjacentes, typique d'applications telles que FTV. Nous avons proposé un schéma de codage de source à débit adaptatif basé sur des codes LDPC. Nous avons développé deux algorithmes de construction de codes LDPC binaires adaptatifs qui surpassent les constructions de l'état de l'art (LDPCA) en termes de taux d'erreur binaire. Nous avons appliqué notre schéma dans une chaîne de traitement sans pertes de vidéo en 360 degrés et obtenu des résultats proches des limites théoriques.

UNIVERSITE BRETAGNE LOIRE / MATHSTIC

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Titre : Codage de sources de Slepian-Wolf utilisant des codes LDPC et application à la télévision interactive

Mots clés : Codes LDPC, Codage de source, Rendement compatible, Codage, FTV.

Résumé : La Télévision Interactive (FTV) est un service de vidéo à la demande qui permet au client de choisir l'angle de vue de la vidéo. Le défi principal est de stocker un d'énorme volume de données et d'extraire une petite partie de ces données à la demande et en temps réel. Pour améliorer le décodage de l'information, on peut supposer que les vues précédemment reçues sont conservées par l'utilisateur. Le problème ainsi posé devient un problème de codage de sources avec information adjacente du côté de l'utilisateur. Cette thèse s'inscrit dans ce contexte. Elle s'intègre au projet CominLabs InterCom dont l'objectif est de proposer des solutions pour l'accès massif aléatoire à des sous-ensembles de données corrélées. Dans cette thèse, nous proposons des schémas pratiques de codage de sources sans perte sous l'hypothèse d'une information adjacente. Ces schémas sont basés sur des codes de type Low Density Parity Check (LDPC). Les méthodes de l'état de l'art utilisent des solution adaptatives en débit s'appuyant sur des codes LDPC de type Rateless ou LDPC Accumulés (LDPCA). Mais les codes Rateless fonctionnent mal aux débits faibles, et les codes LDPCA ne sont pas adaptés aux débits élevés. Dans cette thèse, on combine les deux méthodes pour construire des codes LDPC aux débits adaptables offrant une large gamme de débits. Cependant, la technique LDPCA ne permet pas d'optimiser la distribution des degrés du code, ni de contrôler le nombre de cycles courts pour tous les rendements. C'est pourquoi nous proposons deux nouvelles méthodes de construction pour remplacer le code LDPCA. Les résultats de la simulation montrent une amélioration des performances par rapport à LDPCA. Enfin, nous intégrons la construction de codes sans perte dans un schéma complet de codage de source avec pertes développé pour l'application FTV dans le cadre du project InterCom.

Title : Slepian-Wolf source coding using LDPC codes for Free Viewpoint Television

Keywords : LDPC codes, Source coding, Rate adaptive, Video coding, FTV.

Abstract : Many multimedia applications such as Free Viewpoint Television (FTV) use a distant service provider that offers customized services depending on the user request. The main challenge is the efficient storage of a huge amount of data and the real-time extraction of a small fraction of these data upon request. In some applications such as FTV, the requests previously addressed by the user can help to optimize both the storage and the extraction. The problem can thus be seen as a source coding problem with side information at the user side. This PhD thesis fits into this context. It is part of the CominLabs project InterCom that focuses on solutions for massive random access to subsets of correlated data. In this thesis, we investigate practical lossless source coding schemes with side information based on Low Density Parity Check (LDPC) codes. State-of-the-art approaches use rate-adaptive LDPC codes such as rateless codes and LDPC accumulate (LDPCA) codes. Rateless codes perform poorly at low coding rates while LDPCA is not adapted to high-rates. In this thesis, we combine both methods to construct rate-adaptive LDPC codes offering a wide range of rates. However LDPCA does not allow to optimize the code degree distribution, nor to control the amount of short cycles at all rates. This is why we propose two novel rate-adaptive LDPC code constructions to replace the LDPCA part. Simulation results show improved performance compared to LDPCA. Finally, we incorporate the proposed lossless code construction into a complete lossy source coding scheme that was developped for FTV in the framework of the InterCom project.