



**Université catholique de Louvain**

Ecole Polytechnique de Louvain  
(Faculté des Sciences Appliquées)

LABORATOIRE DE TÉLÉCOMMUNICATIONS  
ET  
TÉLÉDÉTECTION

B - 1348 Louvain-la-Neuve

Belgique

## **Joint Source-Channel Turbo Techniques and Variable Length Codes**

Xavier Jaspar

Thesis presented for the Ph.D. degree  
in Applied Sciences

Ph.D. committee:

Luc VANDENDORPE	(UCL) - Supervisor
Sergio BENEDETTO	(Politecnico di Torino - Italy)
Christophe DE VLEESCHOUWER	(UCL)
Philippe DELSARTE	(UCL)
Christine GUILLEMOT	(INRIA - France)
Auguste LALOUX	(UCL) - Formal Chairman
Jean-Pierre RASKIN	(UCL) - Chairman

April 2008  
(revision 1.00 - April 2008)



*Imagination is more than knowledge*

Albert Einstein



# Remerciements

La croisée des chemins, le fruit des rencontres, quelles inestimables richesses! Nombreuses sont les personnes que je souhaite remercier.

Mes plus vifs remerciements à Luc Vandendorpe, mon promoteur. Que de ténacité, de rigueur, de pédagogie et d'enthousiasme! Merci d'avoir suscité ma curiosité durant les études. Merci de m'avoir invité à commencer ce travail et surtout de m'en avoir donné l'envie. C'est une expérience hors pair qu'il a partagée à travers ces quatre années. Merci pour la confiance qu'il a placée en moi et pour la grande liberté d'action qu'il accorde à ses chercheurs.

Je tiens également à remercier le F.R.S.-FNRS (fonds de la recherche scientifique) pour le soutien financier qui a permis la réalisation de ce travail dans un cadre optimal.

Un grand merci aux membres de mon jury pour leurs questions, leurs commentaires et les améliorations qu'ils ont suggérées: Sergio Benedetto, Christophe De Vleeschouwer, Philippe Delsarte, Christine Guillemot, Auguste Laloux et Jean-Pierre Raskin.

Ma reconnaissance va également aux professeurs qui ont marqué mon parcours. Je pense notamment à Carlo Cordaro et à Daniel Lefèvre en humanités, qui enseignent les mathématiques et transmettent leur passion avec brio. Quel talent! Parmi les professeurs à l'université, je remercie particulièrement Philippe Delsarte et Paul Van Dooren pour leur rigueur exceptionnelle et Auguste Laloux pour sa pédagogie sans égal, qui ont contribué de façon significative à ma formation pré-doctorale. Quel succès si ne fût-ce qu'une infime part de leur rigueur et pédagogie se retrouve dans ce travail!

Jamais je n'oublierai l'ambiance qui règne au labo TELE. Merci à tous. En particulier, Damien, Harold, Pierre, Bertrand, Marilena, Mika, Maria, Onur et Rony ont été bien plus que des compagnons de bureau exceptionnels. Et Jon, Xa & Xa, Ced & Ced, Seb & Seb, Val, Jacek, Lyazid, Rosa, David, Annabelle,

Damien, Fod, Antonin, Thierry, Antoine, Jérôme, Vincent, Patrice, Benoît, Piotr, Christophe, Benjamin, Etienne, Lio, Achraf, Ali, Félix. . . toute la ribambelle TELE. . . merci pour tous les beaux moments partagés, discussions, tranches de fou rire, soirées. D'ailleurs, vivement une prochaine soirée, tous on the dance floor! ;-) Un tout grand merci aussi à Isabelle, Marie-Hélène, Christian, Jean et Robert! Enfin, le partage des points de vue avec Harold a été "salvateur" sur bien des sujets (humains). Et les conseils de Jacek et Christophe m'ont été très précieux; ils sont loin, très loin, d'imaginer à quel point. Merci beaucoup.

Je tiens particulièrement à remercier mes chers amis. Merci pour tous les moments fabuleux, les discussions, les soirées, les week-ends, les vacances, . . . passés ensemble. Merci d'avoir ainsi contribué.

De tout coeur, je remercie vivement mes proches. Ah mes chers parents et mes deux chères soeurs, vous êtes les arcs de Khalil Gibran. Oh combien m'avez-vous apporté et que de souvenirs; oh combien m'apportez-vous et qu'il est bon de se retrouver! Ainsi en est-il aussi avec ma future belle-famille. Merci de m'avoir si bien accueilli, entouré et soutenu comme un fils.

Ah, Marie-Aude, Pioupiou, Muse de chaque instant. . . Merci pour ta patience d'ange! pour ton soutien, ainsi que pour ta contribution inestimable à cette thèse, à notre bonheur et à notre quotidien. Infiniment, merci.

# Abstract

Efficient multimedia communication over mobile or wireless channels remains a challenging problem. To deal with that problem so far, the industry has followed mostly a divide and conquer approach, by considering separately the source of data (text, image, video, etc.) and the communication channel (electromagnetic waves across the air, a telephone line, a coaxial cable, etc.). The goal is always the same: to transmit (or store) more data reliably per unit of time, of energy, of physical medium, etc. With today's applications, the divide and conquer approach has, in a sense, started to show its limits.

Let us consider, for example, the digital transmission of an image. At the transmitter, the first main step is data compression, at the source level. The number of bits that are necessary to represent the image with a given level of quality is reduced, usually by removing details in the image that are invisible (or less visible) to the human eye. The second main step is data protection, at the channel level. The transmission is made ideally resistant to deteriorations caused by the channel, by implementing techniques such as time/frequency/space expansions. In a sense, the two steps are quite antagonistic — we first compress then expand the original signal — and have different goals — compression enables to transfer more data per unit of time/energy/medium while protection enables to transfer data reliably. At the receiver, the “reversed” operations are implemented.

This separation in two steps dates back to Shannon's source and channel coding separation theorem in 1948 and has encouraged the division of the research community in two groups, one focusing on data compression, the other on data protection. This separation has also seduced the industry for the design, thereby supported by theory, of layered communication protocols. But this theorem holds only under asymptotic conditions that are rarely satisfied with today's multimedia content and mobile channels. Therefore, it is usually wise in practice to drop this strict separation and to allow at least some cross-layer cooperation between the source and channel layers.

This is what lies behind the words *joint source-channel* techniques. As the name suggests, these techniques are optimized jointly, without a strict separation. Intuitively, since the optimization is less constrained from a mathematical standpoint, the solution can only be better or equivalent.

In this thesis, we investigate a promising subset of these techniques, based on the *turbo principle* and on *variable length codes*. The potential of this subset has been illustrated for the first time in 2000, with an example that, since then, has been successfully improved in several directions. Unfortunately, most decoding algorithms have been so far developed on an ad hoc basis, without a unified view and often without specifying the approximations made. Besides, most code-related conclusions are based on simulations or on extrinsic information analysis. A theoretical framework on the error correcting properties of variable length codes in turbo systems is lacking.

The purpose of this work, in three parts, is to fill in these gaps up to a certain extent. The first part presents the literature in this field and attempts to give a unified overview. The second part proposes a transmission system that generalizes previous systems from the literature, with the simple addition of a repetition code. While most previous systems are designed for bit streams with a high level of residual redundancy, the proposed system has the interesting flexibility to handle easily different levels of redundancy. Its performance is then analyzed for small levels of redundancy, which is a case not tackled extensively in the literature. This analysis leads notably to the discovery of surprising interleaving gains with reversible variable length codes.

The third part develops the mathematical framework that was motivated during the second part but skipped on purpose for the sake of clarity. We first clarify several issues that arise with non-uniform bits and the extrinsic information charts, and propose and discuss two methods to compute these charts. Next, several theoretical results are stated on the robustness of variable length codes concatenated with linear error correcting codes. Notably, an approximate average distance spectrum of the concatenated code is rigorously developed. Together with the union bound, this spectrum provides upper bounds on the symbol and frame/packet error rates. These bounds are then analyzed from an interleaving gain standpoint and it is proved that the variable length code improves the interleaving gain if its spectrum is bounded.

# Contents

<b>Abstract</b>	<b>vii</b>
<b>Table of contents</b>	<b>ix</b>
<b>1 Introduction</b>	<b>1</b>
<b>2 Joint Source-Channel Turbo Techniques: a Unified View</b>	<b>7</b>
2.1 Introduction . . . . .	8
2.2 Turbo source-channel transmission chain . . . . .	11
2.3 Soft source decoding: principles . . . . .	12
2.3.1 Different distortion measures - different decision criteria	13
2.3.2 State Models and Graphical Interpretation with FLC . .	14
2.3.3 Decoding algorithms . . . . .	15
2.4 Turbo decoding . . . . .	22
2.4.1 Interleaver and cycles in the factor graph . . . . .	23
2.4.2 Turbo decoder structure . . . . .	24
2.4.3 Convergence of turbo decoding . . . . .	25
2.5 Soft decoding of different source codes . . . . .	27
2.5.1 Variable Length Code (VLC) . . . . .	28
2.5.2 Arithmetic Code (AC) . . . . .	29
2.5.3 Quasi-Arithmetic (QA) Code . . . . .	30
2.5.4 Complexity issues and suboptimal decoding . . . . .	31
2.6 Synchronization and Robustness of Entropy codes . . . . .	34
2.6.1 Synchronization mechanisms . . . . .	34
2.6.2 Resynchronization properties of VLC . . . . .	35
2.6.3 Error correction capabilities of VLC . . . . .	36
2.6.4 Recent advances in source coding . . . . .	37

2.7	Related areas . . . . .	38
2.8	Performance Illustrations . . . . .	40
2.8.1	Soft source decoding of real signals . . . . .	41
2.8.2	JSC turbo decoding of theoretical sources and real signals . . . . .	42
2.9	Conclusions . . . . .	46
<b>3</b>	<b>VLCs and Irregular Turbo Codes: an Example of Synergy</b>	<b>49</b>
3.1	Introduction . . . . .	50
3.2	System Overview . . . . .	52
3.2.1	System description and notations . . . . .	52
3.2.2	Comments on the interleaver $\Pi_0$ . . . . .	54
3.3	Joint Iterative/Turbo Decoder . . . . .	55
3.3.1	The decoding algorithm . . . . .	55
3.3.2	The decoding complexity . . . . .	57
3.4	Interleaving Gain Analysis . . . . .	58
3.4.1	Background on distance spectrum and interleaving gains . . . . .	59
3.4.2	Interleaving gains with irregular RCs . . . . .	61
3.4.3	Interleaving gains with regular RCs . . . . .	62
3.4.4	Interleaving gains with RVLCs . . . . .	62
3.4.5	Interleaving gains with FLCs . . . . .	63
3.4.6	Interleaving gains with a serial turbo code . . . . .	64
3.5	Further insight with EXIT Charts . . . . .	65
3.5.1	Preliminaries . . . . .	65
3.5.2	Comparison/analysis of three system examples . . . . .	68
3.5.3	Optimization of the convergence threshold . . . . .	70
3.6	Related works . . . . .	72
3.6.1	Regular turbo codes and VLCs . . . . .	72
3.6.2	Regular turbo codes and FLCs . . . . .	74
3.6.3	Regular turbo codes and binary sources . . . . .	74
3.7	Simulation Results . . . . .	76
3.7.1	Comparisons with the English alphabet source . . . . .	77
3.7.2	Comparisons with other parameters . . . . .	82
3.7.3	Comparisons with a binary source . . . . .	84
3.8	Conclusion . . . . .	86
Appendix 3.A	Fast computation of the VLC-RC EXIT chart . . . . .	87
Appendix 3.B	Proofs of the theoretical results . . . . .	89

<b>Contents</b>	<b>xi</b>
3.B.1 Proof of Corollary 3.2 . . . . .	89
3.B.2 Proof of Theorem 3.3 . . . . .	90
3.B.3 Proof of Theorem 3.4 . . . . .	91
3.B.4 Proof of Theorem 3.5 . . . . .	91
3.B.5 Proof of Theorem 3.7 . . . . .	94
<b>4 Non-Uniform Binary Sources and EXIT charts</b>	<b>97</b>
4.1 Introduction . . . . .	97
4.2 Computation of the EXIT charts . . . . .	100
4.2.1 Assumptions, notations and consistency . . . . .	100
4.2.2 BEXIT charts, Fig. 4.1: biased bits . . . . .	104
4.2.3 FEXIT charts, Fig. 4.2: flipped bits . . . . .	110
4.3 Transformations, equivalence and discussion . . . . .	111
4.3.1 Transformations and equivalence . . . . .	111
4.3.2 Simulation results . . . . .	112
4.3.3 Discussion . . . . .	112
4.3.4 Approximations of JLU(.) . . . . .	115
4.3.5 Comments on the EXIT chart and the turbo decoder . . .	117
4.4 Applications . . . . .	117
4.5 Conclusion . . . . .	120
<b>5 Performance Analysis of VLCs in Turbo/Concatenated Systems</b>	<b>121</b>
5.1 Introduction . . . . .	122
5.2 Background . . . . .	125
5.2.1 Prefix VLCs and discrete source of symbols . . . . .	125
5.2.2 Markov encoding process and Balakirsky trellis . . . . .	126
5.2.3 Levenshtein symbol distance . . . . .	128
5.2.4 Union bound . . . . .	128
5.2.5 Distance spectrum . . . . .	129
5.3 Transmission system . . . . .	130
5.3.1 Semi-infinite VLC stream . . . . .	130
5.3.2 Serial concatenation of a VLC block with a linear code .	131
5.3.3 Link with the proposed performance bounds . . . . .	132
5.4 Assumptions . . . . .	133
5.4.1 Assumptions . . . . .	133
5.4.2 ML versus MAP . . . . .	134

5.5	Performance bounds for VLC streams . . . . .	134
5.5.1	Definition of an error event . . . . .	134
5.5.2	Spectrum and bounds for VLC streams . . . . .	137
5.5.3	Numerical evaluation of the VLC spectrum . . . . .	143
5.6	Performance bounds for VLC blocks . . . . .	145
5.6.1	Framing rule $F_s$ , definition and properties . . . . .	146
5.6.2	Spectrum of VLC blocks with framing rule $F_s$ . . . . .	147
5.6.3	Framing rule $F_b$ , definition and properties . . . . .	148
5.6.4	Spectrum of VLC blocks with framing rule $F_b$ . . . . .	150
5.6.5	Spectrum comparison between the rules $F_b$ and $F_s$ . . . . .	152
5.6.6	VLC block concatenated with a linear code . . . . .	153
5.6.7	Remarks about non-concatenated VLC blocks . . . . .	154
5.6.8	Related work . . . . .	155
5.7	Statistical synchronization, bounded spectrum and interleaving gains . . . . .	157
5.7.1	Bounded spectrum and statistically synchronizable VLC . . . . .	158
5.7.2	Some examples . . . . .	162
5.7.3	Bounded spectrum test algorithm . . . . .	164
5.7.4	Bounded spectrum and non-catastrophic VLC . . . . .	165
5.7.5	Toward a link between bounded spectra and interleaving gains . . . . .	166
5.7.6	Bounded spectrum and guaranteed interleaving gains . . . . .	168
5.8	Discussion and further work . . . . .	170
5.8.1	Other framing rules . . . . .	170
5.8.2	SER for $N_s$ unknown . . . . .	173
5.8.3	Bounds for $N_s$ known at the decoder . . . . .	175
5.8.4	Extension to the MAP decoder . . . . .	177
5.8.5	Extension to sources with memory . . . . .	181
5.9	Simulation results . . . . .	182
5.9.1	Tightness of the estimations . . . . .	182
5.9.2	Interleaving gains . . . . .	182
5.9.3	Bounded spectrum and interleaving gains . . . . .	184
5.10	Conclusions . . . . .	190
Appendix 5.A	Root state probability, $P(\rho_i)$ . . . . .	192
Appendix 5.B	Proofs related to VLC streams . . . . .	197

<b>Contents</b>	<b>xiii</b>
5.B.1 Union bounds on Arbitrary Distortions . . . . .	197
5.B.2 Proof of Theorem 5.18 . . . . .	200
5.B.3 Proof of Theorem 5.21 . . . . .	201
5.B.4 Proof of Theorem 5.22 . . . . .	203
Appendix 5.C Proofs related to VLC blocks . . . . .	203
5.C.1 Proof of Theorem 5.26 . . . . .	203
5.C.2 The framing rule $P_{F_b}$ , properties . . . . .	205
5.C.3 Proof of Theorem 5.31 . . . . .	210
Appendix 5.D Proofs on statistical synchronization and bounded spectrum . . . . .	222
5.D.1 Proof of Lemma 5.37 and Lemma 5.39 . . . . .	222
5.D.2 Proof of Theorem 5.40 . . . . .	223
5.D.3 Proof of Proposition 5.45 . . . . .	230
5.D.4 Proof of Proposition 5.51 . . . . .	230
5.D.5 Proof of Proposition 5.52 . . . . .	230
5.D.6 Proof of Theorem 5.54 . . . . .	232
 <b>6 Conclusions</b>	 <b>241</b>
 <b>List of Publications</b>	 <b>247</b>
 <b>Bibliography</b>	 <b>249</b>



# Introduction

# 1

Efficient multimedia communication over time-varying mobile or wireless channels remains a challenging problem.

## Joint source-channel techniques

In any digital communication system, some transmitter wishes to send some data to a receiver. The goal of digital communication theory is to maximize the amount of data that can be transmitted reliably per unit of resources (time, energy, frequency, physical medium, etc.), or put differently, to minimize the consumption of resources to transmit a given amount of data.

To reach this goal so far, the industry has followed mostly a divide and conquer approach, by considering separately the source of data (text, image, video, etc.) and the communication channel (electromagnetic waves across the air, a telephone line, a coaxial cable, etc.) — recall the simplified example of the digital transmission of an image in the Abstract. Following this divide and conquer approach, the transmitter is in general composed of two antagonistic coders, a source coder which implements data compression and a channel coder which implements data protection. Similarly, the receiver is composed of the reversed operations, a channel decoder and a source decoder. In the communication protocol, the source coder and decoder constitute the source layer. The channel coder and decoder constitute the channel layer.

This separation in two layers, which has been prevailing so far for designing communication systems, relies on Shannon's source and channel coding separation theorem [1]. This theorem states that source and channel optimum performance bounds can be approached as close as desired by designing independently the source and channel coding strategies. Techniques where the source layer and the channel layer are designed and optimized independently of each other, are commonly referred to as *tandem techniques*.

Tandem techniques have in practice their limitations. Shannon's theorem holds indeed only under asymptotic conditions, where both codes are allowed infinite length and complexity. If the design of the system is constrained in terms of delay and complexity, if the sources are not stationary, or if the channels are non-ergodic, separate design and optimization of the source and channel coders can be largely suboptimal, making efficient multimedia communication over mobile or wireless channels particularly challenging. For example, the source layer design in tandem techniques assumes that the error probability at the output of the channel decoder is zero. A non-zero error probability at the output of the channel decoder, which is generally unavoidable for finite-length channel codes in practice, may desynchronize the source decoder and have a strong impact on the end-to-end source distortion. Therefore, it is usually wise in practice to drop the strict separation and to allow at least some cross-layer cooperation between the source layer and the channel layer.

This is what lies behind the words *joint source-channel* techniques. Unlike tandem techniques, joint source-channel techniques are optimized jointly, as their name suggests, without a strict separation and allow some cross-layer cooperation. Since the underlying optimization is less constrained from a mathematical standpoint, joint source-channel techniques can only be better than or equal to tandem techniques. They may thus reduce the end-to-end source distortion and achieve better performance in the case of practical systems with constrained delay and complexity.

Nevertheless, maintaining a certain level of separation/independence between the source layer and the channel layer is often desirable too, which is worth emphasizing. Consider for example the design of a mobile device that must process  $M$  different sources of data and face up  $N$  different types of channels. With a completely joint approach, this represents  $MN$  different

(de)coding strategies. If we maintain a certain level of independence between layers, this represents  $M$  strategies for the source layer and  $N$  strategies for the channel layer. This example is self-speaking.

In the following, the scope is therefore restricted to joint source-channel techniques that maintain a certain level of separation/independence between the source layer and the channel layer.

## Joint source-channel turbo techniques

A new family of joint source-channel techniques has started to show up over the last ten years, based on the so successful *turbo principle*. We refer to those techniques as *joint source-channel turbo techniques*.

The turbo principle is perhaps the most groundbreaking contribution emanating from the discovery of turbo codes in [2]. This principle concerns essentially the receiver and consists in the particular cooperation that is implemented between the component decoders that make up the receiver. This cooperation is iterative: The component decoders are used several times and exchange information about the transmitted signal at each iteration. Loosely speaking, each component decoder has some a priori knowledge about the transmitted signal that the other component decoders have not, and is thus able to correct errors that the other component decoders cannot. Therefore, by using the component decoders several times and by making them cooperate, the end-to-end distortion may be improved over the iterations.

This is much like a group of people, with different historic backgrounds, that would have to discuss about a particular problem. Each person would contribute based on his own background, i.e., based on his own a priori knowledge, and the discussion would certainly become iterative so as to find a consensus.

In the context of joint source-channel techniques, the turbo principle has three interesting key properties. Firstly, it is cross-layer by nature. If the turbo receiver includes indeed component decoders from different layers of the communication protocol, these layers will cooperate at the receiver through the iterative exchange of information. Secondly, the cooperation between the

layers due to the turbo principle is relatively simple; it amounts in practice to the exchange of probabilities, which requires simply an interface of real numbers between the layers. Such an interface still allows a great separation/independence between layers, which is often desirable (see end of previous section). Thirdly, the turbo principle is very effective at combining different layers as the success of turbo techniques has attested so far.

These properties clearly motivate to investigate further the application of the turbo principle to joint source-channel techniques.

The potential of joint source-channel turbo techniques has been illustrated for the first time in [3], with a system based on a variable length code as source code and a convolutional code as channel code. This system has then been successfully improved in several directions. Unfortunately, most decoding algorithms have been so far developed on an ad hoc basis, without unified view and often without specifying the approximations made. Besides, most code-related conclusions are based on simulations or on extrinsic information analysis. A theoretical framework on the error correcting properties of source codes in turbo systems is lacking.

The purpose of this work is to fill in these gaps up to a certain extent, when the source code is a variable length code or a fixed length code.

## Outline

This thesis is made up of four main chapters. Several readings are possible because each chapter is mostly self-contained and independent.

*Chapter 2.* This chapter is both an introduction to joint source-channel turbo techniques and a unified overview of the literature in this field. In particular, the algorithms underlying joint source-channel turbo decoding are described for fixed length codes, for variable length codes and for arithmetic codes, as instances of the Sum-Product algorithm [4–7] on normal factor graphs. Synchronization and robustness issues are also discussed.

*Publication related to Chapter 2: [8].*

*Chapter 3.* This chapter proposes a transmission system that generalizes previous systems from the literature, with the simple addition of a repetition

code. While most previous systems are designed for bit streams with a high level of residual redundancy, the proposed system has the interesting flexibility to handle easily different levels of redundancy. We then analyze its performance for small levels of redundancy, which is a case not tackled extensively in the literature. This analysis leads notably to the discovery of surprising interleaving gains with reversible variable length codes.

*Publications related to Chapter 3: [9–11].*

**Chapter 4.** To assess the convergence of turbo decoders, the extrinsic information transfer (EXIT) charts are very useful and have been introduced in [12] for uniform bits. Unfortunately, numerous applications deal in practice with non-uniform binary sources, i.e., with bits that take their values not equally likely. With such sources, a naive application of the EXIT charts might lead to incorrect results. This chapter extends the EXIT charts to non-uniform binary sources.

*Publication related to Chapter 4: [13].*

**Chapter 5.** This chapter states several theoretical results on the robustness of variable length codes (VLCs) concatenated with linear error correcting codes. In a first part, we rigorously develop notably an approximate average distance spectrum of the concatenated code, based on pioneering results on variable length code bit streams [14] and on turbo codes [15,16]. Together with the union bound, this spectrum provides upper bounds on the symbol and frame/packet error rates. In a second part, we analyze these bounds from an interleaving gain standpoint. We introduce the important concept of bounded VLC spectrum and we prove that the variable length code improves or contributes to the interleaving gain if its spectrum is bounded. At last, this concept of bounded VLC spectrum is proved to be closely related, under certain assumptions, to the well known concept of statistically synchronizable variable length codes [17], which allows us the reuse previously known results from the literature.

*Publications related to Chapter 5: [18–21].*

It is worth noting that Chapters 4 and 5 develop the mathematical framework that is motivated in Chapter 3 but that is skipped on purpose in Chapter 3 for the sake of clarity. Actually, Chapter 3 can be considered as the application (and the small extension) of the theoretical results developed in

Chapters 4 and 5. This order of presentation is rather unusual. But it allows the reader to discover the practical results first, without the need to read the lengthy theoretical results. Let us hope discovering these practical results will motivate the reader to go much beyond them. . .

# Joint Source-Channel Turbo Techniques: a Unified View

## 2

*The content of this chapter is basically the reproduction of a journal paper [8], with a few enhancements.*

As emphasized in the introduction, the principles which have been prevailing so far for designing communication systems rely on Shannon's source and channel coding separation theorem [1]. This theorem states that source and channel optimum performance bounds can be approached as close as desired by designing independently the source layer and the channel layer. However, this theorem holds only under asymptotic conditions, where both codes are allowed infinite length and complexity. If the design of the system is constrained in terms of delay and complexity, if the sources are not stationary, or if the channels are non-ergodic, separate design and optimization of the source and channel layers can be largely suboptimal. For practical systems, it is usually wise to drop the strict separation and to allow at least some cross-layer cooperation between the source layer and the channel layer, i.e., to consider joint source-channel techniques.

This chapter gives a unified overview of recent developments in the field of joint source-channel turbo techniques, which are described in the framework of normal factor graphs. It is intended for readers that have already some basic knowledge in digital communication.

## 2.1 Introduction

A communication system is in general composed of a source coder, which maps the source symbols into a compressed binary representation, and of a channel coder, which maps the binary representation into coded bits or waveforms for transmission. Similarly, the receiver is composed of a channel decoder and a source decoder. The source coder aims at reducing the size of the binary representation to save bandwidth while the channel coder reintroduces some redundancy in order to make the transmission resistant to channel degradations. The source coder and decoder constitute the source layer. The channel coder and decoder constitute the channel layer.

Techniques where the source layer and the channel layer are designed and optimized independently are called *tandem techniques*. The source layer design in this case assumes that the error probability at the output of the channel decoder is zero. Unfortunately, a non-zero error probability at the output of the channel decoder, which is generally unavoidable for finite-length channel codes, may have a strong impact on the end-to-end distortion.

Unlike tandem techniques, *joint source-channel* techniques, as their name suggests, are optimized by considering the source and channel layers jointly, i.e., without a strict separation, and allow some cross-layer cooperation. Since the design is less constrained, joint source-channel techniques may reduce the end-to-end distortion and achieve better performance.

This chapter focuses on joint source-channel turbo techniques [3,11,21–41] and attempts to give a unified overview in this field.

The different decoding algorithms will be described as instances of the Sum-Product algorithm [4–7] on normal factor graphs [4]. Factor graphs form indeed a unifying framework to present turbo and turbo-like techniques. A factor graph is a graphical representation of a factorization, e.g., the factorization of the joint probability of all random variables involved in a transmission chain. As such, it is a powerful tool to analyze graphically the structure of stochastic dependencies and constraints between variables, as well as for reading out conditional independence relations in a model. In a sense, a factor graph is a substitute for equations and provides a clear overview of the transmission chain. Applying the Sum-Product algorithm on factor graphs

provides efficient (iterative) estimation algorithms. Actually, many decoding algorithms from the turbo/soft literature can be “re-discovered” with it and new ones can be developed. On Markov chains for example, it is equivalent to the BCJR algorithm [42], historically discovered before.

The presentation starts with several source estimation algorithms, considering initially only the source layer (without channel layer). In the first place, these algorithms are described considering the example of a fixed length code (FLC) as source code. FLCs make indeed the problem much simpler because the codewords have all the same (fixed) length, by definition, and the symbol positions in the noisy bit stream are thus fixed and known *a priori*. In the second place, the extension to variable length codes (VLCs) and arithmetic codes (ACs) is considered. With these codes, due to their variable length nature, the symbol positions in the bit stream depend on the symbols themselves and are thus random. As we will see, capturing this randomness requires decoders of higher complexity and suggests thus the use of suboptimal decoders. Note that recovering a transmitted sequence of source symbols from the received signal is equivalent to infer the sequence of the transmission chain model states. This problem has already been addressed for Hidden Markov Models in non-turbo systems in the past, for memoryless [43] and Markov [44] sources.

The complete transmission chain is then considered, with both the source layer and the channel layer. From a probabilistic model standpoint, the transmitted bit stream can then be seen as the output of a global state model, product of the state models of the source and channel coders. Unfortunately, the state space of this global model explodes in practice, making the implementation of the optimal decoder [45,46] intractable. Nonetheless, if an interleaver is introduced between the source coder and the channel coder, the optimal decoder can be efficiently approximated in many cases by using the turbo principle, with a much lower decoding complexity.

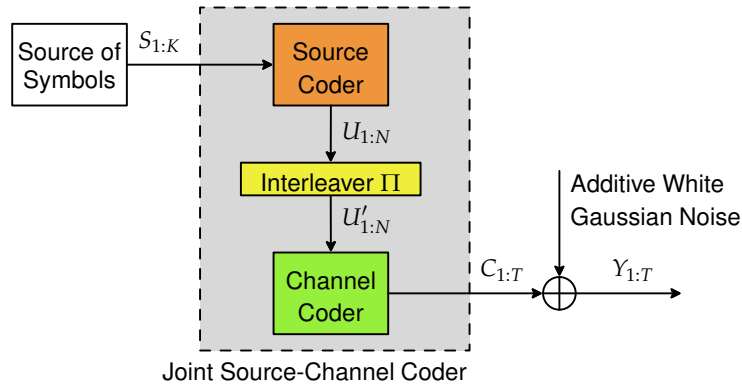
Basically, the turbo decoder is usually composed of a source decoder and a channel decoder, which are used iteratively, several times, and which cooperate by exchanging probabilities about the transmitted signal. Ideally, this iterative cooperation improves the quality of the estimations and the quality of the decoded source samples over the iterations. Already at this stage, it is

worth emphasizing that the cooperation between the source decoder and the channel decoder requires only an iterative interface of probabilities, i.e., of real numbers; such a simple interface allows a great separation/independence between the source and channel layers in the communication protocol, which is often desirable and valuable from an industrial standpoint.

At last, since the channel layer is generally not perfect, residual bit errors may remain in the decoded bit stream. Unfortunately, VLCs and ACs are particularly sensitive to residual bit errors: Even a single bit error can desynchronize the decoder and have a dramatic impact on the end-to-end distortion. Several techniques have been developed in the literature to make VLCs and ACs more resistant to residual bit errors. For example, the desynchronization of the decoder can be limited by means of soft synchronization markers [26] for VLCs, or with a forbidden symbol [47] for ACs. Other source codes can be envisaged as well, with improved error correcting capabilities, e.g., reversible VLCs [48,49], or with inherent synchronization properties [50,51].

The remainder of this chapter is structured as follows. The transmission chain used throughout the chapter is described in Section 2.2. Soft source decoding is introduced in Section 2.3 for a simple source code, an FLC. This is a necessary building block of joint source-channel turbo decoding whose principle is presented in Section 2.4. Soft source decoding is further elaborated in Section 2.5 for other source codes and practical complexity issues are discussed. Section 2.6 summarizes a few techniques and source codes able to improve the resilience, along with some analysis results. Related to joint source-channel turbo decoding, some other techniques of great interest exist and are presented in Section 2.7. Performance illustrations are provided in Section 2.8 for both theoretic and practical sources, notably real images and video signals.

In the following, capital letters indicate random variables and small letters realizations of these. A sub-sequence of variables  $Z_i$  is denoted by  $Z_{m:n} \triangleq (Z_m, Z_{m+1}, \dots, Z_n)$ . The expectation of  $Z$  is denoted by  $E\{Z\}$  and the indicator function by  $\mathbb{I}\{\cdot\}$ , i.e.,  $\mathbb{I}\{a\}$  equals 1 if  $a$  is true, 0 otherwise. The probability  $P(Z = z)$  is abbreviated as  $P(z)$ .



**Figure 2.1** Joint source-channel coder. A forward error correcting code is chosen as channel code. Examples of source codes are given in Section 2.5. The word “joint” refers in this example to the joint design of the source and channel codes.

## 2.2 Turbo source-channel transmission chain

Let us consider the transmission chain depicted in Fig. 2.1. For the sake of clarity, both the source and the channel considered are simple elements. We consider a source of discrete symbols taking their values in a finite alphabet  $\mathcal{A}$ , with or without memory, and a memoryless additive white Gaussian noisy (AWGN) discrete channel. The source outputs sequences of  $K$  symbols  $S_k$  in each packet, where  $k$  is the time index with a symbol clock. The source coder then encodes the sequence  $S_{1:K}$  into a sequence of bits  $U_{1:N}$ . Examples of source codes will be given in Section 2.5.

The source coder can be concatenated with the channel coder either via a serial or a parallel turbo structure. In the sequel, we will focus on a serial turbo structure as depicted in Fig. 2.1. However, a parallel structure has also been considered in the literature [33]. The bits  $U_{1:N}$  produced by the source coder are thus permuted through the interleaver  $\Pi$  to give the bits  $U'_{1:N}$ . The role of the interleaver is twofold. Firstly, it increases the global code spreading, the so-called interleaving gain, thus the global code performance, under certain conditions on the source and channel codes. Secondly, it allows the use of iterative decoding as a low complexity approach to near-optimal decoding [15]. In the channel coder, a forward error correcting code is used to protect the se-

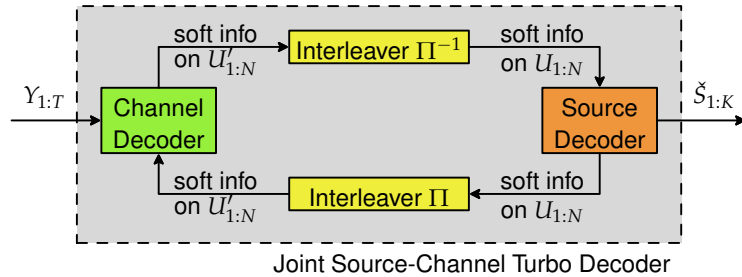


Figure 2.2 Example of a joint source-channel turbo decoder.

quence of interleaved bits before transmission across the channel. The coded bits  $C_{1:T}$  are finally sent across the channel.

At the receiver side, in Fig. 2.2, the joint decoder aims at recovering the transmitted sequence  $S_{1:K}$  of source symbols from the noisy measurements  $Y_{1:T}$  at the output of the channel. We assume in the following that both  $N$  and  $K$  are known at the receiver, unless otherwise stated. As explained in the introduction, to keep the decoding complexity within a tractable range, the optimal joint decoder is approximated by an iterative decoder with, as inputs, the noisy measurements  $Y_{1:T}$  of the coded bits  $C_{1:T}$ . The channel and source decoders are iteratively used and exchange refined information (probabilities) on the bits  $U_{1:N}$  or on  $U'_{1:N}$ . As explained later, each decoder is a Soft-In/Soft-Out module based on, e.g., the BCJR algorithm [42] or the Sum-Product algorithm [7]. After a certain number of iterations, the source decoder computes an estimation  $\hat{S}_{1:K}$  of the symbols  $S_{1:K}$ . If the number of iterations is 1, note that the decoder is roughly equivalent to a tandem decoder. The turbo decoder is further detailed in Section 2.4.

### 2.3 Soft source decoding: principles

At the receiver, the goal is to minimize the distortion of the decoded/estimated signal w.r.t. the original signal. In practice, several distortion measures may be used. With them are associated different decision criteria, different decoding algorithms and different decoding complexities. This section tries to summarize these aspects which are the foundations of Soft-In/Soft-Out source decoding used in turbo decoding.

### 2.3.1 Different distortion measures - different decision criteria

In the tandem receiver, the channel decoder is generally aimed at minimizing either the frame error rate (FER) or the bit error rate (BER) and the source decoder assumes no error in the decoded bits. In the joint source-channel receiver, the source and channel decoders are jointly optimized to minimize the desired distortion together, and the distortions are not limited to the BER or FER. The source symbol error rate (SER) and the mean square error (MSE) are measures which better reflect the quality of the reconstructed sequence of source symbols, especially when a VLC is used, in which case a single bit error can have a dramatic impact on the SER and on the MSE.

$$\begin{aligned} \text{FER} &\triangleq \mathbf{E}\{\mathbb{I}\{\check{S}_{1:K} \neq S_{1:K}\}\} \\ &\rightarrow \check{s}_{1:K} = \underset{s_{1:K}}{\operatorname{argmax}} P(s_{1:K}|y_{1:T}) = \underset{s_{1:K}}{\operatorname{argmax}} P(s_{1:K}, y_{1:T}), \end{aligned} \quad (2.1)$$

$$\begin{aligned} \text{SER} &\triangleq \frac{1}{K} \sum_{k=1}^K \mathbf{E}\{\mathbb{I}\{\check{S}_k \neq S_k\}\} \\ &\rightarrow \check{s}_k = \underset{s_k}{\operatorname{argmax}} P(s_k|y_{1:T}) = \underset{s_k}{\operatorname{argmax}} P(s_k, y_{1:T}), \end{aligned} \quad (2.2)$$

$$\begin{aligned} \text{BER} &\triangleq \frac{1}{N} \sum_{n=1}^N \mathbf{E}\{\mathbb{I}\{\check{U}_n \neq U_n\}\} \\ &\rightarrow \check{u}_n = \underset{u_n}{\operatorname{argmax}} P(u_n|y_{1:T}) = \underset{u_n}{\operatorname{argmax}} P(u_n, y_{1:T}), \end{aligned} \quad (2.3)$$

$$\begin{aligned} \text{MSE} &\triangleq \frac{1}{K} \sum_{k=1}^K \mathbf{E}\{|\check{S}_k - S_k|^2\} \\ &\rightarrow \check{s}_k = \sum_{s_k} s_k P(s_k|y_{1:T}) = \frac{\sum_{s_k} s_k P(s_k, y_{1:T})}{\sum_{s_k} P(s_k, y_{1:T})}. \end{aligned} \quad (2.4)$$

These distortions, FER, SER, BER and MSE, can be minimized by running respectively the frame-MAP (maximum *a posteriori*), symbol-MAP, bit-MAP and minimum mean square error (MMSE) estimations, as summarized in (2.1)–(2.4). Techniques to solve these estimations are presented in Section 2.3.3. Another interesting distortion measure is the symbol error rate computed with the Levenshtein distance ( $\text{SER}_L$ ), see [14]. The  $\text{SER}_L$  is less sensitive than the SER to symbol deletion or insertion, which makes it useful for some audio applications, e.g., when a symbol deletion is acceptable because it is considered as a simple time shift of the original signal.

### 2.3.2 State Models and Graphical Interpretation with FLC

In the following, we use the normal (Forney-style) factor graphs [4] for the presentation of the different state models, as explained in the introduction. Good tutorials about these can be found in [6,7]. Consider the factor graph in Fig. 2.3(a). This represents the transmission chain of Fig. 2.1 with an FLC as source code and no channel code. Each edge represents a variable and each node a function involving the variables connected to it. The only variables here are the symbols  $S_k$  at the top of the graph and the bits  $U_n$  at the bottom. Note that the symbols  $S_k$  are represented by three edges, instead of one, plus an equality constraint (a node with an equality sign at the center). This is a simple artifice from Forney's factor graphs to link a variable to more than two functions:  $S_k$  is linked to  $f_{k-1}$ ,  $f_k$  and  $g_k$ .

Let us review the coding stage step by step. With FLCs, each symbol is associated with a codeword and all codewords have the same fixed length; an example is given in Fig. 2.4. The value of the first symbol  $S_1$  has a probability given by  $P(S_1)$ . That probability is represented by the function  $c_b$  in Fig. 2.3(a). Then, the FLC maps  $S_1$  to a codeword of length  $M$  composed of the bits  $U_{1:M}$ . This mapping is modeled by the function  $g_1 = P(U_{1:M}|S_1)$ . Each bit  $U_n$  is then measured as  $y_n$  at the output of the AWGN channel, which is represented by the black node  $h_n = P(y_n|U_n)$ . Given  $S_1$ , the next symbol  $S_2$  has a probability given by  $P(S_2|S_1)$  — we assume that the memory of the source is modeled by a Markov chain. That probability is represented by the function  $f_1$ . And the process is repeated for each symbol. All the node definitions are summarized on the first row of Tab. 2.1.

This factor graph uses a symbol clock, i.e., the horizontal dependencies between the states  $S_k$  are defined symbol by symbol. A second model of the same transmission chain is possible and is depicted in Fig. 2.3(b), using a bit clock this time. The dependencies between the states  $X_n$  are defined bit by bit. Instead of considering a source of symbols which is transcoded into bits, we consider now a Markov chain producing a bit at each time instant and a symbol every  $M$  bits. If the source has no memory, the different values of the Markov states  $X_n$  (where  $n$  is the bit time index) are the internal nodes of the FLC code tree, i.e.,  $X_n = x_n$  where  $x_n$  takes its values in  $\mathcal{T}_n$ , the set of possible internal nodes at time  $n$ . An example of code tree is given in Fig. 2.4. Note

that  $P(x_{n+1}|x_n)$  can be deduced from the code tree. In the case of a Markov source (with memory), the states  $X_n$  are extended with the previous symbol, i.e., the  $X_n$  are the pairs  $(T_n, S_k)$  where  $T_n \in \mathcal{T}_n$  and  $k$  is the integer division  $n/M$ . The function definitions are given on the second row of Tab. 2.1. The initial constraint  $c_b$  in Tab. 2.1 indicates that we begin in the root node of the code tree, i.e.,  $X_0 = R$ . Note that one particularity is the strict emission of only one source symbol  $S_k$  every  $M$  bit positions, hence a different expression of  $f_n$  when  $n$  is a multiple of  $M$ . Either model, symbol clock or bit clock, can be used almost equivalently. But the latter may allow simplifications and complexity reductions in some cases, as explained later.

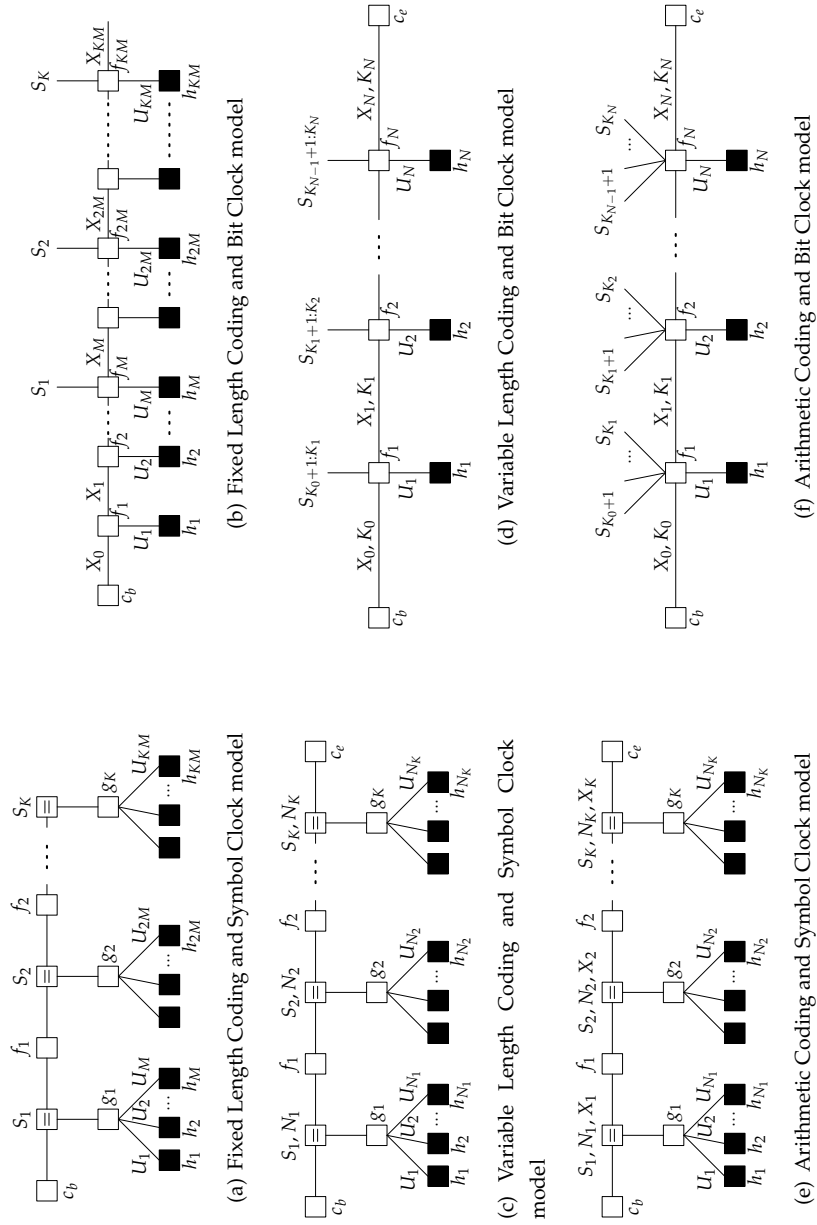
For Markov chains, another interesting graph is the *trellis*. An example is given on the top of Fig. 2.5, using the bit clock model of Fig. 2.3(b) and the FLC of Fig. 2.4. There is a node for each possible realization  $x_n$  of each random variable/state  $X_n$  and there is a branch or edge linking two nodes  $x_n, x_{n+1}$  if a transition from  $x_n$  to  $x_{n+1}$  is possible, i.e., if it has a non-zero probability,  $P(x_{n+1}|x_n) > 0$ . Note the differences and similarities between trellises and factor graphs: Each variable/edge  $X_n$  of the factor graph is exploded into all its possible realizations  $x_n$ . During the decoding algorithm, a metric is evaluated and associated with each branch, here  $P(x_{n+1}, y_{n+1}|x_n)$ , measuring the joint probability of observing the current channel measure  $y_{n+1}$  with the current transition  $x_n \rightarrow x_{n+1}$ .

### 2.3.3 Decoding algorithms

In equations (2.2), (2.3) and (2.4), the symbol-MAP, the bit-MAP and the MMSE all make use of the same *a posteriori* or joint probabilities, either on the symbols  $S_k$  or on the bits  $U_n$ . These probabilities can be efficiently and elegantly computed by the *Sum-Product Algorithm* (SPA) [7] on the factor graph of the transmission chain.

#### 2.3.3.1 Sum-Product algorithm

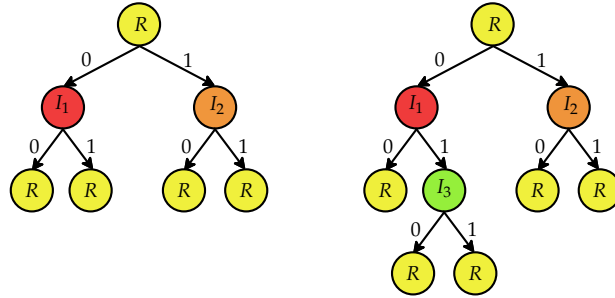
The SPA is based on the *Sum-Product rule* (SPR) which can be stated as follows: The message out of some node/function  $g(x, y_1, \dots, y_n)$  along the



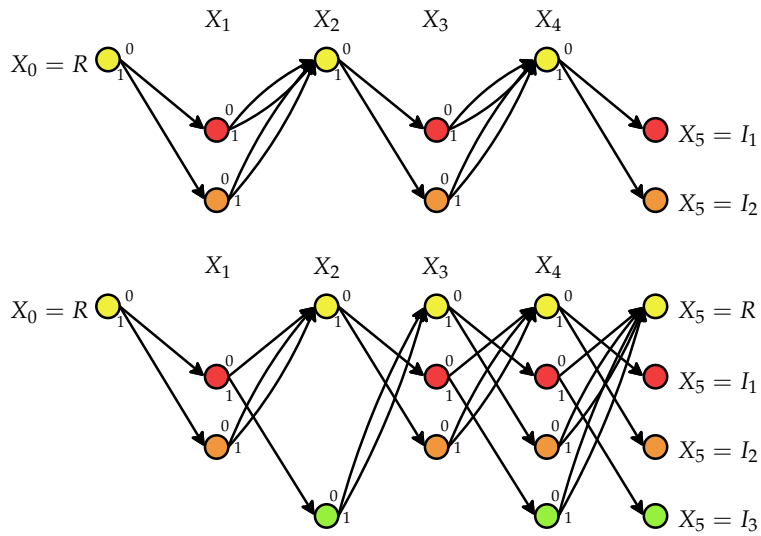
**Figure 2.3** Normal factor graph of a source with memory, different codes and different clock models. The function definitions are given in Tab. 2.1.

**Table 2.1** Details on the different factor graphs of Fig. 2.3.

	$c_b$	$c_e$	$f$	$g_k$	$h_n$
Fig. 2.3(a)	$P(S_1)$		$f_k = P(S_{k+1} S_k)$	$P(U_{(k-1)M+1:kM} S_k)$	$P(y_n U_n)$
Fig. 2.3(b)	$\mathbb{I}\{X_0 = R\}$		$f_{n \neq kM} = P(X_n X_{n-1}) P(U_n X_n, X_{n-1})$ $f_{n=kM} = P(X_n X_{n-1}) P(U_n X_n, X_{n-1})$ $\times P(S_k U_n, X_{n-1})$		$P(y_n U_n)$
Fig. 2.3(c)	$P(S_1)$ $\times \mathbb{I}\{N_1 = I(S_1)\}$	$\mathbb{I}\{N_k = N\}$	$f_k = P(S_{k+1} S_k)$ $\times \mathbb{I}\{N_{k+1} = N_k + I(S_{k+1})\}$	$P(U_{N_{k-1}+1:N_k} N_k, S_k)$	$P(y_n U_n)$
Fig. 2.3(d)	$\mathbb{I}\{K_0 = 0\}$ $\times \mathbb{I}\{X_0 = R\}$	$\mathbb{I}\{K_N = K\}$ $\times \mathbb{I}\{X_N = \text{start}$ $\text{of codeword}\}$	$f_n = P(X_n X_{n-1}) P(K_n K_{n-1}, X_n)$ $\times P(U_n X_n, X_{n-1})$ $\times P(S_{K_{n-1}+1:K_n} K_n, U_n, X_{n-1})$		$P(y_n U_n)$
Fig. 2.3(e)	$P(S_1) P(N_1 S_1)$ $\times \mathbb{I}\{X_1 = \emptyset\}$	system specific	$f_k = P(S_{k+1} S_k) \mathbb{I}\{X_{k+1} = X_k \cup \{S_k\}\}$ $\times P(N_{k+1} N_k, S_{k+1}, X_{k+1})$	$P(U_{N_{k-1}+1:N_k} N_k, S_k, X_k)$	$P(y_n U_n)$
Fig. 2.3(f)	$\mathbb{I}\{K_0 = 0\}$ $\times \mathbb{I}\{X_0 = \emptyset\}$	system specific	$f_n = P(X_n X_{n-1}) P(K_n K_{n-1}, X_n)$ $\times \mathbb{I}\{U_n \in X_n \setminus X_{n-1}\}$ $\times P(S_{K_{n-1}+1:K_n} K_n, U_n, X_{n-1})$		$P(y_n U_n)$



**Figure 2.4** Code trees of the FLC made up of the codewords (00, 01, 10, 11) on the left and of the VLC made up of the codewords (00, 10, 11, 010, 011) on the right. The internal nodes of the FLC are  $R$ ,  $I_1$  and  $I_2$ . The internal nodes of the VLC are  $R$ ,  $I_1$ ,  $I_2$  and  $I_3$ .



**Figure 2.5** Example of the first five sections of two trellises, for an FLC on top and a VLC at the bottom. The codes and code trees are given in Fig. 2.4. These trellises correspond respectively to the factor graphs of Fig. 2.3(b) and 2.3(d), without the counter  $K_n$  and for a source without memory so that  $x_n \in \mathcal{T}_n$ .

edge/variable  $x$  is the function

$$\mu_{g \rightarrow x}(x) \triangleq \sum_{y_1} \dots \sum_{y_n} g(x, y_1, \dots, y_n) \mu_{y_1 \rightarrow g}(y_1) \dots \mu_{y_n \rightarrow g}(y_n), \quad (2.5)$$

where  $\mu_{y_k \rightarrow g}(y_k)$  is the message that arrives at  $g$  along the edge/variable  $y_k$ . Notice that the definition of the SPR is recursive: Computing the message  $\mu_{g \rightarrow x}$  requires to compute the message  $\mu_{y_n \rightarrow g}$ . The SPA is actually the recursive application of the SPR onto all variables/edges in the graph. This recursive application is executed according to a certain schedule, called [4,6] *message-passing schedule*, that states in which order the variables/edges in the graph are processed.

After the recursion of the SPR, if the graph is acyclic, we can compute the joint probability

$$P(x, y_{1:T}) = \mu_{f_1 \rightarrow x}(x) \mu_{f_2 \rightarrow x}(x) \quad (2.6)$$

for each variable/edge  $x$  in the graph, where  $f_1, f_2$  are the only two functions/nodes connected to  $x$ . We can notably compute the joint probabilities  $P(u_n, y_{1:T})$  and  $P(s_k, y_{1:T})$  and thus perform one of the estimations (2.2)–(2.4) of the original signal presented in Section 2.3. Note the complexity of the SPA depends on the number of executions of the SPR (i.e., on the factor graph size and structure), on the state space associated with each variable/edge  $x$  and on the complexity of each function/node in the graph.

An important property of the SPA is that its message-passing schedule (the order of the edges processed by the SPA) can be chosen freely, up to a certain extent. On acyclic graphs, the cheapest schedule in number of operations is to process the graph in two phases, generally a centripetal phase from the leaf nodes of the graph to the root nodes and a centrifugal phase from the root nodes to the leaf nodes. On Markov chains, as in Fig. 2.3, the common practice is to perform the *forward/backward propagation*: a forward phase  $\alpha$  from left to right and a backward phase  $\beta$  from right to left. This is the BCJR algorithm.

### 2.3.3.2 BCJR algorithm

This forward/backward propagation is exactly the BCJR algorithm [42] on the corresponding trellis. Here is a detailed example, considering a Markov source encoded with an FLC and the factor graph of Fig. 2.3(b). The two

phases are given by the application in both directions (left to right, right to left) of the SPR onto the edges/variables  $x_n$ , i.e., they are given by the messages  $\mu_{f_n \rightarrow x_n}$  and  $\mu_{f_{n+1} \rightarrow x_n}$ . By applying the SPR (2.5),

$$\begin{aligned} \mu_{f_n \rightarrow x_n}(x_n) &= \sum_{x_{n-1}} \sum_{u_n} f_n(x_n, x_{n-1}, u_n) \\ &\quad \times \mu_{f_{n-1} \rightarrow x_{n-1}}(x_{n-1}) \mu_{h_n \rightarrow u_n}(u_n), \end{aligned} \quad (2.7)$$

$$\begin{aligned} \mu_{f_{n+1} \rightarrow x_n}(x_n) &= \sum_{x_{n+1}} \sum_{u_{n+1}} f_{n+1}(x_{n+1}, x_n, u_{n+1}) \\ &\quad \times \mu_{f_{n+2} \rightarrow x_{n+1}}(x_{n+1}) \mu_{h_{n+1} \rightarrow u_{n+1}}(u_{n+1}). \end{aligned} \quad (2.8)$$

Substituting into these expressions the values from Tab. 2.1 and denoting  $\alpha_n(x_n) = \mu_{f_n \rightarrow x_n}(x_n)$  and  $\beta_n(x_n) = \mu_{f_{n+1} \rightarrow x_n}(x_n)$  delivers the recurrences

$$\begin{aligned} \alpha_n(x_n) &= P(x_n, y_{1:n}) \\ &= \sum_{x_{n-1}} \underbrace{P(x_{n-1}, y_{1:n-1})}_{\alpha_{n-1}(x_{n-1})} P(x_n | x_{n-1}) \\ &\quad \times \sum_{u_n=0,1} P(u_n | x_n, x_{n-1}) P(y_n | u_n), \end{aligned} \quad (2.9)$$

$$\begin{aligned} \beta_n(x_n) &= P(y_{n+1:N} | x_n) \\ &= \sum_{x_{n+1}} \underbrace{P(y_{n+2:N} | x_{n+1})}_{\beta_{n+1}(x_{n+1})} P(x_{n+1} | x_n) \\ &\quad \times \sum_{u_{n+1}=0,1} P(u_{n+1} | x_{n+1}, x_n) P(y_{n+1} | u_{n+1}), \end{aligned} \quad (2.10)$$

where we have highlighted the fact that  $\alpha_n(x_n)$  and  $\beta_n(x_n)$  are respectively the probabilities<sup>1</sup>  $P(x_n, y_{1:n})$  and  $P(y_{n+1:N} | x_n)$ . The recurrence (2.9) constitutes the forward phase of the BCJR, the recurrence (2.10) the backward phase. They are initialized with  $\alpha_0(x_0) = \mathbb{I}\{x_0 = R\}$  and  $\beta_N(x_N) = 1$ . After the two recurrences, we can calculate the joint probabilities  $P(u_n, y_{1:N})$  and  $P(s_k, y_{1:N})$  using (2.6), to get the marginals required by (2.2)–(2.4) (with  $N = T$ ),

$$P(u_n, y_{1:N}) = \mu_{f_n \rightarrow u_n}(u_n) \mu_{h_n \rightarrow u_n}(u_n), \quad (2.11)$$

$$P(s_k, y_{1:N}) = \mu_{f_{kM} \rightarrow s_k}(s_k). \quad (2.12)$$

<sup>1</sup>Or, more properly, values of probability density functions.

Using the SPR again, these can be developed as

$$\begin{aligned} P(u_n, y_{1:N}) &= \mu_{f_n \rightarrow u_n}(u_n) \mu_{h_n \rightarrow u_n}(u_n) \\ &= \sum_{x_n} \beta_n(x_n) \sum_{x_{n-1}} \alpha_{n-1}(x_{n-1}) \\ &\quad \times P(x_n | x_{n-1}) P(u_n | x_n, x_{n-1}) P(y_n | u_n), \end{aligned} \quad (2.13)$$

$$\begin{aligned} P(s_k, y_{1:N}) &= \mu_{f_{kM} \rightarrow s_k}(s_k) \\ &= \sum_{x_{kM}} \beta_{kM}(x_{kM}) \sum_{x_{kM-1}} \alpha_{kM-1}(x_{kM-1}) \\ &\quad \times P(x_{kM} | x_{kM-1}) \sum_{u_{kM}} P(u_{kM} | x_{kM}, x_{kM-1}) \\ &\quad \times P(s_k | u_{kM}, x_{kM-1}) P(y_{kM} | u_{kM}). \end{aligned} \quad (2.14)$$

### 2.3.3.3 Viterbi algorithm

The frame-MAP, in eq. (2.1), is a special case which is tackled differently. Essentially, the frame-MAP is the problem of finding in the trellis the path with the highest accumulated metric  $P(s_{1:K}, y_{1:N})$ , or equivalently  $P(x_{0:N}, y_{1:N})$ , during a single forward phase. This can be formulated as a Shortest Path Problem on the trellis. It is solved efficiently by the Viterbi algorithm [52] and requires less computations than the BCJR at the receiver (a forward phase instead of two forward/backward phases). For the trellises in Fig. 2.5, it recursively computes

$$P(x_{0:n}, y_{1:n}) = P^*(x_{0:n-1}, y_{1:n-1}) P(x_n | x_{n-1}) P(y_n | x_n, x_{n-1}), \quad (2.15)$$

$$P^*(x_{0:n}, y_{1:n}) = \max_{x_{n-1}} P(x_{0:n}, y_{1:n}), \quad (2.16)$$

initialized with  $P^*(x_0) = \mathbb{I}\{x_0 = R\}$ . At the end of the recursion, the quantity  $P^*(x_{0:n}, y_{1:n})$  is the highest accumulated metric among all paths that lead to state  $x_n$ , i.e., the recursion (2.15)–(2.16) computes  $P^*(x_{0:n}, y_{1:n}) = \max_{x_{0:n-1}} P(x_{0:n}, y_{1:n})$ .

### 2.3.3.4 Soft-In/Soft-Out decoders

In most cases, the messages  $\mu_{g \rightarrow x}(x)$ ,  $\mu_{y_n \rightarrow g}(y_n)$  at the heart of the SPR/SPA in (2.5) are probabilities or values of probability density functions. Recall for example the BCJR equations from Section 2.3.3.2.

In the literature, these messages or probabilities are commonly referred to as *soft decisions* or as *soft information*. This terminology contrasts with the “hard” decisions exchanged in classical tandem decoders. Let us consider for example the estimation of a bit, say  $x \in \{0, 1\}$ . A hard decision on  $x$  is either the value 0 or 1. By contrast, a soft decision on  $x$  is a probability, e.g., the probability that  $x$  is equal to 0 given a certain statistical model. Thus, compared to the hard decision, the soft decision is able to convey more information about  $x$ .

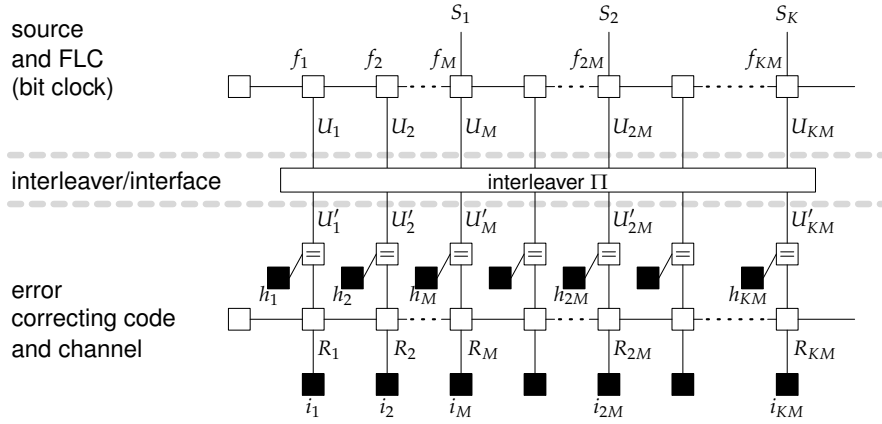
Using this terminology, the SPA consists in the recursive exchange of soft information between the functions/nodes in the factor graph. As a by-product, decoders based on the SPA belong to the family of the so-called Soft-In/Soft-Out decoders, as they accept soft information at their input and produces soft information at their output. For example, the BCJR algorithm described in Section 2.3.3.2 accepts the probabilities  $P(y_n|u_n)$  of the channel measures as input in (2.9), and produces the probabilities  $P(s_k, y_{1:N})$  as output in (2.14). Note that, for the same reasons, decoders based on the Viterbi algorithm are also Soft-In/Soft-Out decoders.

### 2.3.3.5 Summary

To conclude this section, let us summarize that the bit-MAP, symbol-MAP and MMSE can be solved by applying the Sum-Product algorithm on the factor graph. For the source codes considered in this chapter, this is equivalent to the application of the BCJR algorithm on the corresponding trellis. As for the frame-MAP, it can be solved by the Viterbi algorithm on the trellis. At last, recall that decoders based on the Sum-Product algorithm or on the Viterbi algorithm belong to the family of Soft-In/Soft-Out decoders.

## 2.4 Turbo decoding

To resist the multiple distortions potentially caused by the channel, the bits produced by the source coder are protected, after interleaving, by a channel coder in the transmission chain in Fig. 2.1. The factor graph of the complete transmission chain is depicted in Fig. 2.6. For this example, the source code on the top is an FLC, modeled with the bit clock Markov chain of Fig. 2.3(b).



**Figure 2.6** Factor graph of the transmission chain of Fig. 2.1, with an FLC as source code [bit clock model, Fig. 2.3(b)] and a systematic convolutional encoder as channel code. Only the main variables are indicated. Channel measures are available on the systematic bits  $U'_n$  and on the parity bits  $R_n$ , as represented by the black functions/nodes  $h_n$  and  $i_n$ , respectively.

The channel code at the bottom is a convolutional code (CC), modeled also as a Markov chain.

### 2.4.1 Interleaver and cycles in the factor graph

As explained in the introduction, one of the roles of the interleaver is to allow the use of iterative decoding as a low complexity approach to near-optimal decoding — another role, code spreading, is explained later. Without interleaver, indeed, either the SPA behaves badly [6] due to the many short cycles in the factor graph between the two Markov chains (of the FLC and of the CC), or the two Markov chains are to be merged into a new chain of generally intractable complexity. With the interleaver, the short cycles become long and instead of computing, as in the ideal cycle-free case, the exact probabilities in (2.6) for any edge/variable in the factor graph, we get approximations of these. In many cases, the longer the cycles, the better the reliability of the approximations will be. The SPA can thus be used as a good (approximate) alternative to the optimal estimation, as the success of turbo techniques has attested so far. Still, there is one inevitable effect of the (short and long) cycles on the SPA: The cycles introduce infinite algorithmic loops in the recursion of

the SPR (2.5). As a consequence, the algorithm becomes iterative [6] and the resulting joint source-channel decoder belongs to the family of *turbo/iterative decoders*.

### 2.4.2 Turbo decoder structure

The variables/edges processed by the SPA are processed in a certain order, according to the so-called *message-passing schedule* which can be chosen freely up to a certain extent. One common schedule is for the factor graph in Fig. 2.6 to consider first the forward/backward propagation (Section 2.3.3.2) on the CC Markov chain, then the SPR on the variables/edges  $U'_{1:N}$  going through the interleaver  $\Pi$ , which produces the messages  $\mu_{\rightarrow u'_n}$ . Next a second forward/backward propagation is considered on the FLC Markov chain, then the SPR on the variables/edges  $U'_{1:N}$  through the interleaver  $\Pi$  in the reverse direction, which produces the messages  $\mu_{f_n \rightarrow u_n}$ . This schedule constitutes one iteration of the turbo decoder and is theoretically repeated an infinite number of times (due to the cycles). In practice, though, it is repeated until some stopping condition is satisfied.

This schedule can be interpreted in terms of Soft-In/Soft-Out decoding modules. Since the Markov chains are processed separately, we can indeed consider three parts in the factor graph in Fig. 2.6: the source decoding module with the FLC on the top, the channel decoding module with the CC at the bottom, and an interface/interleaver in between. This separation in three parts leads to the common structure illustrated in Fig. 2.2, where the turbo decoder is composed of two decoding modules, one for the FLC (source module) and one for the CC (channel module), separated by the (de)interleavers. At the first iteration, the channel module applies the forward/backward propagation (Section 2.3.3.2) on the CC Markov chain. This generates a first estimation of the bits  $U'_{1:N}$ , the messages  $\mu_{\rightarrow u'_n}$ , which are sent to the source module across the interleaver. The source module then uses these messages as input (after deinterleaving) and applies the forward/backward propagation on the FLC Markov chain. This generates the messages  $\mu_{f_n \rightarrow u_n}$  which are sent to the channel module across the interleaver. At the second iteration, the channel module benefits from the messages  $\mu_{f_n \rightarrow u_n}$  from the source module, uses them as input (after interleaving) to further improve the estimation and

sends new messages  $\mu_{\rightarrow u'_n}$  across the interleaver. The source module is then used again. This process is repeated several times until some stopping condition is satisfied. After that, the source module performs one of the estimations of the original signal presented in Section 2.3.

The turbo decoder consists thus essentially in the iterative exchange of messages across the interleaver between the source module and the channel module. As we have seen, these messages are actually probabilities, thus real numbers. So the cooperation between the source module and the channel module is relatively simple, as it requires simply an iterative interface of real numbers. Such an interface enables a great separation/independence between the source and channel layers in the communication protocol, which is often desirable and valuable from an industrial standpoint.

The success of joint source-channel turbo decoders has been shown in different contexts by many contributions in the literature [3, 11, 21–41]. A few selected simulation results corroborate this success in Section 2.8.

### 2.4.3 Convergence of turbo decoding

With an iterative process is always associated the concept of convergence. In the context of turbo decoding, the convergence of the turbo decoder depends intuitively on the amount of redundancy present at both sides of the interleaver. If, indeed, one module/decoder (on one side of the interleaver) has neither *a priori* information (redundancy) on the transmitted signal nor channel measures available on the received signal, then this module cannot improve the quality of the estimation, i.e., it cannot contribute to the iterations, thus the iterations are almost useless and the turbo decoder does not perform much better than a tandem decoder.

The convergence of turbo decoding can be analyzed with the extrinsic information transfer (EXIT) charts, which have been introduced originally in [12] for parallel turbo codes. The key idea underlying EXIT charts is to measure the reliability of the estimation of a bit by the mutual information between that estimation and the original bit. More precisely, we measure the

reliability at the output of the source decoding module as

$$I_S = I \left\{ U_n ; \log \left( \frac{\mu_{f_n \rightarrow u_n}(0)}{\mu_{f_n \rightarrow u_n}(1)} \right) \right\} \in [0, H_U], \quad (2.17)$$

and at the output of the channel decoding module as

$$I_C = I \left\{ U_n ; \log \left( \frac{\mu_{\rightarrow u_n}(0)}{\mu_{\rightarrow u_n}(1)} \right) \right\} \in [0, H_U], \quad (2.18)$$

where  $I\{A;B\}$  is the mutual information between  $A$  and  $B$ . The maximum value  $H_U$  is the stationary entropy of a bit, i.e.,  $H_U = I(U_n; U_n) = H_b(P(U_n = 1))$ , which we assume here independent of  $n$ , where  $H_b(\cdot)$  is the binary entropy function. If  $I_S$  and  $I_C$  are close to  $H_U$  (resp. 0), it means that the estimation is highly (resp. is not) correlated with the original signal, i.e., the estimation is good (resp. bad). The turbo decoder starts with no estimation,  $I = 0$ . The channel decoder outputs a first estimation of reliability  $I_C^1$ . With that estimation, the source decoder is able to produce  $I_S^1$ . At the  $j$ th iteration, we have  $I_C^j$  and  $I_S^j$ . We are then interested in having/designing a system such that the sequence  $(I_C^1, I_C^2, I_C^3, \dots)$  or  $(I_S^1, I_S^2, I_S^3, \dots)$  or both converge to a value close to  $H_U$  with the fewest iterations as possible.

Interestingly, and this is the second key idea underlying EXIT charts, the evolution of  $I_C$  can be characterized by a transfer function, a transfer “chart”, depending only on  $I_S$ , and vice versa. In many applications indeed, given a channel code and a fixed level of noise on the channel, the value of  $I_C^j$  tends to depend only on the value of  $I_S^{j-1}$ , for large interleavers, independently of the iteration  $j$  and independently of the source code. Vice versa, given a source and a source code,  $I_S^j$  tends to depend only on  $I_C^j$ , independently of  $j$  and independently of the channel code. We can therefore write  $I_C^j = T_C(I_S^{j-1})$  and  $I_S^j = T_S(I_C^j)$ , where  $T_C(\cdot)$  and  $T_S(\cdot)$  are the EXIT charts of the channel decoder and of the source decoder, respectively.

Given these charts, assessing the convergence of the turbo decoder is straightforward. At the first iteration, the mutual information is given by  $I_C^1 = T_C(0)$  at the output of the channel decoder and by  $I_S^1 = T_S(I_C^1) = T_S(T_C(0)) = T_S \circ T_C(0)$  at the output of the source decoder. At the second iteration,  $I_S^2 = T_S \circ T_C \circ T_S \circ T_C(0)$ . By induction, it is self-evident that the turbo decoder converges to the maximum value  $H_U$  if  $T_S \circ T_C \circ \dots \circ T_S \circ T_C(0)$

tends to  $H_U$ , i.e., if the function  $T_S \circ T_C(\cdot)$  is strictly increasing on the interval  $[0, H_U)$ .

The EXIT charts  $T_C(\cdot)$  and  $T_S(\cdot)$  can be computed independently since  $T_C(\cdot)$  tends to be independent of the source code, and  $T_S(\cdot)$  independent of the channel code. This is generally done with Monte-Carlo simulations [12], by feeding the considered decoder with artificially generated input soft information and by measuring the mutual information at the output of the decoder — note that analytical expressions exist for some codes [53]. In practice, given a family of source codes and a family of channel codes, we can compute the EXIT charts of all corresponding decoders in each family, independently, and then choose the most promising combination of codes in terms of the function  $T_S \circ T_C(\cdot)$ . This is perhaps the most common application of EXIT charts.

In the field of joint source-channel turbo techniques, analysis and/or optimization of the transmission system have been carried out with EXIT charts notably in [21, 23, 31, 36, 39, 40]. Most of these contributions, however, neglect or do not take into account the value of the entropy  $H_U$  (or of the probability  $P(U_n = 1)$ ) in (2.17)–(2.18). A preliminary solution to take this value into account is proposed in Section 3.5. It is then further developed in Chapter 4.

## 2.5 Soft decoding of different source codes

One of the main advantages of FLCs, considered in Section 2.3.2, is their robustness thanks to their inherent synchronization: The symbol positions in the bit stream are perfectly known (and fixed) *a priori*. For arbitrary source distributions, however, the level of residual redundancy in the bit stream may be high and may constitute a significant drawback in terms of bandwidth efficiency.

To solve this bandwidth inefficiency, variable length source codes can be used instead, at the expense though of some sensitivity to desynchronization at the decoder.

Only the state models and the corresponding factor graphs are described. The decoding algorithms can be derived by running the SPA on the graphs, as we have illustrated it with FLCs in Section 2.3.3.

### 2.5.1 Variable Length Code (VLC)

VLCs, also known as prefix VLCs or Huffman-like VLCs [54] assign a variable length codeword to each source symbol. To achieve compression, shorter codewords are associated with more frequent symbols — in this chapter, only the stationary probabilities  $P(S_k)$  are considered for compression, not  $P(S_k|S_{k-1})$ . Unfortunately, VLCs are sensitive to desynchronization: a single bit error can indeed desynchronize the VLC decoder and generate a burst of several symbol errors. In terms of state models and factor graphs, VLCs are just an extension of FLCs.

Since the VLC coder produces a variable number of bits per symbol with the symbol clock model in Fig. 2.3(c), the bit position of the current symbol is not known without storing the number  $N_k$  of bits encoded up to the current symbol  $S_k$ . So the Markov states are now the pairs  $(S_k, N_k)$ , see the detailed model in Tab. 2.1. While this does not increase the complexity at the encoder, it really does at the decoder under soft source decoding. Besides, notice that this affects the graph in a very unusual way: Its structure, i.e., the number of edges and the interconnections between the nodes, is now random and depends on  $N_{1:K}$ .

With the bit clock model in Fig. 2.3(d), we have the symmetrical problem: At each bit position, either no symbol is output or only one, so the number of symbols is variable. Therefore, the symbol position of the current bit  $u_n$  is not known without storing the number  $K_n$  of symbols output up to the current bit position  $n$ . So the Markov states are now the pairs  $(X_n, K_n)$  (see the function definitions in Tab. 2.1). The states  $X_n$  correspond to those of the Balakirsky trellis [55]: The different values of  $X_n$  are the internal nodes of the VLC code tree, extended with the source memory if necessary (as with FLCs). An example of code tree and the corresponding trellis are given on the right of Fig. 2.4 and at the bottom of Fig. 2.5. Note that for a source without memory, the final constraint  $c_e$  can be written as  $c_e = \mathbb{I}\{K_N = K\}\mathbb{I}\{X_N = R\}$ , where  $\mathbb{I}\{K_N = K\}$  accounts for the fact that the decoder knows that the number of symbols in the transmitted sequence is  $K$ , and  $\mathbb{I}\{X_N = R\}$  accounts for the fact that the bit  $U_N$  is the last bit of a codeword and that we are thus in one of the leaf nodes of the VLC binary tree (Fig. 2.4), i.e.,  $X_N = R$ . Finally, as

explained in Section 2.5.4, note that the bit clock model may be simplified in the case of the frame-MAP and the bit-MAP estimations.

### 2.5.2 Arithmetic Code (AC)

While VLCs and FLCs assign a codeword to each source symbol, arithmetic coding [56] encodes the entire symbol sequence  $S_{1:K}$  into a single number  $x$  between 0.0 and 1.0. The binary representation of  $x$  is the sequence of transcoded bits  $U_{1:N}$ . Optimal AC encoding proceeds by successively subdividing a probability interval  $[\text{low}_k, \text{up}_k[$ , initialized to  $[\text{low}_0, \text{up}_0[ = [0.0, 1.0[$ , according to the symbol probabilities: Each possible realization  $s_k$  of the current symbol  $S_k$  uniquely defines a sub-interval  $[\text{low}_{k+1}, \text{up}_{k+1}[$  of the current interval  $[\text{low}_k, \text{up}_k[$ , proportional to  $P(s_k)$ :  $\frac{\text{up}_{k+1} - \text{low}_{k+1}}{\text{up}_k - \text{low}_k} = P(s_k)$ . The number  $x$  is any fraction falling in the final interval. At the decoder,  $x$  identifies the final interval from which all symbols can be reconstructed unambiguously. ACs are able to reach near-optimal compression for a given set of symbols and probabilities but are unfortunately very sensitive to desynchronization. Except in some rare cases, almost no resynchronization is actually possible (on the contrary to VLCs) and a single bit error generally invalidates the rest of the frame.

In terms of state models, compared to VLCs, the AC coder not only produces a variable amount of bits per symbol but that amount can be zero (if  $N_{k+1} = N_k$ ). In addition, the internal state of the arithmetic coder, hence the coded bits, depend on all previously encoded symbols. As a consequence, compared to VLCs, there is an additional dimension to the model: We have to store the whole sequence of symbols coded up to the current symbol. For the symbol clock model, depicted in Fig. 2.3(e), the Markov states are now the triplets  $(S_k, N_k, X_k)$  where  $X_k = \{S_1, S_2, \dots, S_{k-1}\}$ . Note that  $N_k$  is unnecessary here since it can be deduced from  $(S_k, X_k)$ ; however, we keep it so as to ease the comparison with VLCs. The model dependencies are summarized on the fifth row of Tab. 2.1. Note that the final constraint  $c_e$  is system specific: It depends on the way  $x$  is chosen within the final interval.

The bit clock model can be constructed in a similar manner, as an extension of the VLC bit clock model (see Fig. 2.3(f) and Tab. 2.1). Instead of storing the internal nodes of the VLC code tree, i.e., the bits of the current codeword, the

state  $X_n$  stores the whole sequence of bits output up to the current bit position  $n$ ,  $X_n = \{u_1, u_2, \dots, u_n\}$ . The Markov states are the pairs  $(X_n, K_n)$ . This model is associated with a high increase in complexity compared to VLCs, like the symbol clock model.

These two simple models, bit clock and symbol clock, allow us to illustrate the very high complexity of optimal decoding of ACs. The state space of both models grows exponentially in the sequence length. A direct application of the SPA and BCJR algorithm is thus intractable. One has to rely instead on suboptimal decoding techniques such as sequential decoding with pruning (see Section 2.5.4) to keep the complexity within a tractable range. An interesting alternative to AC, avoiding these complexity problems and presented hereafter, is quasi-arithmetic code.

At last, note that in practice, the bit clock and symbol clock models presented above are not used because they require infinite precision. Instead, probability intervals are considered as Markov states. Though these intervals do not really diminish the decoding complexity, they solve the numerical precision problem. Basically, as soon as all real numbers within the current interval share a common binary representation, that representation is output and the interval is rescaled accordingly. Further rescaling is also possible when the interval falls into  $[0.25, 0.75[$ .

### 2.5.3 Quasi-Arithmetic (QA) Code

A reduced precision implementation of arithmetic coding, called quasi-arithmetic (QA) coding, has been proposed in [57]. The QA coder operates integer subdivisions of an integer interval  $[0, Q[$ . The integer interval subdivisions lead to some approximation of the source distribution. The parameter  $Q$  controls both the state space dimension, hence the decoding complexity, and the source distribution approximation. It has been shown in [58] that, for a binary source, the variable  $Q$  can be limited to a small value (down to 4) at a small cost in terms of compression efficiency.

The number of possible subdivisions of the interval  $[0, Q[$  being finite, QA codes can be modeled as Finite State Machines. Trellis decoding, using e.g., the SPA or the BCJR algorithms, can thus be applied [35] with a reasonable com-

plexity. As with AC, probability intervals can be considered as Markov states:  $[\text{low}_k, \text{up}_k[$  for the symbol clock model,  $[\text{low}U_n, \text{up}U_n[$  and  $[\text{low}S_{K_n}, \text{up}S_{K_n}[$  for the bit clock model. But, compared to ACs, the different probability intervals define integer segments of the interval  $[0, Q[$  rather than fractional segments of the interval  $[0, 1[$ .

## 2.5.4 Complexity issues and suboptimal decoding

### 2.5.4.1 Bit-level trellis and bit/symbol clock models

Both the symbol clock and the bit clock models can be used for the bit, symbol and frame optimal estimations presented in Section 2.3.1. However, for VLCs, these models in Fig. 2.3(c) and 2.3(d) have a complexity growing as a quadratic function of the sequence length, and even higher for ACs. That complexity is actually not tractable for typical sequence lengths.

Some simplifications are possible with the bit clock model in the case of the frame-MAP or the bit-MAP estimations if the number  $K$  of symbols is not known at the receiver. In that case, all the symbols  $S_k$  and the variables  $K_n$  can be removed from the factor graphs (Fig. 2.3(d) and 2.3(f)). For VLCs, these simplifications enable an optimal frame-MAP or bit-MAP decoding with a complexity approximately growing linearly as the product of  $K$  and the source alphabet size  $|\mathcal{A}|$ . The corresponding trellis is often called the “bit-level” trellis, or Balakirsky trellis, in the literature and was originally proposed in [55] — an example of such a trellis is given at the bottom of Fig. 2.5 for a source without memory.

Unfortunately, in the case of symbol-MAP or MMSE, or if the pair  $(K, N)$  is known and used as a termination constraint, the complexity hurdle remains for the optimal estimation. In order to overcome it, most authors consider suboptimal methods. They (a) either use the bit-level trellis, (b) or apply suboptimal estimation methods such as sequential decoding [59] to the bit or symbol clock models. Method (a), together with the SPA or BCJR algorithm, amounts to optimally estimating with a suboptimal Hidden Markov model. Method (b) processes a suboptimal estimation on an optimal model which fully represents the whole transmission chain.

#### 2.5.4.2 Aggregated State Model

To lower the decoding complexity, an aggregated state model is introduced in [60], based on the bit clock model  $(X_n, K_n)$ . It is defined by both the internal state  $X_n$  of the VLC decoder (i.e., the internal node of the VLC code tree) and  $M_n = K_n \bmod T$ , the rest of the Euclidean division of the symbol clock  $K_n$  by a fixed parameter  $T$ , such that  $1 \leq T \leq K$ . The state model is thus defined by the set of tuples  $(X_n, M_n)$ . When the state  $X_n$  reaches a leaf node in the VLC binary tree ( $X_n = R$ ), it means that the current codeword has just been terminated and the modulo  $M_n$  is therefore updated as  $M_n = M_{n-1} + 1 \bmod T$ . In other words, the model consists in merging/aggregating the states of the bit/symbol trellis which are distant of  $T$  symbol clocks. If  $T = 1$ , the resulting trellis is equivalent to the bit-level trellis or Balakirsky trellis proposed in [55]. If  $T$  is greater than or equal to the symbol sequence length  $K$ , the trellis is equivalent to the full bit-clock trellis  $(X_n, K_n)$ . The intermediate values of  $T$  (between 1 and  $K$ ) enables to trade the decoding complexity against the estimation accuracy. The intuition behind this state aggregation is that a desynchronization error will be detected if the difference between the numbers of transmitted and decoded symbols is a quantity which is not a multiple of  $T$ . The choice of the parameter  $T$  depends therefore on the capability of the considered VLC to quickly resynchronize. We come back on this issue in Section 2.6.

The aggregation above deals with  $K_n$ . A similar principle has also been developed for  $X_n$  in [61] by compacting the VLC table. The codewords are grouped/aggregated into a minimum number of classes. The decoding algorithm may then work on a reduced number of classes, hence on a reduced number of states  $X'_n$ , instead of working on the whole set of codewords. Note that these two state aggregations,  $K_n \rightarrow M_n$  and  $X_n \rightarrow X'_n$ , are complementary and can be combined.

#### 2.5.4.3 Pruning

To lower the decoding complexity, simplifying the model is sometimes not sufficient, especially with ACs. In such cases, suboptimal decoding has to be considered, as in [34, 36, 59]. In particular, several techniques suggested

originally for convolutional codes and based on the Viterbi algorithm can be used, often called *sequential decoding*. A few of them are covered here.

To find in the trellis the path with the highest accumulated metric, the Viterbi algorithm recursively implements an exhaustive search by maintaining one best path, called survivor, in each state of the current trellis section. Each survivor is then extended in (2.15) with the transitions of the next trellis section and only the best resulting paths are kept in (2.16) in each state. Unfortunately, the number of survivors to keep in memory may be too large and may lead to an intractable complexity. To avoid that, a part of the survivors have to be pruned (discarded) in some ways. For example, the M-algorithm [62] keeps only the  $M$  best survivors in each trellis section and the T-algorithm [63] keeps only the survivors whose likelihood is above a given threshold  $T$ , both algorithms according to a breadth first strategy. The stack algorithm [64] (metric first), and similarly the Fano algorithm [65] (depth first), always expand the best survivor only. For the T-algorithm and the stack algorithm, the number of survivors is further limited by a predefined maximum so as to bound the complexity. The stack algorithm has been extended in [66] to make it suitable for iterative decoding. Note finally that the same pruning principles can be applied to the SPA and to the BCJR algorithm. This is roughly equivalent to consider only the dominant terms in the SPR in (2.5).

#### 2.5.4.4 Complexity-Performance Issues

Let us review shortly the complexity of the *optimal* estimations (2.1)–(2.4). They can be deduced from Fig. 2.3 and Tab. 2.1. For FLCs, all estimations have a complexity growing linearly with  $K$ , the number of symbols. For VLCs, they grow quadratically with  $K$ , except in one case: If  $K$  is not known at the receiver (Section 2.5.4.1), the frame- and bit-MAP grow linearly with  $K$  — compared to FLCs, the decoding complexity is then multiplied roughly by the codeword length of the FLC. For ACs, all estimations grow exponentially with  $K$ ; only suboptimal estimations are affordable. In all cases, the complexity can be of course reduced by using suboptimal decoding techniques such as described above.

Concerning the choice of the estimation, the symbol-MMSE and symbol-MAP are usually the most interesting ones for the source distortion, while the

bit- and frame-MAP have the lowest complexity. But such a comparison is simplistic. For example, when VLCs or ACs are used, the main concern at the receiver to reduce the distortion is usually the decoder desynchronization, Section 2.6. And reducing the desynchronization is already handled by the frame-MAP decoder, even if not optimally. Another example is when the application tolerates a time shift of the source signal. Such a possible time shift is not taken into account by any of the estimations (2.1)–(2.4). In that case, no estimation is *a priori* better than the others.

At last, concerning the performance of the different source codes, a comparison is beyond the scope of this chapter. Let us summarize that there is no universal solution. On the one hand, FLCs are more robust and not sensitive to desynchronization. But on the other hand, VLCs and ACs save bandwidth which can be used to improve their robustness against channel errors.

## 2.6 Synchronization and Robustness of Entropy codes

In the case of VLCs and ACs, the decoder desynchronization problem prevails in the performance of the end-to-end transmission chain. Different solutions can be considered to fight against this phenomenon and to increase the robustness. We can for example introduce more redundancy in the bit stream or better structure the bit stream. Some of these techniques are now reviewed.

### 2.6.1 Synchronization mechanisms

When soft decoding is used for VLCs or ACs, several mechanisms can be incorporated in the (de)coding process to help the decoder resynchronization in the presence of bit errors.

#### 2.6.1.1 Termination constraint

If the number of transmitted symbols and/or bits are known by the decoder, termination constraints can be incorporated in the decoding process. For example, one can ensure that the decoder produces the right number of

symbols ( $K_N = K$ ), if  $K$  is known. All the paths in the trellis which do not lead to a valid sequence length are discarded. This is exactly the purpose of the termination constraints,  $c_b$  and  $c_e$  in Fig. 2.3, which enable to synchronize the decoder at both ends of the sequence.

### 2.6.1.2 Soft synchronization

Extra bits (small patterns), often called synchronization markers, can be incorporated at some known positions in the symbol stream to help achieving a proper segmentation of the received noisy bit stream into segments corresponding to the transmitted symbols. This extra information can take the form of dummy symbols (in the spirit of the techniques described in [47, 67]) or of dummy bit patterns, which are inserted in the symbol stream or in the bit stream, respectively, at some known symbol clock positions. The procedure amounts to extending symbols at known positions with a predetermined suffix. These suffixes favor the likelihood of correctly synchronized sequences (i.e., paths in the trellis), and penalize the others.

### 2.6.1.3 Forbidden symbol and cyclic code

To detect and prune erroneous paths in soft AC decoding, the authors in [67] use a reserved interval corresponding to a so-called *forbidden symbol*. All paths hitting this interval are considered erroneous and pruned. This technique is analyzed in [68]. Similarly, the authors in [29] append to the VLC stream a cyclic redundancy check code. At the receiver, a list Viterbi decoding is applied and all paths that do not respect the cyclic code constraint are considered erroneous and pruned.

## 2.6.2 Resynchronization properties of VLC

The respective performance of the different mechanisms depends notably on the synchronization or error recovery properties of the VLC codes being used. Error recovery properties of VLCs have been first studied in [69], where a method is proposed to compute the so-called *expected error span*  $E_s$ , i.e., the expected number of source symbols on which a single bit error propagates. For a given VLC, the lower  $E_s$ , the better the error resilience of this code is

when hard decoding is applied at the decoder side. Later, this method has been adapted in [70] to compute the so-called *synchronization gain/loss*, i.e., the probability that the number of symbols in the transmitted and decoded sequences differ by a given amount  $\Delta S$  when a single bit error occurs during the transmission. In [71] it is shown that the probability mass function (p.m.f.) of the synchronization gain/loss is a key feature of a VLC to analyze the error resilience of such codes when soft decoding with length constraint is applied at the decoder side. Transfer functions defined on an error state diagram enable to estimate the probability that the number of symbols in the transmitted and decoded sequences differ by a given amount  $\Delta S$ . One notable result concerns the probability of resynchronization and the state aggregation technique described in Section 2.5.4.2, that is, the probability that the VLC decoder does not resynchronize in a strict sense (or equivalently  $P(\Delta S \neq 0)$ ) is almost not altered by the state aggregation. Also, and surprisingly, the codes offering the best error resilience under soft decoding with constraint length are not those having the highest resynchronization probability, i.e., the highest  $P(\Delta S = 0)$ .

### 2.6.3 Error correction capabilities of VLC

Besides the possibility to resynchronize after a bit error, VLCs may also exhibit error correction capabilities. These capabilities are obtained through an increased Hamming distance between the binary representations of any two different symbol sequences, at the expense of a compression loss (more redundancy in the bit stream). In other words, several bit errors must occur before the decoder selects a wrong path. Such VLCs are sometimes called variable length error correcting codes (VLECCs).

The analysis of the error correction capabilities of VLCs was notably developed in [14] and is based on the concept of distance spectrum, as with error correcting codes. This analysis was then generalized and extended to turbo systems, in [72,73] and in [20] independently, to provide relatively tight (at least in [20]) performance bounds. Unfortunately, the expression of the spectrum is given without development in these contributions, as a simplistic combination of previous results from [14–16], and some important details are missing. In Chapter 5, we will carry these contributions one step further by

developing more accurate results as well as new results, and by proving them rigorously.

In particular, one promising property of VLECCs in turbo systems is the possibility to improve the so-called interleaving gain, under certain assumptions. The interleaving gain is a key property of parallel turbo codes that was analyzed in [15], which results in a decrease of the BER (bit error rate) as  $N^{-1}$ , where  $N$  is the sequence length. In Chapters 3 and 5, we will investigate which interleaving gains can be expected with VLCs, on the  $SER_L$ , the SER and the FER.

## 2.6.4 Recent advances in source coding

### 2.6.4.1 Reversible Variable Length Codes (RVLC)

RVLCs [48] are a special case of VLECCs, increase the resilience at a small cost in compression and are used in recent audio and video standards, such as AAC [74] and MPEG-4 [75]. RVLCs are both prefix- and suffix-free. They can thus be decoded in both directions. This helps the hard decoder to recover from a short error burst: As soon as an error is detected, we can restart decoding in the reverse direction from the end of the sequence and recover as much information as possible.

In turbo systems, most RVLCs with a free distance equal to 1 can show additional resilience capabilities thanks to particular interleaving gains, compared to other VLCs of free distance 1. In particular, the probability of desynchronization of the decoder can tend to zero with long interleavers, i.e., with long sequences. Indeed, with any RVLC, a single bit error leads only to a single symbol error. In other words, two bit errors at least are necessary to desynchronize the decoder and to produce a burst of symbol errors. The probability of such two bit error events diminish at least as  $N^{-2}$  for most RVLCs if we take, e.g., a parallel turbo code as channel code in Fig. 2.1. Then, the probabilities of symbol error bursts ( $N^{-2}$ ) and of symbol errors ( $N^{-1}$ ) tend to zero with long sequences (large  $N$ ), which is a useful resilience property. Further details on these interleaving gains are given in Chapter 3.

#### 2.6.4.2 Multiplexed Codes

Effort has been dedicated in [50, 51] to design codes that are less sensitive to the decoder desynchronization while at the same time able to approach the source entropy. A family of codes has been introduced, called *multiplexed codes*. These codes are appropriate when we have to encode two (or more) sources of information with different levels of priority. The key idea relies on the fact that lossy compression systems of real signals generate very often such sources, e.g., texture and motion information for a video signal. The risk of “desynchronization” is then confined to the low priority information so as to make the high priority information almost insensitive to the decoder desynchronization.

A high priority source and a low priority source are considered and referred to respectively as  $S_H$  and  $S_L$ . The idea consists in creating an FLC of  $N = 2^c$  codewords for the source  $S_H$ , in partitioning the set of codewords into subsets (or *classes*)  $\mathcal{C}_i, i = 1 \dots \Omega$ , where the class  $\mathcal{C}_i$  is associated with the symbol  $a_i$  of the source alphabet  $\mathcal{A}$ . Each *class*  $\mathcal{C}_i$  contains  $N_i$  codewords. A symbol  $S_t = a_i$  of the source  $S_H$  can be encoded with any codeword  $c_{i,q}$  belonging to the *class*  $\mathcal{C}_i$ . The redundancy inherent to each class is then used to represent information of the low priority source  $S_L$ . With the realization of the sequence of symbols  $S_H$ , one associates a sequence of  $N_i$ -valued variables indexing the codewords in the classes. In order to be multiplexed with the symbols of  $S_H$ , the lower priority bit stream must be mapped into the sequence of  $N_i$ -valued variables associated with the realization of the high priority source  $S_H$  [51]. To reach the source entropy, the cardinalities of the classes  $\mathcal{C}_i$  must be chosen such that  $N_i = \mu_i 2^c$  where  $\mu_i$  denotes the stationary probability of the symbol  $a_i$ . Multiplexed codes can be also be designed to approach higher order entropies by conditioning the construction of the different classes to the realization of the previous symbols or to a given context [51].

## 2.7 Related areas

Surrounding joint source-channel turbo techniques, various subjects of interest have been developed in the literature. In this section, we select and

present briefly four of them: continuous-valued sources, source statistics estimation, multiple descriptions and joint source-channel turbo coding.

So far, we have considered only discrete-valued sources. Several works have studied and integrated continuous-valued sources too in turbo systems, see [22, 27, 32, 39] and references therein. The quantization step is then part of the problem to improve the end-to-end distortion. As source code, these works have considered a fixed length code, so there is no desynchronization problem such as described in Section 2.6. The optimization of the code is made through the optimization of the bit mapping at the output of the quantizer, with the goal of improving the iterative decoding convergence, either by EXIT chart optimization [39] or by an error analysis under a perfect *a priori* assumption [32]. Great performance gains are obtained while the decoding complexity is kept low.

We have also implicitly assumed, so far, that the source statistics are known by the receiver. Most studies make this assumption. In many practical situations, however, the source statistics must be estimated from the noisy channel measures. This is then a joint problem of source-channel decoding and source statistics estimation. If the source can be modeled with a few parameters, then only those parameters need to be estimated. Otherwise, the source statistics  $P(S_k)$  and  $P(S_{k+1}|S_k)$  are to be estimated, which is a bit more complex. An estimation based on the Baum-Welch algorithm and integrated into a turbo decoder has been proposed in [28, 76]. The increase in complexity is kept quite small by using only by-products of the SPA or BCJR algorithm.

On especially bad channels, when the channel layer cannot handle the level of data corruption, it is sometimes very useful to consider particular source transformations that introduce much redundancy at the source level. Multiple descriptions coding, for example, creates several correlated representations of the signal and transmits them across different channels. Basically, the setup can be optimized to achieve a sufficient quality of the reconstructed signal when a single representation is received, i.e., when a single channel worked, and to maximize the quality when all representations are received. This coding technique is easily integrated in the factor graph of the transmission chain, for example in [77] for soft source decoding. Moreover, if each description is followed by an interleaver and a channel coder, the SPA leads to



(a) Classical decoding, PSNR = 16.43 dB.

(b) Sequential decoding, PSNR = 31.91 dB.

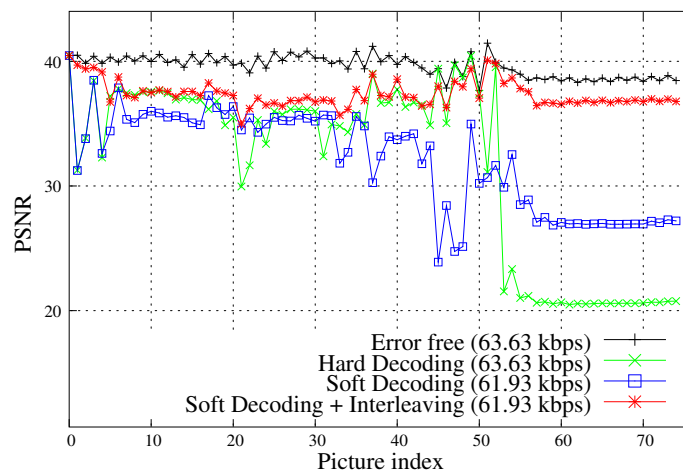
**Figure 2.7** Performance of sequential decoding of arithmetic codes with JPEG-2000 coded images (courtesy of [34]), AWGN channel with  $E_b/N_0 = 5$  dB. The PSNR of the coded image is 37.41 dB.

a turbo decoder iterating between the different descriptions, providing great performance gains over tandem receivers in [78].

At last, let us mention that compression techniques, using as source code either the syndrome former of an error correcting code or the parity bits of a high rate error correcting code, have been developed recently with low density parity check (LDPC) codes and turbo codes. See [79, 80] and references therein. These compression techniques either have built-in support for or are extensible to joint source-channel coding.

## 2.8 Performance Illustrations

A few scenarios are selected in this section to illustrate some of the performance improvements that can be provided by both soft source decoding and joint source-channel turbo decoding.



**Figure 2.8** Performance of hard and soft sequential decoding of arithmetic codes with H.264 [75] and IEEE 802.11b error traces. For soft decoding, forbidden interval of 0.15 and  $W = 32$ .

### 2.8.1 Soft source decoding of real signals

The first attempts to use soft decoding of VLC in practical image and video coding systems have been made considering Huffman VLCs and reversible VLCs (RVLCs). The authors in [81] and [82] show the benefits of MAP decoding of RVLC and VLC encoded texture information in an MPEG-4 video compressed stream. The authors in [83] also apply sequential decoding with both soft and hard channel values to the decoding of startcodes and overhead information in an MPEG-4 video compressed stream. The predominance of ACs in emerging systems has led the community to address the problem of soft decoding of these codes in actual compression systems. For example, sequential soft decoding of ACs has been introduced in the JPEG-2000 decoder in [34] and is an informative annex of JPEG-2000 part 11 dedicated to wireless applications. Fig. 2.7 shows the decoding results obtained with the Lena image encoded at 0.5 bpp and transmitted over an AWGN channel with a signal-to-noise ratio ( $E_b/N_0$ ) of 5dB. The standard JPEG-2000 decoder is compared against the sequential decoding technique with  $W = 20$  surviving paths.

Soft sequential arithmetic decoding has also been considered for resilient decoding of H.264 video streams encoded with the CABAC (context-based adaptive binary arithmetic coding) algorithm [84]. The approach is used



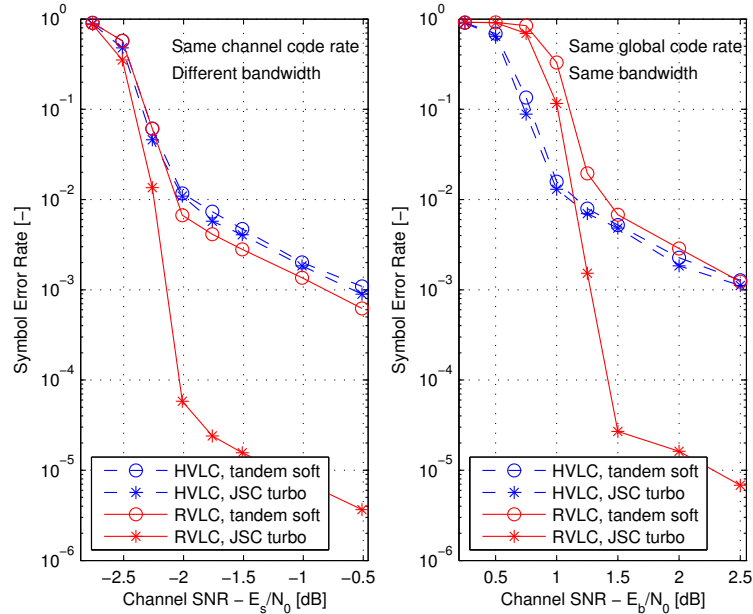
**Figure 2.9** Picture number 15 from the sequence FOREMAN. Quality obtained with WIFI traces, with hard decoding on the left and soft decoding (forbidden interval of 0.09,  $W = 32$ ) on the right.

together with the data partitioning mode of the extended profile of H.264. The data partitioning mode consists in separating elements of a slice in three classes depending on their sensitivity to bit errors. The current data partitioning mode supports three partitions: The first one, referred to as data partition A (DP-A) contains header and motion vector information which have a high sensitivity to bit errors while the two others are less sensitive and contain residual data, transform coefficients of Intra-coded blocks for DP-B, and coefficients of Inter-coded blocks for DP-C. Partitions B and C are thus decoded only if partition A has been received correctly. Fig. 2.8 shows the PSNR values obtained with the Foreman sequence with traces of bit errors induced by the IEEE 802.11b physical layer. The patterns corresponding to 2Mbps data rate and considered in [85] have been used. The average PSNR as well as the visual quality of the decoded sequence (Fig. 2.9) remain much better.

### 2.8.2 JSC turbo decoding of theoretical sources and real signals

The benefits of soft source decoding over hard decoding have just been exemplified. When a channel code is used, these benefits can be further increased by JSC turbo decoding.

JSC turbo decoding was firstly envisaged in [3]. The good results obtained by testing different VLCs with a memoryless source, a convolutional code and an AWGN channel, rapidly motivated further research in the literature. To summarize the most related contributions, binary sources are consid-



**Figure 2.10** Performance comparison of JSC turbo decoding w.r.t. tandem decoding. Note that the channel SNR (signal to noise ratio) is measured as  $E_s/N_0$  on the left and  $E_b/N_0$  on the right, where  $E_s$  is the energy per coded bit,  $E_b$  the energy per entropy bit and  $N_0$  the noise spectral density.

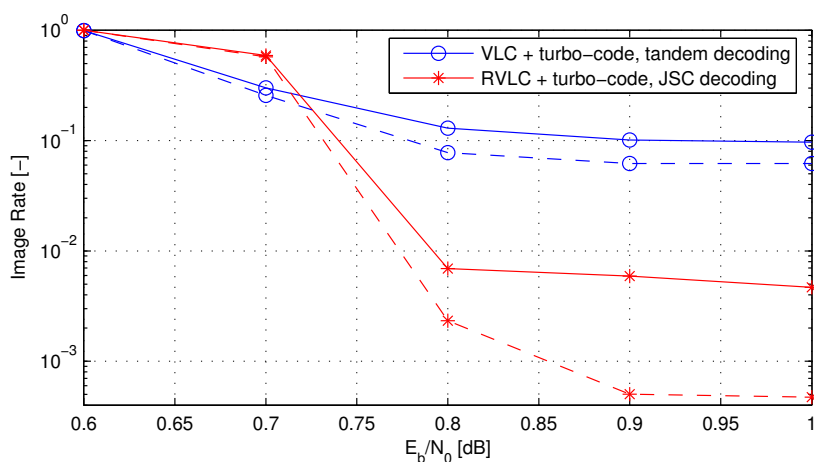
ered in [28, 37, 41]. Sources with memory are considered in [26–28, 31, 41] and source semantics in [25, 38]. As source code, FLCs are used in [27, 39], VLCs in [3, 24, 26, 33], resilient VLCs in [11, 20, 23, 29–31, 40] and ACs in [34–36]. As channel code, capacity approaching turbo codes are considered in [24, 25, 30] with no interleaver between the source code and the turbo code (unlike Fig. 2.1). Turbo codes with an interleaver between the source code and the turbo code, as well as LDPCs, are suggested in [9, 11, 40] and in Chapter 3 — the source and channel decoding modules are better separated in the turbo decoder and the decoding complexity is lower. A parallel concatenation with a convolutional code is proposed in [33]. Suboptimal decoding algorithms, for lower receiver complexity, are proposed in [24, 31, 34, 36]. An analysis and/or optimization with EXIT charts is provided in [21, 23, 31, 36, 39] and in Chapter 4. Performance analysis and prediction with distance spectra are given in [20] and in Chapter 5.

Let us consider firstly a theoretical source of independent symbols, based on the 26 letters of the English alphabet as in [30]. The symbols are encoded by a VLC. The resulting bits are framed to maximum  $N = 4000$  bits, interleaved and protected by a rate- $\frac{1}{2}$  parallel systematic turbo code before transmission across an AWGN channel.

Let us have a look at how changing only the redundancy of the source/VLC may affect the global performance. This is illustrated on the left in Fig. 2.10, for two different VLCs: a HVLC (Huffman VLC) with a code rate  $r_s = 0.99$ , i.e., 1% of redundancy; an RVLC (reversible VLC) with  $r_s = 0.95$ , or 5% of redundancy, and with a free distance (minimum Hamming distance)  $d_f = 2$ . Under tandem decoding, i.e., under iterative decoding of the turbo code and separate soft decoding of the source/VLC, the RVLC provides some improvement over the HVLC, as expected. This is the reason why RVLCs are used in standards with error resilience needs, such as AAC and MPEG-4. Under JSC turbo decoding, the resilience improvement of the RVLC becomes impressively huge, two orders of magnitude in SER. This is the consequence of an increased interleaving gain in the global code (RVLC + turbo code) that the JSC turbo decoder is able to exploit (see Section 2.6.4.1, [20] and Chapters 3 and 5). However, as the RVLC has 4% more of redundancy than the HVLC, the RVLC-based system needs more bandwidth than the HVLC-based system.

Therefore, we are also interested in the influence of the VLC when the global code rate  $r = r_s r_c = 1/2$  is fixed, i.e., when the bandwidth and the transmission energy are kept identical. To keep  $r$  constant, the channel code must thus be adjusted so that its code rate, hence its redundancy, is changed according to  $r_c = r/r_s$ . In such a comparison setup, a redundancy allocation problem exists between source and channel codings, whose optimal solution is not obvious<sup>2</sup> for a given family of source and channel codes. In our example, as illustrated on the right in Fig. 2.10, the RVLC still provides the lowest error

<sup>2</sup>There has not been any significant contribution on that problem in the literature so far and this problem is not considered in this thesis. Exhaustive search among all possible rate allocations and testing each is of course one possibility. But it is a quite heavy one, even if the testing step is reduced to EXIT chart [12] computation or distance spectrum enumeration (see Chapter 5). Still, drastic simplifications may apply in particular cases (i.e., for some family of codes), e.g., the EXIT charts of repetition codes and parity check codes have approximate analytical expressions [53], and the EXIT chart of a pseudo-randomly punctured convolutional code admits an analytical expression given the EXIT chart of the mother (unpunctured) convolutional code.



**Figure 2.11** Solid: image error rate, i.e., rate of images with a PSNR degradation above 0dB. Dashed: rate of images with a PSNR degradation above 3dB.

rates under JSC turbo decoding, but only for channel SNRs (signal to noise ratio) above 1.2dB. There is thus a trade-off whose solution depends on the application needs: In our example, the RVLC scheme is the good candidate except for applications working at critically low channel SNRs.

An interesting comment can be made at this stage. In both plots in Fig. 2.10, the tandem receiver performs quite badly on the RVLC scheme w.r.t. the JSC turbo receiver while, on the HVLC scheme, it performs very close to the JSC turbo receiver. This is related to the respective level of redundancy in the RVLC and in the HVLC, as it was already noted by Shannon in [1]: If some redundancy is left or intentionally introduced at the source code level (as in the RVLC scheme here), and if its distribution is known to the receiver, the tandem receiver could lead to a huge loss in performance w.r.t. the JSC receiver. On the contrary, if there is no redundancy left, these two receivers perform equally (as in the HVLC case).

Finally, the transmission of images across an AWGN channel is considered in Fig. 2.11 with the parallel turbo code of [2] (interleavers of length  $N = 65536$ ) — note that using another turbo code would change/improve the error floors but not the general trends and the conclusions hereafter. This simulation illustrates the potentiality of JSC techniques with a capacity approach-

ing error correcting code. As we can see, great improvements<sup>3</sup> are possible even when we are close to the assumptions of Shannon's separation theorem. Each image is processed by a DCT (discrete cosine transform) based compression, with a compression ratio similar to the JPEG standard. The DC component is coded with an FLC and decoded by an MMSE estimation (2.4). The AC components are run-length coded with HVLCs in the tandem system and with RVLCs in the JSC system, and decoded by a frame-MAP estimation (2.1). RVLCs have been chosen for the JSC system in order to benefit from the near zero desynchronization property described in Section 2.6.4.1, a property that is useless with the tandem decoder. As we can see, the JSC system provides not only a better image error rate, but also a much smaller rate of images with a PSNR distortion above 3dB — actually, most error images of the JSC system have a PSNR quality almost not affected by the error(s).

## 2.9 Conclusions

The success of joint source-channel turbo decoding, attested by many contributions in the literature, benefits greatly from the respective usual suboptimality, in practical contexts, of source and channel codes. Combining indeed the residual redundancy left by the source code with the non-perfect error correcting capability of the channel code offers generally great advantages in terms of end-to-end distortion. In this chapter, much attention has been devoted to present the subject in a unified way, using the extensible framework of factor graphs. Based on this framework, the turbo principle can be easily introduced and detailed in the joint source-channel context, with different well-known source codes and several optimal soft source decoding algorithms. Some source codes, however, raise specific difficulties, such as high decoding complexity or decoder desynchronization, for which suboptimal decoding solutions and ways to improve resilience have been discussed.

It is worth recalling that the cooperation between the source decoding module and the channel decoding module in the turbo decoder amounts to the iterative exchange of probabilities (Section 2.4.2), which requires simply

---

<sup>3</sup>In this example, the improvement provided by the JSC system comes from a better interleaving gain, i.e., the image error rate converges faster to zero as the length of the interleaver(s) increase(s), at the expense of a negligible performance loss at low SNRs.

an interface of real numbers. Such a simple interface allows a great separation/independence between the source and channel layers in the communication protocol, which is often desirable and valuable from an industrial standpoint.

To summarize, the literature has addressed many problems in this field. One of the next challenges is the adoption of such techniques in more standards, with notably a stronger cross-layer interaction between the source layer and the channel layer.

The next chapter proposes a new transmission system for applications based on variable length codes. This system generalizes a few previous systems from the literature and provides significant advantages in terms of performance, flexibility and decoding complexity.



# Variable Length Codes, Irregular Turbo Codes: an Example of Synergy

## 3

*The content of this chapter is basically the reproduction of a journal paper [9], enhanced with a few additional details, especially in the proofs. Parts of this chapter have been published also in [10, 11].*

As we have seen in Chapter 2, variable length codes (VLCs), used in data compression, are very sensitive to error propagation in the presence of noisy channels. Addressing this problem with joint source-channel turbo techniques has been proposed in the literature and looks quite promising. So far, however, contributions in this field have focused on sources/VLCs that generate bit streams containing a large amount of residual redundancy. When the amount of residual redundancy is low, i.e., when there is a good match between the source statistics and the length distribution of the VLC, the classical concatenation of the VLC with a convolutional code is not satisfying.

To solve this issue, this chapter proposes to introduce a repetition code between the VLC and the convolutional code, or equivalently, to concatenate the VLC serially with an irregular turbo code. Through EXIT chart and interleaving gain analysis, this concatenation is shown to be a beautiful example of source-channel synergy, especially with reversible VLCs. The proposed system provides indeed significant advantages in terms of flexibility, performance and decoding complexity, compared to previous systems based on a convolutional code or on a turbo code.

### 3.1 Introduction

Variable length codes (VLCs) offer a compression advantage over fixed length codes (FLCs) but, unlike FLCs, they are very sensitive to error propagation. Several approaches have been proposed in the literature to improve their robustness. Among them, joint source-channel turbo techniques look quite promising. As we have seen in Chapter 2, joint source-channel turbo decoders are elegant approximations of the optimal joint receiver. They can exploit and benefit both from the redundancy in the source bit stream, residual or intentionally introduced, and from the error correcting capabilities of the channel code. Depending on the amount of redundancy after source coding, joint receivers can decrease considerably the distortion of the estimation as compared to tandem receivers, as already stated by Shannon in [1].

The first turbo decoder for the concatenation of a VLC and a convolutional code (CC) has been proposed in [3]. That setup has illustrated the potential of joint source-channel turbo decoding and has then been improved in several directions, notably the VLC decoder [26,86], the VLC itself [14,20,23,39,48,49,87] and the inner error correcting code [20,29–31,37,40,41,88].

This chapter focuses on sources/VLCs that generate bit streams containing a small amount of residual redundancy. It is a case not covered extensively in the turbo literature: Works so far have focused on medium to high redundancy sources/VLCs — a frequent example is a quantized autoregressive (Gauss-Markov) process with correlation coefficient 0.9, where the correlation is not taken into account for compression (only the stationary probabilities are). Low redundancy sources/VLCs are not as well suited for the original concatenation [3] with a CC. The VLC decoder does not produce enough information and it is quite difficult to get a tunnel, free of any intersection between the EXIT charts, at signal to noise ratios (SNRs) close to the capacity.

To address this problem, this chapter suggests to introduce an irregular repetition code (RC) between the VLC and the interleaver. The system is then equivalent to a VLC serially concatenated with an irregular turbo code [89]. Despite its simplicity, this RC enables a better match of the EXIT charts, therefore potentially better performance. Based on distance spectra, we give theoretical rules that increase the interleaving gains of the global code. These

gains explain the particularly good performance at medium to high SNRs in the simulation results.

Among the benefits of the present work, let us emphasize three of them before starting the main matter. Firstly, since the repetition code is a simple component that can be adapted easily, the resulting global code offers some interesting flexibility to handle different levels of source/VLC redundancy. For example, when the redundancy is high, a rate-1 repetition code (i.e., no repetition) can be used, as shown by previous works in the literature where no repetition code is considered (see [26, 31], e.g.). When the redundancy is low, a rate-1/2 regular repetition is generally suitable, as shown by the analysis in this chapter. That kind of flexibility is interesting when dealing with heterogeneous sources of data. Secondly, when a rate-1/2 regular repetition is used, the VLC is equivalently concatenated with a parallel turbo code. The proposed concatenation offers two significant advantages over previous combinations [30, 88] of VLCs and turbo codes, thanks to an additional interleaver: an improvement of the error rates with error correcting VLCs and, at the same time, a reduction of the decoding complexity. Thirdly, we prove that there exists a strong interaction between reversible VLCs and the irregular turbo code when a joint source-channel decoder is used, under certain assumptions; that interaction disappears when a tandem decoder is used. This gives us one possible approach to lower the error floors that is complementary to (and can be combined with) other existing methods from the literature. This approach is supported by the fact that it should hold for other turbo-like codes and that reversible VLCs have been adopted in some video/audio standards (see AAC [74] and MPEG-4 [75]).

As previous works in this field [3, 20, 23, 26, 29–31, 37, 39, 41, 86, 88], we focus on a system whose encoder is of low complexity, based on simple finite state machines, and whose decoder is not necessarily of low complexity — for the up-link connection of a small device to a base station, for example. For applications that can afford an encoder of higher complexity, it might be interesting to envisage a design other than the proposed one. For example, one can resort to a strong separate design with arithmetic source coding [56] and AR4JA-LDPC channel coding [90]. At the expense of a significant increase in encoding complexity, such a setup maintains the aforementioned flexibility to handle different kinds of sources and should offer better performance

than the proposed system, especially for long block lengths. This possibility is however outside the scope of this chapter.

The remainder of the chapter is organized as follows. The system encoder and turbo decoder are presented in Sections 3.2–3.3. Interleaving gains and EXIT charts are given in Sections 3.4–3.5. At last, related works and simulation results are considered in Sections 3.6–3.7.

In the following, random variable are written with capital letters and realizations with small letters.  $P(z)$  is the abbreviation of the probability  $P(Z = z)$ . The sub-sequence  $(Z_m, Z_{m+1}, \dots, Z_n)$  is written  $Z_{m:n}$ , or  $Z$ ; when  $m, n$  can be omitted.  $\mathcal{A}$  is the alphabet of the source symbols.  $l(s)$  is the bit length of the symbol sequence  $s$ .  $\bar{l}$  is the average bit length of the VLC considered,  $\bar{l} = \sum_{s \in \mathcal{A}} P(s) l(s)$ .  $l_{\text{gcd}}$  is the greatest common divisor of the lengths of the VLC considered,  $l_{\text{gcd}} = \text{gcd}\{l(s) : s \in \mathcal{A}\}$ .  $\lfloor a \rfloor$  is the only integer  $i$  such that  $a - 1 < i \leq a$ .  $\mathbb{I}\{a\}$  is the indicator function, i.e.,  $\mathbb{I}\{a\} = 1$  if  $a$  is true, 0 otherwise.  $|\{\cdot\}|$  is the cardinality of the set  $\{\cdot\}$ . At last, let  $\mathbb{Z}$  be the set of integers  $\{\dots, -2, -1, 0, 1, 2, \dots\}$ , and<sup>1</sup> let  $\mathbb{N}_{\geq 0} = \{n \in \mathbb{Z} : n \geq 0\}$  and  $\mathbb{N}_{> 0} = \{n \in \mathbb{Z} : n > 0\}$ .

## 3.2 System Overview

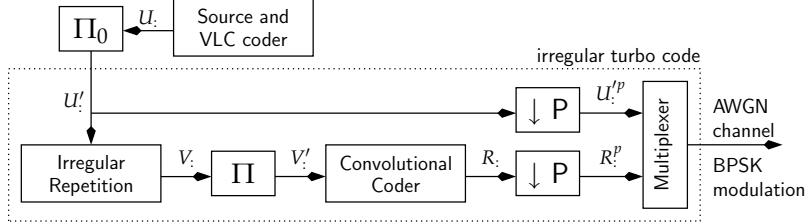
### 3.2.1 System description and notations

The system encoder is depicted in Fig. 3.1. Let us consider a source of discrete symbols and a memoryless discrete channel with additive white Gaussian noise (AWGN). Let  $N$  be the number of bits at the input of the RC. At symbol time  $k$ , the symbol  $S_k$ , belonging to alphabet  $\mathcal{A}$ , is transcoded into a variable length codeword of length  $l(S_k)$ . A frame, or packet, is composed of the first  $K$  symbols such that

$$M \triangleq \sum_{k=1}^K l(S_k) \leq N \quad \text{and} \quad M + l(S_{K+1}) > N. \quad (3.1)$$

If  $M < N$ , zeros are appended to  $U_{1:M}$  to get  $U_{1:N}$ , which are then interleaved by  $\Pi_0$  into  $U'_{1:N}$ . In the RC, a fraction  $f_i$  of  $U'_{1:N}$  is repeated  $i$  times, for  $i =$

<sup>1</sup>We introduce the notations  $\mathbb{N}_{\geq 0}$  and  $\mathbb{N}_{> 0}$  to avoid any possible confusion in the following, instead of the more common  $\mathbb{N}$ ,  $\mathbb{N}_0$ ,  $\mathbb{N}^*$  and  $\mathbb{N}^+$ , which are less explicit and used differently by authors in the literature.

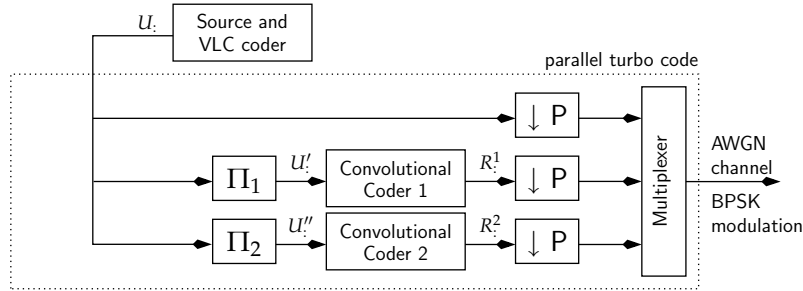


**Figure 3.1** System encoder with an irregular turbo code.

$1, 2, \dots$ . A bit repeated  $i$  times is said to be of degree  $i$  and the distribution  $\{f_i \geq 0, i = 1, 2, \dots : \sum_{i \geq 1} f_i = 1\}$  is the degree distribution of the RC. The RC rate is given by  $r_{rc} = 1/\bar{d}$ , where  $\bar{d} = \sum_{i \geq 1} f_i i$  is the average degree. We assume  $N\bar{d}$  is an integer. The RC output bits  $V_{1:N\bar{d}}$  are interleaved by  $\Pi$  into  $V'_{1:N\bar{d}}$ . Then the CC, based on a non-systematic recursive rate-1 CC encoder with zero-tail (terminated trellis) and memory  $m_{cc}$ , generates the parity bits  $R_{\cdot}$ . Finally, after a possible puncturing ( $\downarrow P$ ), the input bits  $U'_{1:N}$  and the parity bits  $R_{\cdot}$  are multiplexed and sent across the channel. At the receiver, the number of VLC bits ( $M$ ) and the number of RC input bits ( $N$ ) are known but the number of symbols ( $K$ ) is not. Besides, the receiver does not know that the bits  $U_{M+1:N}$  are zeros. The global code rate is given by

$$R_c = r_s r_c = r_s \left( r_{sp}^{-1} + (r_{rc} r_{cc} r_{pp})^{-1} \right)^{-1}, \quad (3.2)$$

where  $r_s$  is the source code rate (defined as the ratio of the source entropy over the average number of VLC bits used to encode the source),  $r_c$  is the channel code rate (systematic bits + RC + CC + puncturing),  $r_{rc}$  is the irregular RC rate,  $r_{cc} \triangleq \left(1 + \frac{2m_{cc}}{N\bar{d}}\right)^{-1}$  is the unpunctured CC rate,  $r_{sp} \triangleq l(U_{\cdot})/l(U_{\cdot}^p)$  is the systematic puncturing rate,  $r_{pp} \triangleq l(R_{\cdot})/l(R_{\cdot}^p)$  is the parity puncturing rate. Note that  $r_{cc}$  is close to 1 for large  $N$  (it equals 1 if we neglect the trellis termination or if we use tail-biting) and  $r_{cc} r_{pp}$  is the code rate of the punctured CC. The allocation of the bit rate among the different components in Fig. 3.1, i.e., the search for the best values (performance-wise) of the parameters involved in (3.2), is not envisaged in this thesis. Instead, we consider  $R_c$  and  $r_s$  are given, good values for all other parameters but  $r_{cc} r_{pp}$  are available (subject to the interleaving gain analysis in Section 3.4) and the value of  $r_{cc} r_{pp}$  is deduced from (3.2).



**Figure 3.2** Equivalent system if a rate-1/2 regular repetition code is used and if  $\Pi(V_{1:2N}) = [\Pi_1(U'_{1:N}), \Pi_2(U''_{1:N})]$  in Fig. 3.1. The interleaver  $\Pi_0$  has no impact with a regular repetition and has been removed, see Section 3.2.2.

A particular setup that proves in the following to be well suited for low redundancy sources/VLCs, say  $r_s > 0.9$ , is a setup using a rate-1/2 regular RC ( $f_2 = 1$ ,  $\bar{d} = 2$ ). In that case, the irregular turbo code is equivalent [89] to a parallel turbo code if we design the interleaver  $\Pi$  such that  $\Pi(V_{1:2N}) = [\Pi_1(U_{1:N}), \Pi_2(U_{1:N})]$ , where  $\Pi_1$  and  $\Pi_2$  are two interleavers of length  $N$ , see Fig. 3.2 — the only difference, which is negligible, has to do with the trellis termination.

### 3.2.2 Comments on the interleaver $\Pi_0$

Let us discuss the motivation for  $\Pi_0$  in Fig. 3.1. In the RC, two bits repeated a different number of times have different levels of protection. In the VLC, we may have also classes of bits with different levels of protection, e.g., a VLC whose bit-level trellis is stationary and periodic with a period longer than one (e.g., an FLC). Unless we are interested in unequal error protection, the purpose of  $\Pi_0$  is to distribute as uniformly as possible the different classes of repetition among the different classes of the VLC bits and among the different bits within a class.

In some cases, the different classes can be uniformly distributed without  $\Pi_0$  and thus  $\Pi_0$  is sometimes not necessary. For example, consider a VLC whose bit-level trellis (Balakirsky trellis [55], extended with the previous source symbol(s) if the source has memory) is stationary and periodic with period  $T_{\text{vlc}}$ , and an RC that is built periodically from a uniform repetition pat-

tern of length  $T_{rc}$ . Then we have a uniform (but deterministic) distribution for sufficiently long frames if  $T_{vlc}$  and  $T_{rc}$  are small and coprime. The coprime condition is trivially satisfied when the RC is regular, then  $T_{rc} = 1$ , or when<sup>2</sup>  $T_{vlc} = 1$ . For example,  $\Pi_0$  is not needed in Fig. 3.2 because the RC is regular.

### 3.3 Joint Iterative/Turbo Decoder

At the receiver, a good decision criterion with VLCs is the frame error rate (FER) minimization. A single bit error can indeed desynchronize the VLC decoder and make the whole frame useless, it is therefore good (though not optimal) to minimize the number of frames with errors. The solution to this minimization is the frame-MAP (maximum a posteriori) detection rule

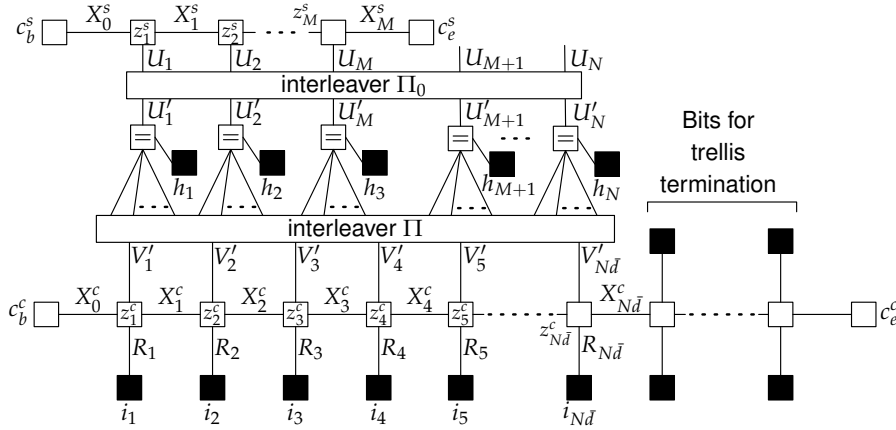
$$\hat{u}_{1:M} = \underset{u_{1:M}}{\operatorname{argmax}} \{P(u_{1:M}|\hat{\mathcal{Y}})\} = \underset{u_{1:M}}{\operatorname{argmax}} \{P(u_{1:M}, \hat{\mathcal{Y}})\}, \quad (3.3)$$

where  $\hat{\mathcal{Y}}$  is the set containing all available channel measures.

#### 3.3.1 The decoding algorithm

The decoder is now concisely described using the factor graph framework presented in Chapter 2. The factor graph of the transmission chain is given in Fig. 3.3. The source and the VLC coder are modeled by the Markov chain  $X^s$  at the top, see the bit clock model in Fig. 2.3(d) in Section 2.5.1. Its state space is the set of nodes of the Balakirsky trellis [55], i.e., the internal nodes of the VLC binary tree (plus the previous symbol(s) if the source has memory). The transitions on  $X^s$  are represented by the functions/nodes  $z_n^s$ , follow from the state space and from the source statistics, and generate the bits  $U_{1:M}$ . The few appended bits  $U_{M+1:N}$  have no associated constraint since their values are assumed to be unknown at the decoder. The repetition of the interleaved bits  $U'_{1:N}$  is represented by the equality nodes and the edges beneath. The convolutional encoder is modeled by the Markov chain  $X^c$  whose states are defined by the coder shift registers. The transitions on  $X^c$ , represented by the functions/nodes  $z_n^c$ , are triggered by the bits  $V'_i$  and generate the parity bits

<sup>2</sup>For example, we have  $T_{vlc} = 1$  with a memoryless source when two lengths of the VLC,  $l(a)$ ,  $l(b)$ ,  $a \neq b$ ,  $a, b \in \mathcal{A}$ , are coprime. A convincing example is to look at the Balakirsky trellis [55] for the VLC  $\{00, 01, 10, 1100, 1101, 1110, 1111\}$  ( $T_{vlc} = 2$ ) and for  $\{00, 01, 10, 110, 1110, 1111\}$  ( $T_{vlc} = 1$ ).



**Figure 3.3** Forney-style factor graph of the complete transmission scheme, with the notations in Fig. 3.1.

$R_n$ . The channel measures are represented by the black nodes: the functions  $h_n(U'_n) = P(Y_{U'_n}|U'_n)$  and  $i_n(R_n) = P(Y_{R_n}|R_n)$ , where  $Y_{U'_n}$  and  $Y_{R_n}$  are the channel measures on  $U'_n$  and  $R_n$ , respectively. For the punctured bits, these black nodes must be removed. At last,  $c_b$  and  $c_e$  are constraints representing the trellis initialization and termination. A few bits are added and transmitted across the channel to terminate the CC trellis.

Now that we have the factor graph, the maximization in (3.3) can be approached efficiently by the succession of:

- the Sum-Product (belief-propagation) algorithm (see Section 2.3.3 and [6]) on the factor graph, in order to get the approximated marginal probabilities of any variable involved in the graph. One possible message-passing schedule is to process, at each iteration, the Markov chain  $X_n^c$  forward/backward, the equality nodes (repetition code), the Markov chain  $X_n^s$  forward/backward, and finally the equality nodes again. Note that for the Markov chains  $X_n^s$  (VLC) and  $X_n^c$  (CC), this is equivalent to the BCJR algorithm (Section 2.3.3.2).
- the Viterbi algorithm on the Markov chain  $X_n^s$  of the source/VLC in order to get a solution to (3.3), based on the marginal probabilities computed by the Sum-Product algorithm.

See Chapter 2, [6] and references therein for Forney-style factor graphs and the Sum-Product algorithm. Note that the same decoding algorithm holds when  $\Pi_0$  is not used, because there is no cycle in the graph between the nodes  $X_s^s$  and the equality nodes.

### 3.3.2 The decoding complexity

By considering the state spaces of the Markov chains  $X_s^c$  and  $X_s^s$ , or the sizes of the trellises, as a measure of complexity, the computational decoding complexity per bit of entropy is

$$r_s^{-1} M^{-1} N_{\text{iter}} \left( \underbrace{N \bar{d} |X^c|}_{\text{trellis size of CC}} + \underbrace{M \overline{|X^s|}}_{\text{trellis size of source/VLC}} \right) \quad (3.4)$$

$$\approx r_s^{-1} N_{\text{iter}} (\bar{d} |X^c| + \overline{|X^s|}), \quad (3.5)$$

where  $N_{\text{iter}}$  is the number of decoding iterations,  $|X^c| = 2^{m_{cc}}$  the number of states of the CC and  $\overline{|X^s|}$  the average number of states of the VLC decoder (averaged over all trellis sections). The tail effects in the trellises are neglected. To give a rough idea, note that the quantities  $|X^c|$  and  $\overline{|X^s|}$  have approximately the same order of magnitude in the simulations in Section 3.7, with  $\bar{d} |X^c| = 32$  and  $3 \leq \overline{|X^s|} \leq 41$ .

In (3.5), the quantity  $\overline{|X^s|}$  depends on the source statistics and on the VLC. It is straightforward to develop a lower bound for a memoryless source,

$$\overline{|X^s|} \geq \frac{|\mathcal{A}| - 1}{l_{\text{gcd}}}, \quad (3.6)$$

where  $|\mathcal{A}|$  is the size of the source alphabet. Note that this inequality holds with equality if and only if Kraft's inequality holds with equality, i.e., if the VLC is complete, in particular if Huffman VLCs are used.

Note that the decoding complexity is generally lower with FLCs. This is mainly because  $l_{\text{gcd}}$  is generally larger with FLCs, thus  $\overline{|X^s|}$  is smaller, which tends to decrease the decoding complexity by (3.5). This tendency is however a bit softened by  $r_s^{-1}$  which is typically larger with FLCs.

Though not investigated in this thesis, it is very often possible to lower the decoding complexity by optimizing the message-passing schedule of the

Sum-Product algorithm. For example, compared to the schedule given in Section 3.3.1, one possibility could be not to process the Markov chain  $X^s$  (source/VLC) at each iteration, notably at iterations where the RC alone<sup>3</sup> provides already a significant improvement — note that the improvement can be predicted in terms of mutual information by EXIT chart analysis.

Another example is to process the Markov chain  $X^c$  (CC) differently. The schedule in Section 3.3.1 proposes to apply the forward/backward recursion on the whole chain  $X^c$  and only then to process the equality nodes (RC decoder) in Fig. 3.3. Instead, let us divide the chain  $X^c$  into  $\bar{d}$  segments and apply the forward/backward recursion on the first segment, then process the equality nodes, then the second segment, then the equality nodes, and so forth. When the  $\bar{d}$  segments have been processed, let us process at last the Markov chain  $X^s$  (source/VLC decoder) and again the equality nodes. This constitutes one iteration of the turbo decoder. By processing the equality nodes after each segment (which has a negligible complexity), this schedule has the advantage of making the results produced by the current segment immediately available to all subsequent segments, instead of waiting until the end of the iteration to make them available. In average thus, the segments work with fresher/better information. Note that this schedule is particularly efficient when the RC is regular. When  $\bar{d} = 2$ , it corresponds roughly to the schedule of a parallel turbo code.

### 3.4 Interleaving Gain Analysis

Interleaving gains quantify the asymptotic decrease of the error rates when the interleaver length  $N$  increases. Interleaving gains are very important in practice as they govern performance for moderate to large interleaver length. In this section, we will calculate the interleaving gains that can be expected with the systems given in Fig. 3.1–3.2, under the following assumptions.

**Assumption 3.1.** *optimal frame-ML (maximum likelihood) decoding, the number of symbols ( $K$ ) unknown at the decoder, random<sup>4</sup> uniform interleaving, recursive non-*

<sup>3</sup>Note that, when the source/VLC decoder is not used at a given iteration, the output from the previous iteration is used, in agreement with the Sum-Product algorithm.

<sup>4</sup>Or more properly “pseudo-random” as the interleavers are chosen randomly for each new frame *but* assumed known at the receiver. Note that this assumption is especially useful for the

*catastrophic CC encoder and bounded VLC spectrum*<sup>5</sup>. Besides, we assume that the interleaving gain exponent conjecture<sup>6</sup> [91] holds for the considered systems.  $\square$

As both the turbo receiver described above and the frame-ML decoder tend to the frame-MAP decoder at high SNRs (signal to noise ratios) on the channel, we can use these interleaving gains as design rules at high SNRs. In the following, we are interested in the interleaving gains on the symbol, symbol with Levenshtein distance, and frame error rates (SER, SER<sub>L</sub>, FER, resp.). Note that the FER is a useful performance measure for lossless transmission.

### 3.4.1 Background on distance spectrum and interleaving gains

Let us introduce briefly (and in a simplified way) a few selected results from Chapter 5. Given two bit sequences  $u, u'$  of the same length, let  $d_H(u, u')$  be the Hamming distance between these sequences. Given two symbol sequences  $s, s'$ , let  $d_S(s, s')$  and  $d_{S_L}(s, s')$  be the symbol and the Levenshtein symbol distances between  $s$  and  $s'$ . Given the global code and the notations in Fig. 3.1, let  $\mathcal{S}$  be the set of all possible source symbol sequences subject to (3.1). Given  $s \in \mathcal{S}$ , let  $v(s)$  be the codeword produced by the VLC and let  $g(s)$  be the output codeword produced by the global code (VLC + RC + CC).

Then, the average distance spectrum of the global code in Fig. 3.1 can be defined as follows. Given the conditional spectrum  $A_{w,s,h|s}$ , which denotes the number of sequences  $s' \in \mathcal{S}$  subject to  $d_H(v(s), v(s')) = w$ ,  $d_{S_L}(s, s') = s$  and  $d_H(g(s), g(s')) = h$ , the average distance spectrum is given by  $A_{w,s,h} = \sum_{s \in \mathcal{S}} P(s) A_{w,s,h|s}$ .

---

validity of the theoretical results; in practice, the same pseudo-random interleaver may be used for all frames and the theoretical results still apply for most pseudo-random interleavers.

<sup>5</sup>At this stage, let us consider simply that VLCs with bounded spectrum have similar properties as non-catastrophic convolutional encoders. This concept will be clearly defined in Section 5.5.2 and analyzed in Section 5.7.

<sup>6</sup>This conjecture has been proved so far for parallel and serial turbo codes in [92]. In Chapter 5, it will be proved for the serial concatenation of a VLC with a CC. Note the proof proposed in Chapter 5 is extensible to other types of concatenation. So, including the IGE conjecture in Assumption 3.1 is reasonable and maybe not restrictive.

The union bound can then be used with this spectrum to get upper bounds on the performance of the frame-ML decoder, e.g.,

$$\text{SER}_L \leq \sum_{w,s,h} \frac{s}{N/\bar{l}} A_{w,s,h} P_h, \quad (3.7)$$

$$\text{FER} \leq \sum_{w,s,h} A_{w,s,h} P_h, \quad (3.8)$$

where  $P_h$  is the pairwise error probability and  $\bar{l}$  is the average length of the VLC. Several remarks can be made. 1) The bound on the  $\text{SER}_L$  is approximate<sup>7</sup>. 2) So far and to the best of our knowledge, no useful upper bound on the SER has been developed in the literature, though the SER is of course bounded by the FER. An upper bound on the SER will be discussed in Section 5.8.2. 3) For the AWGN channel with BPSK modulation,  $P_h$  is given by  $\frac{1}{2} \operatorname{erfc} \left( \sqrt{hR_c \frac{E_b}{N_0}} \right)$ , where  $R_c$  is the global code rate in (3.2),  $N_0/2$  the double-sided noise spectral density and  $E_b$  the energy per bit of entropy.

In this chapter, we are not interested in the computation of  $A_{w,s,h}$  but rather in its asymptotic behavior when  $N$  increases. As originally developed in [16] for a serial turbo code, this behavior can be predicted by upper bounding  $A_{w,s,h}$  with a series in  $N$  of the form (see [16, eq. (17)])

$$A_{w,s,h} \leq \sum_{\alpha=-\infty}^{\infty} a_{\alpha}(w,s,h) N^{\alpha} \quad (3.9)$$

where the coefficients  $a_{\alpha}(w,s,h)$  are independent of  $N$ , and by finding the maximum exponent of  $N$ ,  $\alpha_M = \max\{\alpha : \exists w,s,h, a_{\alpha}(w,s,h) > 0\}$ . If  $\alpha_M < 0$ , the interleaving gain exponent conjecture [91] states that the upper bound (3.8) on the FER decreases as  $N^{\alpha_M}$  for increasing values of  $N$  (provided that the channel noise is below a certain threshold); we say that there is an *interleaving gain* on the FER. Likewise, the  $\text{SER}_L$  decreases as  $N^{\alpha_M-1}$  and there is an interleaving gain on the  $\text{SER}_L$  if  $\alpha_M < 1$ , since  $A_{w,s,h}$  is further divided by  $N$  in (3.7). To summarize, the interleaving gains are given by

$$\alpha_{\text{ser}_L} \leq \alpha_M - 1, \quad \alpha_{\text{fer}} \leq \alpha_M. \quad (3.10)$$

For the SER, without any additional information, we have only the general inequalities

$$\alpha_{\text{ser}_L} \leq \alpha_{\text{ser}} \leq \alpha_{\text{fer}}. \quad (3.11)$$

<sup>7</sup>Given (3.1),  $K$  is a random variable that is approximated in (3.7) by  $N/\bar{l}$ , which is asymptotically valid for long  $N$ .

In practice, the impact of these interleaving gains with a turbo decoder is observed at SNRs beyond<sup>8</sup> the so-called waterfall region.

Finally, a useful relation to what follows is the interleaving gain of a serial concatenation between an outer code of free distance  $d_f^o$  and an inner recursive CC encoder [16]:

$$\alpha_M = 1 - \left\lfloor (d_f^o + 1)/2 \right\rfloor \mathbb{I}\{d_f^o \geq 2\}. \quad (3.12)$$

Let  $d_f^{\text{vlc}}$  denote the free distance of the VLC and  $d_f^{\text{rc}}$  the free distance of the RC,  $d_f^{\text{rc}} = \min\{i : f_i > 0\}$ .

### 3.4.2 Interleaving gains with irregular RCs

In the global code in Fig. 3.1, each VLC bit is repeated by the RC at least  $d_f^{\text{rc}}$  times. Therefore, if we consider the VLC and the RC jointly, the pair VLC/RC has a free distance given by  $d_f^{\text{vlc}} d_f^{\text{rc}}$ . Besides, if we neglect the systematic branch, the global code is a serially concatenated code, with the VLC/RC as outer code and the CC as inner code. We have therefore straightforwardly the following corollary with  $d_f^o = d_f^{\text{vlc}} d_f^{\text{rc}}$ , as a consequence of (3.10) and (3.12).

**Corollary 3.2** (Interleaving gains with irregular RCs). *Under Assumption 3.1, we have the following interleaving gains for the global code in Fig. 3.1:*

$$\alpha_{\text{serL}} \leq \begin{cases} 0, & \text{if } d_f^{\text{vlc}} = d_f^{\text{rc}} = 1, \\ -\left\lfloor \frac{d_f^{\text{vlc}} d_f^{\text{rc}} + 1}{2} \right\rfloor, & \text{otherwise,} \end{cases} \quad (3.13)$$

$$\alpha_{\text{ser}} \leq \alpha_{\text{fer}} \leq \begin{cases} 1, & \text{if } d_f^{\text{vlc}} = d_f^{\text{rc}} = 1, \\ 1 - \left\lfloor \frac{d_f^{\text{vlc}} d_f^{\text{rc}} + 1}{2} \right\rfloor, & \text{otherwise.} \end{cases} \quad (3.14)$$

See proof in Appendix 3.B.1.  $\square$

Let us discuss these gains for  $d_f^{\text{vlc}} \leq 2$ . When  $d_f^{\text{vlc}} = 1$ , there is no guaranteed interleaving gain if the bits are not repeated ( $d_f^{\text{rc}} = 1$ ). This corresponds

<sup>8</sup>At asymptotically high SNRs better interleaving gains can be observed. This corresponds to the behavior of  $A_{w,s,h=d_f}$  at the free distance  $d_f$  of the global code, i.e.,  $\alpha'_M = \max\{\alpha : \exists w, s, a_n(w, s, h = d_f) > 0\}$ . These interleaving gains are not discussed in this thesis because they require very high SNRs to have an impact, e.g., 10 to 20dB beyond the waterfall region.

almost to a CC alone; the  $\text{SER}_L$  is approximately independent of the frame length  $N$  and the FER increases linearly with  $N$  as long as it is smaller than 1. When  $d_f^{\text{vlc}} = 1$ , there is an interleaving gain on the  $\text{SER}_L$  when the bits are repeated at least 2 times, and on the FER when they are repeated at least 3 times. When  $d_f^{\text{vlc}} = 2$ , a repetition of 1 is sufficient for the  $\text{SER}_L$ , and a repetition of 2 for the FER. In other words, depending on the application, an RC with  $d_f^{\text{rc}} = 2$  might be sufficient since it offers already interesting gains with small  $d_f^{\text{vlc}}$ .

### 3.4.3 Interleaving gains with regular RCs

When a regular RC is used, the interleaving gains of Corollary 3.2 can be improved by carefully designing the interleaver.

**Theorem 3.3** (Interleaving gains with regular RCs). *Under Assumption 3.1, if we use a regular RC with the interleaver*

$$\Pi(V_{1:Nd_f^{\text{rc}}}) = [\Pi_1(U_{1:N}), \Pi_2(U_{1:N}), \dots, \Pi_{d_f^{\text{rc}}}(U_{1:N})], \quad (3.17)$$

where  $\Pi_i$  are random<sup>9</sup> interleavers of length  $N$  — we get a multiple parallel turbo code —, we have the following interleaving gains for the global code in Fig. 3.1:

$$\alpha_{\text{ser}_L} \leq \begin{cases} 1 - d_f^{\text{rc}}, & \text{if } d_f^{\text{vlc}} = 1, \\ -d_f^{\text{rc}} \lfloor (d_f^{\text{vlc}} + 1)/2 \rfloor, & \text{otherwise,} \end{cases} \quad (3.18)$$

$$\alpha_{\text{ser}} \leq \alpha_{\text{fer}} \leq \begin{cases} 2 - d_f^{\text{rc}}, & \text{if } d_f^{\text{vlc}} = 1, \\ 1 - d_f^{\text{rc}} \lfloor (d_f^{\text{vlc}} + 1)/2 \rfloor, & \text{otherwise,} \end{cases} \quad (3.20)$$

See proof in Appendix 3.B.2.  $\square$

### 3.4.4 Interleaving gains with RVLCs

Compared to other VLCs with a free distance of 1,  $d_f^{\text{vlc}} = 1$ , reversible VLCs (RVLCs) can lead to better interleaving gains on the SER in turbo systems. This is mainly the consequence of the following theorem.

**Theorem 3.4** (Resilience of RVLCs). *Since RVLCs are both prefix- and suffix-free, a single bit error in the frame leads only to a single symbol error, with both the Levenshtein and the non-Levenshtein symbol distances. See proof in Appendix 3.B.3.  $\square$*

<sup>9</sup>See footnotes associated with Assumption 3.1.

In other words, two bit errors at least are necessary to desynchronize the VLC decoder (when an RVLC is used) and to generate a burst of symbol errors. Interestingly and loosely speaking, with many turbo-like codes, the probability of two bit error events is much smaller than the probability of single bit error events. When this is the case, RVLCs benefit from an improved interleaving gain on the probability of desynchronization and thus on the probability of symbol error burst, hence a better SER. This is formalized in the next theorem.

**Theorem 3.5** (Interleaving gains with RVLCs). *With RVLCs,  $d_f^{\text{vlc}} = 1$  and  $d_f^{\text{rc}} \geq 2$ , we have under the assumptions of Corollary 3.2*

$$\alpha_{\text{ser}} \leq -\lfloor (d_f^{\text{rc}} + 1)/2 \rfloor, \quad (3.22)$$

and under the assumptions of Theorem 3.3

$$\alpha_{\text{ser}} \leq 1 - d_f^{\text{rc}}. \quad (3.23)$$

With RVLCs and  $d_f^{\text{vlc}} \geq 2$ ,  $\alpha_{\text{ser}}$  is given by Corollary 3.2 and Theorem 3.3. See proof in Appendix 3.B.4.  $\square$

This theorem deals with the global codes considered in Fig. 3.1–3.2. But it can be extended to other turbo-like codes as well, e.g., a serial turbo code in Theorem 3.7. Indeed, the proof in Appendix 3.B.4 relies on the property that the bit error rate is due mainly to low weight error events. So it is possible to extend Theorem 3.5 to other turbo-like codes satisfying this property.

### 3.4.5 Interleaving gains with FLCs

At last, with FLCs (which are particular RVLCs), it is well known that the decoder resynchronizes automatically after each symbol and there is thus no desynchronization of the symbol clock. As a corollary,

$$\alpha_{\text{ser}} = \alpha_{\text{ser}_L}, \quad (3.24)$$

i.e., the interleaving gains on the SER are equal to those on the  $\text{SER}_L$  given in (3.13), (3.14), (3.18) and (3.19).

**Corollary 3.6** (Interleaving gains with FLCs). *With an FLC of free distance  $d_f^{\text{vlc}}$ , we have under the assumptions of Corollary 3.2*

$$\alpha_{\text{ser}} \leq \begin{cases} 0, & \text{if } d_f^{\text{vlc}} = d_f^{\text{rc}} = 1, \\ -\lfloor \frac{d_f^{\text{vlc}} d_f^{\text{rc}} + 1}{2} \rfloor, & \text{otherwise,} \end{cases} \quad (3.25)$$

and under the assumptions of Theorem 3.3

$$\alpha_{\text{ser}} \leq \begin{cases} 1 - d_f^{\text{rc}}, & \text{if } d_f^{\text{vlc}} = 1, \\ -d_f^{\text{rc}} \lfloor (d_f^{\text{vlc}} + 1)/2 \rfloor, & \text{otherwise.} \end{cases} \quad (3.27)$$

For interleaving gains on the  $\text{SER}_{\text{L}}$  and on the FER, see Corollary 3.2 and Theorem 3.3. ■

The comparison with VLCs is interesting. At first sight, FLCs provide indeed better interleaving gains on the SER than VLCs. But this comparison is simplistic. On the one hand, when  $d_f^{\text{vlc}} = 1$ , FLCs and RVLCs provide the same interleaving gain on the SER if  $d_f^{\text{rc}} \geq 2$ . On the other hand, VLCs can be designed with a larger free distance  $d_f^{\text{vlc}}$  than FLCs while keeping in most cases a compression advantage over FLCs (at the expense of some design cost). So in practice VLCs are usually capable of better interleaving gains on the  $\text{SER}_{\text{L}}$ , on the SER and on the FER. This will be observed in the simulation results in Section 3.7.

To complete this comparison, recall that the decoding complexity is generally lower with FLCs (Section 3.3.2), which might be a crucial point in some applications.

### 3.4.6 Interleaving gains with a serial turbo code

A multiple parallel turbo code was envisaged in Th. 3.3. Let us now briefly address the case of a serial turbo code [16], by replacing the RC in Fig. 3.1 with a CC (with a non-catastrophic encoder). We then have two CCs; let  $\text{CC}^i$  be the inner CC and  $\text{CC}^m$  be the middle CC (which replaces the RC) of free distance  $d_f^{\text{cc},m}$ .

**Theorem 3.7** (Interleaving gains with a serial turbo code). *Under Assumption 3.1, if  $\text{CC}^i$  is recursive, then we have for both recursive and non-recursive  $\text{CC}^m$*

with  $d_f^{\text{cc,m}} \geq 2$ ,

$$\alpha_{\text{serL}} \leq 1 - d_f^{\text{vlc}} - \lfloor (d_f^{\text{cc,m}} + 1)/2 \rfloor, \quad (3.29)$$

$$\alpha_{\text{ser}} \leq \alpha_{\text{fer}} \leq 2 - d_f^{\text{vlc}} - \lfloor (d_f^{\text{cc,m}} + 1)/2 \rfloor. \quad (3.30)$$

If, besides, the VLC is an RVLC with  $d_f^{\text{vlc}} = 1$ , then

$$\alpha_{\text{ser}} \leq -\lfloor (d_f^{\text{cc,m}} + 1)/2 \rfloor, \quad (3.31)$$

See proof in Appendix 3.B.5.  $\square$

Note that these interleaving gains are equal to or less attractive than those associated with an RC for  $d_f^{\text{rc}} = d_f^{\text{cc,m}}$  — make the comparison with (3.14) and (3.16) —, which could have been predicted since the RC is a particular non-recursive  $\text{CC}^{\text{m}}$ . In practice, however, at equal code rate, the CC can be designed with a larger free distance than the RC, i.e.,  $d_f^{\text{cc,m}} \geq d_f^{\text{rc}}$  — note we have the upper bound  $d_f^{\text{rc}} \leq r_{\text{rc}}^{-1}$  for the RC. The CC can therefore lead to better interleaving gains in practice.

But using a CC instead of the RC in Fig. 3.1 brings up also some disadvantages — if we exclude the particular case of CCs that are actually RCs. For example, the equality nodes must be replaced by a Markov chain in the factor graph in Fig. 3.3 and, consequently, the turbo decoder requires an additional (BCJR) module to decode  $\text{CC}^{\text{m}}$ . Also, this  $\text{CC}^{\text{m}}$  does not provide the same level of flexibility as the RC to handle different levels of source/VLC redundancy. In particular, given a source/VLC, the optimization of the global code through EXIT charts is more difficult: It requires three-dimensional EXIT charts [93], which are heavier to compute, and there is no analytical expression for the EXIT chart of the CC (while there is for the RC).

In the following, only the RC is considered.

## 3.5 Further insight with EXIT Charts

### 3.5.1 Preliminaries

The extrinsic information transfer (EXIT) charts have been originally developed in [12] as a tool to assess the convergence of turbo decoding, i.e., to

assess the convergence of the iterative exchange of information between two Soft-In/Soft-Out (SISO) decoders. An introduction to these charts was given in Section 2.4.3.

For the system given in Fig. 3.1, the two SISO decoders of interest can be identified with the factor graph given in Fig. 3.3. The interleaver  $\Pi$  divides indeed the graph into two acyclic subgraphs. By applying the Sum-Product algorithm on each subgraph, we get two SISO decoders: a CC decoder below and a joint VLC-RC decoder above.

We consider the exchange of information between these two decoders through the interleaver  $\Pi$ . As in Section 2.4.3, the output of the decoders is measured by the mutual information as

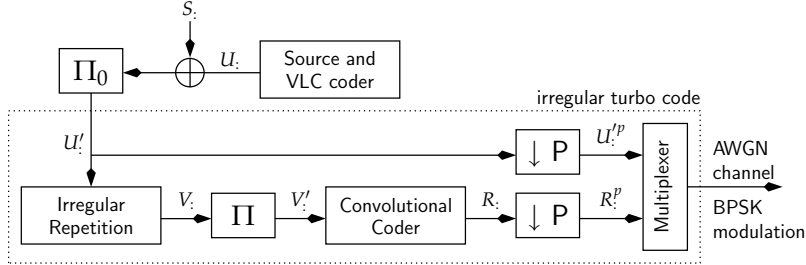
$$\begin{aligned} I_{\text{cc}}^e &= I\{V'_j; L_{E,j}^{\text{cc}}\} \leq H(V'_j), \\ I_{\text{vlc-rc}}^e &= I\{V'_j; L_{E,j}^{\text{vlc-rc}}\} \leq H(V'_j), \end{aligned} \quad (3.32)$$

where  $L_{E,j}^{\text{cc}}$  and  $L_{E,j}^{\text{vlc-rc}}$  are the so-called extrinsic log-likelihood ratios (LLRs) on the bit  $V'_j$  at the output of the CC decoder and at the output of the VLC-RC decoder, respectively,

$$L_{E,j}^{\text{cc}} = \log \left( \frac{\mu_{z_j^c \rightarrow V'_j}(0)}{\mu_{z_j^c \rightarrow V'_j}(1)} \right), \quad (3.33)$$

$$L_{E,j}^{\text{vlc-rc}} = \log \left( \frac{\mu_{V'_j \rightarrow z_j^c}(0)}{\mu_{V'_j \rightarrow z_j^c}(1)} \right), \quad (3.34)$$

where  $I\{A;B\}$  is the mutual information between  $A$  and  $B$ ,  $H(V'_j) = I\{V'_j; V'_j\} = H_b(P(V'_j = 1))$  is the entropy of  $V'_j$ ,  $H_b(\cdot)$  is the binary entropy function,  $\mu_{z_j^c \rightarrow V'_j}(v)$  is the message computed by the Sum-Product algorithm (see Chapter 2 and [6]) out of the node  $z_j^c$  along the edge  $V'_j$  for the realization  $V'_j = v$ ,  $\mu_{V'_j \rightarrow z_j^c}(v)$  is the message that arrives at  $z_j^c$  along  $V'_j$  for  $V'_j = v$ . In the turbo terminology,  $I_{\text{cc}}^e$  (resp.  $I_{\text{vlc-rc}}^e$ ) is the extrinsic information produced by the CC decoder (resp. VLC-RC decoder) and used as a priori information by the VLC-RC decoder (resp. CC decoder). At last, note that since  $V'_j$  is the repetition of some bit  $U_i$ , we have  $P(V'_j = 1) = P(U_i = 1)$ , which is the unconditional probability of  $U_i$  in the VLC stream — it can be extracted from the source statistics and the VLC binary tree if we neglect the tail effects.



**Figure 3.4** Equivalent system as Fig. 3.1, with a pseudo-random bit flipping represented by the modulo-2 addition with  $S_i$ . Note that the value of  $S_i$  is assumed known by both the encoder and the decoder.

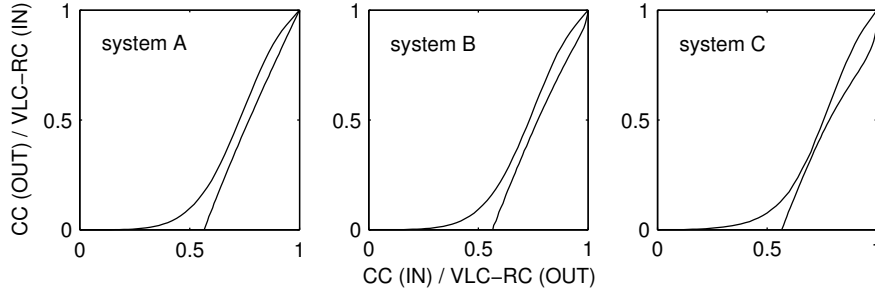
Though the computation of the EXIT charts this way is possible, and is equivalent to [12] when the bits are uniform ( $P(U_i = 1) = 1/2$ ), it has some drawbacks when they are not uniform, see Chapter 4 for details. For example, it follows from (3.32) that the EXIT chart of the CC depends on  $H(V'_j)$ , i.e., on  $P(U_i = 1)$ , and therefore on the source/VLC, which is undesirable. Besides, the function  $J(\cdot)$  in [12, eq. (15)] and the generation of the a priori LLRs must be adapted to take the value of  $P(U_i = 1)$  into account. At last and to the best of our knowledge, there is no trivial way to handle the case where  $P(U_i = 1)$  depends<sup>10</sup> on  $i$ .

To circumvent these issues, we can capitalize on the linearity of the irregular turbo code and on the symmetry of the channel. Thanks to this linearity/symmetry, the system in Fig. 3.1 is totally equivalent performance-wise and convergence-wise to the new system given in Fig. 3.4, where a pseudo-random bit flipping has been introduced between the VLC and the interleaver  $\Pi_0$ ,

$$U'_i = \Pi_0(U_i \oplus S_i), \quad (3.35)$$

where  $S_i$  is a sequence of random independent equiprobable bits, which is known by both the encoder and the decoder. Thanks to this pseudo-random flipping, the bits  $U'_i$ ,  $V_i$  and  $V'_i$  are uniform, i.e., they take their values (0 or 1) equally likely, thus  $P(V'_j = 1) = 1/2$  and  $H(V'_j) = 1$ . We can therefore compute the EXIT charts of the flipped system given in Fig. 3.4 with the clas-

<sup>10</sup>When  $T_{\text{vlc}} > 1$  (see Section 3.2.2),  $P(U_i = 1)$  can take up to  $T_{\text{vlc}}$  different values, e.g., an FLC of length  $l$  can lead to  $l$  different values.



**Figure 3.5** EXIT charts for three different VLCs, rate-1/2 regular RC,  $E_b/N_0 = 1.0\text{dB}$ . From left to right: A) HVLC, B) RVLC  $d_f = 1$  and C) RVLC  $d_f = 2$ .

sical method of [12], which holds for uniform bits, without any of the issues mentioned above. Since the flipped system in Fig. 3.4 and the original system in Fig. 3.1 are equivalent both performance-wise and convergence-wise, these EXIT charts (of the flipped system) can be used and will be used in the following to predict the convergence of the original system. We will come back to this pseudo-random bit flipping technique in Chapter 4.

### 3.5.2 Comparison/analysis of three system examples

EXIT chart examples are given in Fig. 3.5, for three systems considered in the simulation results, labeled system A, system B and system C.

These systems are based on the global code depicted in Fig. 3.1, with the following parameters. The source considered is the theoretical source of independent symbols known as the “English alphabet” (see [49], for example). As source code, three different VLCs are envisaged: system A uses a Huffman VLC (HVLC), system B uses an RVLC with  $d_f^{\text{vlc}} = 1$  and system C uses an RVLC with  $d_f^{\text{vlc}} = 2$ . The source code rates are, respectively,  $r_s = 0.992$ ,  $r_s = 0.984$  and  $r_s = 0.948$ . The RC is a regular rate-1/2 RC, thus with free distance  $d_f^{\text{rc}} = 2$ . The systematic bits are not punctured ( $r_{sp} = 1$ ). The coded bits of the CC are punctured so as to obtain the global code rate  $R_c = 1/2$  in (3.2) for all systems (same bandwidth usage), i.e.,  $r_{pp} = r_{cc}^{-1} (r_s - 1/2)^{-1}$ . Further details on the parameters are given in Section 3.7.

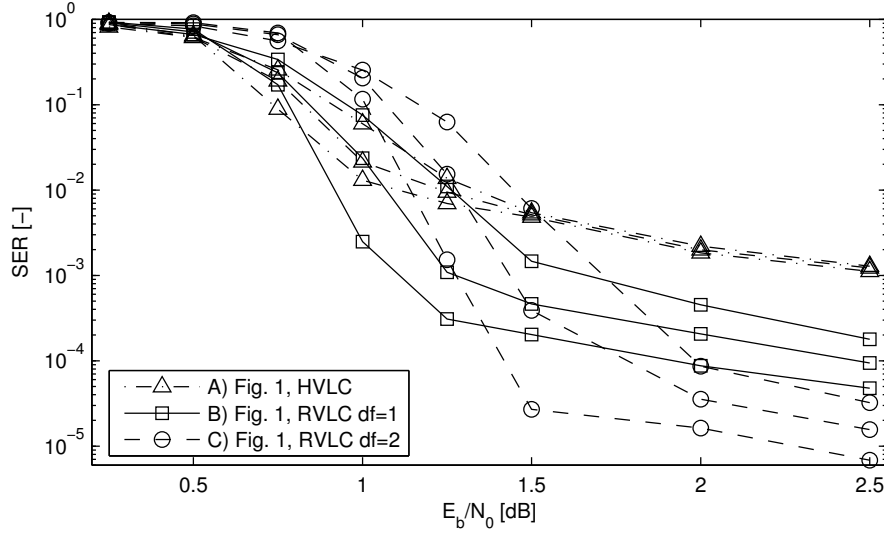
Let us compare these three systems in the case of arbitrarily long interleavers and in the case of short interleavers.

**Table 3.1** Convergence thresholds and interleaving gains of different systems.

system	A	B	C	D	E
source code	HVLC	RVLC $d_f = 1$	RVLC $d_f = 2$	RVLC $d_f = 2$	RVLC $d_f = 2$
$f: =$	[0,1,0]	[0,1,0]	[0,1,0]	[.5,0,.5]	[.4,.4,.2]
conv. th. [dB]	0.59	0.64	0.87	0.52	0.55
$\alpha_{\text{ser}_L}$	-1	-1	-2	-1	-1
$\alpha_{\text{ser}}$	0	-1	-1	0	0
$\alpha_{\text{fer}}$	0	0	-1	0	0

With arbitrarily long interleavers, the convergence threshold gives us an idea of the waterfall region. This threshold is the SNR above which the turbo decoder converges toward an information level of  $H(V'_j)$ , if infinitely long interleavers are used. The bit error rate (BER) is then guaranteed to get arbitrarily small. And so is the  $\text{SER}_L$  when the VLC spectrum is bounded. However, we cannot draw any conclusion on the SER and on the FER, based solely on the convergence threshold and on the EXIT charts. To predict the SER and the FER behavior, we have to take into account the existence or on the non-existence of the interleaving gains. For the systems A, B and C considered in Fig. 3.5, the values of the convergence thresholds and of the interleaving gains are summarized in the first three columns in Table 3.1. From these values, assuming infinitely long interleavers, we can conclude that system A will provide arbitrarily small  $\text{SER}_L$ , system B arbitrarily small  $\text{SER}_L$  and SER, and system C arbitrarily small  $\text{SER}_L$ , SER and FER.

With short interleavers, the error rates cannot be made arbitrarily small and we are thus interested in the residual error floors. As a first approximation, the interleaving gains can be used again to compare the different systems in terms of error floors. For example, from Table 3.1, the best interleaving gain on the SER is achieved by both systems using the RVLCs, systems B and C. Thus, above a certain interleaver length and above a certain SNR threshold, these systems are ensured to have the lowest residual floors for the SER. This is illustrated in Fig. 3.6 where, as expected, above a certain SNR threshold, the SER improves steadily (at least as fast as  $N^{-1}$ ) with systems B and C as



**Figure 3.6** Illustration of the interleaving gains on the SER for the systems A, B and C considered in the first three columns in Table 3.1 and for different interleaver sizes  $N \in \{1000, 2000, 4000\}$ .

the interleaver length  $N$  increases. But it does not improve with system A. At last, regarding the  $SER_L$  and the FER, note that the best interleaving gains in Table 3.1 are achieved by system C.

### 3.5.3 Optimization of the convergence threshold

As mentioned, we can optimize the system notably by choosing appropriately the coefficients  $f_i$  of the irregular RC. To make this kind of optimization easier by EXIT charts, note that a fast computation of the EXIT chart of the VLC-RC is given in Appendix 3.A.

Let us consider for example the minimization of the convergence threshold of system C (with the RVLC  $d_f^{vlc} = 2$ ), for long interleaver applications, subject to an interleaving gain on the  $SER_L$  (but not necessarily on the SER or on the FER). Let us limit the search to  $f_{1:3}$ , i.e., let us consider  $f_{i \geq 4} = 0$ , subject to the same RC rate  $r_{rc} = 1/2$ . According to (3.14), there is always an interleaving gain on the  $SER_L$ , independently of  $f_i$ , since  $d_f^{vlc} = 2$ . So there is no restriction on  $d_f^{rc}$ , thus  $f_1$  can be greater than zero. A good solution satisfying

all constraints is  $f_c = [0.5, 0.0, 0.5]$ . The corresponding EXIT charts are indeed close to each other in Fig. 3.7 and the convergence threshold is 0.52dB. In the following, this solution is labeled system D.

Besides this solution, and comparatively, we might be interested in a solution characterized by a larger distance between the EXIT charts in the upper right corner. A larger distance between the EXIT charts in the upper right corner helps indeed in practice to correct residual errors and to lower slightly the error floors. By increasing  $r_{rc}$  (and decreasing  $r_{pp}$  to maintain  $R_c$  constant in (3.2)), we find for example  $f_c = [0.403296, 0.4, 0.196704]$ , labeled system E in the following. The distance between the EXIT charts is indeed slightly larger in the upper right corner in Fig. 3.7, at the expense of a small increase of the convergence threshold, 0.55dB, see Table 3.1.

At last, note that a general and systematic method to optimize the match between the two EXIT charts is given in [94]. This method, originally developed for error correcting codes — where no symbol error rate is taken into account —, can be enhanced to better meet the current system, by adding to the optimization problem one or more constraints on the minimum desired interleaving gain(s), based on the results from Section 3.4. Of course, requiring an increased interleaving gain will generally deteriorate the convergence threshold because of fewer degrees of freedom in the optimization problem. Nevertheless, additional degrees of freedom are possible if we include other elements in the optimization problem. For example, a mixture of CCs (or irregular CC), possibly of different code rates, can be considered instead of the CC and the puncturing module used in Fig. 3.1; we then have a double<sup>11</sup> irregular system.

---

<sup>11</sup>The optimization problem of [94] can be extended straightforwardly to such a system and can still be modeled as a quadratic programming problem as long as the rate allocation strategy is given, i.e., as long as the average code rate of the irregular RC, the average code rate of the irregular CC and the code rates of the other components in Fig. 3.1 are given (see (3.2)). Unfortunately, there is no elegant method known to the authors to jointly optimize the match of the EXIT charts and the rate allocation subject to a global code rate  $R_c$ . One suboptimal method could be to solve the optimization problem for different rate allocations and to keep the best one. If the purpose is only to maximize the global code rate  $R_c$ , another suboptimal method could be to fix the rate allocation except for one component code, to maximize  $R_c$  as in [94] by linear programming and to repeat this in a round-robin fashion for all component codes.

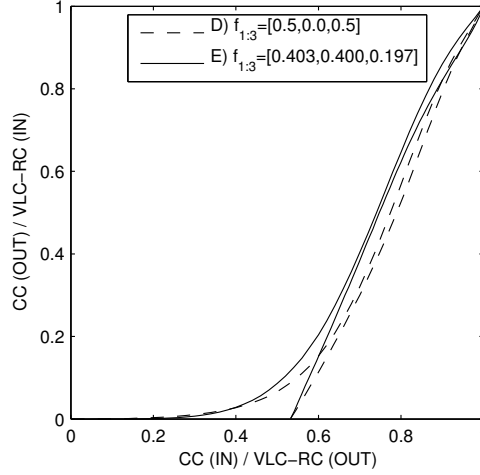


Figure 3.7 Low SNR optimization by EXIT chart matching,  $E_b/N_0 = 0.56\text{dB}$ .

## 3.6 Related works

### 3.6.1 Regular turbo codes and VLCs

If we use a rate-1/2 regular RC and the interleaver given in (3.17), the channel code in Fig. 3.1 is equivalent to the parallel turbo code used in Fig. 3.2. Such a parallel turbo code has already been proposed in [30, 88] to protect VLC bit streams. Still, compared to the proposed system, there is a small but significant difference: Compared to Fig. 3.2, these contributions do not use the interleaver  $\Pi_1$ , and without this interleaver, the global code and the associated decoder are completely different.

Let us consider firstly the impact of removing  $\Pi_1$  on the global code. Without the interleaver  $\Pi_1$ , there is no code spreading between the VLC and the first CC in Fig. 3.2. Therefore, the system cannot benefit from the same interleaving gains when error correcting VLCs are used. More precisely, let us assume as in most cases that if the VLC has a free distance  $d_f^{\text{vlc}} \geq 2$ , at least one error event of the VLC at distance  $d_f^{\text{vlc}}$  is also an error event of the first CC in Fig. 3.2. Then we have the following interleaving gains for  $d_f^{\text{vlc}} \geq 1$ :

$$\alpha_{\text{serL}} \leq -\lfloor (d_f^{\text{vlc}} + 1)/2 \rfloor, \quad (3.36)$$

$$\alpha_{\text{ser}} \leq \alpha_{\text{fer}} \leq 1 - \lfloor (d_f^{\text{vlc}} + 1)/2 \rfloor. \quad (3.37)$$

**Table 3.2** Computational decoding complexity per bit of entropy for several variants of a VLC + a regular rate-1/2 turbo code.

Global Code	Turbo decoder	Decoder Complexity
Fig. 3.1	(joint VLC-RC) + CC, Section 3.3	$r_s^{-1} N_{\text{iter}} (2 X^c  +  \overline{X^s} )$
Fig. 3.2, no $\Pi_1$	(joint VLC-CC <sub>1</sub> ) + CC <sub>2</sub> , [30]	$r_s^{-1} N_{\text{iter}} ( X^c  +  X^c   \overline{X^s} )$
Fig. 3.2, no $\Pi_1$	(subopt. VLC-CC <sub>1</sub> ) + CC <sub>2</sub> , [88]	$r_s^{-1} N_{\text{iter}} (2 X^c  + \text{neglig. VLC})$
Fig. 3.1	(hard VLC) + RC + CC, tandem	$r_s^{-1} N_{\text{iter}} (2 X^c ) + \text{neglig. VLC}$

As expected, these interleaving gains are less attractive for  $d_f^{\text{vlc}} \geq 2$  than the gains (3.18)–(3.21) offered by the system in Fig. 3.2 with  $\Pi_1$ . Furthermore, when  $\Pi_1$  is not used, it is straightforward to show that there is no gain improvement when RVLCs are used, unlike (3.22), (3.23). These small differences in interleaving gains will have a significant impact in terms of performance and error floors in the simulation results in the next section.

Let us consider now the impact of removing  $\Pi_1$  on the decoding complexity. When  $\Pi_1$  is not used, the factor graph of the global code has many short cycles between the Markov chain of the source/VLC and the Markov chain of the first CC. Unfortunately, in the presence of short cycles, the Sum-Product algorithm does not behave nicely and we have thus to merge the two Markov chains (so as to remove the short cycles), i.e., to decode the VLC and the first CC jointly. Merging the two Markov chains is equivalent to consider the product of the corresponding state spaces,  $|X^c| |\overline{X^s}|$ , which makes the optimal joint VLC-CC decoder developed in [30] quite complex. A first approach proposed in [30] to lower that complexity is to consider a CC of lower memory, e.g., an accumulator of memory 1,  $m_{cc} = 1$ , thus  $|X^c| = 2^{m_{cc}}$  is smaller. Another approach is to use the suboptimal VLC-CC decoder developed in [88], which is based on the MAX-LOGMAP algorithm on a reduced trellis. However, both approaches tend to lower significantly the robustness of the system, as we will see in the simulation results.

For the sake of clarity, the decoding complexities are summarized in Table 3.2, where the decoding complexity of the tandem decoder is given as a reference.

### 3.6.2 Regular turbo codes and FLCs

It is interesting to compare also the proposed system in Fig. 3.2 with the system proposed in [30] (which is the system in Fig. 3.2 without the interleaver  $\Pi_1$ ) when FLCs are used.

Straightforwardly, without the interleaver  $\Pi_1$ , by (3.24), (3.36) and (3.37), the interleaving gains with an FLC of free distance  $d_f^{\text{vlc}}$  are

$$\alpha_{\text{ser}} = \alpha_{\text{ser}_L} \leq -\lfloor (d_f^{\text{vlc}} + 1)/2 \rfloor, \quad (3.38)$$

$$\alpha_{\text{fer}} \leq 1 - \lfloor (d_f^{\text{vlc}} + 1)/2 \rfloor. \quad (3.39)$$

Note that these interleaving gains are less attractive for  $d_f^{\text{vlc}} \geq 2$  than the interleaving gains obtained when the interleaver  $\Pi_1$  is used (see Theorem 3.3 and Corollary 3.6 with  $d_f^{\text{rc}} = 2$ ). It is worth emphasizing, though, that most FLCs used in practice have a free distance  $d_f^{\text{vlc}} = 1$ , in which case the interleaving gains are identical and independent of the interleaver  $\Pi_1$ .

Still, using or not the interleaver  $\Pi_1$  has a significant impact on the decoding complexity. Indeed, the complexity comparison given for VLCs in Section 3.6.1 and in Table 3.2 holds with FLCs. To summarize, when the interleaver  $\Pi_1$  is not used, the FLC and the first CC of the turbo code must be considered jointly in the turbo decoder, which dramatically increases the decoding complexity if an optimal joint FLC-CC decoder is considered (second row in Table 3.2).

### 3.6.3 Regular turbo codes and binary sources

A parallel turbo code has also been proposed in [37] to protect a binary memoryless source, where the source bits are directly fed uncompressed into the turbo code. When the source is not uniform, some redundancy is thus left and it is suggested in [37] to use it at the receiver so as to improve the performance of the turbo code. This technique has been extended to Markov binary sources in [41].

This system has been compared in [37] to a tandem benchmark system based on a  $k$ th-order Huffman VLC serially concatenated with a parallel turbo code. The  $k$ th-order VLC consists in segmenting the source bit stream into segments of  $k$  bits and in replacing each segment by a VLC codeword. Since a segment of  $k$  bits can take  $2^k$  different values, the VLC must be composed of  $2^k$  codewords.

Though the present work is not specifically optimized for binary sources, it is interesting to investigate how the interleaving gain analysis enables to improve the tandem benchmark system used in [37] and how it compares with the system proposed in [37]. To this end, let  $\alpha_{\text{sber}}$  be the interleaving gain on the source BER, i.e., on the BER affecting the source bits.

In the system proposed in [37], the source bits are directly fed uncompressed into the turbo code. This means (i) the source BER is equal to the BER of the turbo code, (ii) the interleaving gains are determined solely by the turbo code (since there is no source code). Therefore,

$$\alpha_{\text{sber}} \leq -1, \quad \text{and} \quad \alpha_{\text{fer}} \leq 0, \quad (3.40)$$

which are the interleaving gains offered by parallel turbo codes, see [15].

Let us now consider the system given in Fig. 3.2 with a  $k$ th-order VLC and a parallel turbo code, assuming at the receiver the joint source-channel turbo decoder described in Section 3.3. The source BER of this system is straightforwardly proportional to the SER of its  $k$ th-order VLC. Therefore,

$$\alpha_{\text{sber}} \leq \alpha_{\text{fer}} \leq \begin{cases} 0, & \text{if } d_f^{\text{vlc}} = 1, \\ 1 - 2\lfloor (d_f^{\text{vlc}} + 1)/2 \rfloor, & \text{otherwise,} \end{cases} \quad (3.41)$$

by Theorem 3.3 with  $d_f^{\text{rc}} = 2$ . Besides, if the VLC is reversible with  $d_f^{\text{vlc}} = 1$ ,

$$\alpha_{\text{sber}} \leq -1 \quad (3.43)$$

by Theorem 3.5 with  $d_f^{\text{rc}} = 2$ .

Thus, on the one hand, depending on the chosen  $k$ th-order VLC, the proposed system may have an advantage in terms of interleaving gains. On the other hand, though, it is worth noting the system proposed in [37] does not

need a (heavy) VLC decoder and has therefore an advantage in terms of decoding complexity. In the simulation results, a performance comparison of the two systems will be given, using an RVLC of free distance  $d_f^{\text{vlc}} = 2$  with the proposed system.

### 3.7 Simulation Results

Simulation results are now reported for the systems considered in Sections 3.4–3.6 and summarized in Table 3.3. Four VLCs are considered: an FLC, an HVLC (Huffman VLC), an RVLC (reversible VLC) with  $d_f^{\text{vlc}} = 1$  built as in [87] and an RVLC with  $d_f^{\text{vlc}} = 2$  built as in [49]. The systematic bits are not punctured. The CC is punctured<sup>12</sup> to keep the global code rate constant among the different systems,  $R_c = 1/2$  in (3.2). The generators of the CC are given by  $(037, 021)_8$ , where  $037_8$  is the feedback. Note that using other generators or another puncturing would change the system performance. But it would not alter the discussion since all systems would be affected almost similarly. These particular generators have been chosen because they are weak. The global code in Fig. 3.1 is indeed capable of low error rates, thanks to its better interleaving gains. Combined with strong generators, it leads to error rates that are difficult to measure accurately by Monte-Carlo simulations. Consequently, stronger generators will be considered only in Fig. 3.10.

The primary goal of the simulation results hereafter is to illustrate the benefits of the proposed system in terms of interleaving gains, especially compared to the systems based on turbo codes and proposed in [30, 88] (systems I to N in Table 3.3). In particular, the simulation results are not aimed at comparing the considered VLCs. Nevertheless, for the reader interested in a comparison between them, it is worth emphasizing the considered VLCs have different code rates  $r_s$  which are compensated in this chapter by adapting accordingly the rate  $r_{pp}$  of the CC puncturing in order to compare all systems at the same global code rate  $R_c$  in (3.2) (same global bandwidth usage).

<sup>12</sup>The puncturing is performed as follows. Let  $r_{cc} r_{pp}$  be the code rate of the punctured CC, see (3.2), and let us neglect the trellis termination ( $r_{cc} \approx 1$ ), then the bit  $R_i$  is *not* punctured iff  $\exists n \in \mathbb{N}_{\geq 0} : i = \lfloor nr_{pp} + 3/2 \rfloor$ .

### 3.7.1 Comparisons with the English alphabet source

We start with a theoretical source of independent symbols known as the “English alphabet”, see [49] and references therein. For that source, the source code rate  $r_s$  and the size  $|\overline{X^s}|$  of the state space of the source/VLC Markov chain are given by  $(r_s, |\overline{X^s}|) = (0.992, 25)$  for the HVLC,  $(0.984, 27)$  for the RVLC  $d_f^{\text{vlc}} = 1$ ,  $(0.948, 41)$  for the RVLC  $d_f^{\text{vlc}} = 2$ , and  $(0.824, 5.4)$  for the FLC.

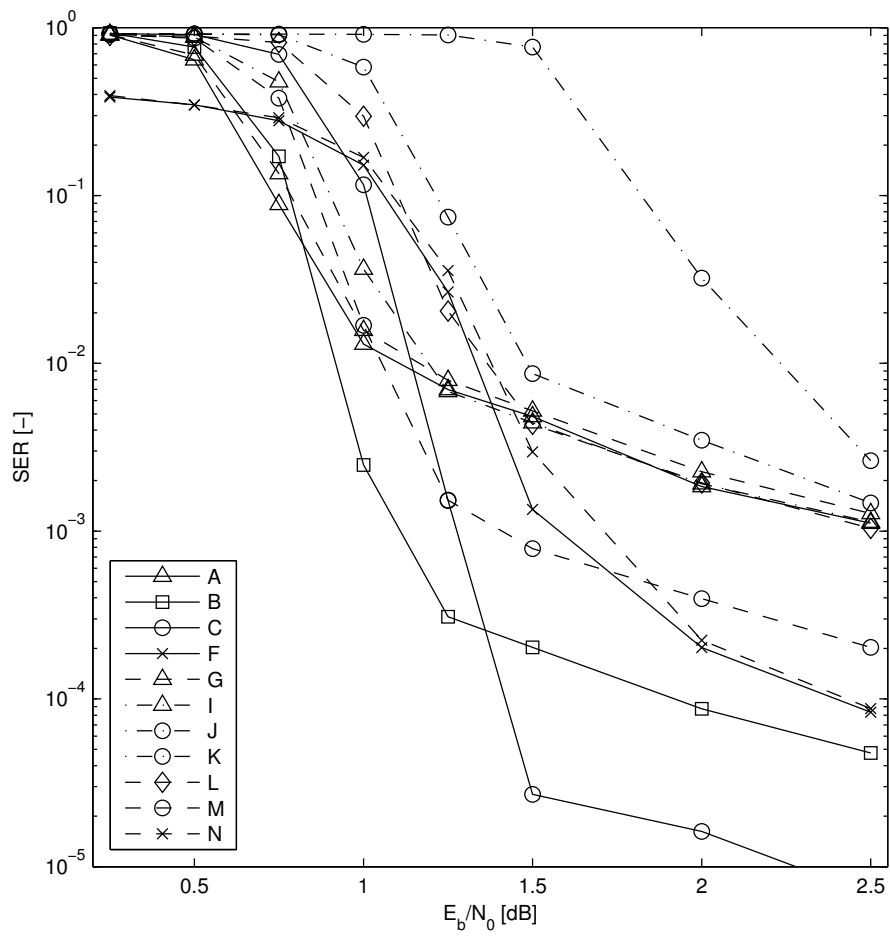
As we can see, the HVLC leaves almost no redundancy in the bit stream ( $r_s = 0.992$ ). This situation is here interesting for two reasons. Firstly, it is in favor of the tandem approach. So it makes the tandem approach a good reference point to which the joint source-channel approach can be compared and validated. Secondly, it helps to better isolate the gain due to the RVLCs since the gain due to the residual redundancy in the HVLC is almost zero.

The SER achieved by several systems is given in Fig. 3.8 for  $N = 4000$  and allows a few pertinent observations. Firstly, systems A, G and I, which use the HVLC, perform all almost equally. As mentioned, the redundancy contained in the HVLC bit stream is nearly zero with this source; so the way the VLC is concatenated or decoded does not change much the global performance. Secondly, system M with the RVLC  $d_f^{\text{vlc}} = 2$  and system N with the FLC bring a significant improvement. However, as already discussed, they use a joint VLC-CC (FLC-CC) decoder of high complexity. A first approach proposed in [30] to lower that complexity is to consider a CC of lower memory, e.g., an accumulator of memory 1, system L, at the expense of some decrease in performance. Another approach is to use the suboptimal VLC-CC decoder developed in [88], of low complexity, which is known to perform nicely with HVLCs (here, system I). Unfortunately, this suboptimal VLC-CC decoder appears to be affected by severe degradations<sup>13</sup> when RVLCs are used, systems J and K.

<sup>13</sup>One possible explanation is the following. This algorithm is based on the MAX-LOGMAP algorithm. But, instead of keeping one surviving path for each state of the joint VLC-CC trellis, i.e., one path for each VLC state in each CC state, it keeps only one surviving path (for only one VLC state) in each CC state. With RVLCs, or other error correcting VLCs [14], some states of the VLC binary tree lead only to one possible transition (either 0 or 1). When a surviving path falls into such a VLC state, the only possible transition according to the VLC may be very improbable according to the CC, making this path useless for subsequent operations. That issue is not a problem with the MAX-LOGMAP algorithm on the joint VLC-CC trellis, because of the other surviving paths in the other VLC states. Of course, increasing the number of surviving paths

**Table 3.3** Summary of the systems compared in the simulation results.

Global Code	Particularities	Literature
A) Fig. 3.2	HVLC	
B) Fig. 3.2	RVLC $d_f^{vlc} = 1$	
C) Fig. 3.2	RVLC $d_f^{vlc} = 2$	
D) Fig. 3.1	RVLC $d_f^{vlc} = 2$ , $f: = [0.5, 0.0, 0.5]$	
E) Fig. 3.1	RVLC $d_f^{vlc} = 2$ , $f: = [0.403, 0.4, 0.197]$	
F) Fig. 3.2	FLC	
G) Fig. 3.2	HVLC, tandem decoder with soft VLC decoder	
H) Fig. 3.2	RVLC $d_f^{vlc} = 2$ , tandem decoder with soft VLC decoder	
I) Fig. 3.2, no $\Pi_1$	HVLC, SUBMAP VLC-CC decoder	[88]
J) Fig. 3.2, no $\Pi_1$	RVLC $d_f^{vlc} = 2$ , SUBMAP VLC-CC decoder	[88]
K) Fig. 3.2, no $\Pi_1$	RVLC $d_f^{vlc} = 2$ , SUBMAP VLC-CC decoder 5 survivors	[88]
L) Fig. 3.2, no $\Pi_1$	RVLC $d_f^{vlc} = 2$ , accumulator as $CC_1$	[30]
M) Fig. 3.2, no $\Pi_1$	RVLC $d_f^{vlc} = 2$	[30]
N) Fig. 3.2, no $\Pi_1$	FLC	[30]
O) Fig. 3.2	memoryless binary source, 4th order RVLC $d_f^{vlc} = 2$	
P) [37, Fig. 3]	memoryless binary source (directly fed into the turbo code)	[37]
Q) [37, Fig. 8]	memoryless binary source, 4th order HVLC, tandem decoder	see tandem (35,23) in [37, Fig. 8]



**Figure 3.8** English alphabet source,  $N = 4000$  VLC bits,  $N_{\text{iter}} = 50$  iterations, random interleaving. See Table 3.3 for system parameters.

Systems B, C and F do not suffer from these issues (high complexity or poor performance). The SER they achieve is indeed much lower, which is in agreement with the fact that they benefit from interleaving gains on the SER while other systems do not (except system N), see Section 3.6. Besides, recall that systems B, C and F have an advantage in decoding complexity compared to systems M and N, since the VLC (or FLC) and the first CC do not need to be decoded jointly.

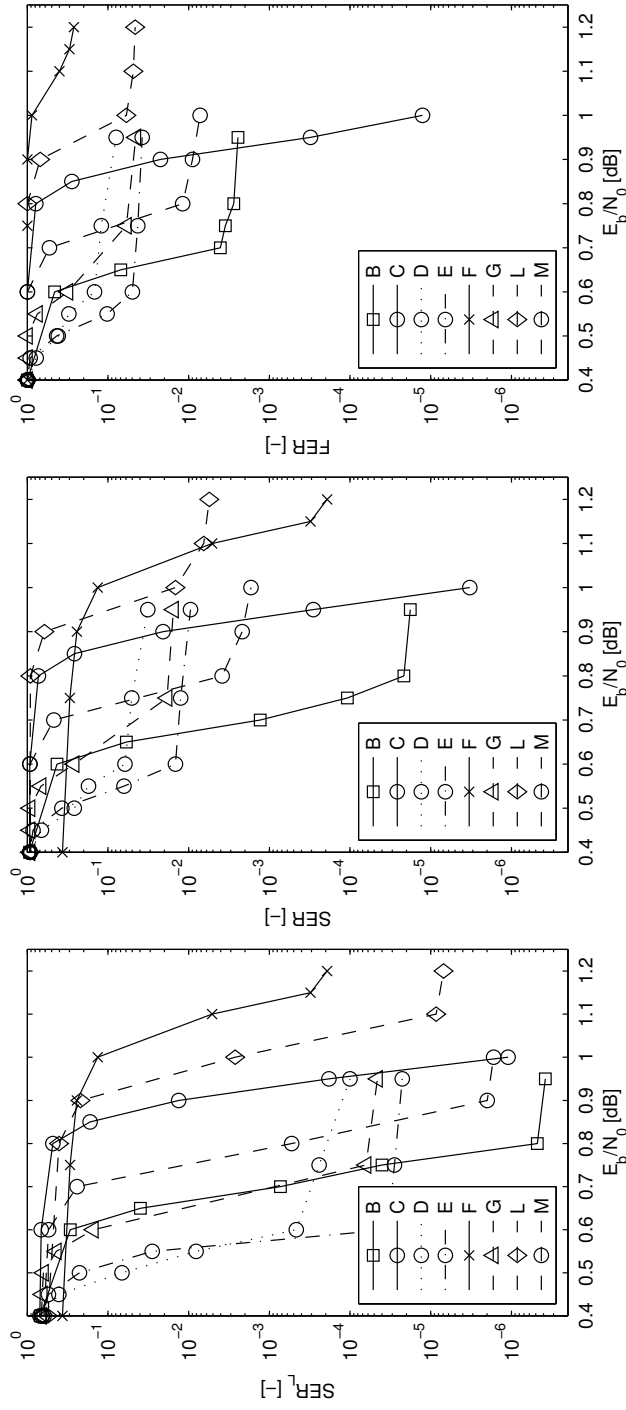
To better illustrate the impacts of the interleaving gains, longer interleavers are considered in Fig. 3.9. All systems have the same interleaving gain on the  $SER_L$ ,  $\alpha_{ser_L} = -1$ , and the  $SER_L$  error floors are indeed close to each other, except system C whose interleaving gain is  $\alpha_{ser_L} = -2$ . On the contrary, significant differences exist in SER error floors because only systems B, C (with RVLCs) and F (with FLCs) have an interleaving gain on the SER,  $\alpha_{ser} = -1$ . At last, the best FER is achieved by system C which is the only system offering an interleaving gain on the FER, with  $\alpha_{fer} = -1$ . These comparisons illustrate the significance of interleaving gains and, as a by-product, the interesting interleaving gain that can be obtained on the SER with RVLCs  $d_f = 1$  and turbo-like codes (see Theorem 3.5).

Regarding the waterfall regions, note that the convergence thresholds evaluated in Section 3.5 are good predictions of these. In this context, note also that systems I and J which have been optimized for  $SER_L$  performance at low  $E_b/N_0$  in Section 3.5.3, have indeed the best  $SER_L$  in the waterfall region at  $E_b/N_0 = 0.6\text{dB}$ .

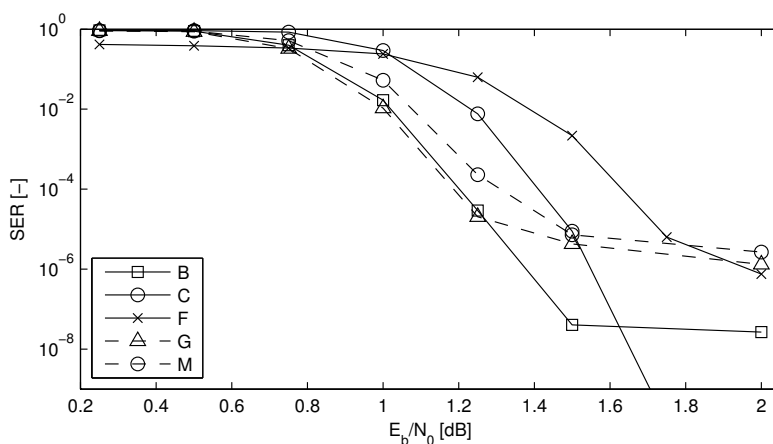
At last, note that the proposed system is definitely superior to the system proposed in [30]. Indeed, the comparison between systems M and L, and system C is self-speaking. Either we make the comparison at the same decoding complexity and system C provides a huge improvement in performance over system L. Or we make the comparison with the same CC generators and system C provides both a significant improvement in performance (especially in SER and in FER) and a significant reduction<sup>14</sup> of the decoding complexity

for each CC state improves performance, system K in Fig. 3.8; however, the decoding complexity rapidly increases because one has to maintain a list of the best surviving paths in each CC state.

<sup>14</sup>For both systems C and M,  $r_s = 0.948$ ,  $|\overline{X^s}| = 41$  and  $|X^c| = 16$ . Therefore, the decoding complexity per bit of entropy is  $77.0 \times N_{iter}$  for system C and  $708.9 \times N_{iter}$  for system N, see the first two rows in Table 3.2.



**Figure 3.9** English alphabet source,  $N = 65536$  VLC bits,  $N_{iter} = 50$  iterations, random interleaving. See Table 3.3 for system parameters. Note that the statistical accuracy is lower for system C at  $E_b/N_0 = 1$ dB: Only 12 error frames were measured among  $10^6$  tested frames.



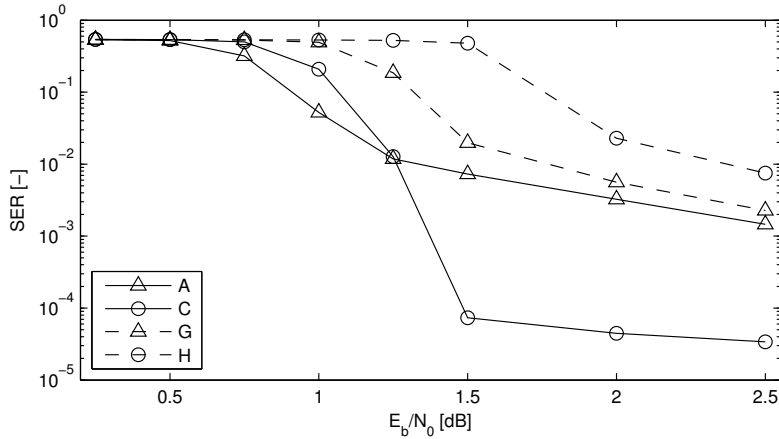
**Figure 3.10** English alphabet source, CC generators  $(023, 035)_8$ , S-random interleaving of spread 25,  $N = 4000$  VLC bits,  $N_{\text{iter}} = 50$  iterations. See Table 3.3 for system parameters. Note that the statistical accuracy is smaller than in Fig. 3.8. More precisely, the results were averaged over  $10^7$  frames or 450 error frames, with two exceptions: 1) system C, only 1 error frame was measured among  $15 \times 10^7$  tested frames at  $E_b/N_0 = 1.75\text{dB}$ ; 2) due to its higher decoding complexity, the results for system M were averaged over  $2 \times 10^6$  frames or 80 error frames.

over system M. The reader is invited to make the same comparison between systems C and M hereafter in Fig. 3.10.

### 3.7.2 Comparisons with other parameters

So far, random uniform interleaving was used to satisfy Assumption 3.1 and the CC generators  $(037, 021)_8$  were chosen to get error rates that are easier to measure by Monte-Carlo simulations. Let us now compare the SER achieved by a few systems when better components are used, that is, S-random interleavers of spread 25 and CC generators  $(023, 035)_8$ . As we can see in Fig. 3.10, the tandem system G is much more robust than in Fig. 3.8 — though not given in the plot, note that the BER is around  $10^{-9}$  for that system at 1.5dB. Nevertheless the observations made previously still hold: Systems B and C do keep a clear advantage in terms of error floors over other systems.

As mentioned above, the HVLC for the English alphabet source leaves almost no redundancy in the bit stream. In Fig. 3.11, another source is consid-

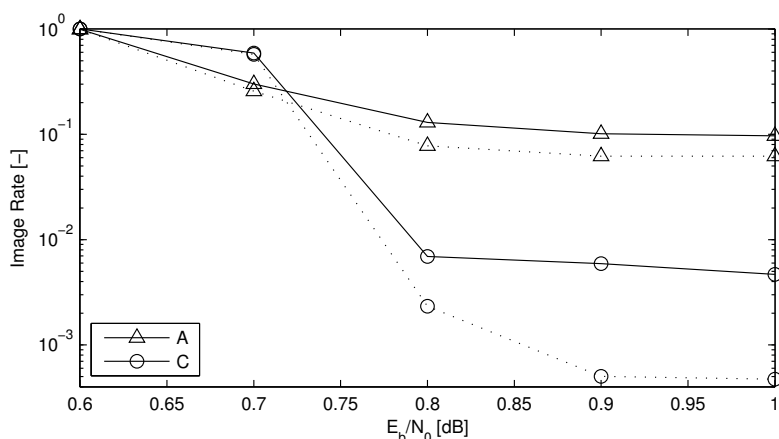


**Figure 3.11** Memoryless source with 3 possible values (probabilities  $\{0.6, 0.3, 0.1\}$ ),  $N = 4000$ ,  $N_{\text{iter}} = 50$  iterations, random interleaving. See Table 3.3 for system parameters.

ered for which the HVLC ( $r_s = 0.925$ ) and the RVLC  $d_f^{\text{vlc}} = 2$  ( $r_s = 0.864$ ) leave comparatively a larger amount of redundancy. The purpose of this simulation is simply to illustrate that, when there is some residual redundancy in the bit stream, the joint turbo decoder provides an improvement of the coding gain over the tandem decoder, in addition to a better error floor with the RVLC.

At last, the transmission of an image across an AWGN channel is considered in Fig. 3.12 for different transmission systems. These results corroborate once again the observations made so far, with a real signal this time. In addition, note that the improvement provided by system C over system A is larger when a small PSNR distortion (between the transmitted and the decoded images) is allowed. The image considered is Lena and the image compression codec<sup>15</sup> is based on the discrete cosine transform.

<sup>15</sup>The image codec uses the  $8 \times 8$  discrete cosine transform (DCT). The DC coefficients are encoded with FLCs. The non-zero values of the 63 AC coefficients are encoded with VLCs, while the differential positions of these non-zero values are encoded in a separate sequence with VLCs. The difference between adjacent (both vertically and horizontally) DC coefficients are assumed to follow a Gaussian distribution, while the AC coefficients are assumed to be Laplacian [95]. The variances of the distributions are estimated at the coder, and sent well protected to the decoder. These parameters are then used to create both coding and decoding tables. Note that the VLCs are constrained such that the codewords with occurrence probability below a given threshold  $P_{\text{pr}}$  have the same fixed length. We call such VLCs *least probable fixed - variable length codes*. They are



**Figure 3.12** Solid: image error rate, i.e., rate of images with a PSNR degradation above 0dB. Dotted: rate of images with a PSNR degradation above 3dB.  $N = 65536$  bits,  $N_{\text{iter}} = 30$  iterations. See Table 3.3 for system parameters.

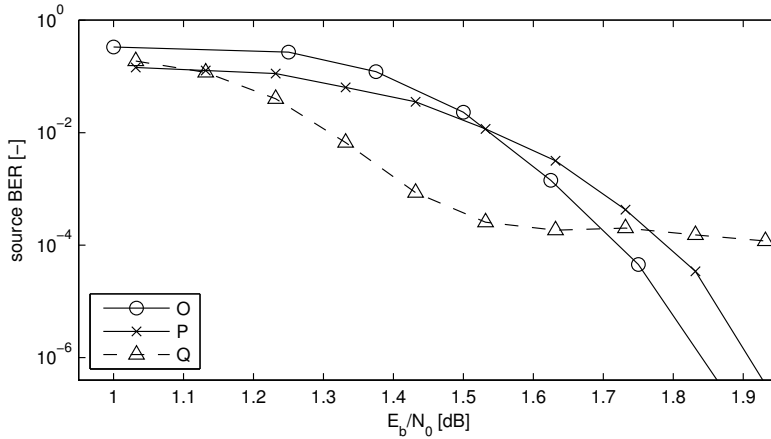
### 3.7.3 Comparisons with a binary source

To illustrate the comparison discussed in Section 3.6.3, a non-uniform memoryless binary source characterized by the probability  $P(U_i = 0) = 0.83079$  is considered with three different systems in Fig. 3.13.

Among these systems, system P is the system proposed in [37]. It uses a non-systematic asymmetric turbo code with CC generators  $(035, 023)_8$  and  $(035, 025)_8$ , without source coding, that is, the source bits are directly fed uncompressed into the turbo code. The size of the interleaver for this system is fixed to 12000 bits, which corresponds to 12000 source bits. System Q is the tandem system used as benchmark in [37, Fig. 8]. It uses a fourth-order HVLC and a symmetric turbo code with CC generators  $(035, 023)_8$ . For this system, the number of source bits is fixed to 12000 bits in each packet; the length of the interleaver is thus variable with an average of 8000 bits. System O is the system proposed in this chapter, based on the global code given in Fig. 3.2, with CC generators  $(035, 023)_8$  and with a fourth-order RVLC  $d_f^{\text{vlc}} = 2$ . For this system, the size of the interleaver is fixed to  $N = 8458$ , which corresponds

---

less sensitive to desynchronization in the case of an imperfect source model (as in this codec). An example is given in [10].



**Figure 3.13** Binary memoryless source, Rayleigh fast fading channel, S-random interleaving of spread 10,  $N_{\text{iter}} = 20$  iterations. See Table 3.3 for system parameters.

to 8458 VLC bits or to an average of 12000 source bits. At last, the source code rates<sup>16</sup> are respectively  $r_s = 0.931$ ,  $0.656$ , and  $0.984$ , for systems O, P and Q.

As we can see in Fig. 3.13, system Q suffers from a high error floor. By contrast, systems O and P are quite robust and offer roughly the same level of performance. This is in agreement with the conclusions drawn in Section 3.6.3; by (3.40) and (3.42), systems O and P have indeed an interleaving gain on the source BER, given by  $\alpha_{\text{sber}} = -1$ , while system Q has not.

Though systems O and P perform almost equally in Fig. 3.13, one system may still be preferred over the other, depending on the application. On the one hand, for example, system O has an advantage over system P in terms of interleaving gains, as it offers an interleaving gain on the FER, by (3.42), while system P does not. On the other hand, as mentioned in Section 3.6.3, system P has an advantage over system O in terms of decoding complexity, since system P does not need a (heavy) VLC decoder — this is however slightly compensated by a longer turbo code since the interleaver length is 12000 for system P, compared to 8458 for system O. More precisely, for system P, the total compu-

<sup>16</sup>Note that the source code rate  $r_s$  is not computed as in [37]. The source code rate is here defined as the ratio of the source entropy over the average number of bits used to encode the source messages.

tational decoding complexity per bit of entropy is given by  $r_s^{-1} N_{\text{iter}} \bar{d} |X^c| = 48.8 \times N_{\text{iter}}$ , for system O by  $r_s^{-1} N_{\text{iter}} (\bar{d} |X^c| + |\overline{X^s}|) = 57.0 \times N_{\text{iter}}$ .

### 3.8 Conclusion

To protect VLC bit streams containing a small amount of redundancy, this chapter proposes a repetition code concatenated with a convolutional code, i.e., an irregular turbo code. In the case of non-binary sources, this provides significant advantages over previous solutions from the literature, in terms of flexibility, performance and decoding complexity. These advantages have been successfully analyzed through interleaving gains, EXIT chart and complexity analysis, and finally corroborated by simulation results. One notable result of this analysis is the discovery and proof of surprising interleaving gains with reversible VLCs of free distance 1, on the probability of desynchronization of the VLC decoder and on the (non-Levenshtein) symbol error rate.

The analysis in this chapter has however limitations. We focused on a particular VLC decoder; other interesting design rules might be deduced when, e.g., the decoder knows the number  $K$  of transmitted symbols. In addition, the scope has been restricted to irregular turbo codes. Extension of the design rules to LDPC codes, as used with VLCs and optimized by mutual information evolution in [40], would be interesting.

Last but not least, we focused only on the error correcting capabilities of the VLC in terms of free distance. Consequently, the proposed analysis does not distinguish two VLCs having the same free distance and the same compression efficiency. Of course, the EXIT charts in Section 3.5 can be used to get further insight but this is not sufficient. Further investigation on VLC bit streams containing a large amount of redundancy is necessary.

For convenience, some theoretical details were skipped on purpose in Sections 3.4–3.5. These details are developed and proved in the next two chapters.

## Appendix 3.A Fast computation of the VLC-RC EXIT chart

To make the optimization of the RC degree distribution  $f$ : more efficient by EXIT charts, a fast computation of the VLC-RC chart is now described. The VLC-RC chart can be indeed computed analytically as a function of

- $I_{\text{vlc}}(\cdot)$ , which is the EXIT chart of the VLC decoder alone;
- and  $J_0(\cdot)$ , which is the function  $J_{L_U=0}(\cdot)$  in (4.16) in Chapter 4 — note  $J_0(\cdot)$  corresponds also to the function  $J(\cdot)$  in [12, eq. (15)] with  $J_0(\mu) = J(\sqrt{2\mu})$  or  $J_0(\sigma^2/2) = J(\sigma)$ .

So the heavy computation of the VLC chart  $I_{\text{vlc}}(\cdot)$  needs to be done only once. Subsequently, if the functions  $I_{\text{vlc}}(\cdot)$  and  $J_0(\cdot)$  are implemented by look-up tables, the computation of the VLC-RC chart for different RC distributions  $f$ : using the analytical expression hereafter is very fast.

Let  $g_i$  be the fraction of bits in  $V$ : ( $V'$ ) of degree  $i$ , i.e., belonging to the class of bits repeated  $i$  times,  $g_i = f_i i / \bar{d}$ . Recall the definition (3.32) of  $I_{\text{cc}}^e$  and  $I_{\text{vlc-rc}}^e$ . Moreover, let  $I_{\text{vlc}}(\cdot)$  be the EXIT chart of the VLC, i.e., let  $I_{\text{vlc}}(I_{\text{vlc}}^a)$  be the level of extrinsic information at the output of the VLC decoder (with the pseudo-random bit flipping included, see Section 3.5.1) for a given a priori information  $I_{\text{vlc}}^a$  at the input of the VLC decoder. This function is made independent of the channel by considering that the VLC decoder has no access to the channel and by including the systematic channel measures on  $U$ : in the a priori information  $I_{\text{vlc}}^a$  — through parameter  $\mu_{\text{ch}}$  in (3.50) hereafter.

Then, the EXIT chart  $I_{\text{vlc-rc}}(\cdot)$  of the VLC-RC decoder can be developed as a function of  $I_{\text{cc}}^e$  as

$$I_{\text{vlc-rc}}^e = \sum_i g_i I_{\text{vlc-rc},i}^e \quad \text{by [53, eq. (19)],} \quad (3.44)$$

where

$$I_{\text{vlc-rc},i}^e = J_0(\mu_{\text{ch}} + \mu_{\text{vlc}} + (i-1)\mu_{\text{cc}}), \quad \text{by [53, Example 3],} \quad (3.45)$$

$$\mu_{\text{ch}} = 2 R_c E_b / (N_0/2), \quad \text{by [12, eq. (7)],} \quad (3.46)$$

$$\mu_{\text{cc}} = J_0^{-1}(I_{\text{cc}}^e), \quad \text{by definition of } J_0(\cdot), \quad (3.47)$$

$$\mu_{\text{vlc}} = J_0^{-1}(I_{\text{vlc}}(I_{\text{vlc}}^{\text{a}})), \quad \text{by definition of } J_0(\cdot), \quad (3.48)$$

$$I_{\text{vlc}}^{\text{a}} = \sum_i f_i I_{\text{vlc},i}^{\text{a}}, \quad \text{by [53, eq. (19)]}, \quad (3.49)$$

$$I_{\text{vlc},i}^{\text{a}} = J_0(\mu_{\text{ch}} + i \mu_{\text{cc}}), \quad \text{by [53, Example 3]}, \quad (3.50)$$

where  $R_c$  is the global code rate,  $N_0/2$  the double-sided noise spectral density,  $E_b$  the energy per bit of entropy, and  $I_{\text{vlc-rc},i}^{\text{e}}$  and  $I_{\text{vlc},i}^{\text{a}}$  are respectively the amount of extrinsic information produced by the VLC-RC decoder and the amount of a priori information at the input of the VLC decoder, for a bit of degree  $i$ . After substitutions, we obtain the VLC-RC chart  $I_{\text{vlc-rc}}(\cdot)$  as

$$\begin{aligned} I_{\text{vlc-rc}}^{\text{e}} &= I_{\text{vlc-rc}}(I_{\text{cc}}^{\text{e}}) \\ &= \sum_i g_i J_0 \left( \mu_{\text{ch}} + J_0^{-1} \left( I_{\text{vlc}} \left( \sum_j f_j J_0 \left( \mu_{\text{ch}} + j J_0^{-1}(I_{\text{cc}}^{\text{e}}) \right) \right) \right) \right. \\ &\quad \left. + (i-1) J_0^{-1}(I_{\text{cc}}^{\text{e}}) \right) \end{aligned} \quad (3.51)$$

with  $\mu_{\text{ch}} = 2 R_c E_b / (N_0/2)$ .

Equations (3.44), (3.45), (3.49) and (3.50) are based on two properties. Firstly, the EXIT chart of a mixture of codes, or irregular code, here the RC, is given [94], [53, eq. (19)] by the average of the component EXIT charts, which explains the two summations over  $i$  in (3.44) and (3.49). The sum is weighted by  $g_i$  in (3.44) (resp. by  $f_i$  in (3.49)) because, at the output of the VLC-RC decoder (resp. input of the VLC decoder), we consider the bits  $V_i$  (resp.  $U_i'$ ) and a fraction  $g_i$  (resp.  $f_i$ ) of these are of degree  $i$ . Secondly, as the LLRs associated with a given bit  $V \in \{+1, -1\}$  are assumed [12] to be independent and Gaussian ( $\mathcal{N}(V\mu, 2\mu)$  for some parameter  $\mu$ ) given the bit  $V$ , a sum of LLRs is also Gaussian, with a mean and a variance given by the sum of the means and variances of these LLRs. Let  $\mu_{\text{vlc}}$  and  $\mu_{\text{cc}}$  be the parameters characterizing the extrinsic LLRs of the VLC decoder and of the CC decoder, respectively; let  $\mu_{\text{ch}}$  be the parameter of the LLRs from the channel on the systematic bits. For a bit  $V$  of degree  $i$ , the LLR produced by the VLC-RC decoder is the sum of one LLR  $\mathcal{N}(V\mu_{\text{ch}}, 2\mu_{\text{ch}})$  from the channel, one LLR  $\mathcal{N}(V\mu_{\text{vlc}}, 2\mu_{\text{vlc}})$  from the VLC decoder, and  $(i-1)$  LLRs  $\mathcal{N}(V\mu_{\text{cc}}, 2\mu_{\text{cc}})$  from the CC decoder; the LLR produced by the VLC-RC decoder is thus distributed as  $\mathcal{N}(V(\mu_{\text{vlc}} + \mu_{\text{ch}} + (i-1)\mu_{\text{cc}}), 2(\mu_{\text{vlc}} + \mu_{\text{ch}} + (i-1)\mu_{\text{cc}}))$ , which

explains (3.45). For a bit of degree  $i$ , the LLR received by the VLC decoder is the sum of one LLR from the channel and  $i$  LLRs from the CC decoder; this LLR is thus  $\mathcal{N}(V(\mu_{\text{ch}} + i \mu_{\text{cc}}), 2(\mu_{\text{ch}} + i \mu_{\text{cc}}))$  in (3.50). Note that (3.45) and (3.50) are straightforward extensions of [53, Example 3].

The relations above rely notably on the pseudo-randomness of the interleaver  $\Pi_0$  between the VLC and the RC. When  $\Pi_0$  is not random, no average can be taken in (3.49) over the different repetition degrees. Nevertheless, similar expressions can be developed in some cases. For example, let us consider the interleaver  $\Pi_0$  is such that the bit  $U_n$  is of degree  $i$  if and only if

$$n_i < n \leq n_{i+1} \quad \text{where } n_i = N \sum_{j=1}^i f_j. \quad (3.52)$$

Then, the a priori information received by the VLC decoder for a bit  $U_n$  of degree  $i$  is

$$I_{\text{vlc},i}^{\text{a}} = J_0(\mu_{\text{ch}} + i \mu_{\text{cc}}), \quad (3.53)$$

and the extrinsic information produced by the VLC decoder on  $U_n$  is  $I_{\text{vlc}}(I_{\text{vlc},i}^{\text{a}})$  — if we neglect the tail effects near  $n \approx n_{i-1}$  and near  $n \approx n_i$ . Accordingly, the LLR produced by the VLC-RC decoder for a bit of degree  $i$  is the sum of one LLR of degree  $i$  from the VLC decoder, one LLR from the channel and  $(i-1)$  LLRs from the CC decoder; this LLR is thus distributed as  $\mathcal{N}(V(\mu_{\text{vlc},i} + \mu_{\text{ch}} + (i-1)\mu_{\text{cc}}), 2(\mu_{\text{vlc},i} + \mu_{\text{ch}} + (i-1)\mu_{\text{cc}}))$  with  $\mu_{\text{vlc},i} = J_0^{-1}(I_{\text{vlc}}(I_{\text{vlc},i}^{\text{a}}))$ . Finally the EXIT characteristic of the VLC-RC is given by

$$I_{\text{vlc-rc}}^{\text{e}} = \sum_i g_i J_0(\mu_{\text{vlc},i} + \mu_{\text{ch}} + (i-1)\mu_{\text{cc}}). \quad (3.54)$$

This relation is different from (3.44), except when the RC is regular ( $f_i = g_i = 1$  for some  $i$ ), which corresponds to the intuition.

## Appendix 3.B Proofs of the theoretical results

### 3.B.1 Proof of Corollary 3.2

**Proof.** If we neglect the systematic branch in Fig. 3.1, we have a serial concatenation between the outer VLC/RC and the inner CC. The gains (3.14), (3.16) follow therefore from (3.10), (3.12) with  $d_f^{\text{o}} = d_f^{\text{vlc}} d_f^{\text{cc}}$ . The systematic branch

does not alter these gains — though this branch might be necessary in some setups, notably to ensure the global code is invertible, e.g., when the puncturing rate  $r_{pp}$  is larger than  $(r_{rc} r_{cc} r_s)^{-1}$  in (3.2). At last, (3.13) and (3.15) correspond to a CC alone. ■

### 3.B.2 Proof of Theorem 3.3

**Proof.** If  $d_f^{rc} = 1$ , the interleaving gains are given by Corollary 3.2. Let us assume  $d_f^{rc} \geq 2$ . By using the tools developed in [16], we find that one possible set of exponents  $\alpha$  to upper bound  $A_{w,s,h}$  in (3.9) is

$$\left\{ \alpha \in \mathbb{Z} : \alpha = n^0 + \sum_{i=1}^{d_f^{rc}} n^i - d_f^{rc} w ; n^0, n^i, w \in \mathbb{N}_{>0} \right\}, \quad (3.55)$$

where  $w, n^0$  and  $n^i$  are subject to several constraints, notably

$$w \geq d_f^{vlc}, \quad \text{by definition}^{17} \text{ of } d_f^{vlc}, \quad (3.56)$$

$$w \geq 2, \quad \text{by recursiveness}^{17} \text{ of the CC}, \quad (3.57)$$

$$n^0 \leq \lfloor w/d_f^{vlc} \rfloor, \quad \text{by definition}^{17} \text{ of } d_f^{vlc}, \quad (3.58)$$

$$n^i \leq \lfloor w/2 \rfloor, \quad \text{by recursiveness}^{17} \text{ of the CC}, \quad (3.59)$$

where  $w$  is the number of bit errors in the detected sequence of information bits,  $n^0$  is the number of error events (i.e., the number of times the detected path in the trellis diverges and merges back with the correct one) in the source/VLC trellis, and  $n^i$  is the number of error events in the portion of the CC trellis corresponding to the interleaved bits  $\Pi_i(U'_{1:N})$ . As explained in Section 3.4.1, we are interested only in the maximum of that set. To find an upper bound on that maximum, we can use (3.58)–(3.59), which gives us

$$\alpha \leq \left\lfloor \frac{w}{d_f^{vlc}} \right\rfloor + d_f^{rc} \left( \left\lfloor \frac{w}{2} \right\rfloor - w \right) = \left\lfloor \frac{w}{d_f^{vlc}} \right\rfloor - d_f^{rc} \left\lfloor \frac{w+1}{2} \right\rfloor. \quad (3.60)$$

If  $w = 2q$  is even for some integer  $q$ , then

$$\alpha \leq \left\lfloor \frac{2q}{d_f^{vlc}} \right\rfloor - d_f^{rc} q, \quad (3.61)$$

<sup>17</sup>An error event of a VLC is associated to at least  $d_f^{vlc}$  bit errors. An error event of a recursive CC is associated to at least 2 input bit errors.

which is maximized for small  $q$  since  $d_f^{rc} \geq 2$ . The smallest value of  $q$  can be deduced from (3.56)–(3.57),  $q \geq \lfloor (d_f^{vlc} + 1)/2 \rfloor$ , so

$$\alpha \leq \begin{cases} 1 - d_f^{rc} \lfloor (d_f^{vlc} + 1)/2 \rfloor, & \text{if } d_f^{vlc} \geq 2, \\ 2 - d_f^{rc}, & \text{if } d_f^{vlc} = 1. \end{cases} \quad (3.62)$$

If  $w = 2q - 1$  is odd, then

$$\alpha \leq \left\lfloor \frac{2q - 1}{d_f^{vlc}} \right\rfloor - d_f^{rc} q, \quad (3.63)$$

which is again maximized for small  $q$ . The smallest value of  $q$  can be deduced from (3.56)–(3.57),  $q \geq \max\{2; 1 + \lfloor d_f^{vlc}/2 \rfloor\}$ , so

$$\alpha \leq \begin{cases} 1 - d_f^{rc} \lfloor (d_f^{vlc} + 2)/2 \rfloor, & \text{if } d_f^{vlc} \geq 2, \\ 3 - 2d_f^{rc}, & \text{if } d_f^{vlc} = 1. \end{cases} \quad (3.64)$$

which is smaller than (3.62). So (3.62) is an upper bound on  $\alpha$  and, together with (3.10), it proves (3.18)–(3.21). ■

### 3.B.3 Proof of Theorem 3.4

**Proof.** Consider a single error event in the frame. Without loss of generality, we assume that the error event starts at the first source symbol  $S_1$ , i.e.,  $\check{S}_1 \neq S_1$  where  $\check{S}_1$  is the decision taken on  $S_1$ . If  $l(\check{S}_1) = l(S_1)$ , it is trivial: There is only one symbol error in the frame,  $\check{S}_1 \neq S_1$ , and at least one binary error. If  $l(\check{S}_1) \neq l(S_1)$ , several symbols are involved in the error event. Let  $m, n$  be the smallest integers such that  $l(\check{S}_{1:m}) = l(S_{1:n})$ , where  $\check{S}_{1:m}$  is the decision taken on the first  $m$  symbols. We have of course  $\check{S}_1 \neq S_1$  and  $\check{S}_m \neq S_n$ , so  $d_S(\check{S}, S) \geq d_L(\check{S}, S) \geq 2$ . Due to the prefix-free property, there is at least a bit difference between the prefixes of the codewords of  $\check{S}_1$  and  $S_1$ . Due to the suffix-free property, there is at least a second bit difference between the suffixes of the codewords of  $\check{S}_m$  and  $S_n$ . ■

### 3.B.4 Proof of Theorem 3.5

**Proof.** We start with the proof of (3.22). As in (3.55), one possible set of exponents  $\alpha$  to upper bound  $A_{w,s,h}$  in (3.9) is

$$\left\{ \alpha \in \mathbb{Z} : \alpha = n^0 + n^i - d_f^{rc} w ; n^0, n^i, w \in \mathbb{N}_{>0} \right\}. \quad (3.65)$$

where  $n^i$  is the number of error events in the CC trellis and is subject to  $n^i \leq \lfloor d_f^{\text{rc}} w / 2 \rfloor$  by recursiveness of the CC encoder. Let us split  $w$  into  $w = w_1 + w_2$ , where  $w_1$  is the number of bit errors associated with single-bit error events in source/VLC trellis and where  $w_2$  is consequently the number of bit errors associated with error events involving more than 2 bit errors each in the source/VLC trellis; by definition,  $w_2 \neq 1$ . Then, given  $w_1$  and  $w_2$ , we have this constraint on  $n^o$ :  $n^o \leq w_1 + \lfloor w_2 / 2 \rfloor$ , which is a generalization of (3.58). Finally, by Theorem 3.4, the number of symbol errors is proportional to the number of Levenshtein symbol errors when  $w_2 = 0$ ; by using (3.10) and (3.11), with some abuse of notation,

$$\begin{aligned} \alpha_{\text{ser}} &\leq \begin{cases} \alpha_{\text{ser}_L}, & \text{if } w_2 = 0, \\ \alpha_{\text{fer}}, & \text{otherwise,} \end{cases} \\ &= \alpha_M - \mathbb{I}\{w_2 = 0\}. \end{aligned} \quad (3.66)$$

Therefore,

$$\begin{aligned} \alpha_{\text{ser}} &\leq w_1 + \left\lfloor \frac{w_2}{2} \right\rfloor + \left\lfloor \frac{d_f^{\text{rc}}(w_1 + w_2)}{2} \right\rfloor - d_f^{\text{rc}}(w_1 + w_2) - \mathbb{I}\{w_2 = 0\} \\ &= w_1 + \left\lfloor \frac{w_2}{2} \right\rfloor - \left\lfloor \frac{d_f^{\text{rc}}(w_1 + w_2) + 1}{2} \right\rfloor - \mathbb{I}\{w_2 = 0\}. \end{aligned} \quad (3.67)$$

To bound this expression, we can do a case-by-case analysis.

For  $w_1$  even,  $w_2$  even (then  $w_1 + w_2 \geq 2$ ),

$$\alpha_{\text{ser}} \leq w_1 + \frac{w_2}{2} - d_f^{\text{rc}} \frac{w_1 + w_2}{2} - \mathbb{I}\{w_2 = 0\} \quad (3.68)$$

$$= \frac{w_1}{2}(2 - d_f^{\text{rc}}) + \frac{w_2}{2}(1 - d_f^{\text{rc}}) - \mathbb{I}\{w_2 = 0\}, \quad (3.69)$$

which is maximized for small  $w_1$  and  $w_2$  since  $d_f^{\text{rc}} \geq 2$ , e.g.,  $w_1 = 2$  and  $w_2 = 0$ ,

$$\alpha_{\text{ser}} \leq 1 - d_f^{\text{rc}}. \quad (3.70)$$

For  $w_1$  odd,  $w_2$  odd (then  $w_1 \geq 1$ ,  $w_2 \geq 3$ ),

$$\alpha_{\text{ser}} \leq w_1 + \frac{w_2 - 1}{2} - d_f^{\text{rc}} \frac{w_1 + w_2}{2} \quad (3.71)$$

$$= \frac{w_1}{2}(2 - d_f^{\text{rc}}) + \frac{w_2}{2}(1 - d_f^{\text{rc}}) - \frac{1}{2}, \quad (3.72)$$

which is again maximized for small  $w_1$  and  $w_2$ , e.g.,  $w_1 = 1$  and  $w_2 = 3$ ,

$$\alpha_{\text{ser}} \leq 2 - 2d_f^{\text{rc}}. \quad (3.73)$$

For  $w_1$  even,  $w_2$  odd (then  $w_2 \geq 3$ ):

$$\alpha_{\text{ser}} \leq w_1 + \frac{w_2 - 1}{2} - \left\lfloor \frac{d_f^{\text{rc}}(w_1 + w_2) + 1}{2} \right\rfloor \quad (3.74)$$

$$= -\frac{1}{2} + \frac{2w_1 + w_2}{2} - \left\lfloor \frac{d_f^{\text{rc}}(w_1 + w_2) + 1}{2} \right\rfloor, \quad (3.75)$$

which is maximized for small  $w_1$  and  $w_2$ , e.g.,  $w_1 = 0$  and  $w_2 = 3$ ,

$$\alpha_{\text{ser}} \leq 1 - \left\lfloor \frac{3d_f^{\text{rc}} + 1}{2} \right\rfloor. \quad (3.76)$$

For  $w_1$  odd,  $w_2$  even (then  $w_1 \geq 1$ ):

$$\alpha_{\text{ser}} \leq w_1 + \frac{w_2}{2} - \left\lfloor \frac{d_f^{\text{rc}}(w_1 + w_2) + 1}{2} \right\rfloor - \mathbb{I}\{w_2 = 0\} \quad (3.77)$$

$$= \frac{2w_1 + w_2}{2} - \left\lfloor \frac{d_f^{\text{rc}}(w_1 + w_2) + 1}{2} \right\rfloor - \mathbb{I}\{w_2 = 0\}, \quad (3.78)$$

which is maximized for small  $w_1$  and  $w_2$ , e.g.,  $w_1 = 1$  and  $w_2 = 0$ ,

$$\alpha_{\text{ser}} \leq - \left\lfloor \frac{d_f^{\text{rc}} + 1}{2} \right\rfloor. \quad (3.79)$$

All cases are upper bounded by (3.22).

The proof of (3.23) can be done similarly. Starting with (3.55) and (3.66) under (3.56)–(3.59) and using again  $w = w_1 + w_2$  and  $n^o \leq w_1 + \lfloor w_2/2 \rfloor$ , we have

$$\alpha_{\text{ser}} \leq w_1 + \left\lfloor \frac{w_2}{2} \right\rfloor - d_f^{\text{rc}} \left\lfloor \frac{w_1 + w_2 + 1}{2} \right\rfloor - \mathbb{I}\{w_2 = 0\}.$$

We do the same case-by-case analysis as previously.

For  $w_1$  even,  $w_2$  even (then  $w_1 + w_2 \geq 2$ ),

$$\alpha_{\text{ser}} \leq w_1 + \frac{w_2}{2} - d_f^{\text{rc}} \frac{w_1 + w_2}{2} - \mathbb{I}\{w_2 = 0\} \quad (3.80)$$

$$= \frac{w_1}{2} (2 - d_f^{\text{rc}}) + \frac{w_2}{2} (1 - d_f^{\text{rc}}) - \mathbb{I}\{w_2 = 0\}, \quad (3.81)$$

which is maximized for small  $w_1$  and  $w_2$  since  $d_f^{\text{rc}} \geq 2$ , e.g.,  $w_1 = 2$  and  $w_2 = 0$ ,

$$\alpha_{\text{ser}} \leq 1 - d_f^{\text{rc}}. \quad (3.82)$$

For  $w_1$  odd,  $w_2$  odd (then  $w_1 \geq 1$  and  $w_2 \geq 3$ ),

$$\alpha_{\text{ser}} \leq w_1 + \frac{w_2 - 1}{2} - d_f^{\text{rc}} \frac{w_1 + w_2}{2} \quad (3.83)$$

$$= \frac{w_1}{2}(2 - d_f^{\text{rc}}) + \frac{w_2}{2}(1 - d_f^{\text{rc}}) - \frac{1}{2}, \quad (3.84)$$

which is again maximized for small  $w_1$  and  $w_2$ , e.g.,  $w_1 = 1$  and  $w_2 = 3$ ,

$$\alpha_{\text{ser}} \leq 2 - 2d_f^{\text{rc}}. \quad (3.85)$$

For  $w_1$  even,  $w_2$  odd (then  $w_2 \geq 3$ ),

$$\alpha_{\text{ser}} \leq w_1 + \frac{w_2 - 1}{2} - d_f^{\text{rc}} \frac{w_1 + w_2 + 1}{2} \quad (3.86)$$

$$= \frac{w_1}{2}(2 - d_f^{\text{rc}}) + \frac{w_2}{2}(1 - d_f^{\text{rc}}) - \frac{1}{2}(1 + d_f^{\text{rc}}), \quad (3.87)$$

which is maximized for small  $w_1$  and  $w_2$ , e.g.,  $w_1 = 0$  and  $w_2 = 3$ ,

$$\alpha_{\text{ser}} \leq 1 - 2d_f^{\text{rc}}. \quad (3.88)$$

For  $w_1$  odd,  $w_2$  even (then  $w_1 \geq 1$ , and  $w_1 + w_2 \geq 2$  by (3.57)),

$$\alpha_{\text{ser}} \leq w_1 + \frac{w_2}{2} - d_f^{\text{rc}} \frac{w_1 + w_2 + 1}{2} - \mathbb{I}\{w_2 = 0\} \quad (3.89)$$

$$= \frac{w_1}{2}(2 - d_f^{\text{rc}}) + \frac{w_2}{2}(1 - d_f^{\text{rc}}) - \frac{d_f^{\text{rc}}}{2} - \mathbb{I}\{w_2 = 0\}, \quad (3.90)$$

which is maximized for small  $w_1$  and  $w_2$ , e.g.,  $w_1 = 3$  and  $w_2 = 0$ ,

$$\alpha_{\text{ser}} \leq 2 - 2d_f^{\text{rc}}. \quad (3.91)$$

All cases are upper bounded by (3.23). ■

### 3.B.5 Proof of Theorem 3.7

**Proof.** Eq. (3.29) and (3.30) are corollaries of [96, eq. (35)]. Eq. (3.31) can be proved as follows, similarly as the proof of Theorem 3.5.

As in (3.55), one possible set of exponents  $\alpha$  to upper bound  $A_{w,s,h}$  in (3.9) is

$$\left\{ \alpha \in \mathbb{Z} : \alpha = n^o + n^m + n^i - w - d^m; n^o, n^m, n^i, w, d^m \in \mathbb{N}_{>0} \right\}. \quad (3.92)$$

where  $w$  is the number of bit errors in the decoded sequence of VLC bits  $U$ ,  $d^m$  is the number of bit errors in the decoded sequence of coded bits  $V$  of  $CC^m$ , and  $n^i$ ,  $n^m$  and  $n^o$  are the numbers of error events in the trellises of  $CC^i$ , of  $CC^m$  and of the VLC. They are subject to several constraints, notably

$$w \geq d_f^{\text{vlc}}, \quad \text{by definition of } d_f^{\text{vlc}}, \quad (3.93)$$

$$d^m \geq d_f^{\text{cc,m}}, \quad \text{by definition of } d_f^{\text{cc,m}}, \quad (3.94)$$

$$n^o \leq \lfloor w/d_f^{\text{vlc}} \rfloor, \quad \text{by definition of } d_f^{\text{vlc}}, \quad (3.95)$$

$$n^m \leq \lfloor d^m/d_f^{\text{cc,m}} \rfloor, \quad \text{by definition of } d_f^{\text{cc,m}}, \quad (3.96)$$

$$n^i \leq \lfloor d^m/2 \rfloor, \quad \text{by recursiveness of } CC^i, \quad (3.97)$$

As in the proof of Theorem 3.5, we split  $w$  into  $w = w_1 + w_2$ , where  $w_1$  is the number of bit errors generated by single-bit error events in the VLC trellis and where  $w_2$  is consequently the number of bit errors generated by error events involving more than 2 bit errors each; by definition,  $w_2 \neq 1$ . Then, the constraint (3.95) becomes  $n^o \leq w_1 + \lfloor w_2/2 \rfloor$ . Finally, together with (3.66), we have

$$\alpha_{\text{ser}} \leq w_1 + \left\lfloor \frac{w_2}{2} \right\rfloor + \left\lfloor \frac{d^m}{d_f^{\text{cc,m}}} \right\rfloor + \left\lfloor \frac{d^m}{2} \right\rfloor - (w_1 + w_2) - d^m - \mathbb{I}\{w_2 = 0\} \quad (3.98)$$

$$= - \left\lfloor \frac{w_2 + 1}{2} \right\rfloor - \mathbb{I}\{w_2 = 0\} + \left\lfloor \frac{d^m}{d_f^{\text{cc,m}}} \right\rfloor + \left\lfloor \frac{d^m}{2} \right\rfloor - d^m. \quad (3.99)$$

The first two terms are trivially upper bounded by  $-1$ . The last three terms are maximized for small  $d^m$  since  $d_f^{\text{cc,m}} \geq 2$ . The smallest value of  $d^m$  follows from (3.94), and therefore

$$\alpha_{\text{ser}} \leq -d_f^{\text{cc,m}} + \left\lfloor \frac{d_f^{\text{cc,m}}}{2} \right\rfloor = - \left\lfloor \frac{d_f^{\text{cc,m}} + 1}{2} \right\rfloor. \quad (3.100)$$

■



# Non-Uniform Binary Sources and EXIT charts



*The content of this chapter has not been published yet and will be submitted soon as a journal paper [13].*

As we have seen in Chapter 2, to assess the convergence of turbo decoders, we can use the extrinsic information transfer (EXIT) charts which have been introduced in [12] for uniform bits. Unfortunately, numerous applications deal in practice with non-uniform binary sources. With such sources, several issues arise and a naive application of the EXIT charts might lead to incorrect results.

This chapter clarifies these issues and extends the EXIT charts to non-uniform binary sources. Two methods to compute the charts are proposed. The first one is a generalization of the original EXIT charts to non-uniform bits. The second one uses a pseudo-random bit flipping to make the bits virtually uniform. These methods provide different but equivalent EXIT charts, under some assumptions. The equivalence as well as advantages and limitations of each method are discussed. An illustrative example is given with a basic (non-optimized) transmission system.

This chapter completes the EXIT chart analysis made in Chapter 3.

## 4.1 Introduction

Consider the generic system in Fig. 4.1. It involves a serial concatenation at the transmitter and a serial turbo decoder [16] at the receiver. Despite its

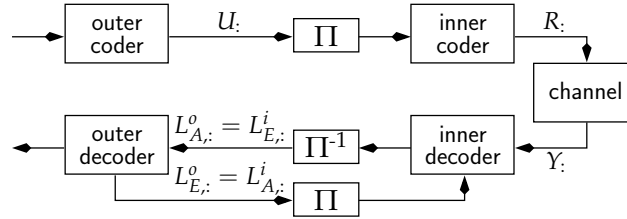
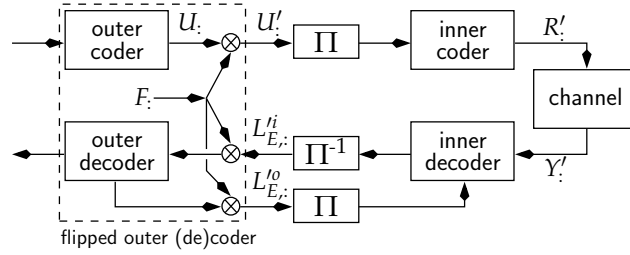


Figure 4.1 System under consideration.

simplicity, this system is sufficiently general to describe several issues that arise with EXIT charts [12] when the bits  $U_i$  are not uniform.

In the following, let us consider that the outer code in Fig. 4.1 produces a sequence of  $N$  bits  $U_{1:N}$ , or  $U_i$ , that take values in  $\{+1, -1\}$ , not necessarily equally likely. We say that the bits  $U_i$  are *biased* or *non-uniform*. Note that the outer code can include an error correcting code in addition to an entropy code and a source of data. The sequence of  $U_i$  is pseudo-randomly interleaved by  $\Pi$ . From the interleaved sequence, the inner code produces a sequence of coded bits  $R_i$  which are sent across the channel. We assume the inner code is linear and the channel is binary, symmetric, memoryless and time invariant. At the receiver, an iterative decoder is used. It is based on two decoders, one for each code. They exchange extrinsic log-likelihood ratios (LLRs) iteratively, in a typical serial configuration [16], see Sections 2.2, 2.4 and 3.5.1. In the following, let  $L_{E_i}^o$  be the extrinsic LLRs of the outer decoder, let  $L_{E_i}^i$  be the deinterleaved extrinsic LLRs of the inner decoder, and let  $L_{A_i}^i = L_{E_i}^o$  and  $L_{A_i}^o = L_{E_i}^i$  be the corresponding a priori LLRs.

To assess the convergence of this iterative decoder, one very useful tool is the EXIT chart introduced in [12] for uniform bits, see Sections 2.4.3 and 3.5. However, when the bits are not uniform, a naive application of this tool that would neglect the bias might lead to incorrect results. If the bias cannot be neglected, the computation of the EXIT charts raises a few issues that are clarified in this chapter. To the best of our knowledge, the first extension of the EXIT charts to non-uniform bits has been developed in [97], with a biased memoryless binary source and an irregular turbo code, in the context of distributed source coding.



**Figure 4.2** Equivalent system as Fig. 4.1 with a pseudo-random bit flipping,  $U_k, U'_k, F_k \in \{+1, -1\}$ . As in Fig. 4.1, we define  $L^o_{A,:} = L^i_{E,:}$  and  $L^o_{E,:} = L^i_{A,:}$ .

In this chapter, we provide two methods to analyze the system in Fig. 4.1 by EXIT charts. The first method in Section 4.2.2 is an extension of [12] to non-uniform bits, and is a generalization of [97]. Unfortunately, the resulting EXIT chart of the inner code depends on the bias of the outer code and must be recomputed if the bias changes. To circumvent this drawback, the second method in Section 4.2.3 introduces a pseudo-random bit flipping in the system that makes the bits virtually uniform, see Fig. 4.2. Both methods lead actually to different but equivalent EXIT charts, under some assumptions, as it is shown by semi-analytical transformations in Section 4.3.1. Advantages and limitations of each method are discussed in Section 4.3.3. A comparison with previous methods for uniform bits is also given and leads to a surprising and interesting result: Though the methods in [94, 98, 99] were developed for uniform bits and neglect the source bias, we prove that they are equivalent to the method proposed in Section 4.2.3 with the bit flipping and can give therefore a correct prediction of convergence with non-uniform bits.

In the remainder, random variables are written with capital letters and realizations with small letters.  $P(z)$  is the abbreviation of the probability  $P(Z = z)$ . The sub-sequence  $(Z_m, Z_{m+1}, \dots, Z_n)$  is written  $Z_{m:n}$ , or  $Z$ ; when  $m, n$  can be omitted.  $\mathbb{I}\{a\}$  is the indicator function, i.e.,  $\mathbb{I}\{a\}$  equals 1 if  $a$  is true, 0 otherwise.  $\mathbf{E}\{Z\}$  is the expectation of  $Z$ .  $I(Y; Z) = \mathbf{E}\{SI(Y; Z)\}$  is the mutual information between  $Y$  and  $Z$ , where  $SI(y; z) = \log_2 \frac{p(y,z)}{p(y)p(z)}$  is the point-wise mutual information.  $H(Z) = I(Z; Z)$  is the entropy of  $Z$ .  $H_b(p)$  is the binary entropy function, i.e.,  $H_b(p) = -p \log_2(p) - (1-p) \log_2(1-p)$ .

## 4.2 Computation of the EXIT charts

This section presents two methods to compute the charts. For convenience, the former method which is based on the original system in Fig. 4.1 and thus on biased bits, will be referred to as the *BEXIT method* and computes *BEXIT charts*. The latter, which is based on the flipped system in Fig. 4.2 and thus on flipped bits, will be referred to as the *FEXIT method* and computes *FEXIT charts*.

The main characteristic equations of these methods are summarized for clarity in Table 4.1 and in Table 4.2.

For the sake of conciseness, some familiarity with the results from Section 2.4.3 and with the original contribution [12] on EXIT charts is assumed hereafter.

### 4.2.1 Assumptions, notations and consistency

This subsection summarizes several assumptions, on which the BEXIT and FEXIT methods rely, and several properties characterizing the LLRs.

We assume that the inner code is linear and the channel is binary, symmetric, memoryless and time invariant. This assumption is important to make the original system in Fig. 4.1 and the flipped system in Fig. 4.2 equivalent.

Besides, we assume that the inner and outer decoders, taken apart, are optimal. Specifically, let  $\mathcal{R}_k^i = (Y, L_{A,1:k-1}^i, L_{A,k+1:N}^i)$  and  $\mathcal{R}_k^o = (L_{A,1:k-1}^o, L_{A,k+1:N}^o)$  and assume that the elements in  $\mathcal{R}_k^i$  and in  $\mathcal{R}_k^o$  are independent. Then the inner and outer extrinsic LLRs on  $U_k$ , produced by the inner and outer decoders, respectively, are assumed to satisfy the optimal expressions

$$L_{E,k}^i = \log \frac{p(\mathcal{R}_k^i | U_k = +1)}{p(\mathcal{R}_k^i | U_k = -1)}, \quad (4.1)$$

$$L_{E,k}^o = \log \frac{p(\mathcal{R}_k^o | U_k = +1)}{p(\mathcal{R}_k^o | U_k = -1)} = \log \frac{p(\mathcal{R}_k^o | U_k = +1)}{p(\mathcal{R}_k^o | U_k = -1)} + L_{U,k}, \quad (4.2)$$

where the source bias  $L_{U,k}$  is defined as

$$L_{U,k} \triangleq \log \frac{P(U_k = +1)}{P(U_k = -1)}, \text{ with } 0 < P(U_k = +1) < 1. \quad (4.3)$$

In (4.1),  $p(\mathcal{R}_k^i|U_k)$  is the likelihood of  $\mathcal{R}_k^i$  given  $U_k$  according to the structure of the inner code only, as if there was no outer code. Similarly,  $p(\mathcal{R}_k^o|U_k)$  in (4.2) is related only to the structure of the outer code, as if there was no inner code. This explains the difference between (4.1) and (4.2): Since the source model is included only in the outer code, only the outer decoder knows the a priori information on  $U$ ; and thus the source bias in (4.2).

When the decoders are optimal, i.e., when (4.1) and (4.2) are satisfied, the probability density functions of the extrinsic LLRs  $L_{E,k}^i$  and  $L_{E,k}^o$  satisfy some symmetry condition. To characterize this symmetry, the concepts of L-consistency and P-consistency are now introduced. In a sense, these concepts are generalizations of the consistency/symmetry conditions introduced in [98,100,101].

**Definition 4.1** (P-consistency and L-consistency).

$L_P$  is posterior-consistent or P-consistent with  $U$  if

$$p(L_P = l, U = +1) = e^l p(L_P = l, U = -1). \quad (4.4)$$

$L_L$  is likelihood-consistent or L-consistent with  $U$  if

$$p(L_L = l|U = +1) = e^l p(L_L = l|U = -1). \quad (4.5)$$

□

Note that if  $L_L$  is L-consistent, then  $L_L + L_U$  is P-consistent.

**Proposition 4.2.** The extrinsic LLR of the inner decoder,  $L_{E,k}^i$  in (4.1), is L-consistent with  $U_k$ . The extrinsic LLR of the outer decoder,  $L_{E,k}^o$  in (4.2), is P-consistent with  $U_k$ . □

**Sketch of Proof.** The likelihoods  $p(L_{E,k}^i = l|U_k = +1)$  and  $p(L_{E,k}^i = l|U_k = -1)$  are the integrals of  $p(\mathcal{R}_k^i|U_k = +1)$  and  $p(\mathcal{R}_k^i|U_k = -1)$ , respectively, over the set of  $\mathcal{R}_k^i$  that lead to  $L_{E,k}^i = l$ . By using the definition of  $L_{E,k}^i$  in (4.1), we obtain immediately the L-consistency (4.5). The P-consistency of  $L_{E,k}^o$  can be proved similarly. ■

Note the L-consistency is equivalent to the “exponential symmetry” [100, eq. (9)] but is slightly different from the consistency in [98, 101], where the authors introduce both a consistency condition  $p(L = l|U = +1) = e^l p(L = -l|U = +1)$  and a symmetry condition  $p(L = -l|U = +1) =$

$p(L = l|U = -1)$ . When this symmetry condition is satisfied, the consistency in [98, 101] is straightforwardly equivalent to the L-consistency (4.5). Here, however, the symmetry is not necessarily satisfied, owing to the non-linearity of the outer code, and we have thus only the L- and P-consistencies formalized in Definition 4.1 and in Proposition 4.2.

Let us assume that  $L_U = L_{U,k}$  is independent of  $k$ , i.e., the bits  $U_k$  have the same entropy  $H_U = H_{U,k} \triangleq H_b(P(U_k = +1))$  independently of  $k$ . Let us then measure the a priori and extrinsic levels of information as

$$I_A = \lim_{N \rightarrow +\infty} \frac{1}{N} \sum_{k=1}^N I(U_k; L_{A,k}) \in [0, H_U], \quad (4.6)$$

$$I_E = \lim_{N \rightarrow +\infty} \frac{1}{N} \sum_{k=1}^N I(U_k; L_{E,k}) \in [0, H_U], \quad (4.7)$$

where  $(I_A, L_{A,k})$  is either  $(I_A^i, L_{A,k}^i)$  or  $(I_A^o, L_{A,k}^o)$ , and  $(I_E, L_{E,k})$  is either  $(I_E^i, L_{E,k}^i)$  or  $(I_E^o, L_{E,k}^o)$ .

Given statistical models for  $L_A^i$  and for  $L_A^o$ , we can characterize the inner and outer decoders by single-parameter information transfer functions, namely their *EXIT charts* [12], respectively  $I_E^i = T_i(I_A^i)$  and  $I_E^o = T_o(I_A^o)$ , see the introduction given in Section 2.4.3. As in [12], we will compute these functions by feeding the decoders with a priori LLRs subject to the chosen statistical model, for different levels of a priori information, and by measuring at the decoder output the corresponding levels of extrinsic information. This is emphasized in the following remark.

**Remark 4.3.** *The inner (resp. outer) decoder will be fed with a priori LLRs that are independent, P-consistent (resp. L-consistent) and Gaussian.*

Note though that the real LLRs in Fig. 4.1 have a distribution that can be asymmetric, far from Gaussian, and heavily dependent on the (non-linear) outer code. For example, variable length codes can lead to non-smooth distributions that contain several high peaks. EXIT charts, therefore, can only be used as an approximate prediction/analysis tool. Still, this tool is quite robust in practice, see [12].

We assume furthermore that the process  $(U_k, L_{E,k})$  — both  $(U_k, L_{E,k}^i)$  and  $(U_k, L_{E,k}^o)$  — satisfies

$$I_E = \lim_{N \rightarrow +\infty} \frac{1}{N} \sum_{k=1}^N SI(u_k; l_{E,k}). \quad (4.8)$$

This assumption is closely related to ergodicity and typicality. For example, it is satisfied when we have an ergodic process or a mixture of ergodic processes (see mixture of codes in [94, 98]) whose fractions are independent of  $N$ . Loosely speaking, we assume in (4.8) that the most probable (typical) realizations of  $L_{E,1:N}$  carry the same amount of extrinsic information when  $N$  becomes large. Note that this assumption is important for convergence prediction purposes: When it is satisfied, the most probable snapshot trajectories [12] match the EXIT charts for large  $N$  and can be therefore predicted by the EXIT charts.

Remark 4.3 asks for one last comment. The *requirement* for the P- and L-consistencies, which is in agreement with Proposition 4.2, is related notably to the reliability of the a priori information. Indeed, when the LLRs are consistent, their absolute values are a measure of reliability, see Corollary 4.5 hereafter. Without consistency, on the contrary, the link with the reliability is broken. Furthermore, when  $L_A^i$  is P-consistent, the inner decoder can extract the likelihoods  $p(L_A^i, U = +1)$  and  $p(L_A^i, U = -1)$  up to a normalizing factor from (4.4), given the value of  $L_A^i$ . And these likelihoods are precisely needed when the inner decoder factorizes and evaluates (4.1). Similarly, when  $L_A^o$  is L-consistent, the outer decoder can extract  $p(L_A^o|U = +1)$  and  $p(L_A^o|U = -1)$  from (4.5), which are needed to evaluate (4.2). Without consistency, the decoders cannot extract these likelihoods correctly.

**Proposition 4.4.** *If  $L_P$  is P-consistent with  $U$ , then*

$$P(U = u|L_P = l) = 1/(1 + e^{-ul}). \quad (4.9)$$

*If  $L_L$  is L-consistent with  $U$ , then*

$$P(U = u|L_L = l) = 1/(1 + e^{-u(Lu+l)}). \quad (4.10)$$

□

**Proof.** From the P-consistency (4.4), it follows that

$$P(U = +1|L_P = l) = e^l P(U = -1|L_P = l). \quad (4.11)$$

Since  $P(U = +1|L_P = l) + P(U = -1|L_P = l) = 1$ , we obtain  $P(U = +1|L_P = l) = 1/(1 + e^{-l})$  and  $P(U = -1|L_P = l) = 1/(1 + e^l)$ , which proves (4.9). Let  $L = L_U + L_L$ . Eq. (4.10) follows from the application of (4.9) to  $L$ , by noting  $L$  is P-consistent and  $P(U = u|L_L = l) = P(U = u|L = L_U + l)$ . ■

We have straightforwardly the following corollary, which is an extension of [100].

**Corollary 4.5.** *For  $L$ -consistent  $L_L$  and P-consistent  $L_P$ , the bit error rates associated with the decisions  $\check{d}(L_L)$ ,  $\check{d}(L_L + L_U)$  and  $\check{d}(L_P)$ , are given by*

$$P(U \neq \check{d}(l)|L_L = l) = \frac{1}{1 + e^{|l| + \check{d}(l)L_U}}, \quad (4.12)$$

$$P(U \neq \check{d}(l + L_U)|L_L = l) = \frac{1}{1 + e^{|l + L_U|}}, \quad (4.13)$$

$$P(U \neq \check{d}(l)|L_P = l) = \frac{1}{1 + e^{|l|}}, \quad (4.14)$$

where  $\check{d}(l) = \mathbb{I}\{l > 0\} - \mathbb{I}\{l < 0\} + k\mathbb{I}\{l = 0\}$  and  $k \in \{+1, -1\}$  is the arbitrary<sup>1</sup> decision taken when  $l = 0$ . ■

**Proof.** Let us prove (4.12). Eq. (4.13) and (4.14) can be proved similarly. We have

$$\begin{aligned} P(U \neq \check{d}(l)|L_L = l) &= P(U = -\check{d}(l)|L_L = l) \\ &= \frac{1}{1 + e^{\check{d}(l)(l + L_U)}} = \frac{1}{1 + e^{|l| + \check{d}(l)L_U}}, \end{aligned}$$

where the second equality follows from Proposition 4.4. ■

## 4.2.2 BEXIT charts, Fig. 4.1: biased bits

Let us start with the description of the BEXIT method. The FEXIT method will then follow as a particular case.

Though analytical expressions of EXIT charts exist for some codes [53] and some models of a priori information, the EXIT charts are generally computed by means of Monte-Carlo simulations.

<sup>1</sup>Of course, the value  $k = \mathbb{I}\{L_U \geq 0\} - \mathbb{I}\{L_U < 0\}$  minimizes the bit error rate in the case of (4.12).

Considering the EXIT chart  $T(\cdot)$  of a linear code, the Monte-Carlo method in [12] consists in feeding the considered decoder with artificially generated a priori information  $I_A$  and in measuring afterward the extrinsic information  $I_E$  at the output of the decoder. Repeating this for different values of  $I_A$  provides several values of  $I_E$ , thus several points of the EXIT chart  $T(\cdot)$  since  $I_E = T(I_A)$ . In other words, the Monte-Carlo method provides a quantized version of the EXIT chart  $T(\cdot)$ .

The BEXIT method (and the FEXIT method afterward) follows this approach. Given any fixed level  $I_A$  of a priori information, Section 4.2.2.2 explains how to generate the a priori LLRs  $L_{A,k}$  to feed the input of the decoders, such that  $I(U_k; L_{A,k}) = I_A$  in (4.6). Next, Section 4.2.2.3 describes how to measure the extrinsic information  $I_E = I(U_k; L_{E,k})$  in (4.7) at the output of the decoders. An example of BEXIT charts will be given in Section 4.3.2.

#### 4.2.2.1 The invertible function $J_{L_U}(\cdot)$

The function  $J_{L_U}(\cdot)$  is an important function that is used hereafter to generate the a priori LLRs. It is an extension to biased bits of the function  $J(\cdot)$  given in [12, eq. (15)] — actually, when the bias  $L_U$  equals zero,  $J(\sqrt{2\mu}) = J_{L_U}(\mu)$  or equivalently  $J(\sigma) = J_{L_U}(\sigma^2/2)$  — and is defined as follows.

Given a bit  $U$ , let  $L_U = \log(P(U = +1)/P(U = -1))$  be the bias, let  $H_U = H_b(P(U = +1))$  be the entropy of  $U$ , and let  $L$  be the Gaussian-like variable

$$L = \mu U + N + k \quad (4.15)$$

where  $\mu$  and  $k$  are constants and  $N$  is a centered Gaussian random variable of variance  $2\mu$ ,  $N \sim \mathcal{N}(0, 2\mu)$ . Note the mutual information  $I(U; L)$  between  $U$  and  $L$  is independent of  $k$  (since  $k$  is an additive constant) and depends only on  $\mu$ . In other words,  $I(U; L)$  is a function of  $\mu$ . Let then  $J_{L_U}(\mu)$  be that function, i.e.,  $J_{L_U}(\mu) = I(U; L)$ .

Table 4.1 Summary of the two Monte-Carlo methods

	BEXIT charts: with biased bits, Fig. 4.1
Generation of the a priori LLRs.	<p>Given a level <math>I_A</math> of a priori information, let <math>\mu_L = J_{L_U}^{-1}(I_A)</math> and <math>N_L \sim \mathcal{N}(0, 2\mu_L)</math>. Then, the a priori LLRs can be generated as</p> $L_A^o = \mu_L U + N_L, \quad (\text{T-4.1-1})$ <p>for the outer code, and as</p> $L_A^i = \mu_L U + N_L + L_U, \quad (\text{T-4.1-2})$ <p>for the inner code. Note <math>L_A^o</math> is L-consistent and <math>L_A^i</math> is P-consistent.</p>
Measurement of the extrinsic information for consistent LLRs	<p>Given the extrinsic LLR <math>L_E</math>, the amount of extrinsic information on the bit <math>U</math> is given by</p> $I_E = H_U - \mathbf{E} \left\{ \log_2 \left( 1 + e^{-U \left( L_U + \log \frac{p(L_E U=+1)}{p(L_E U=-1)} \right)} \right) \right\} \quad (\text{T-4.1-3})$ <p>for both the inner code and the outer code, where the expectation can be evaluated by approaching <math>p(L_E U)</math> with histogram measurements as in [12]. Besides, since <math>L_E^o</math> is P-consistent and <math>L_E^i</math> is L-consistent, we have also</p> $I_E^o = H_U - \mathbf{E} \{ \log_2(1 + e^{-U L_E^o}) \}, \quad (\text{T-4.1-4})$ $= H_U - \mathbf{E} \{ \text{H}_b(1/(1 + e^{L_E^o})) \}, \quad (\text{T-4.1-5})$ <p>for the outer code, and</p> $I_E^i = H_U - \mathbf{E} \{ \log_2(1 + e^{-U(L_U + L_E^i)}) \}, \quad (\text{T-4.1-6})$ $= H_U - \mathbf{E} \{ \text{H}_b(1/(1 + e^{L_U + L_E^i})) \}, \quad (\text{T-4.1-7})$ <p>for the inner code, where the expectations can be approached by time averages as in [98].</p>
Link between BEXIT charts and FEXIT charts	$I_A^o \approx J_{L_U}(J_0^{-1}(I_A^o)), \quad I_E^i \approx J_{L_U}(J_0^{-1}(I_E^i)),$ $I_A^i = I_A^o - (1 - H_U), \quad I_E^o = I_E^i - (1 - H_U).$

**Table 4.2** (Table 4.1 continued) Summary of the two Monte-Carlo methods

	FEXIT charts: with flipped bits, Fig. 4.2
Generation of the a priori LLRs.	<p>Given a level <math>I'_A</math> of a priori information, let <math>\mu'_L = J_0^{-1}(I'_A)</math> and <math>N'_L \sim \mathcal{N}(0, 2\mu'_L)</math>. Then, the a priori LLRs can be generated as</p> $L'_A = \mu'_L U' + N'_L \quad (\text{T-4.2-1})$ <p>for both the inner code and the outer code.</p>
Measurement of the extrinsic information for consistent LLRs	<p>Given the extrinsic LLR <math>L'_E</math>, the amount of extrinsic information on the bit <math>U'</math> is given by</p> $I'_E = 1 - \mathbf{E} \left\{ \log_2 \left( 1 + e^{-U' \log \frac{p(L'_E U'=+1)}{p(L'_E U'=-1)}} \right) \right\} \quad (\text{T-4.2-2})$ <p>for both the inner code and the outer code, where the expectation can be evaluated by approaching <math>p(L'_E U')</math> with histogram measurements as in [12]. Besides,</p> $I'_E = 1 - \mathbf{E} \{ \log_2(1 + e^{-U' L'_E}) \}, \quad (\text{T-4.2-3})$ $= 1 - \mathbf{E} \{ \text{H}_b(1/(1 + e^{L'_E})) \}, \quad (\text{T-4.2-4})$ <p>where the expectations can be approached by time averages as in [98].</p>
Link between BEXIT charts and FEXIT charts	$I_A^o \approx J_0(J_{L_u}^{-1}(I_A^o)), \quad I_E^i \approx J_0(J_{L_u}^{-1}(I_E^i)), \quad (\text{T-4.2-5})$ $I_A^i = I_A^o + (1 - H_U), \quad I_E^o = I_E^i + (1 - H_U). \quad (\text{T-4.2-6})$

By developing analytically the expression of  $I(U;L)$ , we can define the function  $J_{L_U}$  more formally as the function  $J_{L_U} : [0, +\infty) \rightarrow [0, H_U]$ ,

$$J_{L_U}(\mu) = H_U - \sum_{u \in \{+1, -1\}} P(U = u) \times \int_{-\infty}^{+\infty} \frac{e^{-(\xi - \mu u)^2 / (4\mu)}}{\sqrt{4\pi\mu}} \log_2(1 + e^{-u(L_U + \xi)}) d\xi. \quad (4.16)$$

This function is strictly increasing and thus invertible.

#### 4.2.2.2 Generating the a priori LLRs

As mentioned here above, the first step of the BEXIT method is to generate a priori LLRs that correspond to a certain given level  $I_A$  of a priori information. To this end, let us consider the sequence of bits  $U$ : generated by the outer coder and let us focus on one of these bits, say  $U$ . Given a certain level  $I_A$  of a priori information, and recalling the expression (4.6) of the a priori information, we need thus to generate an a priori LLR  $L_A$  such that the mutual information between the bit  $U$  and this LLR equals  $I_A$ , i.e., such that  $I(U;L_A) = I_A$ .

For the outer decoder, the a priori LLR  $L_A^o$  is generated using a Gaussian-like model as in [12],

$$L_A^o = \mu_L U + N_L + k \quad (4.17)$$

where  $\mu_L$  and  $k$  are constants, and  $N_L$  is a centered Gaussian random variable of variance  $2\mu_L$ . By Remark 4.3,  $L_A^o$  must be L-consistent. It is straightforward to show that this a priori LLR  $L_A^o$  is L-consistent for any positive constant  $\mu_L$  if  $k = 0$ . At last, the constant  $\mu_L$  is determined by the equality  $I(U;L_A^o) = I_A$ ; thus, by definition of the function  $J_{L_U}(\cdot)$  in (4.16),  $\mu_L = J_{L_U}^{-1}(I_A)$ . This is summarized in (T-4.1-1) in Table 4.1.

Similarly, for the inner decoder, the a priori LLR  $L_A^i$  is generated as  $L_A^i = \mu_L U + N_L + k$ . By Remark 4.3, the a priori LLR  $L_A^i$  must be P-consistent, which is satisfied for all positive  $\mu_L$  if  $k = L_U$ . The constant  $\mu_L$  then follows from the equality  $I(U;L_A^i) = I_A$ , i.e.,  $\mu_L = J_{L_U}^{-1}(I_A)$ . This is summarized in (T-4.1-2) in Table 4.1.

### 4.2.2.3 Measuring the extrinsic information

The second step of the BEXIT method is to measure the extrinsic information as  $I_E = I(U_k; L_{E,k})$  in (4.7) at the output of the decoders. To this end, let us consider the sequence of bits  $U_i$  and the corresponding sequence of extrinsic LLRs  $L_{E,i}$ . For both decoders, we can calculate  $I(U_k; L_{E,k})$  as

$$\begin{aligned} I(U_k; L_{E,k}) &= H_U - \mathbf{E} \left\{ \log_2 \frac{1}{P(U_k | L_{E,k})} \right\} \\ &= H_U - \mathbf{E} \left\{ \log_2 \left( 1 + e^{-U_k \left( L_U + \log \frac{p(L_{E,k} | U_k = +1)}{p(L_{E,k} | U_k = -1)} \right)} \right) \right\}. \end{aligned} \quad (4.18)$$

This expression can be evaluated by approaching the distribution  $p(L_{E,k} | U_k)$  with histogram measurements as in [12], with one histogram for each position  $k$ . Note that if there are groups of positions with the same distribution, the computation of the histograms is simpler since the same histogram can be used for all positions belonging to the same group.

By assuming the consistency (4.4)–(4.5) is satisfied, measuring the amount of extrinsic information becomes simpler. Indeed, let us introduce two random variables,  $L_E$  and  $U$ , whose joint distribution is

$$p(L_E = l, U = u) = \frac{1}{N} \sum_{k=1}^N p(L_{E,k} = l, U_k = u). \quad (4.19)$$

This joint distribution corresponds to the merging of all histograms into one common histogram. From the (P- or L-) consistency of all  $L_{E,k}$ , it follows that for all  $k$

$$\frac{p(L_E = l | U = +1)}{p(L_E = l | U = -1)} = \frac{p(L_{E,k} = l | U_k = +1)}{p(L_{E,k} = l | U_k = -1)}. \quad (4.20)$$

Consequently, (4.7) and (4.18) can be rewritten as

$$\begin{aligned} I_E &= H_U - \frac{1}{N} \sum_{k=1}^N \sum_u \int_{-\infty}^{+\infty} p(L_{E,k} = l, U_k = u) \\ &\quad \times \log_2 \left( 1 + e^{-u \left( L_U + \log \frac{p(L_{E,k} = l | U_k = +1)}{p(L_{E,k} = l | U_k = -1)} \right)} \right) dl \end{aligned} \quad (4.21)$$

$$\begin{aligned}
&= H_U - \sum_u \int_{-\infty}^{+\infty} \underbrace{\frac{1}{N} \sum_{k=1}^N p(L_{E,k} = l, U_k = u)}_{p(L_E=l, U=u)} \\
&\quad \times \log_2 \left( 1 + e^{-u \left( L_U + \log \frac{p(L_E=l|U=+1)}{p(L_E=l|U=-1)} \right)} \right) dl, \quad (4.22)
\end{aligned}$$

which simplifies into (T-4.1-3) in Table 4.1. It is simpler and requires only one histogram for  $p(L_E|U)$ . As a by-product, this result shows that the so-called “two-step” and “three-step” methods in [99] are equivalent when “LogAPP” decoders (i.e., optimal decoders, thus consistent) are used, even when  $p(L_{E,k}|U_k)$  depends on  $k$ .

We can further simplify (T-4.1-3) by using the consistency. We get immediately<sup>2</sup> (T-4.1-4) by P-consistency (4.4) for the outer decoder and (T-4.1-6) by L-consistency (4.5) for the inner decoder. Besides, we can simplify (T-4.1-4) into (T-4.1-5) since

$$\begin{aligned}
\mathbb{E} \left\{ \log_2(1 + e^{-UL_E^o}) | L_E^o \right\} &= \sum_{u \in \{+1, -1\}} P(U = u | L_E^o) \log_2(1 + e^{-uL_E^o}) \quad (4.23) \\
&= H_b(1/(1 + e^{L_E^o})) = H_b(1/(1 + e^{-L_E^o})),
\end{aligned}$$

where  $P(U = u | L_E^o)$  is given in (4.9). Similarly, we can simplify (T-4.1-6) into (T-4.1-7), with  $P(U = u | L_E^i)$  given in (4.10). Note that (T-4.1-4) is equivalent to [97, eq. (4)]; (T-4.1-4) and (T-4.1-6) are extensions of [98, eq. (4)]; and (T-4.1-5) and (T-4.1-7) are extensions of [99, Th. 7].

Finally, by assuming (4.8), the expectations in (T-4.1-4)–(T-4.1-7) can be approached as in [94, 98] by time averages for large  $N$ . For example, (T-4.1-7) can be evaluated as

$$I_E^i \approx H_U - \frac{1}{N} \sum_{k=1}^N H_b \left( 1/(1 + e^{L_U + I_{E,k}^i}) \right). \quad (4.24)$$

### 4.2.3 FEXIT charts, Fig. 4.2: flipped bits

Let us now consider the system in Fig. 4.2, where a pseudo-random bit flipping is introduced before the interleaver  $\Pi$  to make the bits  $U'_k$  uniform,

<sup>2</sup>Note that (T-4.1-4) and (T-4.1-6) follow also straightforwardly from (4.18) and Proposition 4.4.

i.e.,

$$U'_k = U_k F_k \quad \text{for all } k, \quad (4.25)$$

where the bits  $F_k \in \{+1, -1\}$  are independent and uniformly distributed. At the receiver, the bits  $F_k$  are known and the corresponding LLRs are flipped accordingly. By linearity of the inner code and symmetry of the channel, the flipped system in Fig. 4.2 is equivalent to the original system in Fig. 4.1.

Consequently, the EXIT charts of the flipped system can be used to characterize the original system. In the following, we refer to these charts as the *FEXIT charts* of the original system. For clarity, all symbols related to the FEXIT charts and to the flipped system use a prime ( $'$ ) notation.

With FEXIT charts, we are interested in the exchange of information about the flipped bits  $U'_k$  between the inner decoder and the flipped outer decoder that is highlighted by a dashed box in Fig. 4.2. Since these two decoders are still optimal (taken apart) under the assumptions of Section 4.2.1, and since  $P(U'_k = +1) = P(U'_k = -1) = 1/2$ , computing the FEXIT charts is a particular case of the BEXIT method with  $L_U = 0$ . In other words, we can use the results obtained so far by simply replacing  $L_U$  with 0 and  $H_U$  with 1. This is summarized in Table 4.2. Since the bits  $U'_k$  are uniform, the FEXIT charts are equivalent to the EXIT charts developed in [12].

## 4.3 Transformations, equivalence and discussion

### 4.3.1 Transformations and equivalence

Transformations to obtain the FEXIT chart from the BEXIT chart, and vice-versa, are given in (T-4.2-5), (T-4.2-6). These transformations demonstrate the equivalence between the FEXIT and BEXIT charts under the assumptions of Section 4.2.1.

To prove (T-4.2-5), let us consider  $L_A^o$  given in (T-4.1-1) in the biased case. If we apply the flipping  $F$  (see (4.25)) on  $L_A^o$ , we get  $L_A'^o = L_A^o F = \mu_L U F + N_L F = \mu_L U' + N_L F$ , and it is self-evident that this  $L_A'^o$  is equivalent to (T-4.2-1) if  $\mu_L = \mu_L'$ , i.e., if  $J_{L_U}^{-1}(I_A^o) = J_0^{-1}(I_A'^o)$ , which proves (T-4.2-5) for consistent Gaussian LLRs. For consistent non-Gaussian LLRs, we can invoke the empirical robustness of EXIT charts w.r.t. the a priori statistical model and assume

that (T-4.2-5) is a sufficient approximation, hence the approximation symbol “ $\approx$ ”. To prove (T-4.2-6), we use the equalities

$$I_E^o - H_U = -\mathbf{E}\{\log_2(1 + e^{-UL_E^o})\} \quad (\text{by (T-4.1-4)}) \quad (4.26)$$

$$= -\mathbf{E}\{\log_2(1 + e^{-U'L_E^o})\} \quad (4.27)$$

$$= I_E^o - 1, \quad (\text{by (T-4.2-3)}) \quad (4.28)$$

where (4.27) follows from  $U'L_E^o = (UF)(L_E^oF) = UL_E^o$ .

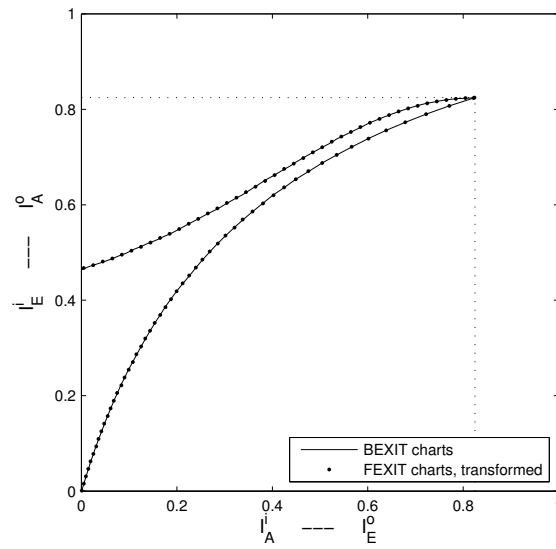
### 4.3.2 Simulation results

To illustrate the equivalence between the FEXIT and BEXIT charts, let us compute the charts of the following (non-optimized) system. The outer code is a memoryless source of 3 symbols with probabilities 0.85, 0.14, 0.01, encoded respectively by the variable length codewords  $(+1)$ ,  $(-1, -1)$ ,  $(-1, +1, -1)$ , leading to  $P(U = +1) = 0.741$  and  $H_U = 0.825$ . The inner code is a rate- $\frac{1}{2}$  recursive systematic convolutional encoder with forward generator  $35_8$  (in octal) and feedback generator  $23_8$ . The channel is an additive white Gaussian noise channel with binary phase-shift keying and  $E_b/N_0 = 1.4\text{dB}$ , where  $E_b$  is the energy per bit of entropy and  $N_0/2$  the double-sided noise spectral density.

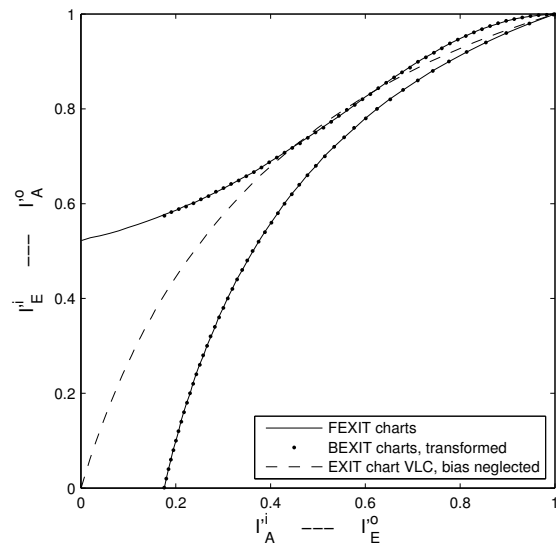
The BEXIT charts of the system are given in Fig. 4.3(a) and the FEXIT charts in Fig. 4.3(b). The solid lines show the charts obtained with the methods described in Sections 4.2.2, 4.2.3. By contrast, the data points in Fig. 4.3(a) show the BEXIT charts obtained from the FEXIT charts (given in solid lines in Fig. 4.3(b)) by applying the transformations (T-4.2-5)–(T-4.2-6). Conversely, the data points in Fig. 4.3(b) show the FEXIT charts obtained from the BEXIT charts (given in solid lines in Fig. 4.3(a)) by applying the transformations (T-4.2-5)–(T-4.2-6). As we can see, there is a good match between the data points and the solid lines, which illustrate the accuracy of the transformations and the equivalence between FEXIT and BEXIT charts.

### 4.3.3 Discussion

Consider the original (biased) system in Fig. 4.1. When the outer decoder does not receive any a priori information from the inner decoder, i.e., when  $I_A^o = 0$ , the outer decoder can only rely on its internal a priori knowledge of



(a) BEXIT charts



(b) FEXIT charts

Figure 4.3 BEXIT and FEXIT charts of the system described in Section 4.3.1.

the transmitted signal and outputs thus only the source bias  $L_E^o = L_U$ . Since this extrinsic LLR is constant, it does not contribute to the mutual information and  $I(U; L_E^o) = 0$ , hence  $I_E^o = 0$ . This explains why  $I_E^o = 0$  in Fig. 4.3(a) when  $I_A^o = 0$ .

By contrast, with the flipped system, though the flipped outer decoder produces also only the bias when  $I_A^o = 0$ , this bias is flipped, that is,  $L_E^o = L_U F$ . This can be rewritten as  $L_E^o = (L_U U) U'$ , which is equivalent to a binary symmetric channel (BSC) with inputs  $U'$ , outputs  $L_U U U'$ , crossover probability  $P(U = -1)$  and therefore capacity  $1 - H_U$ . This explains why  $I_E^o = 1 - H_U$  in Fig. 4.3(b) when  $I_A^o = 0$ . In this context, note that (T-4.2-6) implies the lower bound  $I_E^o = I_E^o + 1 - H_U \geq 1 - H_U$ .

In other words, while the outer decoder does not produce any information in the biased case when  $I_A^o = 0$  — as in typical serial concatenations —, it does (virtually) in the flipped case.

In terms of (dis)advantages of each method, FEXIT charts, unlike BEXIT charts, are restricted to linear inner codes and symmetric channels. Nevertheless, when the inner code is linear and the channel symmetric, FEXIT charts have at least two advantages over BEXIT charts.

Firstly, the FEXIT chart of the inner code is independent of the outer code and can be computed therefore independently of the outer code. In particular, it does not need to be recomputed when the outer code changes. By contrast, the BEXIT chart of the inner code depends on  $L_U$  (see (T-4.1-2), (T-4.1-6) and (T-4.1-7)), thus depends on the outer code and needs to be recomputed when  $L_U$  changes.

Secondly, FEXIT charts can handle very easily outer codes that have an entropy  $H_{U,k}$  varying with  $k$  — recall that we have assumed  $L_{U,k}$ , thus  $H_{U,k}$ , independent of  $k$  in Section 4.2.1. This is essentially because the random bit flipping makes the entropy equal to 1 for all  $k$ . By contrast, to the best of our knowledge, there is no straightforward way to address a varying entropy with BEXIT charts. Unfortunately, sources with a varying entropy  $H_{U,k}$  are not uncommon in practice. We can face such sources with, for example, fixed length codes, certain variable length codes, mixtures of outer codes with different entropies (e.g., mixture of different variable length codes), etc.

Finally, let us examine what happens when we neglect the bias and compute the EXIT charts of the (biased) system in Fig. 4.1 with the original methods of [12, 98, 99] as if the bits  $U_i$  were uniform. With the method of [12, eq. (19)], we obtain actually the FEXIT chart of the inner decoder for the inner decoder, and the chart given with a dashed line in Fig. 4.3(b) for the outer decoder. As we can see, these charts intersect each other and give therefore a prediction of convergence that is too pessimistic.

On the contrary, and interestingly, if we use either the method of [98] or the method of [99], we get actually the FEXIT charts of the system. Indeed, the methods of [98, eq. (5)] and [99, Th. 7] compute the extrinsic information respectively as (with some mathematical rewriting)

$$I_E^{[98]} = 1 - \mathbf{E}\{\log_2(1 + e^{-UL})\}, \quad (4.29)$$

$$I_E^{[99]} = 1 - \mathbf{E}\{H_b(1/(1 + e^L))\}. \quad (4.30)$$

Since  $U'L' = (UF)(LF) = UL$ , (4.29) is equivalent to (T-4.2-3). Since  $H_b(1/(1 + e^L)) = H_b(1/(1 + e^{-L})) = H_b(1/(1 + e^{LF}))$ , (4.30) is equivalent to (T-4.2-4). Let us emphasize this surprising result: Though the methods of [98] and [99] were developed for uniform bits, neglect the bias and do not use a random bit flipping, we have just proved that they give a correct prediction of convergence (when the inner code is linear and the channel symmetric) since they compute implicitly the FEXIT charts of the system.

#### 4.3.4 Approximations of $J_{LU}(\cdot)$

By making a few assumptions, we can develop two approximations of the function  $J_{LU}(\cdot)$  defined in (4.16):

$$J_{LU}(\mu) \approx J_0\left(\mu + J_0^{-1}(1 - H_U)\right) - (1 - H_U) \quad (4.31)$$

$$\approx J_0(\mu)H_U. \quad (4.32)$$

By inverting these expressions, we find approximations of  $J_{LU}^{-1}(\cdot)$ :

$$J_{LU}^{-1}(I) \approx J_0^{-1}(I + 1 - H_U) - J_0^{-1}(1 - H_U) \quad (4.33)$$

$$\approx J_0^{-1}(I/H_U). \quad (4.34)$$

These approximations are accurate for small to moderate biases. More precisely, the relative errors of (4.31) and of (4.32) on  $J_{L_U}(\mu)$  are smaller than 5% when  $P(U = +1) \in [.17, .83]$  and when  $P(U = +1) \in [.30, .70]$ , respectively. If we fix  $\mu = J_0^{-1}(0.5)$ , the relative errors are smaller than 5% when  $P(U = +1) \in [.09, .91]$  and when  $P(U = +1) \in [.19, .81]$ , respectively.

Note polynomial fitting combined with exponential fitting can provide more accurate results — see [102, Appendix] for an example with  $J_0(\cdot)$ .

The approximations (4.31)–(4.32) can be explained as follows. Given  $L_A^i$  in (T-4.1-2) and

$$L_A^i = L_A^i F = \mu_L U' + N_L F + L_U U U', \quad (4.35)$$

let us compute  $I(L_A^i, U')$  by three different ways. Firstly, from the transformation (T-4.2-6), we can compute exactly

$$I(L_A^i, U') = I(L_A^i, U) + 1 - H_U = J_{L_U}(\mu_L) + 1 - H_U. \quad (4.36)$$

Secondly, recall that the term  $L_U U U'$  in the expression of  $L_A^i$  is equivalent to a BSC of capacity  $1 - H_U$ . With some approximation, let us replace this term by an L-consistent Gaussian-like LLR carrying the same amount of information, i.e.,  $\mathcal{N}(\mu_L'' U', 2\mu_L'')$  where  $\mu_L'' = J_0^{-1}(1 - H_U)$ . We can then consider that we have approximately  $L_A^i \sim \mathcal{N}((\mu_L + \mu_L'') U', 2\mu_L + 2\mu_L'')$  and

$$I(L_A^i, U') \approx J_0(\mu_L + \mu_L'') = J_0(\mu_L + J_0^{-1}(1 - H_U)). \quad (4.37)$$

The comparison with (4.36) gives (4.31).

Thirdly, let us replace the first two terms in (4.35),  $\mu_L U' + N_L F$ , by an equivalent binary erasure channel (BEC) carrying an identical amount of mutual information, i.e.,

$$L_A^i = \beta U' + L_U U U', \quad (4.38)$$

where  $P(\beta = 0) = 1 - J_0(\mu_L)$  is the erasure probability and  $P(\beta = +\infty) = J_0(\mu_L)$  is the BEC capacity. We then have

$$I(L_A^i, U') \approx P(\beta = +\infty) + P(\beta = 0)(1 - H_U) \quad (4.39)$$

$$= J_0(\mu_L) + (1 - J_0(\mu_L))(1 - H_U). \quad (4.40)$$

The comparison with (4.36) leads to (4.32).

### 4.3.5 Comments on the EXIT chart and the turbo decoder

So far we have focused on the inner decoder and on the outer decoder separately, by assuming independent elements in  $\mathcal{R}_k^i$  or in  $\mathcal{R}_k^o$ . When the inner and outer decoders are used iteratively inside a turbo decoder as in Fig. 4.1–4.2, these elements are generally not independent any longer. Since the non-independence is not taken into account by the inner and outer decoders, they cannot be assumed optimal any longer and the consistency (4.4)–(4.5) is not guaranteed. In other words, the decoders may behave differently inside a turbo decoder, which explains why EXIT charts cannot characterize accurately the turbo decoder when the interleaver is short — while they can characterize the inner and outer decoders separately.

Still, for asymptotically long interleavers, the elements in  $\mathcal{R}_k^i$  or in  $\mathcal{R}_k^o$  remain [103] fairly independent for certain codes, during a certain number of decoding iterations (that depends on the length of the interleaver), and the EXIT charts can thus characterize the convergence of the turbo decoder.

Note that a second (but less significant) issue related to the inaccuracy of EXIT charts with short interleavers is the non-typicality of the realizations of short sequences. When (4.8) is satisfied and the interleaver is large (i.e.,  $N$  large), the finite summation in (4.8) involves many terms, is thus approximately constant and equal to the limit by the central limit theorem. In other words, most realizations of  $(U, R)$  have the same properties (in the sense of “typicality”, with some abuse) and make the turbo decoder behave similarly: same amount of extrinsic information at a given iteration, same number of decoding iterations to reach a given level of performance, etc. By contrast, when the interleaver is short (i.e.,  $N$  small), the summation in (4.8) involves too few terms and does not converge. There is a large variance among the realizations of  $(U, R)$  and the turbo decoder can behave much differently with each realization, making the prediction of convergence difficult.

## 4.4 Applications

Most applications deal in practice with non-uniform binary sources, see notably [3, 11, 21–24, 26–32, 34–37, 39–41]. When the bias cannot be neglected, the BEXIT and FEXIT charts developed here above can be used to take the

bias into account and to predict the convergence correctly; applications of the BEXIT and FEXIT charts have already been given in Sections 3.5 and 4.3.2.

In this section, we just point out one possible application of the transformations (T-4.2-5)–(T-4.2-6). Let us focus on the system considered in [37], a memoryless biased binary source which is directly fed into a parallel turbo code, which is optimized for low signal to noise ratios on the channel. The purpose is therefore to lower the convergence threshold. In [37], the optimization is made through an exhaustive search among the possible generators for the convolutional codes; the performance of each generator is assessed by bit error rate (BER) Monte-Carlo simulations. To limit the search, the set of possible generators is restricted by a few constraints. Unfortunately, in spite of these constraints, the complexity of the whole optimization remains high since the exhaustive search and thus the heavy BER simulations are repeated for each value of the source bias  $L_U$ .

To lower this complexity, EXIT charts can be used instead of BER simulations to predict (fairly accurately) the convergence threshold, since only long interleavers are considered in [37]. Besides, the FEXIT charts of the system can be computed efficiently by using the transformations (T-4.2-5)–(T-4.2-6). Let us show how — note the optimization is not performed afterward, only the methodology is described.

Given the source bias  $L_U$ , let  $I'_{L_U}(\cdot)$  be the FEXIT chart of one decoder,

$$I'_E = I'_{L_U}(I'_A), \quad (4.41)$$

where  $I'_E$  is the level of extrinsic information and  $I'_A$  is the level of a priori information provided by the other decoder. Note that in [37], neither the extrinsic nor the a priori LLRs include the bias  $L_U$ . Rather, each decoder knows the bias and uses it only internally.

When there is no bias, the original method of [12] can be used to compute the FEXIT chart  $I'_{L_U}(\cdot)$ , no matter whether the bit flipping is used or not. Let  $I_0(\cdot) = I'_{L_U=0}(\cdot)$  be that FEXIT chart.

When there is a bias, the problem is to discover how the internal knowledge of the bias  $L_U$  changes the FEXIT chart  $I'_{L_U}(\cdot)$  w.r.t. the uniform case  $I_0(\cdot)$ . In other words, we want a transformation to obtain  $I'_{L_U}(\cdot)$  efficiently from  $I_0(\cdot)$ . To determine this transformation, we can do as follows:

- evaluate the total amount of information  $I_A^{\text{tot}}$ , including the internal knowledge of the bias, that the decoder has about each flipped bit  $U'$ ;
- given  $I_A^{\text{tot}}$ , obtain the extrinsic information  $I'_E$  in (4.41) as

$$I'_E = I_0(I_A^{\text{tot}}), \quad (4.42)$$

since the flipped bits  $U'$  have no bias;

- express  $I_A^{\text{tot}}$  as a function of  $I'_A$ ;
- based on this expression of  $I_A^{\text{tot}}$  as a function of  $I'_A$ , find  $I'_{L_U}(\cdot)$  by comparison between (4.41) and (4.42).

Let us thus express  $I_A^{\text{tot}}$  as a function of  $I'_A$ . On each bit  $U$ , the decoder has the a priori LLR  $L$  from the other decoder and it has (internally) the bias  $L_U$ . Equivalently, on each flipped bit  $U' = UF$ , it has the flipped LLR  $LF$  and the flipped bias  $L_U F$ , where  $F$  is the random bit flipping. We have therefore

$$I_A^{\text{tot}} = I(U'; LF + L_U F). \quad (4.43)$$

Since  $L + L_U$  is P-consistent (4.4), we can use (T-4.2-6) to calculate

$$I(U'; LF + L_U F) = I(U; L + L_U) + 1 - H_U. \quad (4.44)$$

Since  $L_U$  is a constant, we have  $I(U; L + L_U) = I(U; L)$ . Since  $L$  is L-consistent (4.5), we can use (T-4.2-5) to get

$$I(U; L) \approx J_{L_U} \left( J_0^{-1} (I(U'; LF)) \right), \quad (4.45)$$

where  $I(U'; LF) = I'_A$  by definition of  $I'_A$ . Finally,

$$I_A^{\text{tot}} \approx J_{L_U} (J_0^{-1} (I'_A)) + 1 - H_U. \quad (4.46)$$

At last, given (4.41) and (4.42), we obtain eventually the FEXIT chart  $I'_{L_U}(\cdot)$  as a function of  $I_0(\cdot)$  as

$$I'_{L_U}(I'_A) = I_0(I_A^{\text{tot}}) \approx I_0 \left( J_{L_U} (J_0^{-1} (I'_A)) + 1 - H_U \right). \quad (4.47)$$

Coming back to the exhaustive search performed in [37], it can be simplified in the following way. Instead of performing a BER simulation for each

$L_U$  and for each generator, the FEXIT chart  $I'_0(\cdot)$  needs to be computed only once for each generator. Then, to assess the performance of a turbo code for a given  $L_U$ , the FEXIT charts  $I_{L_U}(\cdot)$  can be obtained efficiently from (4.47). This method is much faster than performing BER simulations but it certainly comes at the expense of a small inaccuracy in the prediction of the convergence threshold.

## 4.5 Conclusion

Two methods have been developed in this chapter to handle non-uniform bits in the computation of EXIT charts, namely BEXIT and FEXIT charts. Though equivalent for the prediction of convergence under certain assumptions, they have different advantages and disadvantages. To summarize these, let us consider a serial concatenation. On the one hand, the FEXIT method is restricted to linear inner codes and symmetric channels while the BEXIT method is not. On the other hand, the FEXIT method handles very easily a mixture of bits having different entropies and offers a FEXIT chart for the inner decoder that is independent of the outer code, unlike the BEXIT method. In practice, both methods are thus complementary and might help in numerous applications.

This chapter completes the EXIT chart analysis made in Chapter 3.

# Performance Analysis of Variable Length Codes in Turbo/Concatenated Systems

## 5

*The content of this chapter will be submitted soon as two journal papers [18, 19]. Parts of it have already been published in [20,21].*

As we have seen in Chapter 2, variable length codes (VLCs), used in data compression, are very sensitive to error propagation in the presence of noisy channels. Addressing this problem with joint source-channel turbo techniques has been proposed in the literature, further studied in Chapter 3, and looks quite promising. But to date, most code-related conclusions are based on simulations.

This chapter, in two parts, states several theoretical results about the robustness of prefix VLCs concatenated with linear error correcting codes (ECC), assuming an optimal maximum likelihood decoder. In the first part, an approximate average distance spectrum of the concatenated code (VLC+ECC) is rigorously developed. Together with the union bound, this spectrum provides upper bounds on the symbol and frame/packet error rates. In the second part, these bounds are analyzed from an interleaving gain standpoint. In particular, it is investigated whether the VLC can contribute to the interleaving gain just as a convolutional code with non-catastrophic encoder would. To this end, the important concept of bounded VLC spectrum is introduced and is proved to be a sufficient condition for the VLC to contribute indeed to the interleaving gain. At last, this concept is proved to be closely related, under certain

assumptions, to the well known concept of statistically synchronizable VLCs, which allows us to reuse previously known results from the literature.

This chapter completes the interleaving gain analysis made in Chapter 3.

## 5.1 Introduction

The potential of joint source-channel turbo (de)coding with a variable length code as source code has been illustrated for the first time in [3], based on the serial concatenation of a VLC with a convolutional code (CC). This concatenation has then been improved in several directions, notably the VLC decoder [26,86], the VLC itself [14,20,23,39,48,49,72,73,87] and the inner error correcting code [20,29–31,37,40,41,88]. So far, however, most code-related conclusions have been based on simulations or on extrinsic information analysis, and a theoretical framework on the error correcting properties of VLCs in turbo/concatenated systems has been lacking.

The only attempts toward such a framework have been proposed in [72,73], and later independently in [20], for the serial concatenation of a VLC with a linear error correcting code (ECC). They provide the distance spectrum of the global code (VLC+ECC), based on pioneering results on VLC bit streams [14] and on turbo codes [15,16]. The distance spectrum is then used to upper bound the performance of the global code, assuming an optimal maximum likelihood (ML) decoder. Unfortunately, the expression of the spectrum is given without development in these contributions, as a simplistic combination/extension of previous results from [14–16], and some important details are missing.

*In this chapter*, in two main parts, we carry these contributions one step further by developing more accurate results as well as new results, and by proving them rigorously.

The first part is devoted to the calculation of an approximate average distance spectrum of the global code (VLC+ECC). Together with the union bound, this spectrum provides upper bounds on the bit, Levenshtein symbol and frame error rates, which are tight with simulation results at medium to high signal to noise ratios (SNRs) on the channel.

Capitalizing on these bounds, the second part is devoted to the analysis of the interleaving gains [15, 16] offered by the global code, based on the result from [92]. In particular, it is investigated whether the VLC, as outer code of the serial concatenation, can contribute to the interleaving gain, just as a convolutional code with non-catastrophic encoder would. To this end, the important concept of *bounded VLC spectrum* is introduced and is proved to be a sufficient condition for the VLC to contribute indeed to the interleaving gain. Next, this concept is also proved to be closely related, under certain assumptions, to the well known concept of statistically synchronizable VLCs [17], which gives us a simple way to test whether a given VLC has a bounded spectrum, without actually computing the spectrum. Simulation results will confirm the significance of bounded VLC spectra for the interleaving gains. Interestingly, they will also reveal that the maximum a posteriori (MAP) decoder, unlike the ML decoder, is sometimes still able to offer the interleaving gains when the VLC spectrum is not bounded, just as if it was bounded.

The remainder of this chapter is structured as follows. Section 5.2 recalls a few necessary background results on prefix VLCs, on the Balakirsky trellis, on the Levenshtein distance and on the distance spectrum of linear concatenated codes. Section 5.3 presents the generic transmission chain and makes already the link with the proposed performance bounds. Section 5.4 introduces the assumptions used throughout the chapter. Sections 5.5–5.6 constitute the main first part of this chapter, by providing the expression of the distance spectrum of VLC streams but especially by providing an approximate expression of the distance spectrum of VLCs concatenated with linear ECCs. Combined with the union bound, these spectra deliver several performance bounds. Related work in this field is discussed at the end of Section 5.6. Section 5.7 constitutes the main second part of this chapter and is essentially devoted to the analysis of the interleaving gains, assuming the VLC spectrum is bounded. At last, possible extensions of this work are discussed in Section 5.8 and simulations illustrating the theoretical results are reported in Section 5.9.

Here is a brief summary of the most used notations. Most of them are clearly defined within the chapter and are just reminded here for the sake of clarity, with only a short explanation. Note that many additional notations are introduced in Sections 5.2–5.3.

$Z$	a random variable (capital letter)
$z$	realization of $Z$ (small letter)
$P(z)$	abbreviation of the probability $P(Z = z)$
$E\{Z\}$	expectation of $Z$
$Z_{m:n}$	sub-sequence $(Z_m, Z_{m+1}, \dots, Z_n)$
$Z_:$	abbreviation of $Z_{m:n}$ when $m, n$ can be omitted
$\check{Z}$	decision taken on $Z$ at the receiver
$\mathcal{A}$	alphabet of the source symbols
$\mathcal{V}$	set of the VLC codewords
$S_:$	a sequence of source symbols
$U_:$	a sequence of VLC bits
$\text{vlc}(s_:$ )	bit sequence produced by the VLC for $s_:$
$\text{vlc}^{-1}(u_:$ )	reverse operation of $\text{vlc}(\cdot)$
$l(u_:$ )	bit length of $u_:$
$l(s_:$ )	bit length of $\text{vlc}(s_:$ )
$l_S(s_:$ )	symbol length of $s_:$
$l_S(u_:$ )	symbol length of $\text{vlc}^{-1}(u_:$ )
$H(S)$	entropy of $S$ , $-\sum_{s \in \mathcal{A}} P(s) \log_2(P(s))$
$\bar{l}$	average of the VLC codeword lengths, $\sum_{s \in \mathcal{A}} P(s) l(s)$
$\sigma_l^2$	variance of the VLC codeword lengths, $\sum_{s \in \mathcal{A}} P(s) (l(s) - \bar{l})^2$
$l_{\text{gcd}}$	greatest common divisor of the VLC codeword lengths, $\text{gcd}\{l(s) : s \in \mathcal{A}\}$
$l_{\text{gcd}}^*$	$\text{gcd}\{l(s) : s \in \mathcal{A}, P(s) > 0\}$ — note $l_{\text{gcd}}^* \geq l_{\text{gcd}}$
$l_{\text{max}}$	maximum VLC codeword length, $\max\{l(s) : s \in \mathcal{A}\}$
$l_{\text{max}}^*$	$\max\{l(s) : s \in \mathcal{A}, P(s) > 0\}$ — note $l_{\text{max}}^* \leq l_{\text{max}}$
$l_{\text{min}}$	minimum VLC codeword length, $\min\{l(s) : s \in \mathcal{A}\}$
$l_{\text{min}}^*$	$\min\{l(s) : s \in \mathcal{A}, P(s) > 0\}$ — note $l_{\text{min}}^* \geq l_{\text{min}}$
$d_H(s_:, \check{s}_:)$	Hamming distance between $\text{vlc}(s_:$ ) and $\text{vlc}(\check{s}_:)$
$d_S(s_:, \check{s}_:)$	symbol distance between $s_:$ and $\check{s}_:$
$d_{S_L}(s_:, \check{s}_:)$	Levenshtein symbol distance between $s_:$ and $\check{s}_:$
$\mathcal{E}(s_:, \check{s}_:)$	set of error events between $s_:$ and $\check{s}_:$
$u_{i:j} \equiv s_{m:n}$	relation of equivalence, that is, $u_{i:j} = \text{vlc}(s_{m:n})$ , $l(s_{1:m-1}) = i - 1$ and $l(s_{1:n}) = j$
$\mathbb{I}\{a\}$	indicator function, $\mathbb{I}\{a\}$ equals 1 if $a$ is true, 0 otherwise
$ \{\cdot\} $	number of elements in the set $\{\cdot\}$

$P_h$	pairwise error probability at Hamming distance $h$
$\mathbb{Z}$	set of integers $\{\dots, -2, -1, 0, 1, 2, \dots\}$
$\mathbb{N}_{\geq 0}, \mathbb{N}_{> 0}$	sets <sup>1</sup> $\mathbb{N}_{\geq 0} = \{n \in \mathbb{Z} : n \geq 0\}$ and $\mathbb{N}_{> 0} = \{n \in \mathbb{Z} : n > 0\}$
$\lfloor a \rfloor$	the floor function, $\max\{z \in \mathbb{Z} : z \leq a\}$
$\lceil a \rceil$	the ceiling function, $\min\{z \in \mathbb{Z} : z \geq a\}$
$\mathcal{O}(f(n))$	set [104] of all $g(n)$ such that there exist positive constants $k$ and $n_0$ with $g(n) \leq kf(n)$ for all $n \geq n_0$ . By extension, $\mathcal{O}(f(m, n))$ as $n \rightarrow +\infty$ denotes the set of all $g(m, n)$ such that, for any fixed $m$ , $g'(n) \in \mathcal{O}(f'(n))$ where $f'(n) = f(m, n)$ and $g'(n) = g(m, n)$ . Also, let $g(n) = \mathcal{O}(f(n))$ mean equivalently $g(n) \in \mathcal{O}(f(n))$ .

## 5.2 Background

### 5.2.1 Prefix VLCs and discrete source of symbols

Let  $\mathcal{B}$  be the binary alphabet  $\{0, 1\}$ . A finite sequence  $w = b_{1:m} = b_1 \dots b_m$  of code letters  $b_j \in \mathcal{B}$  is called a *word* over  $\mathcal{B}$ , of length  $l(w) = m$ . Let  $R$  be the empty word; we have  $l(R) = 0$  and  $Rw = w = wR$  for any word  $w$ . Let  $\mathcal{B}^+ \triangleq \cup_{i \geq 1} \mathcal{B}^i$ , the set of all non-empty words. Let  $\text{prefix}(w) = \{p \in \mathcal{B}^+ : \exists v \in \mathcal{B}^+, w = pv\}$  and  $\text{suffix}(w) = \{s \in \mathcal{B}^+ : \exists v \in \mathcal{B}^+, w = vs\}$ . Each element of  $\text{prefix}(w)$  is a *prefix* of  $w$  and each element of  $\text{suffix}(w)$  a *suffix* of  $w$ . A *code* is a set  $\mathcal{V} \subset \mathcal{B}^+$ . The elements of  $\mathcal{V}$  are called *codewords*. If no codeword is the prefix of another codeword, then the set  $\mathcal{V}$  is called a *prefix variable length code* (VLC). If, besides, no codeword is the suffix of another codeword, then the VLC is said *reversible*. In the following, only prefix VLCs are considered and we omit to specify "prefix" for convenience. Let  $\mathcal{V}^+ \triangleq \cup_{i \geq 1} \mathcal{V}^i$ , the set of all non-empty sequences of VLC codewords. At last, for any code  $\mathcal{U} \in \mathcal{B}^+$ , let  $\text{prefix}(\mathcal{U}) \triangleq \cup_{w \in \mathcal{U}} \text{prefix}(w)$ , the set of all prefixes of  $\mathcal{U}$ .

Let us then consider an information *source* of discrete, independent (memoryless) and identically distributed symbols over the alphabet  $\mathcal{A}$ ,  $|\mathcal{A}| \geq 2$ ,

<sup>1</sup>We introduce the notations  $\mathbb{N}_{\geq 0}$  and  $\mathbb{N}_{> 0}$  to avoid any possible confusion in the following, instead of the more common  $\mathbb{N}$ ,  $\mathbb{N}_0$ ,  $\mathbb{N}^*$  and  $\mathbb{N}^+$ , which are less explicit and used differently by authors in the literature.

and characterized by the probability distribution  $P(S)$ . This source is described by the stationary random sequence of symbols  $S_1, S_2, S_3, \dots$ , where  $S_k \in \mathcal{A}$  and  $P(S_k = s | S_{k-1}) = P(S_k = s) = P(S = s)$ . Let us assume  $P(S = s) > 0, \forall s \in \mathcal{A}$ . Let  $\mathcal{A}^+ \triangleq \cup_{i \geq 1} \mathcal{A}^i$ , the set of all non-empty sequences of symbols.

Given a source and a VLC with  $|\mathcal{A}| = |\mathcal{V}|$ , we create a bijective (invertible) mapping  $\text{vlc}(\cdot)$  by associating to each symbol  $s \in \mathcal{A}$  a unique codeword  $w = \text{vlc}(s) \in \mathcal{V}$ . The inverse mapping is denoted  $\text{vlc}^{-1}(\cdot)$ . In the following, this mapping is always assumed implicitly; we will use concepts associated with codewords equivalently with symbols and vice versa, e.g.,  $l(s) = l(\text{vlc}(s))$ ,  $P(w) = P(\text{vlc}^{-1}(w))$ .

At last, given  $U_{1:\infty} = \text{vlc}(S_{1:\infty})$ , we introduce a relation of equivalence between bit sub-sequences and symbol sub-sequences:  $u_{i:j} \equiv s_{m:n}$  means  $u_{i:j} = \text{vlc}(s_{m:n})$ ,  $l(s_{1:m-1}) = i - 1$  and  $l(s_{1:n}) = j$ . In words,  $u_{i:j}$  and  $s_{m:n}$  refer to the same realization of the source/VLC and  $P(u_{i:j}) = P(s_{m:n})$ . In the following, this equivalence is generally assumed implicitly, unless otherwise stated.

## 5.2.2 Markov encoding process and Balakirsky trellis

Most of the results in this chapter are presented independently of any model. Still, for the familiar reader, an interpretation is sometimes given using the Balakirsky trellis diagram [55].

The concatenation of the source and of the prefix VLC described above can be easily modeled as a Markov binary source whose state space is made up of all prefixes of the VLC:

$$\text{state space:} \quad \mathcal{X} = \{R\} \cup \text{prefix}(\mathcal{V}), \quad (5.1)$$

$$\text{state at (bit) time } n: \quad X_n \in \mathcal{X}, \quad (5.2)$$

$$\text{transition probability:} \quad P(U_{n+1}, X_{n+1} | X_n), \quad (5.3)$$

where the bit  $U_{n+1}$  is associated with the transition from  $X_n$  to  $X_{n+1}$ . The probability of the transition is given by

$$P(u_{n+1}, x_{n+1} | x_n) = P(u_{n+1} | x_n) P(x_{n+1} | x_n, u_{n+1}), \quad (5.4)$$

$$P(u_{n+1}|x_n) = \frac{\sum_{w \in \mathcal{W}(x_n u_{n+1})} P(w)}{\sum_{w \in \mathcal{W}(x_n)} P(w)}, \quad (5.5)$$

$$P(x_{n+1}|x_n, u_{n+1}) = \mathbb{I}\{x_{n+1} = x_n u_{n+1}\} + \mathbb{I}\{x_{n+1} = R, x_n u_{n+1} \in \mathcal{V}\}, \quad (5.6)$$

where  $\mathcal{W}(x)$  is the set of codewords starting with  $x$ ,

$$\mathcal{W}(x) = \{w \in \mathcal{V} : \exists v \in \mathcal{B}^+ \cup \{R\}, w = xv\}. \quad (5.7)$$

Recall  $R$  in (5.1) is the empty word; in the current context, it is the state of the Markov source when it has finished a codeword, or equivalently, when it is ready to start a new one.

In words, starting for example from  $X_0 = R$ , the Markov source appends a bit  $U_{n+1}$  at each time instant to the current state  $X_n$ . If  $X_n U_{n+1} \in \mathcal{V}$ , we have reached a codeword and the next state becomes the empty word  $X_{n+1} = R$ . If  $X_n U_{n+1} \in \text{prefix } \mathcal{V}$ , the next state becomes  $X_{n+1} = X_n U_{n+1}$ . Otherwise (if  $X_n U_{n+1} \notin \text{prefix } \mathcal{V}$ ), the bit  $U_{n+1}$  is discarded because no codeword starts with  $X_n U_{n+1}$ .

This Markov model fully represents the concatenation of the source of symbols with the prefix VLC of Section 5.2.1. In terms of trellis diagram [42], it corresponds to the so-called Balakirsky trellis [55] (see Fig. 2.5 in Chapter 2). In terms of binary tree (see Fig. 2.4 in Chapter 2), the elements of the state space  $\mathcal{X}$  are the internal nodes of the binary tree. In particular, the element  $R$  is the root node of the tree and is the so-called *root state* in the Balakirsky trellis.

When  $l_{\text{gcd}} > 1$ , i.e., when the codewords lengths are multiples of  $l_{\text{gcd}} > 1$ , the Markov model is periodic, independently of the source and of the source/VLC mapping. If we take the source statistics and the source/VLC mapping into account, then the Markov model is periodic when  $l_{\text{gcd}}^* > 1$ , i.e., when the lengths of the codewords of non-zero probability are multiples of  $l_{\text{gcd}}^* > 1$ . This is formally stated in the following proposition.

**Proposition 5.1.** *Let us consider the Markov source starts in  $X_0 = R$  and let us define the instantaneous state spaces*

$$\mathcal{X}_i \triangleq \left\{ p \in \mathcal{X} : l(p) \equiv i \pmod{l_{\text{gcd}}} \right\}, \quad 0 \leq i < l_{\text{gcd}}, \quad (5.8)$$

$$\mathcal{X}_i^* \triangleq \left\{ p \in \mathcal{X} : l(p) \equiv i \pmod{l_{\text{gcd}}^*}, \sum_{w \in \mathcal{W}(p)} P(w) > 0 \right\}, \quad 0 \leq i < l_{\text{gcd}}^*. \quad (5.9)$$

Then, by (5.4)–(5.6),

$$\begin{aligned} P(X_n \in \mathcal{X}_i) &= 1, \quad \text{for } n \equiv i \pmod{l_{\text{gcd}}}, \text{ independently of } P(S), \text{vlc}(\cdot), \\ P(X_n \in \mathcal{X}_i^*) &= 1, \quad \text{for } n \equiv i \pmod{l_{\text{gcd}}^*}. \end{aligned}$$

In particular, we cannot be in the root state at time  $n$ ,  $X_n \neq R \in \mathcal{X}_0$ , for values of  $n$  that are not multiple of  $l_{\text{gcd}}$ ,  $n \not\equiv 0 \pmod{l_{\text{gcd}}}$ . ■

Another property of the Markov chain is the following: There is at most one state  $x_n$  that can reach a given state  $x_{n+1}$ , except if  $x_{n+1}$  is the root state.

**Proposition 5.2.** *Given a state  $x_{n+1} \in \mathcal{X} \setminus \{R\}$ , there is at most one state  $x_n \in \mathcal{X}$  with  $P(X_{n+1} = x_{n+1} | X_n = x_n) > 0$ .* ■

### 5.2.3 Levenshtein symbol distance

The Levenshtein distance has been introduced in [105] as an edit distance to measure the resemblance between two strings. It will be used as a performance measure in this chapter.

**Definition 5.3.** *The Levenshtein symbol distance  $d_{S_L}(s, \check{s})$  between two sequences of symbols  $s, \check{s} \in \mathcal{A}^+$  is given by the minimum number of operations needed to transform one sequence into the other, where an operation is a substitution, an insertion or a deletion of a single symbol.* □

### 5.2.4 Union bound

In probability theory, the *union bound*, also known as *Boole's inequality*, says that the probability of a union of events cannot exceed the sum of the individ-

ual probabilities, i.e., given a countable set of events  $E_1, E_2, E_3, \dots$ ,

$$P(\cup_i E_i) \leq \sum_i P(E_i), \quad (5.10)$$

which is an equality when the events are mutually exclusive (or disjoint). When each  $E_i$  is the event of selecting a specific erroneous codeword at the receiver (instead of the transmitted one), then (5.10) gives an upper bound on the error probability.

### 5.2.5 Distance spectrum

A few principles and notations from [16] about distance spectrum for linear codes are here recalled. Given a linear block code  $C$ , let  $A_{w,h}^C$  denote the number of codewords with weight  $h$  generated by information words of weight  $w$ . In the following, the quantity  $A_{w,h}^C$  is referred to as the *distance spectrum* of the code  $C$ . For convolutional codes, the same notations hold in terms of paths instead of codewords.

The union bounds on the FER and BER for a linear block code  $C$  with  $N$  information bits, are respectively

$$\text{FER} \leq \sum_{w,h} A_{w,h}^C P_h, \quad (5.11)$$

$$\text{BER} \leq \sum_{w,h} \frac{w}{N} A_{w,h}^C P_h, \quad (5.12)$$

where  $P_h$  is the pairwise error probability. These bounds are upper bounds on the error rates obtained with maximum likelihood (ML) decoding. Note that  $P_h$  is the conditional probability that a given codeword  $\check{u}_.$ , at distance  $d_H(u., \check{u}.) = h$  from the transmitted codeword  $u.$ , has a larger likelihood metric than  $u.$ . For the additive white Gaussian noisy (AWGN) channel with the BPSK modulation, it is given by  $P_h = \frac{1}{2} \text{erfc}\left(\sqrt{hR_c \frac{E_b}{N_0}}\right)$ , where  $R_c$  is the global code rate,  $N_0/2$  is the double-sided noise spectral density and  $E_b$  the energy per bit of entropy. For the binary symmetric channel with error probability  $p$ ,  $P_h \leq [4(1-p)p]^{h/2}$ .

For the evaluation of the bounds (5.11) and (5.12), the distance spectrum is required. The abstract concept of *uniform interleaver* is introduced and methods are developed in [16] to compute efficiently the distance spectrum of concatenated linear codes. To summarize, consider a serially concatenated code

$C_s$  whose outer and inner linear component codes are respectively  $C_o$  and  $C_i$ . The length of the interleaver is  $N$ . Then, the distance spectrum of this concatenated code is given by

$$A_{w,h}^{C_s} = \sum_{l=0}^N \frac{A_{w,h=l}^{C_o} A_{w=l,h}^{C_i}}{\binom{N}{l}}. \quad (5.13)$$

The technique proposed in [16] relies on the linearity of the component codes. It cannot be applied directly here due to the non-linearity of the VLC. One contribution of this chapter is precisely to develop and demonstrate similar relations for VLCs.

### 5.3 Transmission system

We consider two generic transmission systems. The former in Fig. 5.1 transmits data streams and is based only on a source and a VLC. The latter in Fig. 5.2 is based on frames: The VLC stream is framed into a VLC block, which is then serially concatenated with a linear error correcting code.

For both systems, a few bit errors at the receiver can desynchronize the VLC decoder and make all subsequent symbols — from a non-Levenshtein symbol distance standpoint — in the stream or in the frame useless. Therefore, focusing on the frame-based system, a good (but non-optimal) decision criterion is the minimization of the frame error rate. The solution to this minimization is the frame-MAP (maximum a posteriori) detection rule

$$\check{s} \triangleq \underset{s}{\operatorname{argmax}} \{P(s|y)\} = \underset{s}{\operatorname{argmax}} \{P(y, s)\}, \quad (5.14)$$

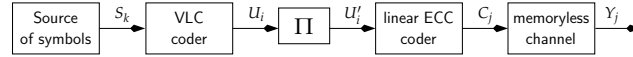
where  $y$  is the received signal. This detection rule extends easily to the streams of the stream-based system; let us call it the sequence-MAP detection.

#### 5.3.1 Semi-infinite VLC stream

Consider in Fig. 5.1 a stationary source producing a semi-infinite stream of discrete symbols  $S_{1,\infty}$ , or *symbol stream*. Consider also a prefix VLC that encodes each symbol  $S_k$  into a codeword  $\operatorname{vlc}(S_k)$ . This results in a semi-infinite *bit stream* or *VLC stream*  $U_{1,\infty} = \operatorname{vlc}(S_{1,\infty})$ .



**Figure 5.1** Generic stream coder, without ECC.



**Figure 5.2** Generic block coder, serially concatenated with a linear block ECC.

At the receiver, the sequence-MAP decoder can be implemented with a reasonable complexity, e.g., the Viterbi algorithm [52] on the Balakirsky trellis (Section 5.2.2).

### 5.3.2 Serial concatenation of a VLC block with a linear code

In Fig. 5.2, the infinite VLC stream is framed into finite-length VLC blocks (frames or packets) before the interleaving by the interleaver  $\Pi$  of length  $N$ . This framing process is done according to a certain *framing rule* which is detailed later. At this state, let us consider that a VLC block is composed of a certain number of VLC codewords, say  $N_s$ , and a certain number of bits, say  $N_b$ , with

$$N_b \triangleq \sum_{k=1}^{N_s} l(S_k) = l(S_{1:N_s}) \leq N. \quad (5.15)$$

How  $N$ ,  $N_b$  and  $N_s$  are determined depends on the chosen framing rule.

After the framing process, zeros are appended to  $U_{1:N_b}$  to get  $U_{1:N}$  if  $N_b < N$ . The bits  $U_{1:N}$  are then interleaved by  $\Pi$  into  $U'_{1:N}$ . For the analysis, the interleaver  $\Pi$  is assumed uniform [16] (or fully random). The ECC then generates the coded bits  $C$ ; and sends them across the channel, which is binary, symmetric, memoryless and time invariant. The ECC can be any linear block code, e.g., framed convolutional-code, turbo-code, low density parity check (LDPC) code. Note that the interleaver may be not necessary with certain codes, e.g., LDPC code, repeat-accumulate code, irregular turbo code, etc.

At the receiver, the frame-MAP detection is prohibitively complex because of the product code source/VLC + ECC. In the simulation results, we will use instead a frame-MAP joint source-channel iterative/turbo decoder, as a first (good) approximation. It can be deduced from the application of the sum-product algorithm (SPA) on the factor graph [5,7] of the complete transmission scheme, followed by the Viterbi algorithm on the Balakirsky trellis — examples can be found in [8,9] and in Chapters 2 and 3. If the constituent codes are “well” separated, here thanks to the interleaver  $\Pi$ , the decoding complexity of the turbo decoder is generally much lower than with the optimal frame-MAP decoder and is proportional to the decoding complexities of the constituent decoders.

### 5.3.3 Link with the proposed performance bounds

The proposed performance bounds in the next sections are based on the ML (maximum likelihood) detection rule

$$\check{s}_i = \underset{s_i}{\operatorname{argmax}}\{P(y_i|s_i)\}, \quad (5.16)$$

which does not take into account the a priori probabilities of the source sequences compared to the MAP detection rule (5.14).

Though the ML detection is suboptimal and different from the MAP detection when the frames  $s_i$  are not uniformly distributed, it tends to the MAP detection under certain conditions (Section 5.4). When these conditions are met, the proposed (ML) performance bounds can be used therefore to analyze the (MAP-based) decoders described here above in Sections 5.3.1–5.3.2.

This conclusion is further supported by the next two facts. Firstly, the union bound, which is at the heart of the proposed upper bounds, is tight and thus useful at moderate to high signal to noise ratios, where, as mentioned, the MAP and ML decoders tend to behave similarly. Secondly, the distribution of the source symbols is sometimes not perfectly known/estimated at the receiver, making it impossible to implement the optimal MAP decoder. When this is case, the proposed performance bounds can be used somehow as a worst-case scenario analysis.

These arguments motivate to focus on the simpler ML detection in the following Sections. Afterward, an extension of the proposed bounds to the MAP detection is discussed in Section 5.8.4.

## 5.4 Assumptions

For the sake of clarity, most assumptions used to develop the theoretical results in this chapter are summarized hereafter. Possible generalizations to other assumptions are discussed in Section 5.8.

### 5.4.1 Assumptions

**Assumption 5.4** (source/VLC). *Let us consider a stationary memoryless source and a prefix VLC, as in Section 5.2.1, with at least two different symbol values of non-zero probability (i.e.,  $\exists s', s'' \in \mathcal{A}$ ,  $s' \neq s''$ ,  $P(s') > 0$ ,  $P(s'') > 0$ ) — this implies  $|\mathcal{A}| \geq 2$ .  $\square$*

**Assumption 5.5** (global systems). *In addition to Assumption 5.4, we make the following assumptions. (i) For the stream-based system in Fig. 5.1, we consider semi-infinite streams  $U_{1:\infty} = \text{vlc}(S_{1:\infty})$  and a sequence-ML decoder (5.16). (ii) For the frame-based system in Fig. 5.2, we consider finite-length frames  $U_{1:N_b} = \text{vlc}(S_{1:N_s})$ , a uniform interleaver [16] of length  $N$ , a linear code and a frame-ML decoder (5.16). We assume furthermore that the decoder knows (and uses) the values of  $N_b$  and  $N$  but not the value of  $N_s$ . If  $N_b < N$ , the decoder knows besides that the bits  $U_{N_b+1:N}$  are zeros.  $\square$*

Note that assuming the decoder knows  $N_b$  in the frame-based system ensures that the transmitted and decoded frames have the same bit length. But it is not ensured that they have the same symbol length since  $N_s$  is assumed unknown.

**Assumption 5.6.** *In addition to Assumption 5.5, let us consider all symbol values have non-zero probabilities,  $P(s') > 0$  for all  $s' \in \mathcal{A}$ .  $\square$*

This framework of assumptions, especially Assumption 5.5, will be used throughout the remainder of this chapter.

### 5.4.2 ML versus MAP

Though based on the ML detection (by Assumption 5.5), the theoretical results developed in the next sections can be used to analyze also the MAP-based decoders proposed in Sections 5.3.1–5.3.2.

It is indeed well known and worth recalling that the ML and MAP detections tend to the same performance level under certain conditions. But besides, even when these conditions are not satisfied, we can often still analyze the MAP-based decoders with the proposed theoretical results by defining a new source and a new VLC. For example, assuming  $P(s') > 0$  for all  $s' \in \mathcal{A}$  in Assumption 5.6 makes the ML detection and the MAP detection straightforwardly equivalent<sup>2</sup> at moderate to high signal to noise ratios on the channel, i.e., when the channel reliability becomes significant compared to the a priori information. But besides, note that we can always trivially satisfy this assumption ( $P(s') > 0$  for all  $s' \in \mathcal{A}$ ) for a stationary memoryless source by simply defining a new source alphabet and a new VLC that exclude the symbols/codewords of zero probability.

Together with the arguments in Section 5.3.3, this motivates to focus on the simpler ML detection from now on and discuss the extension to the MAP detection afterward in Section 5.8.4.

## 5.5 Performance bounds for VLC streams

### 5.5.1 Definition of an error event

To characterize the synchronization between a transmitted stream  $U_{1:\infty} = \text{vlc}(S_{1:\infty})$  and a decoded stream  $\check{U}_{1:\infty} = \text{vlc}(\check{S}_{1:\infty})$ , let us define the events

$$\rho_{k,k'}(S_{1:\infty}, \check{S}_{1:\infty}) \triangleq (l(S_{1:k-1}) = l(\check{S}_{1:k'-1})), \quad (5.17)$$

$$\rho_l(U_{1:\infty}, \check{U}_{1:\infty}) \triangleq (U_{1:l-1} \in \mathcal{V}^+ \cup \{R\}, \check{U}_{1:l-1} \in \mathcal{V}^+ \cup \{R\}). \quad (5.18)$$

<sup>2</sup>To make this clearer, let us illustrate what may happen when Assumption 5.6 is not satisfied. Consider the VLC  $\mathcal{V} = \{0, 11, 10\}$  with  $P(0) > 0$ ,  $P(11) > 0$  and  $P(10) = 0$ . Then, from the ML decoder viewpoint, note that the minimum Hamming distance between any pair of bit streams is 1, since  $d_H(11, 10) = 1$ . But from the MAP decoder viewpoint, it is as if the codeword 10 did not exist; the minimum distance is thus 2 and besides the VLC is reversible.

These events are intrinsically linked, e.g.,  $\rho_{k,k'}(S_{1:\infty}, \check{S}_{1:\infty})$  implies  $\rho_{l=1+l(S_{1:k-1})}(U_{1:\infty}, \check{U}_{1:\infty})$ .

Both events can be used as measures of synchronization. Indeed,  $\rho_l(U_{1:\infty}, \check{U}_{1:\infty})$  is, loosely speaking, the joint event that  $U_l$  is the first bit of a codeword in  $U_{1:\infty}$  and  $\check{U}_l$  is the first bit of a codeword in  $\check{U}_{1:\infty}$ , i.e., that both sequences have a codeword starting at bit position  $l$ . Therefore, if there is no bit difference (no bit error) after  $U_l$ , i.e.,  $\check{U}_{l:\infty} = U_{l:\infty}$ , then, by the prefix property of the VLC, all subsequent codewords are decoded correctly (assuming a possible time shift), i.e.,  $\check{S}_{k':\infty} = S_{k:\infty}$  for  $k-1 = l_S(U_{1:l-1})$  and  $k'-1 = l_S(\check{U}_{1:l-1})$ .

This is the idea underlying the concept of error event hereafter. An error event is generated by some errors in the stream and ends as soon as  $\rho_l(U_{1:\infty}, \check{U}_{1:\infty})$  for some  $l$ .

**Definition 5.7** (Error event). *Under Assumption 5.5, given  $u_{1:\infty} = \text{vlc}(s_{1:\infty})$  and  $\check{u}_{1:\infty} = \text{vlc}(\check{s}_{1:\infty})$ ,  $u_{i:j}$  and  $\check{u}_{i:j}$  form an error event  $e$  if*

$$u_{i:j} \neq \check{u}_{i:j} \\ \text{and } \{i, j+1\} = \{l : i \leq l \leq j+1, \rho_l(u_{\cdot}, \check{u}_{\cdot})\}, \quad (5.19)$$

and we say the error event  $e$  starts with  $u_i$ . Equivalently,  $s_{m:n}$  and  $\check{s}_{m':n'}$  form an error event  $e$  if

$$s_m \neq \check{s}_{m'} \\ \text{and } \{(m, m'), (n+1, n'+1)\} = \\ \{(k, k') : (m, m') \leq (k, k') \leq (n+1, n'+1), \rho_{k,k'}(s_{\cdot}, \check{s}_{\cdot})\}, \quad (5.20)$$

where  $(a, a') \leq (b, b')$  means  $(a \leq b, a' \leq b')$ , and we say the error event  $e$  starts with  $s_m$ .  $\square$

By extension of previous notations, let  $l_S(e) = l_S(u_{i:j})$ ,  $l(e) = l(u_{i:j}) = j - i + 1$ ,  $d_{S_L}(e) = d_{S_L}(\check{u}_{i:j}, u_{i:j})$  and  $d_H(e) = d_H(\check{u}_{i:j}, u_{i:j})$ . Note the two definitions are equivalent but (5.20) states explicitly that there is necessarily at least an error during the first symbol,  $s_m \neq \check{s}_{m'}$ .

**Remark 5.8** (Error events and links with  $l_{\text{gcd}}, l_{\text{gcd}}^*$ ). *It is worth noting that codewords in  $U_{1:\infty}$  and in  $\check{U}_{1:\infty}$ , and thus error events, cannot start at bit positions  $l$  subject to  $l \not\equiv 1 \pmod{l_{\text{gcd}}}$  since all codewords have lengths multiple of  $l_{\text{gcd}}$  (by*

definition of  $l_{\text{gcd}}$ ). Furthermore, the probability that codewords in  $U_{1:\infty}$  (but not necessarily<sup>3</sup> in  $\check{U}_{1:\infty}$ ), and thus error events, start at bit positions  $l$  subject to  $l \not\equiv 1 \pmod{l_{\text{gcd}}^*}$  is zero since all codewords of non-zero probability have lengths multiple of  $l_{\text{gcd}}^*$ . This is related to Proposition 5.1.  $\square$

Note that the streams  $u_{1:\infty} = \text{vlc}(s_{1:\infty})$  and  $\check{u}_{1:\infty} = \text{vlc}(\check{s}_{1:\infty})$  might have several pairs of sub-sequences forming an error event. In the following, we denote by  $\mathcal{E}(u, \check{u})$  (equivalently,  $\mathcal{E}(s, \check{s})$ ) the set of error events between  $u$  and  $\check{u}$ . By definition, there is no bit error “outside” these error events. There is no symbol error either, if we tolerate a possible time shift of the symbol stream (due to the possibility of symbol deletions/insertions in error events). Here is an example.

**Example 5.9.** Consider the source alphabet  $\mathcal{A} = \{a, b\}$ , with  $P(S = a) = p \in (0, 1)$  and  $P(S = b) = 1 - p$ , and the VLC  $\{0, 11\}$  with  $\text{vlc}(a) = 0$  and  $\text{vlc}(b) = 11$ . This simple and theoretical example will be used throughout the chapter to illustrate certain results. Consider the following transmitted stream  $u$  and decoded stream  $\check{u}$ : (bit errors in positions 1, 4, 8 and 14) framed to 14 bits for clarity:

$$\begin{aligned}
 u_{1:12} &= \underset{a}{0} \underset{b}{1} \underset{a}{1} \underset{a}{0} \underset{a}{0} \underset{a}{0} \underset{b}{1} \underset{b}{1} \underset{b}{1} \underset{b}{1} \underset{a}{1} \underset{a}{0}, \\
 \check{u}_{1:12} &= \underset{b}{1} \underset{b}{1} \underset{a}{1} \underset{a}{0} \underset{a}{0} \underset{a}{0} \underset{b}{1} \underset{b}{1} \underset{b}{1} \underset{b}{1} \underset{b}{1} \underset{b}{1}.
 \end{aligned}$$

There is an error event of bit length 4 starting with  $u_1$  or, equivalently, of symbol length 3 starting with  $s_1$ , ( $aba, bb$ ). There is a second error event of bit length 7 starting with  $u_8$  or, equivalently, of symbol length 4 starting with  $s_7$ , ( $bbba, abbb$ ). There is no symbol error in between if we tolerate a symbol time shift ( $\check{u}_{1:4}$  contains one symbol less than  $u_{1:4}$ ).  $\square$

In terms of trellis diagram,  $\rho_i(U_{1:\infty}, \check{U}_{1:\infty})$  is related to the event of being in the root state in the Balakirsky trellis (Section 5.2.2). Given the Markov state sequences  $X_{0:\infty}, \check{X}_{0:\infty}$  associated with  $U_{1:\infty}, \check{U}_{1:\infty}$  respectively,  $\rho_i(U_{1:\infty}, \check{U}_{1:\infty})$  is the event that  $X_{i-1} = \check{X}_{i-1} = R$ . Therefore, by extension,  $x_{i-1:j}$  and  $\check{x}_{i-1:j}$

<sup>3</sup>Under Assumption 5.5,  $\check{U}_{1:\infty}$  may have codewords starting at bit positions  $l$  subject to  $l \not\equiv 1 \pmod{l_{\text{gcd}}^*}$  if  $l_{\text{gcd}}^* > l_{\text{gcd}}$  since the ML decoder does not take into account the a priori probabilities of the source sequences and may thus decode codewords of zero probability.

form an error event if

$$x_{i-1:j} \neq \check{x}_{i-1:j} \text{ and } \{i-1, j\} = \{l : i-1 \leq l \leq j, x_l = \check{x}_l = R\} \quad (5.21)$$

At last, strictly speaking, an error event is usually considered as the *probabilistic* event that the sequence decoder (or Viterbi decoder) on a given trellis diverges at a given position from the path representing the transmitted sequence and merges back with it later, while the words “error event” in Definition 5.7 refer rather to a *non-probabilistic* code property. This language abuse is common because there is a one-to-one mapping between the error events of the decoder and the “error events” in Definition 5.7. Indeed, each error event of the decoder is associated with a specific pair of paths. For example,  $x_{i':j}, \check{x}_{i':j}$  form an error event in the Balakirsky trellis, or *Balakirsky error event*, if

$$u_{i'+1} \neq \check{u}_{i'+1} \text{ and } \{i', j\} = \{l : i' \leq l \leq j, x_l = \check{x}_l\}. \quad (5.22)$$

In words, the decoder is synchronized in  $x_{i'} = \check{x}_{i'}$ , makes a bit error in  $u_{i'+1}$ , diverges immediately after if  $j \geq i' + 2$  (i.e.,  $x_{i'+1} \neq \check{x}_{i'+1}$ ) and merges back for the first time in  $x_j = \check{x}_j$ . The link with Definition 5.7 is straightforward. Firstly indeed, by Proposition 5.2, diverging paths can merge only in the root state, i.e.,  $x_l = \check{x}_l = R$  in (5.22) when  $l = j$ ; secondly, by (5.6),  $x_i = R$  with  $i = i' - l(x_{i'})$ ; thirdly, by Proposition 5.2, there is only one path that can lead to  $x_{i'}$  from  $x_i = R$ , i.e.,  $x_l = \check{x}_l$  for  $i \leq l \leq i'$ . So for each Balakirsky error event, i.e., for each error event of the decoder, there is one and only one “error event” in (5.19)–(5.21), and vice versa.

### 5.5.2 Spectrum and bounds for VLC streams

For  $u: \equiv s: \equiv s_{m:n}$ , let  $P(s:)$  and  $P(u:)$  be the probability of observing  $s:$  among all sequences of the same symbol length, i.e.,  $P(u:) = P(s:) = \prod_{k=m}^n P(S = s_k)$ .

**Definition 5.10.** *The average distance spectrum of a VLC stream, or VLC stream spectrum, is under Assumption 5.5*

$$A_{s_L, h, l_s, l_b}^{\text{vlc}} \triangleq \sum_{\substack{s_{1:l_s} \in \mathcal{A}^{l_s} \\ \text{s.t. } l(s_{1:l_s}) = l_b}} P(s_{1:l_s}) A_{s_L, h | s_{1:l_s}}^{\text{vlc}}, \quad (5.23)$$

where  $A_{s_L, h | s}^{\text{vlc}}$  is the conditional spectrum given  $s$ ,

$$A_{s_L, h | s}^{\text{vlc}} \triangleq |\mathcal{A}_{s_L, h | s}^{\text{vlc}}|, \quad (5.24)$$

$$\mathcal{A}_{s_L, h | s}^{\text{vlc}} \triangleq \left\{ \check{s} \in \mathcal{A}^+ : s \text{ and } \check{s} \text{ form an error event, } d_{s_L}(s, \check{s}) = s_L, d_H(s, \check{s}) = h \right\}. \quad (5.25)$$

□

Given a sequence of symbols  $s_{1:l_s}$ , the conditional spectrum is the number of sequences  $\check{s}$  forming an error event with  $s_{1:l_s}$  — in the Balakirsky trellis, given the path representing  $s_{1:l_s}$ , it is the number of paths diverging from it during the first symbol  $s_1$  and merging back with it for the first time at the end of the last symbol  $s_{l_s}$ . That number is then weighted by  $P(s_{1:l_s})$  in (5.23). The numerical evaluation of (5.23) is discussed in Section 5.5.3.

**Remark 5.11.** *Despite its name, the VLC stream spectrum in Definition 5.10 is more properly the spectrum of the triplet formed by the VLC, the source/VLC mapping  $\text{vlc}(\cdot)$  and the source. In particular, it follows from (5.23) that the VLC spectrum depends on the source statistics. This is really worth emphasizing: Many properties that we relate commonly (with some abuse) to the the VLC depend often also on the source statistics, e.g., the concept of bounded (VLC) spectrum hereafter.* □

In the following, unnecessary subscripts will be sometimes discarded for convenience, e.g.,

$$A_{s_L, h}^{\text{vlc}} = \sum_{l_s, l_b} A_{s_L, h, l_s, l_b}^{\text{vlc}}. \quad (5.26)$$

An important concept in the theory of codes is the free distance.

**Definition 5.12.** *The free distance of a VLC is given by  $d_f^{\text{vlc}} = \min\{h \in \mathbb{N}_{>0} : A_h^{\text{vlc}} \neq 0\}$ .* □

From the definition of the VLC stream spectrum  $A_h^{\text{vlc}}$ , it follows that the free distance is the minimum Hamming distance between any pair of admissible binary sequences of the same bit length, among the pairs including at least one sequence of non-zero probability.

It is well known that most VLCs are very sensitive to residual bit errors, in the sense that a few bit errors are often able to desynchronize the decoder and to generate an arbitrary long error event. In terms of spectrum, this sensitivity to residual bit errors, or equivalently the existence of arbitrarily long

error events, translates into the existence of arbitrarily large values of  $l_b$  (or  $l_s$ ) such that the spectrum  $A_{h,l_b}^{\text{vlc}}$  (or  $A_{h,l_s}^{\text{vlc}}$ ) is non-zero for small  $h$  (small number of residual bit errors). Fortunately, later on we will show that the probability of long error events is usually very small and that the VLC spectrum  $A_{h,l_b}^{\text{vlc}}$  decreases exponentially toward zero for sufficiently large  $l_b$ . To characterize this, we introduce the novel concept of bounded spectrum.

**Definition 5.13.** A VLC spectrum is bounded if  $\exists c < 1, \forall h, \exists l'_h, \forall l_b \geq l'_h, A_{h,l_b}^{\text{vlc}} \leq c^{l_b}$ , i.e., if  $\exists c < 1, \forall h, A_{h,l_b}^{\text{vlc}} = \mathcal{O}(c^{l_b})$ .  $\square$

**Definition 5.14.** A VLC spectrum is strongly bounded if  $\exists c < 1, \exists \eta, \forall h, \forall l_b \geq \eta h, A_{h,l_b}^{\text{vlc}} \leq c^{l_b}$ .  $\square$

In Theorem 5.40, we will show that bounded and strongly bounded spectra are equivalent concepts under Assumption 5.5. Note that the “boundness” of the VLC spectrum might depend on the source statistics (and thus also on the source/VLC mapping) for some VLCs, recall indeed Remark 5.11 and see Example 5.48.

It is worth noting the analogy between the concept of strongly bounded VLC spectrum and the concept of CC with non-catastrophic encoder. Indeed, for a non-catastrophic CC encoder, it has been shown in [106, Section I.B] and in [92, Theorem A.1]<sup>4</sup> that there exists a constant  $\eta$  such that there is no error event of weight  $h$  longer than  $\eta h$ . Put differently, there exists a constant  $\eta$  such that the spectrum of the CC equals zero,  $A_{h,l_b}^{\text{cc}} = 0$ , for all  $l_b \geq \eta h$ . The analogy with strongly bounded VLC spectrum is straightforward: Instead of being equal to zero above  $\eta h$ , a strongly bounded VLC spectrum converges exponentially toward zero above  $\eta h$ , which is equivalent asymptotically. In this context, VLCs with bounded spectrum will be proved in Section 5.7.4 to be non-catastrophic in average with the Levenshtein symbol distance.

The following two straightforward properties, given without proof, state the equivalence between  $A_{h,l_b}^{\text{vlc}}$  bounded and  $A_{h,l_s}^{\text{vlc}}$  bounded.

**Property 5.15.** A VLC spectrum is bounded if and only if  $\exists c < 1, \forall h, A_{h,l_s}^{\text{vlc}} = \mathcal{O}(c^{l_s})$ .  $\blacksquare$

<sup>4</sup>There is a small typo in [92, Theorem A.1]: read  $d\eta \geq L_0$  instead of  $d \geq L_0\eta$ .

**Property 5.16.** A VLC spectrum is strongly bounded if and only if  $\exists c < 1$ ,  $\exists \eta$ ,  $\forall h$ ,  $\forall l_s \geq \eta h$ ,  $A_{h,l_s}^{\text{vlc}} \leq c^{l_s}$ . ■

**Example 5.17.** Consider again the source/VLC of Example 5.9. For this source/VLC, we can calculate  $A_{s_L, h, l_s, l_b}^{\text{vlc}}$  analytically. Indeed, by enumerating all  $s$ : involved in (5.23), it is straightforward to notice that the only  $s$ : associated with a non-empty set  $\mathcal{A}_{s_L, h}^{\text{vlc}}$  have the following form:

$s$ :	$P(s)$	$l(s)$	$l_S(s)$	$\mathcal{A}_{s_L=2, h=2}^{\text{vlc}} _s$ .
$\underline{bb}$	$(1-p)^{m+1}$	$2m+2$	$m+1$	$\{\underline{aba}\}$
$\underline{aba}$	$p^2(1-p)^m$	$2m+2$	$m+2$	$\{\underline{bb}\}$
$\underline{bba}$	$p(1-p)^{m+1}$	$2m+3$	$m+2$	$\{\underline{abb}\}$
$\underline{abb}$	$p(1-p)^{m+1}$	$2m+3$	$m+2$	$\{\underline{bba}\}$

where  $\underline{b}$  is the symbol  $b$  repeated  $m$  times,  $m \geq 0$ . By substituting this into (5.23), we obtain

$$A_{s_L, h, l_s, l_b}^{\text{vlc}} = \begin{cases} p^2(1-p)^{l_s-2}, & \text{if } s_L = h = 2 \text{ and } l_b = 2l_s - 2 \geq 2, \\ 2p(1-p)^{l_s-1}, & \text{if } s_L = h = 2 \text{ and } l_b = 2l_s - 1 \geq 2, \\ (1-p)^{l_s}, & \text{if } s_L = h = 2 \text{ and } l_b = 2l_s \geq 2, \\ 0, & \text{otherwise.} \end{cases} \quad (5.27)$$

Note that this spectrum is bounded (Definition 5.13). Finally, by summing over all  $l_s, l_b$ , we get  $A_{s_L=2, h=2}^{\text{vlc}} = 1 + \frac{1}{p} - p$ . □

### 5.5.2.1 Levenshtein symbol error rate, $\text{SER}_L$

Given an integer  $l_s$  and the transmitted and decoded streams  $S_{1:\infty}$  and  $\check{S}_{1:\infty}$ , let  $\check{l}_s$  be the integer (random variable) that minimizes  $|l(S_{1:l_s}) - l(\check{S}_{1:\check{l}_s})|$ .

**Theorem 5.18** ( $\text{SER}_L$ ). Under Assumption 5.5, an upper bound on the  $\text{SER}_L$  of a VLC stream is

$$\text{SER}_L \triangleq \lim_{l_s \rightarrow +\infty} \mathbf{E} \left\{ \frac{d_{S_L}(S_{1:l_s}, \check{S}_{1:\check{l}_s})}{l_s} \right\} \quad (5.28)$$

$$\leq \sum_{h \geq 1} P_h \sum_{s_L \geq 1} s_L A_{s_L, h}^{\text{vlc}}, \quad (5.29)$$

where the expectation is taken over all semi-infinite streams  $S_{1:\infty}$  and  $\check{S}_{1:\infty}$ , and  $A_{s_L, h}^{\text{vlc}}$  is given in Definition 5.10. See proof in Appendix 5.B.2.  $\square$

Note Theorem 5.18 can be straightforwardly extended to the Damerau-Levenshtein distance [107].

The bound (5.29) on the  $\text{SER}_L$  was originally proposed in [14]. The proof given in Appendix 5.B.2 is however different and more general. In particular, the proof highlights that the inequality (upper bound) in (5.29) is due not only to the union bound, see (5.10), but also to the following property of the Levenshtein distance, which can be proved straightforwardly.

**Proposition 5.19** (Levenshtein symbol distance of events). *The Levenshtein symbol distance between  $s$  and  $\check{s}$  with  $l(s) = l(\check{s})$  is upper bounded by the sum of the Levenshtein symbol distances of the events in  $\mathcal{E}(\check{s}, s)$  (see Section 5.5.1),*

$$d_{S_L}(\check{s}, s) \leq \sum_{e \in \mathcal{E}(\check{s}, s)} d_{S_L}(e). \quad (5.30)$$

■

This bound is rather tight in practice. In most cases, (5.30) is indeed an equality. Still, it is a strict inequality with some particular combinations of error events. Here is an example.

**Example 5.20.** *Consider again Example 5.9. By Definition 5.7, we have two error events given by the pairs  $(aba, bb)$  and  $(bbba, abbb)$ , and a strict inequality in (5.30) since*

$$2 = d_{S_L}(abaaaabbba, bbaaaabbb) < d_{S_L}(aba, bb) + d_{S_L}(bbba, abbb) = 4.$$

$\square$

### 5.5.2.2 Symbol error rate, SER

Except in trivial cases (e.g., with fixed length codes), the concept of SER is generally pointless with (semi-)infinite VLC streams. This is mainly because, without proper synchronization mechanism, the average number of symbol errors is generally infinite given only a few bit errors.

Furthermore, even when the VLC stream is framed into a VLC block (Section 5.6), the SER remains an ambiguous concept. Indeed, since the number of

transmitted symbols  $N_s$  is assumed unknown at the receiver (Assumption 5.5), the frames  $S_i$  and  $\check{S}_i$  may have different symbol lengths, so the meaning of the symbol distance  $d_S(S_i, \check{S}_i)$  is ambiguous. The SER is considered later in Section 5.8.2.

### 5.5.2.3 Bit error rate, BER

The BER is not very informative w.r.t. the source distortion. Nevertheless a bound on the BER offers at least two possibilities: (i) computing the ratios  $\text{SER}_L/\text{BER}$  and  $\text{SER}/\text{BER}$ , which are measures of the desynchronization sensitivity of the VLC; (ii) comparing a VLC against an ECC.

**Theorem 5.21** (BER). *Under Assumption 5.5, an upper bound on the BER<sup>5</sup> of a VLC stream is*

$$\text{BER} \triangleq \lim_{l_b \rightarrow +\infty} \mathbf{E} \left\{ \frac{d_H(U_{1:l_b}, \check{U}_{1:l_b})}{l_b} \right\} \quad (5.31)$$

$$\leq \frac{1}{l} \sum_{h \geq 1} P_h h A_h^{\text{vlc}}, \quad (5.32)$$

where the expectation is taken over all semi-infinite streams  $U_{1:\infty}$  and  $\check{U}_{1:\infty}$ , and  $A_h^{\text{vlc}}$  is given in Definition 5.10. See proof in Appendix 5.B.3.  $\square$

### 5.5.2.4 Error event rate, EER

One performance measure of sequence (Viterbi) decoding on trellises is the EER.

<sup>5</sup>Another equivalent definition of the BER is

$$\text{BER} \triangleq \lim_{l \rightarrow +\infty} \mathbf{E} \left\{ \frac{1}{l_{\text{gcd}}^*} \sum_{i=0}^{l_{\text{gcd}}^* - 1} \mathbb{I}\{\check{U}_{l+i} \neq U_{l+i}\} \right\},$$

which simplifies into the intuitive expression

$$\text{BER} = \lim_{l \rightarrow +\infty} \mathbf{E}\{\mathbb{I}\{\check{U}_l \neq U_l\}\}$$

when  $l_{\text{gcd}}^* = 1$ . Note the limit is used only to avoid the non-stationary effect at the beginning of the semi-infinite stream. With infinite streams,  $U_{-\infty:+\infty}$ , the limit would not be needed. We will come back to this definition of the BER in Appendix 5.B.3.

**Theorem 5.22 (EER).** *Under Assumption 5.5, an upper bound on the probability for a VLC stream to have an error event starting with  $S_k$ , averaged over all  $k$ , is*

$$\text{EER}^s \triangleq \lim_{l_s \rightarrow +\infty} \mathbf{E} \left\{ \frac{d_{EE}(S_{1:l_s}, \check{S}_{1:l_s})}{l_s} \right\} \quad (5.33)$$

$$\leq \sum_{h \geq 1} P_h A_h^{\text{vlc}}, \quad (5.34)$$

where the expectation is taken over all semi-infinite streams  $S_{1:\infty}$  and  $\check{S}_{1:\infty}$ ,  $A_h^{\text{vlc}}$  is given in Definition 5.10 and  $d_{EE}(S_{1:l_s}, \check{S}_{1:l_s})$  counts the number of error events involved in the sub-sequences  $S_{1:l_s}$  and  $\check{S}_{1:l_s}$ , i.e.,  $d_{EE}(S_{1:l_s}, \check{S}_{1:l_s}) = |\{e \in \mathcal{E}(S_{1:\infty}, \check{S}_{1:\infty}) : e \text{ starts with } S_k, k \leq l_s\}|$ .

Similarly, recalling that error events can start only (in probability) with bits  $U_l$  subject to  $l \equiv 1 \pmod{l_{\text{gcd}}^*}$  (see Remark 5.8), an upper bound on the probability to have an error event starting with  $U_l$ , averaged over all  $l \equiv 1 \pmod{l_{\text{gcd}}^*}$ , is<sup>6</sup>

$$\text{EER}^b \triangleq \lim_{l_b \rightarrow +\infty} \mathbf{E} \left\{ \frac{d_{EE}(U_{1:l_b}, \check{U}_{1:l_b})}{\lfloor l_b / l_{\text{gcd}}^* \rfloor} \right\} \quad (5.35)$$

$$\leq \frac{l_{\text{gcd}}^*}{l} \sum_{h \geq 1} P_h A_h^{\text{vlc}}, \quad (5.36)$$

where the expectation is taken over all semi-infinite streams  $U_{1:\infty}$  and  $\check{U}_{1:\infty}$ , and  $d_{EE}(U_{1:l_b}, \check{U}_{1:l_b}) = |\{e \in \mathcal{E}(U_{1:\infty}, \check{U}_{1:\infty}) : e \text{ starts with } U_l, l \leq l_b\}|$ . See proof in Appendix 5.B.4.  $\square$

**Example 5.23.** *Coming back to Example 5.17, we have:  $\text{SER}_L \leq 2f(p)$ ,  $\text{BER} \leq 2\frac{f(p)}{2-p}$ ,  $\text{EER}^b \leq \frac{f(p)}{2-p}$  and  $\text{EER}^s \leq f(p)$ , where  $f(p) = P_{h=2} \frac{1+p-p^2}{p}$ .  $\square$*

### 5.5.3 Numerical evaluation of the VLC spectrum

Due to the non-linearity of the VLC, there is no straightforward simplification to evaluate numerically the VLC spectrum given in Definition 5.10. The expression of the VLC spectrum must be evaluated as is, i.e., by enumerating all possible sequences  $s$ : and by computing the conditional spectrum for all of them.

<sup>6</sup>By contrast, a Balakirsky error event, see (5.22), can start at any bit  $U_{l'}$  and it is straightforward to show that the probability to start a Balakirsky error event with the bit  $U_{l'}$ , averaged over all  $l'$ , is given by  $\text{EER}^b / l_{\text{gcd}}^*$ .

In practice, we limit the complexity by computing the spectrum up to some maximum  $h$ ,  $l_s$  and  $l_b$ . This has an impact on the tightness of the performance bounds since this is equivalent to truncating the bounds to the first few terms. Fortunately, this impact is limited for two reasons. Firstly, the pairwise error probability  $P_h$  decreases exponentially at high signal to noise ratios for increasing  $h$ . Secondly, given a value of  $h$ , the VLC spectrum decreases exponentially for increasing  $l_b, l_s$  above a certain threshold when the VLC spectrum is bounded (Definition 5.13). As we will see in Theorem 5.40, the VLC spectrum is bounded under Assumption 5.5 as soon as the VLC has a synchronizing sequence of strictly positive probability. Besides, the value of  $c$  in Definition 5.13 is smaller, and thus the VLC spectrum decreases faster, if the VLC has a higher probability to resynchronize.

Consequently, for a given level of accuracy, the VLC spectrum evaluation can be stopped/truncated sooner, and thus requires less computations, if the VLC has a higher probability to resynchronize and if the channel signal to noise ratio is higher. As a corollary, the spectrum evaluation requires generally more computation when the source/VLC Markov model has a larger memory. This is because the probability to resynchronize is generally lower when the memory is larger, except if the increase in memory is associated with better synchronizing and error correcting capabilities.

At last, though there is no straightforward simplification to evaluate the spectrum, there are still smart ways to implement the underlying enumeration. We refer the reader to [14, 108], where two algorithms are proposed<sup>7</sup>. The former algorithm follows somehow a breadth first strategy, by exploring all pairs of paths  $(s, \check{s})$  in parallel. It is computationally efficient but memory-intensive. The latter algorithm follows rather a depth first strategy, by exploring sequentially all paths  $s$ : and by enumerating, for each of them, all paths  $\check{s}$ : forming an error event with  $s$ :. It is memory efficient but performs more computations than the former algorithm. In a sense, this latter algorithm is very close to the mathematical definition (5.23)–(5.24) of the VLC spectrum.

---

<sup>7</sup>More precisely, these algorithms compute the quantity  $\sum_{s_L} s_L A_{s_L, h}$  in [14, 108] but can be extended straightforwardly to compute the spectrum  $A_{s_L, h, l_s, l_b}^{\text{vlc}}$  given in Definition 5.10.

## 5.6 Performance bounds for VLC blocks

In Fig. 5.2, the semi-infinite VLC stream is framed into VLC blocks according to a certain *framing rule*, see Section 5.3.2.

**Definition 5.24.** *Under Assumption 5.5, the spectrum of a VLC block  $B_{\text{vlc}}$  framed according to a certain rule  $F$  is*

$$A_{s_L, h}^F \triangleq \sum_{s:} P_F(B_{\text{vlc}} = s:) A_{s_L, h|s:}^{\text{Bvlc}}, \quad (5.37)$$

where the conditional spectrum  $A_{s_L, h|s:}^{\text{Bvlc}}$  is given by

$$A_{s_L, h|s:}^{\text{Bvlc}} = \left| \left\{ \check{s}: \begin{array}{l} d_H(s:, \check{s}:) = h, \\ d_{s_L}(s:, \check{s}:) = s_L, \\ l(\check{s}:) = l(s:) \end{array} \right\} \right|. \quad (5.38)$$

□

In this section, we present two framing rules. The former rule has limited practical applications but the developments behind it are easier to understand and help to introduce the developments of the more practical latter rule. To get the spectra of the resulting VLC blocks — the direct computation of (5.37) being particularly difficult — we give transformations that approximate them based on the spectrum of the VLC stream. These transformations are similar to the method given in [15] for CCs. We then use the approximated spectra to get performance estimations of the frame-based system in Fig. 5.2.

These transformations are possible because the VLC stream and the VLC block share the same error events. Indeed, two different codewords of a VLC block can always be considered as a succession of pairs of sub-sequences that either are identical or form an error event. Because of that, there is obviously a link between the spectrum of a VLC stream and the spectrum of a VLC block.

The approximated spectra of the VLC blocks, resulting from these transformations, are asymptotically tight with the real spectra. However, to guarantee that they give upper bounds when they are used in union bounds, we introduce the concept of upper spectrum.

**Definition 5.25.** *We say  $A_{s_L, h}^{F, \text{upp}}$  is an upper spectrum of the VLC block and we write  $A_{s_L, h}^F \preceq A_{s_L, h}^{F, \text{upp}}$  if*

$$\sum_{s_L \geq s} A_{s_L, h}^F \leq \sum_{s_L \geq s} A_{s_L, h}^{F, \text{upp}} \quad (5.39)$$

for any  $s, h \in \mathbb{N}_{>0}$ .

□

By induction, it follows straightforwardly from (5.39) that

$$\sum_{s_L \geq s} s_L A_{s_L, h}^F \leq \sum_{s_L \geq s} s_L A_{s_L, h}^{F, \text{upp}} \quad (5.40)$$

for any  $s, h \in \mathbb{N}_{>0}$ . The inequalities (5.39), (5.40) imply that  $A_{s_L, h}^{F, \text{upp}}$  can be used to get upper bounds on the BER, on the  $\text{SER}_L$  and on the FER, which is particularly useful in Section 5.7.

### 5.6.1 Framing rule $F_s$ , definition and properties

Assume the VLC stream has been processed up to a certain symbol. The symbol and bit indices have been shifted such that the next symbol to be processed is  $S_1$  and the next bit is  $U_1$ . Then, given a fixed  $N_s$  and a fixed  $N \geq N_s l_{\max}^*$ , let  $F_s$  be the rule that forms the next VLC block with  $S_{1:N_s}$ , that sets  $N_b = l(S_{1:N_s})$ ,  $U_{1:N_b} = \text{vlc}(S_{1:N_s})$  and, if  $N_b < N$ , that appends  $N - N_b$  zeros to  $U_{1:N_b}$  to get  $U_{1:N}$ . In the following,  $U_i$  refers to  $U_{1:N_b}$ , not to  $U_{1:N}$ .

This framing rule uses a constant number  $N_s$  of symbols in each VLC block, a variable number  $N_b$  of VLC bits and a constant length  $N$  (the length of the interleaver in Fig. 5.2). Let  $B_{\text{vlc}}$  be the random variable of the VLC block. Under Assumption 5.5, the probability of a realization  $B_{\text{vlc}} = u_i \equiv s_i$  is trivially given by

$$P_{F_s}(B_{\text{vlc}} = s_i) = \mathbb{I}\{l_S(s_i) = N_s\} \prod_{k=1}^{l_S(s_i)} P(S = s_k). \quad (5.41)$$

Also, by the central limit theorem, the distribution of  $N_b/N_s$  converges toward the normal distribution  $\mathcal{N}(\bar{l}, \sigma_l^2/N_s)$  as  $N_s \rightarrow +\infty$  — recall  $\sigma_l^2$  is the variance of the VLC lengths. Note the variance  $\sigma_l^2/N_s$  converges to 0 as  $N_s \rightarrow +\infty$ . In other words, the most probable values of  $N_b$  are concentrated around  $N_s \bar{l}$  for large  $N_s$ .

This framing rule has limited practical applications for two reasons. Firstly, the number of appended zeros,  $N - N_b$ , can be large and is a waste in bandwidth. Secondly, even though  $N_s$  is constant, the decoder is assumed not to know its value in Assumption 5.5 — however, it is worth noting that a decoder knowing and using its value jointly with the value of  $N_b$  is of much higher complexity for large  $N_s$  (e.g., quadratic in  $N_s$  instead of linear, for the Viterbi algorithm on an optimal source/VLC trellis).

### 5.6.2 Spectrum of VLC blocks with framing rule $F_s$

**Theorem 5.26** (Approximate spectrum of VLC blocks,  $F_s$ ). *Given a VLC stream, the spectrum  $A_{s_L, h}^{F_s}$  of the VLC block can be approximated, under Assumption 5.5, by*

$$A_{s_L, h}^{F_s, \text{app}} \triangleq \sum_{\substack{l_s, l_b, n \\ l_s \leq N_s}} \binom{N_s - l_s + n}{n} T_{s_L, h, l_s, l_b, n}^{\text{vlc}} \quad (5.42)$$

where  $T_{s_L, h, l_s, l_b, n}^{\text{vlc}}$  is the coefficient of the polynomial

$$\begin{aligned} T^{\text{vlc}}(S_L, H, L_s, L_b, \Omega) &\triangleq \sum_{s_L, h, l_s, l_b, n} T_{s_L, h, l_s, l_b, n}^{\text{vlc}} S_L^{s_L} H^h L_s^{l_s} L_b^{l_b} \Omega^n \\ &= \sum_{n \geq 1} \left( A^{\text{vlc}}(S_L, H, L_s, L_b) \Omega \right)^n, \\ A^{\text{vlc}}(S_L, H, L_s, L_b) &\triangleq \sum_{s_L, h, l_s, l_b} A_{s_L, h, l_s, l_b}^{\text{vlc}} S_L^{s_L} H^h L_s^{l_s} L_b^{l_b}, \end{aligned} \quad (5.43)$$

where  $A_{s_L, h, l_s, l_b}^{\text{vlc}}$  is the spectrum of the VLC stream (Definition 5.10). See proof in Appendix 5.C.1.  $\square$

The fact that  $A_{s_L, h}^{F_s, \text{app}}$  is an approximation of the real spectrum  $A_{s_L, h}^{F_s}$  is only due to the upper bound (5.30). It follows trivially that the approximate spectrum  $A_{s_L, h}^{F_s, \text{app}}$  is an upper spectrum, i.e.,  $A_{s_L, h}^{F_s} \preceq A_{s_L, h}^{F_s, \text{app}}$ , see Definition 5.25. Besides, note that this spectrum  $A_{s_L, h}^{F_s, \text{app}}$  satisfies

$$\sum_{s_L} A_{s_L, h}^{F_s} = \sum_{s_L} A_{s_L, h}^{F_s, \text{app}}, \quad (5.44)$$

for any  $h \in \mathbb{N}_{>0}$ .

Here is an interpretation of (5.42), which makes the link with the developments for convolutional codes in [15, Appendix]. Let us take again Examples 5.9 and 5.17. Consider a single error event  $e$  in the VLC block, formed, e.g., by the pair of sub-sequences  $(aba, bb)$  where  $aba$  is the transmitted sub-sequence and  $bb$  the decoded sub-sequence. It has symbol length  $l_s = 3$  and can start therefore at any position  $j \in \{1, 2, \dots, N_s - 2\}$ , i.e., there are  $\binom{N_s - l_s + 1}{1}$  possible positions. Given a position  $j$ , the probability of sequences  $S_{1:N_s}$  subject to  $S_{j:j+2} = aba$  is precisely

$$\sum_{\substack{s_{1:N_s} \text{ s.t.} \\ s_{j:j+2} = aba}} P_{F_s}(S_{1:N_s} = s_{1:N_s}) = P(S_{j:j+2} = aba),$$

i.e.,  $P(S_{1:3} = aba)$  or more simply  $P(aba)$ . By repeating this with all error events  $(s_{1:3}, \check{s})$  of symbol length  $l_s = 3$ , we find

$$\sum_{s_{1:3}} \binom{N_s - l_s + 1}{1} P(s_{1:3}) A_{s_L, h, l_s | s_{1:3}}^{\text{vlc}} = \binom{N_s - l_s + 1}{1} A_{s_L, h, l_s}^{\text{vlc}}.$$

Next, consider two error events in the VLC block, e.g., firstly  $e_1 = (aba, bb)$  and then  $e_2 = (ab, ba)$  (in this order). They have a total symbol length  $l_s = l_S(e_1) + l_S(e_2) = 5$ . The event  $e_1$  can start therefore at any position  $j_1 \in \{1, 2, \dots, N_s - 4\}$ . The event  $e_2$ , of symbol length 2, can start at any of the remaining positions  $j_2$  after  $e_1$ , i.e.,  $j_2 \in \{j_1 + 3, j_1 + 4, \dots, N_s - 1\}$ . By combinatorics, there are thus  $\binom{N_s - l_s + 2}{2}$  possible pairs of positions  $(j_1, j_2)$ . Given a pair of positions  $(j_1, j_2)$ , the probability of sequences  $S_{1:N_s}$  subject to  $S_{j_1:j_1+2} = aba$  and to  $S_{j_2:j_2+1} = ab$  is precisely  $P(aba)P(ab)$ . By repeating this with all pairs of error events  $e_1$  and  $e_2$  subject to  $l_S(e_1) + l_S(e_2) = l_s$ ,  $l(e_1) + l(e_2) = l_b$ ,  $d_H(e_1) + d_H(e_2) = h$ ,  $d_{S_L}(e_1) + d_{S_L}(e_2) = s_L$ , we find

$$\binom{N_s - l_s + 2}{2} T_{s_L, h, l_s, l_b, n=2}^{\text{vlc}}$$

and so on with any number  $n$  of error events.

### 5.6.3 Framing rule $F_b$ , definition and properties

Assume again the VLC stream has been processed up to a certain symbol, and the symbol and bit indices have been shifted such that the next symbol to be processed is  $S_1$  and the next bit is  $U_1$ . Then, given a fixed  $N$ , let  $F_b$  be the rule that forms the next VLC block with  $S_{1:N_s}$  ( $N_s$  is variable) subject to

$$\begin{aligned} N_b = l(S_{1:N_s}) &\leq N, \\ N_b + l(S_{N_s+1}) &> N, \end{aligned} \tag{5.45}$$

where  $S_{N_s+1}$  becomes the first symbol of the next VLC block. If  $N_b < N$ ,  $N - N_b$  zeros are appended to  $U_{1:N_b}$  to get  $U_{1:N}$ . In the following,  $U$  refers to  $U_{1:N_b}$ , not to  $U_{1:N}$ .

This framing rule uses an interleaver  $\Pi$  of fixed length  $N$ . Roughly speaking, given a maximum VLC block size  $N$ , we fill the block with VLC codewords until a codeword that does not fit in the block is found; that codeword becomes the first one of the next VLC block.

By (5.45), the admissible values of  $N_b$  are subject to

$$\begin{aligned} N_b &\equiv 0 \pmod{l_{\text{gcd}}}, \\ N - l_{\text{max}} &< N_b \leq N, \end{aligned} \quad (5.46)$$

and the admissible values of non-zero probability are subject to

$$\begin{aligned} N_b &\equiv 0 \pmod{l_{\text{gcd}}^*}, \\ N - l_{\text{max}}^* &< N_b \leq N. \end{aligned} \quad (5.47)$$

The maximum value of  $N_b$  having a non-zero probability is therefore  $N_b^{\text{max}} \triangleq l_{\text{gcd}}^* \lfloor N/l_{\text{gcd}}^* \rfloor$ . Let  $B_{\text{vlc}}$  be the random variable of the VLC block. Under Assumption 5.5, the probability of a realization  $B_{\text{vlc}} = u: \equiv s$  depends on the value of  $N_b$  of the previous block, which we denote  $N_b^{(-1)}$ . The first VLC block does not depend on any previous block, which is equivalent to  $N_b^{(-1)} = N_b^{\text{max}}$ . We can show, see Appendix 5.C.2, that the framing rule  $F_b$  satisfies the following properties under Assumption 5.5.

**Property 5.27** (conditional distribution of  $B_{\text{vlc}}$ ).

$$\begin{aligned} &P_{F_b}(B_{\text{vlc}} = s: | N_b^{(-1)}) \\ &= \mathbb{I}\{N - l_{\text{max}}^* < l(s) \leq N\} \left( \prod_{k=1}^{l_S(s)} P(S = s_k) \right) \\ &\times \frac{\mathbb{I}\{l(s_1) > N - N_b^{(-1)}\}}{P(l(S) > N - N_b^{(-1)})} P(l(S) > N - l(s)). \end{aligned} \quad (5.48)$$

□

**Property 5.28** (distribution of  $N_b$ ). For  $n_b$  subject to (5.47),

$$P_{F_b}(n_b | N_b^{(-1)}) = \frac{l_{\text{gcd}}^*}{\bar{l}} P(l(S) > N - n_b) \pm \mathcal{O}(\theta^{n_b}), \quad (5.49)$$

$$P_{F_b}(n_b) = \frac{l_{\text{gcd}}^*}{\bar{l}} P(l(S) > N - n_b) \pm \mathcal{O}(\theta^{n_b}), \quad (5.50)$$

where  $0 \leq \theta < 1$ . See Lemma 5.59 for the exact value of  $\theta$ . □

By averaging (5.48) over  $N_b^{(-1)}$  with  $P_{F_b}(N_b^{(-1)} = n_b) = P_{F_b}(N_b = n_b)$ , we obtain the stationary probability of  $B_{\text{vlc}}$ , a good approximation of which for large  $N$  is the following.

**Property 5.29** (stationary distribution of  $B_{\text{vlc}}$ ).

$$P_{F_b}(B_{\text{vlc}} = s_1) \approx \mathbb{I}\{N - l_{\max}^* < l(s_1) \leq N\} \times \left( \prod_{k=1}^{l_s(s_1)} P(S = s_k) \right) \frac{l(s_1)}{\bar{l}} P(l(S) > N - l(s_1)). \quad (5.51)$$

□

**Property 5.30** (admissible values of  $N_b$  and  $N_s$ ). By (5.46), the admissible values of  $N_b$  are close to each other for  $N \gg l_{\max}$  and are concentrated around the average

$$\bar{N}_b \triangleq \sum_{n_b} n_b P_{F_b}(n_b) \approx N_b^{\max} + \frac{l_{\text{gcd}}^*}{2} - \frac{\sigma_l^2}{2\bar{l}} - \frac{\bar{l}}{2}, \quad (5.52)$$

where  $N_b^{\max} = l_{\text{gcd}}^* \lfloor N/l_{\text{gcd}}^* \rfloor$  — recall  $\sigma_l^2$  is the variance of the VLC lengths. Similarly, the most probable values of  $N_s$  are concentrated around  $\bar{N}_b/\bar{l}$  for large  $N$ : the ratio<sup>8</sup>  $N_s/\bar{N}_b$  is asymptotically normal as  $N \rightarrow +\infty$ , with

$$\frac{N_s}{\bar{N}_b} \rightarrow \mathcal{N}\left(\frac{1}{\bar{l}}, \frac{\sigma_l^2}{\bar{l}^3 \bar{N}_b}\right). \quad (5.53)$$

□

#### 5.6.4 Spectrum of VLC blocks with framing rule $F_b$

**Theorem 5.31** (Approximate spectrum of VLC blocks,  $F_b$ ). Given a VLC stream with bounded spectrum  $A_{s_L, h, l_s, l_b}^{\text{vlc}}$ , the spectrum  $A_{s_L, h}^{F_b}$  of the corresponding VLC block can be approximated, for small  $h$  and asymptotically large  $N$ , under Assumption 5.5, by

$$A_{s_L, h}^{F_b, \text{app}} \triangleq \sum_{\substack{l_s, l_b, n \\ l_b \leq N_b^{\max}}} \left( \frac{N_b^{\max}}{l_{\text{gcd}}^*} - \frac{l_b}{l_{\text{gcd}}^*} + n \right) \left( \frac{l_{\text{gcd}}^*}{\bar{l}} \right)^n T_{s_L, h, l_s, l_b, n}^{\text{vlc}} \quad (5.54)$$

where  $T_{s_L, h, l_s, l_b, n}^{\text{vlc}}$  is given in Theorem 5.26. See proof in Appendix 5.C.3. □

Since  $A_{s_L, h}^{F_b, \text{app}}$  is a good approximation for small  $h$  and large  $N$ , it can be used to get approximate performance predictions at high SNRs for concatenated systems with sufficiently long interleavers. But because of the approx-

<sup>8</sup>The ratio  $N_s/\bar{N}_b$  is an arbitrary choice. For example, the ratios  $N_s/N$  and  $N_s/N_b^{\max}$  have distributions that converge also to the same normal distribution as  $N$  approaches  $+\infty$ .

imations made, we cannot guarantee that it is an upper spectrum (Definition 5.25). Nonetheless, the spectrum

$$A_{s_L, h}^{F_b, \text{upp}} \triangleq \frac{l^*_{\max}}{l^*_{\text{gcd}}} \sum_{\substack{l_s, l_b, n \\ l_b \leq N}} \binom{\frac{N_b^{\max}}{l^*_{\text{gcd}}} - \frac{l_b}{l^*_{\text{gcd}}} + n}{n} T_{s_L, h, l_s, l_b, n}^{\text{vlc}} \quad (5.55)$$

is an upper spectrum, see proof in Appendix 5.C.3.4. It can thus be used with the union bound to get strict upper performance bounds (though non-tight).

At last, here is an intuitive interpretation of (5.54), which is very similar to the interpretation given in Section 5.6.2 for the framing rule  $F_s$ . Let us take back Examples 5.9 and 5.17. Consider a single error event in the VLC block, e.g.,  $e = (aba, bb)$ . It has bit length  $l_b = 4$  and can start therefore at any position  $j \in \{1, 2, \dots, N_b^{\max} - 3\}$ , i.e., there are  $\binom{N_b^{\max} - l_b + 1}{1}$  possible positions. The main difference with the framing rule  $F_s$  is that to have an error event starting with  $U_j$ , the bit  $U_j$  must be the first bit of a codeword in  $U$ , i.e., we must have  $U_{1:j-1} \in \mathcal{V}^+$  or, equivalently,  $\rho_j(U, U)$ . Therefore, given a position  $j$ , the probability of sequences  $U$ : subject to  $U_{j:j+3} = \text{vlc}(aba) = 0110$  and such that  $U_j$  is the first bit of a codeword in  $U$ , is

$$\sum_{\substack{u: \text{s.t.} \\ u_{j:j+3} = 0110 \\ \rho_j(u, u)}} P_{F_b}(U = u) = P_{F_b}(U_{j:j+3} = 0110, \rho_j(U, U)).$$

By factorization, this is equal to

$$P_{F_b}(U_{j:j+3} = 0110 | \rho_j(U, U)) P_{F_b}(\rho_j(U, U))$$

where the first factor can be simplified for  $j > 1$  as

$$\begin{aligned} P_{F_b}(U_{j:j+3} = 0110 | \rho_j(U, U)) &= P(U_{1:4} = 0110 | \rho_1(U, U)) \\ &= P(U_{1:4} = 0110) \\ &= P(aba), \end{aligned}$$

and the second factor can be approximated for  $j \gg 1$  by

$$P_{F_b}(\rho_j(U, U)) \approx 1/\bar{l}.$$

For details, see the proof, Lemma 5.59 and Lemma 5.62. By repeating this with all error events  $(u_{1:4}, \check{u}_{1:4})$  of length  $l_b = 4$ , we find

$$\sum_{u_{1:4}} \binom{N_b^{\max} - l_b + 1}{1} \frac{1}{\bar{l}} P(u_{1:4}) A_{s_L, h, l_b | u_{1:4}}^{\text{vlc}} = \binom{N_b^{\max} - l_b + 1}{1} \frac{1}{\bar{l}} A_{s_L, h, l_b}^{\text{vlc}}.$$

Next, consider two error events in the VLC block, e.g., firstly  $e_1 = (aba, bb)$  and then  $e_2 = (ab, ba)$  (in this order). They have total bit length  $l_b = l(e_1) + l(e_2) = 7$ . The event  $e_1$  can start therefore at any position  $j_1 \in \{1, 2, \dots, N_b^{\max} - 6\}$ . The event  $e_2$ , of bit length 3, can start at any of the remaining positions  $j_2$  after  $e_1$ , i.e.,  $j_2 \in \{j_1 + 4, j_1 + 5, \dots, N_b^{\max} - 2\}$ . By combinatorics, there are thus  $\binom{N_b^{\max} - l_b + 2}{2}$  possible pairs of positions  $(j_1, j_2)$ . Given a pair of positions  $(j_1, j_2)$ , the probability of sequences  $U$ , subject to  $U_{j_1:j_1+3} = \text{vlc}(aba)$ ,  $U_{j_2:j_2+2} = \text{vlc}(ab)$ ,  $\rho_{j_1}(U, U)$  and  $\rho_{j_2}(U, U)$ , is approximately  $P(aba) P(ab) (1/\bar{l})^2$  for  $j_2 \gg j_1 \gg 1$ . By repeating this with all pairs of error events  $e_1$  and  $e_2$  subject to  $l_S(e_1) + l_S(e_2) = l_s$ ,  $l(e_1) + l(e_2) = l_b$ ,  $d_H(e_1) + d_H(e_2) = h$ ,  $d_{S_L}(e_1) + d_{S_L}(e_2) = s_L$ , we find approximately

$$\binom{N_b^{\max} - l_b + 2}{2} \left(\frac{1}{\bar{l}}\right)^2 T_{s_L, h, l_s, l_b, n=2}^{\text{vlc}}$$

and so on with any number  $n$  of error events.

### 5.6.5 Spectrum comparison between the rules $F_b$ and $F_s$

Experimental computations show that  $A_{s_L, h}^{F_s, \text{app}}$  in (5.42) and  $A_{s_L, h}^{F_b, \text{app}}$  in (5.54) are approximately equal for small  $h$  and large  $N$ , if we compare them with the same average number of symbols or the same average number of bits per frame.

A short analytical interpretation is the following. Let us take  $N_s = \lfloor \overline{N_b} / \bar{l} \rfloor$  for the framing rule  $F_s$ , where the ratio  $\overline{N_b} / \bar{l}$  is approximately the average number of symbols of the framing rule  $F_b$  (see Property 5.30). This way, the two framing rules are compared with the same average number of symbols per frame. Let us assume  $h$  is small. Then, by definition of  $T_{s_L, h, l_s, l_b, n}^{\text{vlc}}$ , the number  $n \leq h$  of error events is small. Besides, for VLCs with bounded spectrum and for  $n$  small, there exists a threshold on  $l_s, l_b$  above which  $T_{s_L, h, l_s, l_b, n}^{\text{vlc}}$  becomes negligible (for details, see Lemma 5.64). So we can focus on small values of  $n, l_s$  and  $l_b$ . Eventually, for large  $N$ , the following two factors in (5.54) can be developed as

$$\left(\frac{\frac{N_b^{\max}}{l_{\text{gcd}}^*} - \frac{l_b}{l_{\text{gcd}}^*} + n}{n}\right) \left(\frac{l_{\text{gcd}}^*}{\bar{l}}\right)^n \stackrel{(a)}{\approx} \frac{\left(\frac{N_b^{\max}}{\bar{l}} - \frac{l_b}{\bar{l}} + n \frac{l_{\text{gcd}}^*}{\bar{l}}\right)^n}{n!}$$

$$\stackrel{(b)}{\approx} \frac{(N_s - l_s + n)^n}{n!} \stackrel{(a)}{\approx} \binom{N_s - l_s + n}{n},$$

which reminds us the expression (5.42). Note (a) the approximation  $\binom{N}{n} \approx \frac{N^n}{n!}$  holds for  $n \ll N$ ; (b)  $N_b^{\max}$  is close to  $\bar{N}_b$  for large  $N$ , thus  $N_b^{\max} / \bar{N}_b \approx N_s$ .

### 5.6.6 VLC block concatenated with a linear code

Given the spectrum  $A_{s_L, h}^F$  of a VLC block for a given framing rule  $F$ , we can use (5.13) to calculate the spectrum  $A_{s_L, w, h}^{\text{gc}}$  of the global code (GC) in Fig. 5.2, i.e., the spectrum of the VLC block serially concatenated with the linear ECC. Indeed, by linearity of the ECC, it is self-evident that (5.13) holds with the conditional spectrum, i.e.,

$$A_{s_L, w, h|s}^{\text{gc}} = \frac{A_{s_L, h=w|s}^{\text{Bvlc}} A_{w, h}^{\text{ecc}}}{\binom{N}{w}}. \quad (5.56)$$

Note the summation of (5.13) is not used here because we want to keep the dependence w.r.t. the number  $w$  of VLC bit errors. Then, by (5.37) and by linearity of (5.56) in the conditional VLC block spectrum,

$$A_{s_L, w, h}^{\text{gc}} = \sum_{s:} P_F(s) A_{s_L, w, h|s}^{\text{gc}} = \frac{A_{s_L, h=w}^F A_{w, h}^{\text{ecc}}}{\binom{N}{w}}. \quad (5.57)$$

At last, by linearity again, this spectrum can be used in the union bounds (5.11)–(5.12) to get performance bounds on the frame-ML decoding of the global code, as summarized in next theorem.

**Theorem 5.32** (Global code performance bounds). *Under Assumption 5.5, the performance of the frame-ML decoding of the global code in Fig. 5.2 is upper bounded by the union bounds*

$$\text{FER} \leq \sum_h P_h \sum_{s_L, w} A_{s_L, w, h'}^{\text{gc}} \quad (5.58)$$

$$\text{SER}_L \leq \sum_h P_h \sum_{s_L, w} \frac{s_L}{N_s^{\min}} A_{s_L, w, h'}^{\text{gc}} \quad (5.59)$$

$$\text{BER} \leq \sum_h P_h \sum_{s_L, w} \frac{w}{N_b^{\min}} A_{s_L, w, h'}^{\text{gc}} \quad (5.60)$$

where  $N_s^{\min}$  and  $N_b^{\min}$  are the minimum admissible values of  $N_s$  and  $N_b$  having a non-zero probability, respectively, for the given framing rule  $F$ . ■

We have also straightforwardly the following corollary.

**Corollary 5.33.** *The upper bounds (5.59)–(5.60) become approximate performance estimations if we replace  $N_s^{\min}$  and  $N_b^{\min}$  by their average counterparts,  $\overline{N}_s$  and  $\overline{N}_b$ .* ■

Note that Theorem 5.32 and Corollary 5.33 provide the same performance estimation for the  $\text{SER}_L$  with the framing rule  $F_s$  because  $N_s = N_s^{\min} = \overline{N}_s$  since  $N_s$  is fixed. Similarly, they provide the same estimation for the BER with the framing rule  $F_b$  because it follows from (5.47) that  $N_b^{\min} \approx \overline{N}_b$  for large  $N$ .

At last, recall that the approximate spectrum  $A_{s_L,h}^{F_s,\text{app}}$  is an upper spectrum, see Definition 5.25. It can be used therefore in (5.57) instead of the real spectrum  $A_{s_L,h}^{F_s}$  and still lead to upper bounds in Theorem 5.32. By contrast, the approximate spectrum  $A_{s_L,h}^{F_b,\text{app}}$  is not necessarily an upper spectrum. We cannot guarantee therefore that (5.58)–(5.60) are still upper bounds if we use it instead of the real spectrum  $A_{s_L,h}^{F_b}$ . To ensure strict upper bounds (but untight!), the upper spectrum  $A_{s_L,h}^{F_b,\text{upp}}$  from (5.55) must be used instead. In practice, though, experimental results show that the obtained performance estimations with  $A_{s_L,h}^{F_b,\text{app}}$  are tight at moderate SNRs on the channel.

### 5.6.7 Remarks about non-concatenated VLC blocks

Consider a VLC block alone, non-concatenated, which is directly sent across the channel without interleaver and without ECC in Fig. 5.2. Then, note that the performance of the frame-ML decoding of such a VLC block is upper bounded by the bounds given in Section 5.5 for VLC streams. This is because less error events are possible with a VLC block, compared to a VLC stream. For example, an error event starting close to the end of the frame cannot have an arbitrary length since it must fit in the frame.

But the performance of such a VLC block is bounded also by the bounds of Theorem 5.32 here above. It is therefore interesting to make the link between these bounds and the bounds given in Section 5.5 for VLC streams.

Let us consider a VLC with bounded spectrum, the framing rule  $F_s$  and the  $\text{SER}_L$ . The  $\text{SER}_L$  is upper bounded by (5.59), i.e.,

$$\text{SER}_L \leq \sum_{s_L, h} \frac{s_L}{N_s} A_{s_L, h}^{F_s, \text{app}} P_h, \quad (5.61)$$

where we replaced  $N_s^{\min}$  by  $N_s$  since  $N_s$  is fixed, and  $A_{s_L, w, h}^{\text{gc}}$  by  $A_{s_L, h}^{F_s, \text{app}}$  since the VLC is not concatenated. Recall the expression (5.42) of  $A_{s_L, h}^{F_s, \text{app}}$  and let us truncate it to  $n = 1$ . Then, the right-hand side of (5.61) becomes

$$\sum_{s_L, h, l_s} \frac{s_L}{N_s} \binom{N_s - l_s + 1}{1} A_{s_L, l_s, h}^{\text{vlc}} P_h = \sum_{s_L, h, l_s} s_L \frac{N_s - l_s + 1}{N_s} A_{s_L, l_s, h}^{\text{vlc}} P_h. \quad (5.62)$$

Let  $N_s$  approach  $+\infty$ . Since the VLC is bounded, there exists a threshold on  $l_s$  above which the spectrum  $A_{s_L, l_s, h}^{\text{vlc}}$  can be considered negligible. For values of  $l_s$  below this threshold, the fraction  $(N_s - l_s + 1)/N_s$  converges to 1 as  $N_s$  approaches  $+\infty$ . Therefore, for  $N_s \rightarrow +\infty$ , the expression (5.62) becomes

$$\sum_{s_L, h, l_s} s_L A_{s_L, l_s, h}^{\text{vlc}} P_h, \quad (5.63)$$

which is exactly (5.29).

This illustrates two facts. Firstly, to upper bound the performance of the frame-ML decoding of a VLC block that is not concatenated, the bounds for VLC streams are tighter than the bounds for concatenated VLC blocks. Secondly — and this explains the first fact —, the union bound (5.61) contains many non mutually exclusive events. In particular, all terms for  $n \geq 2$  in the expression (5.42) of  $A_{s_L, h}^{F_s, \text{app}}$  are irrelevant and can be expurgated in (5.61). These terms correspond indeed to  $n$  single error events and are therefore already counted by single error events in (5.62). By contrast, when the VLC block is concatenated, no straightforward expurgation technique is known to eliminate the irrelevant terms because of the interleaver.

### 5.6.8 Related work

To the best of our knowledge, the distance spectrum of VLC blocks concatenated with linear ECCs has been first introduced in [72, 73], and later independently in [20].

In [72,73], another framing rule is introduced. Let us denote it by  $F_h$ . This rule is very similar to the rule  $F_s$  defined in Section 5.6.1: Given a fixed  $N_s$ , the rule  $F_h$  forms the next VLC block with  $S_{1:N_s}$  and sets  $N = N_b = I(S_{1:N_s})$ .

Compared to  $F_s$ , this framing rule  $F_h$  does not introduce any waste in bandwidth — recall  $F_s$  appends  $N - N_b$  zeros to  $U_{1:N_b}$ . Nonetheless, it is affected by another drawback in practice: The interleaver and the ECC in Fig. 5.2 need to be adapted to each realization of  $S_{1:N_s}$  since their length  $N$  depends on  $S_{1:N_s}$ . At last, note  $F_h$  shares the same drawback regarding the lack of knowledge of  $N_s$  at the decoder (although  $N_s$  is assumed constant).

Due to the randomness of  $N$ , it is suggested in [72, eq. (5)] to develop the global code spectrum conditionally to  $N$  and then to average it over  $N$ , i.e., with our notations,

$$A_{s_L, w, h}^{\text{gc}} = \sum_{n_b=N_b^{\min}}^{N_b^{\max}} P_{F_h}(n_b) \frac{A_{s_L, h=w|n_b}^{F_h} A_{w, h|n_b}^{\text{ecc}}}{\binom{n_b}{w}} \quad (5.64)$$

where  $N$  has been replaced by  $N_b$  since  $N = N_b$ ,  $A_{w, h|n_b}^{\text{ecc}}$  is the ECC spectrum subject to  $n_b$  input bits, and  $A_{s_L, h|n_b}^{F_h}$  is the conditional VLC block spectrum given  $N_b = n_b$ ,

$$A_{s_L, h|n_b}^{F_h} = \sum_{s: \in \mathcal{S}^{N_s}} P_{F_h}(s|N_b \triangleq I(S) = n_b) A_{s_L, h|s}^{\text{Bvlc}}$$

where  $A_{s_L, h|s}^{\text{Bvlc}}$  is defined in (5.38).

Unfortunately, the computation of (5.64) is difficult. Indeed, to the best of our knowledge, there is no simple approximate transformation such as (5.42) to calculate  $A_{s_L, h|n_b}^{F_h}$ . In [72], the authors suggest as a first approximation to consider only the average interleaver length  $\bar{N} = \bar{N}_b$  in (5.64). However, the calculation of  $A_{s_L, h|n_b}^{F_h}$  for the average interleaver length, which is required, is left unclear.

Based on the developments of this section, we can provide further information on this methodology, in particular how to compute  $A_{s_L, h|n_b}^{F_h}$  for the average interleaver length. By the central limit theorem, the most probable values of  $N_b$  are concentrated around  $N_s \bar{I}$  for large  $N_s$ , see Section 5.6.1. It is therefore reasonable to approximate (5.64) for the average interleaver length

$\bar{N} = \bar{N}_b$  as suggested in [72], i.e.,

$$A_{s_L, w, h}^{\text{gc}} \approx \frac{A_{s_L, h=w}^{F_h, \bar{N}_b} A_{w, h | \bar{N}_b}^{\text{ecc}}}{\binom{\bar{N}_b}{w}}. \quad (5.65)$$

Furthermore, the approximation  $A_{s_L, h}^{F_h, \bar{N}_b}$  of the VLC block spectrum can be calculated with both Theorem 5.26 and Theorem 5.31, since we are interested in the average interleaver length. It is self-evident that we can use Theorem 5.31 by replacing  $N_b^{\text{max}}$  with  $\bar{N}_b$ . But notice that the VLC block spectra for  $F_s$  and  $F_h$  are equal,  $A_{s_L, h}^{F_h} = A_{s_L, h}^{F_s}$ . So we can use also Theorem 5.26, which gives us almost the exact<sup>9</sup> VLC block spectrum — recall that the only source of inaccuracy is the upper bound (5.30). Either computation method is equivalent for small  $h$  and large  $N_s$ , see the comparison in Section 5.6.5.

To summarize, compared to [72,73], we have detailed two methods to compute approximately the VLC block spectrum from the VLC stream spectrum. These methods have been carefully proved and all approximations made have been highlighted. At last, the proposed bounds are tight with the simulation results at high SNRs, see Section 5.9.

## 5.7 Statistical synchronization, bounded spectrum and interleaving gains

The concept of bounded spectrum, Definition 5.13, has already been used several times so far; it is notably an assumption of Theorem 5.31. In the following, this concept will prove to be of even greater importance. In particular, we will show that a bounded spectrum is a sufficient condition to guarantee the so-called interleaving gains for serial concatenations.

It is certainly the right time to recall Remark 5.11, i.e., to recall that many concepts that we relate (with some abuse) to the VLC depend also on the source/VLC mapping and on the source (and thus on the source statistics).

<sup>9</sup>Having the exact spectrum  $A_{s_L, h}^{F_h}$  does not mean that we can calculate the exact spectrum of the concatenated code because the expression (5.65) remains an approximation. By contrast, the exact expression (5.64) requires the exact conditional VLC block spectra  $A_{s_L, h | n_b}^{F_h}$ , which we have not.

For example, for some VLCs, whether the VLC spectrum is bounded or not depends on the source statistics (see Example 5.48); for some other VLCs, the VLC spectrum is always bounded, independently of the source and of the source/VLC mapping.

### 5.7.1 Bounded spectrum and statistically synchronizable VLC

The concepts of statistically synchronizable<sup>10</sup> VLCs and of synchronizing sequences are thoroughly studied in [17]. In this section, we introduce a particular flavor of these concepts and then show that statistical synchronization and bounded spectrum (see Definition 5.13) are equivalent. This equivalence has the nice consequence that the algorithm of [17, Section V], which tests the statistical synchronization, tests also whether the VLC spectrum is bounded.

Compared to [17], the scope is here restricted to prefix VLCs and to binary errors. In particular, neither bit insertion nor bit deletion are considered. These restrictions make the concepts hereafter slightly simpler. In addition, still compared to [17], we make the concept of statistical synchronization depending on the considered decoder in Definition 5.34.

Let  $U_{1:\infty} = \text{vlc}(S_{1:\infty})$  be the transmitted stream, without channel code, and  $\check{U}_{1:\infty} = \text{vlc}(\check{S}_{1:\infty})$  the decoded stream, as in Fig. 5.1.

**Definition 5.34.** *Given a source and a decoder, a prefix VLC is statistically synchronizable, or more properly, statistically  $\rho$ -synchronizable, if*

$$\lim_{s \rightarrow +\infty} P(\tau_i(U_{1:\infty}, \check{U}_{1:\infty}) \leq s \mid \check{U}_{i:s-1} = U_{i:s-1}, U_{1:i-1} = u_{1:i-1}) = 1 \quad (5.66)$$

for any integer  $i$  and any  $u_{1:i-1} \in \text{prefix}(\mathcal{V}^+)$  with<sup>11</sup>  $P(u_{1:i-1}) > 0$ , independently of any channel errors, where  $\tau_i(U_{1:\infty}, \check{U}_{1:\infty})$  is the smallest integer  $t \geq i$  with  $\rho_t(U_{1:\infty}, \check{U}_{1:\infty})$ , see (5.18).  $\square$

Using today's terminology, note that the constraint  $\check{U}_{i:s-1} = U_{i:s-1}$  in (5.66) can be thought of as a perfect a priori constraint or as a genie-aided constraint, i.e., we are interested in the probability of synchronization when the

<sup>10</sup>Other terminologies from the literature are "completely self-synchronizing", "ergodic" or "synchronizing" VLCs.

<sup>11</sup> $P(u_{1:i-1})$  is the probability of starting the semi-infinite VLC stream with  $u_{1:i-1}$ .

decoder perfectly knows the bits  $U_{i:s-1}$ . Loosely speaking, a VLC is thus statistically synchronizable if, independently of any channel bit errors among the bits  $U_{1:i-1}$ , the decoder resynchronizes with high probability with the correct sequence provided that a sufficient number of correct bits ( $\check{U}_{i:s-1} = U_{i:s-1}$ ) are received. Note that a VLC is (deterministically) *synchronizable* if  $\tau_i(U_{1:\infty}, \check{U}_{1:\infty})$  is bounded, i.e., if there exists a finite  $t$  such that

$$P(\tau_i(U_{1:\infty}, \check{U}_{1:\infty}) \leq i + t \mid \check{U}_{i:i+t-1} = U_{i:i+t-1}, U_{1:i-1} = u_{1:i-1}) = 1.$$

In the following, we will focus on the sequence-ML decoder for stationary memoryless sources with at least two symbol values of strictly positive probability, as summarized in Assumption 5.5. Under these assumptions, the key characteristic in common between statistically synchronizable VLCs and VLCs with bounded spectrum is the synchronizing sequence, as stated in Theorem 5.40 hereafter.

**Definition 5.35.** A sequence  $s^s \equiv u^s$  is synchronizing if, for all  $\check{u} \in \mathcal{B}^+$  subject to  $l(\check{u}) \equiv 0 \pmod{l_{\text{gcd}}}$ , we have either  $\check{u}:u^s \in \mathcal{V}^+$  (admissibility) or  $\check{u}:u^s \notin \text{prefix}(\mathcal{V}^+)$  (inadmissibility).  $\square$

In other words, either  $\check{u}:u^s$  is an admissible sequence of VLC codewords, or it is an inadmissible sequence since it is not the prefix of any valid sequence of VLC codewords. Because of this, all decoders that consider only admissible sequences, e.g., the sequence-ML decoder, are forced to resynchronize as soon as a synchronizing sequence is received error-free.

Note that synchronization markers, often used in compression standards and inserted in the VLC bit stream to help the decoder resynchronize after some bit error(s), play at least the same role as the synchronizing sequence. Therefore, many results that follow hereafter when the source/VLC admits a synchronizing sequence of non-zero probability follow generally also when synchronization markers are used.

Roughly speaking, when the VLC has no synchronizing sequence, there exists intuitively at least one sequence from which we can never resynchronize, as stated contrapositively in Lemma 5.37 hereafter. We refer to such a particular sequence as an anti-synchronizing sequence.

**Definition 5.36.** A sequence  $u_{\cdot}^{\text{as}} \in \mathcal{B}^+$  is anti-synchronizing if its length is a multiple of  $l_{\text{gcd}}$  and if  $u_{\cdot}^{\text{as}} u_{\cdot} \in \text{prefix}(\mathcal{V}^+) \setminus \mathcal{V}^+$  for all  $u_{\cdot} \in \mathcal{V}^+$ .  $\square$

**Lemma 5.37.** Given a VLC, the following propositions are equivalent:

- a) the VLC has at least one synchronizing sequence;
- b) there is no sequence  $u_{\cdot}^{\text{as}} \in \mathcal{B}^+$  that is anti-synchronizing;
- c) there is no prefix  $p \in \text{prefix}(\mathcal{V})$  that is anti-synchronizing.

See proof in Appendix 5.D.1.  $\square$

Note that if  $u_{\cdot}^{\text{as}}$  is anti-synchronizing, then straightforwardly  $u_{\cdot} u_{\cdot}^{\text{as}} v_{\cdot}$  is anti-synchronizing for any  $u_{\cdot}, v_{\cdot} \in \mathcal{V}^+$ . Note also that if the prefix  $p$  exists in Lemma 5.37, then it belongs necessarily to the set  $\mathcal{X}_0$  defined in (5.8).

Since the source outputs only sequences  $u_{\cdot}$  of non-zero probability, it is interesting to restrict the concept of anti-synchronization to such sequences. In practice indeed, what matters is the existence of a synchronizing sequence of non-zero probability (otherwise, it never appears in the VLC stream).

**Definition 5.38.** A sequence  $u_{\cdot}^{\text{as}} \in \mathcal{B}^+$  is anti-synchronizing w.r.t. sequences of non-zero probability if its length is a multiple of  $l_{\text{gcd}}$  and if  $u_{\cdot}^{\text{as}} u_{\cdot} \in \text{prefix}(\mathcal{V}^+) \setminus \mathcal{V}^+$  for all  $u_{\cdot} \in \mathcal{V}^+$  with  $P(u_{\cdot}) > 0$ .  $\square$

**Lemma 5.39.** Given a VLC, the following propositions are equivalent under Assumption 5.4:

- a) the VLC has at least one synchronizing sequence of non-zero probability;
- b) there is no sequence  $u_{\cdot}^{\text{as}} \in \mathcal{B}^+$  that is anti-synchronizing w.r.t. sequences of non-zero probability;
- c) there is no prefix  $p \in \text{prefix}(\mathcal{V})$  that is anti-synchronizing w.r.t. sequences of non-zero probability.

See proof in Appendix 5.D.1.  $\square$

**Theorem 5.40.** Given a VLC, the following propositions are equivalent under Assumption 5.5:

- a) the VLC has at least one synchronizing sequence of non-zero probability;

- b) the VLC is statistically synchronizable;
- c) the VLC spectrum is bounded;
- d) the VLC spectrum is strongly bounded.

See proof in Appendix 5.D.2. □

**Corollary 5.41.** Under<sup>12</sup> Assumption 5.6, a VLC has at least one synchronizing sequence if and only if one of the equivalences of Theorem 5.40 is satisfied. ■

The implication “(a) implies (c)” is important. It is used notably in Section 5.7.3 to test whether a given VLC has a bounded spectrum, and deserves some interpretation. The key idea behind this implication is the following. Consider a VLC with a synchronizing sequence  $u^s$  of non-zero probability and consider a pair  $(U_{1:l_b}, \check{U}_{1:l_b})$  that forms an error event of Hamming distance  $h$ . Notice that, on the one hand,  $U_{1:l_b}$  cannot contain more than  $h$  occurrences of  $u^s$ . Otherwise, one of the occurrences is error-free and resynchronizes the decoder before the end of the error event, which contradicts Definition 5.7 of an error event. On the other hand, it is self-evident that the probability of sequences  $U_{1:l_b}$  with at least  $h + 1$  occurrences of  $u^s$  approaches rapidly 1 as  $l_b$  becomes large. Put differently, the probability  $p_{h,l_b}$  of sequences  $U_{1:l_b}$  with less than  $h$  occurrences of  $u^s$  approaches rapidly 0 as  $l_b$  becomes large; actually, we can show that the probability  $p_{h,l_b}$  converges to 0 exponentially, for sufficiently large  $l_b$ . At last, note that the maximum number of sequences  $\check{U}_{1:l_b}$  at Hamming distance  $h$  from  $U_{1:l_b}$  is upper bounded by  $\binom{l_b}{h} \leq l_b^h$ . Multiplied by  $p_{h,l_b}$ , we still have a quantity decreasing exponentially toward 0, which explains intuitively why the spectrum is bounded when the VLC has a synchronizing sequence of non-zero probability.

**Example 5.42.** Consider again the source/VLC of Example 5.9. Both  $\text{vlc}(a) = 0$  and  $\text{vlc}(b) = 11$  are synchronizing sequences. This VLC is therefore statistically synchronizable and its spectrum is bounded — recall the expression of its spectrum in Example 5.17. □

---

<sup>12</sup>Note that the stationary memoryless source with  $P(s) > 0$  for all  $s \in \mathcal{A}$  in Assumption 5.6 is a particular case of an  $\epsilon$ -guaranteed source [17]. In particular, it guarantees that any synchronizing sequence of the VLC has a non-zero probability.

### 5.7.2 Some examples

In this subsection, we consider two classes of VLCs as examples. The former is a subset of statistically synchronizable VLCs. The latter is a subset of non-synchronizable VLCs.

The following proposition is straightforward and provides a sufficient but restrictive condition to guarantee statistical synchronization.

**Proposition 5.43.** *Under Assumption 5.5, if a VLC has at least one codeword of length  $l_{\text{gcd}}$  and of non-zero probability, then the sequence  $u_{1:l_b}^s$  made up of this codeword repeated  $l_{\text{max}}/l_{\text{gcd}}$  times is synchronizing, the VLC is therefore statistically synchronizable and its spectrum is bounded by Theorem 5.40. ■*

With FLCs, all codewords have length  $l_{\text{gcd}}$ . We have therefore the immediate corollary by Proposition 5.43 that all FLCs are statistically synchronizable. More than that, with FLCs, we have always  $\rho_s(U, \check{U})$  for any  $s \equiv 0 \pmod{l_{\text{gcd}}}$ , which implies the following well-known result.

**Proposition 5.44.** *All FLCs are synchronizable<sup>13</sup>. ■*

Let us now consider a class of non-synchronizable VLCs.

**Proposition 5.45.** *Under Assumption 5.5, no complete reversible VLC, FLCs excepted, is statistically synchronizable. See proof in Appendix 5.D.3. □*

A VLC is *complete* if it satisfies Kraft's inequality with strict equality; when a VLC is complete, we have in particular  $\text{prefix}(\mathcal{V}^+) = \mathcal{B}^+$ , i.e., all binary sequences are admissible.

**Example 5.46.** *The reversible VLC given by the codewords 000, 0010, 0011, 01, 100, 1010, 1011, 110 and 111, is complete. This VLC is therefore non-synchronizable and its spectrum is not bounded under Assumption 5.5. Furthermore, any prefix of this VLC is anti-synchronizing. □*

Besides, there exist also incomplete reversible VLCs that are not synchronizable. Here is an example.

**Example 5.47.** *Let us repeat twice each bit of the reversible VLC of Example 5.46: 000000, 00001100, 00001111, 0011, 110000, 11001100, 11001111, 111100 and*

<sup>13</sup>Recall no bit insertion or deletion are considered here. When these are considered, as in [17], FLCs of length strictly larger than 1 ( $l_{\text{gcd}} > 1$ ) are not necessarily statistically synchronizable.

111111. This VLC is reversible, incomplete, non-synchronizable and its spectrum is not bounded under Assumption 5.5.  $\square$

Here is at last an example illustrating that, for some VLCs, statistical synchronization and bounded spectrum depend on the source/VLC mapping  $\text{vlc}(\cdot)$  and on the source (statistics).

**Example 5.48.** Let us append one codeword, the codeword 0010, to the reversible VLC of Example 5.47. Then the resulting VLC contains 10 codewords and has a synchronizing sequence (e.g., the codeword 0010 itself). This VLC is used to encode a source of independent and identically distributed symbols taking values in the alphabet  $\mathcal{A} = \{a, b, c, \dots, h, i, j\}$ . At the receiver, a sequence-ML decoder is used. If  $P(s) > 0$  for all  $s \in \mathcal{A}$ , then, independently of the source/VLC mapping, the source/VLC is statistically synchronizable and the spectrum is bounded by Corollary 5.41. If  $P(s) > 0$  for all  $s \in \mathcal{A} \setminus \{j\}$  and if  $P(j) = 0$ , then by Theorem 5.40, the source/VLC has a synchronizing sequence of non-zero probability, is thus statistically synchronizable and the spectrum is bounded if and only if the source/VLC mapping satisfies  $\text{vlc}(j) \neq 0010$ , i.e., if and only if the (synchronizing) sequence 0010 has a non-zero probability.  $\square$

Proposition 5.45 deserves some further comment. Indeed, in Chapter 3 and in [9], we showed that reversible VLCs with bounded spectrum lead to interesting performance gains in terms of SER. But it follows from Theorem 5.40 and Proposition 5.45 that not all reversible VLCs have a bounded spectrum. Unfortunately, when the spectrum is not bounded, we cannot guarantee that the VLC is non-catastrophic (see Section 5.7.4). In other words, while it is shown in Chapter 3 and in [9] that reversible VLCs with bounded spectrum can lead to interesting SER-performance gains, we observe now that there exist reversible VLCs that have an unbounded spectrum and thus that might be SER-catastrophic.

This contrast, performance-wise, between bounded and unbounded spectra motivates clearly the search for necessary and sufficient conditions for bounded spectrum. So far, no such conditions have been found in the literature and the next subsection provides, instead, an efficient algorithm to test whether the VLC spectrum is bounded, without actually computing the spectrum.

### 5.7.3 Bounded spectrum test algorithm

The equivalences in Theorem 5.40 have the nice consequence that the algorithm of [17, Section V], which tests whether a synchronizing sequence exists, tests also whether the VLC spectrum is bounded (if this sequence has a non-zero probability), without computing the spectrum. This algorithm is now recalled in a slightly modified form that takes into account that we do not consider bit insertion/deletion, and that tests whether a synchronizing sequence of *non-zero probability* exists — in order to test whether the VLC spectrum is bounded under Assumption 5.5. A C++ implementation is provided in [109]. The underlying principle is to test whether there exists  $p \in \mathcal{X}_0$  that is anti-synchronizing w.r.t. sequences of non-zero probability. If there is no such  $p$ , we know from Lemma 5.39 that the VLC has a synchronizing sequence of non-zero probability.

For any  $x \in \mathcal{X}_0$ , let  $\mathcal{D}(x)$  be the union of the sets

$$\begin{aligned} & \{d \in \mathcal{X}_0 : \exists w \in \mathcal{V}, P(w) > 0, \exists v \in \mathcal{V}^+ \cup \{R\}, xw = vd\} \\ & \cup \{\chi : \exists w \in \mathcal{V}, P(w) > 0, xw \notin \text{prefix}(\mathcal{V}^+)\}. \end{aligned}$$

The first set contains the set of states  $d \in \mathcal{X}_0$  that are attainable from the state  $x$  by encoding a codeword  $w$  of non-zero probability. The second set contains the symbol  $\chi$  if there exists some codeword  $w$  of non-zero probability that leads from  $x$  to a non-existing state (represented by  $\chi$ ).

**Algorithm 5.49** (existence of a synch. sequence of non-zero prob.).

Step 1: Let  $F = \{R, \chi\}$ .

Step 2: Let  $F' = \{x \in \mathcal{X}_0 \setminus F : \mathcal{D}(x) \cap F \neq \emptyset\}$ .

Step 3: Appends  $F'$  to  $F$ , i.e., let  $F = F \cup F'$ .

Step 4: If  $F' = \emptyset$ , go to Step 5. Otherwise, go to Step 2.

Step 5: A synchronizing sequence of non-zero probability exists if and only if  $F = \mathcal{X}_0 \cup \{\chi\}$ . □

Step 2 selects all states from which we can attain a state in  $F$  (including a non-existing state) with a codeword of non-zero probability. Based on this,

it is straightforward to show by induction that, at the end of the algorithm,  $\mathcal{X}_0 \setminus F$  contains the set of anti-synchronizing prefixes w.r.t. sequences of non-zero probability. This explains the decision taken at Step 5 by Lemma 5.39.

#### 5.7.4 Bounded spectrum and non-catastrophic VLC

Due to the sensitivity to desynchronization, most VLCs have a catastrophic behavior in the  $\text{SER}_L$ , i.e., they can lead to an arbitrarily large number of Levenshtein symbol errors given a finite number of bit errors. Hopefully, the probability of pairs of sequences generating an infinite number of Levenshtein symbol errors tends most often toward zero exponentially. In this case, the average number of Levenshtein symbol errors is finite and we say the VLC is non-catastrophic in average, as stated hereafter.

**Definition 5.50.** *A VLC is  $\text{SER}_L$ -catastrophic if a finite number of bit errors can generate an infinite number of Levenshtein symbols errors. A VLC is  $\text{SER}_L$ -catastrophic in average if the average number of Levenshtein symbol errors generated by a finite number of bit errors is infinite.*  $\square$

Note that the catastrophic (or non-catastrophic) behavior of a VLC depends generally on the source/VLC mapping and on the source. This is again related to Remark 5.11 and Example 5.48.

**Theorem 5.51.** *No VLC with bounded spectrum is  $\text{SER}_L$ -catastrophic in average. See proof in Appendix 5.D.4.*  $\square$

A stronger result is the following theorem.

**Theorem 5.52.** *Under Assumption 5.5, if the VLC spectrum is strongly bounded, there exists some  $\mu$  such that, for all  $h$ ,*

$$\sum_{s_L \geq 1} s_L A_{s_L, h}^{\text{vlc}} \leq \mu h A_h^{\text{vlc}}. \quad (5.67)$$

*See proof in Appendix 5.D.5.*  $\square$

It follows from Theorem 5.40 and Theorem 5.52 that all bounded spectra satisfy (5.67) under Assumption 5.5. Furthermore, notice that when (5.67) is satisfied, the  $\text{SER}_L$  and the BER of a VLC stream can be upper bounded by the same bound up to a multiplicative constant: By (5.29) and (5.67), we obtain

$\text{SER}_L \leq \mu \sum_{h \geq 1} h A_h^{\text{vlc}} P_h$ , which is proportional to the upper bound (5.32) on the BER.

### 5.7.5 Toward a link between bounded spectra and interleaving gains

Recall the expression of the upper bound (5.58) on the FER,

$$\text{FER} \leq \sum_h P_h \sum_{s_L, w} A_{s_L, w, h}^{\text{gc}}$$

from Theorem 5.32. The spectrum of the global code,  $A_{s_L, w, h}^{\text{gc}}$ , depends on the interleaver size  $N$ . For large values of  $N$ , this dependence on  $N$  is characterized by

$$\alpha_M \triangleq \max_{s_L, w, h} \left\{ \limsup_{N \rightarrow +\infty} \log_N (A_{s_L, w, h}^{\text{gc}}) \right\}. \quad (5.68)$$

The following conjecture from [91] states that  $\alpha_M$  characterizes the so-called interleaving gain.

**Conjecture 5.53** (interleaving gain exponent (IGE), [91]). *There exists some threshold independent of  $N$  such that, for any fixed SNR above that threshold, the FER is  $\mathcal{O}(N^{\alpha_M})$  and the BER is  $\mathcal{O}(N^{\alpha_M-1})$  for asymptotically large  $N$ .*  $\square$

Later on, we will see that the  $\text{SER}_L$  is also  $\mathcal{O}(N^{\alpha_M-1})$  for asymptotically large  $N$ , under certain assumptions. When this conjecture holds and when  $\alpha_M < 0$ , the FER decreases to zero like  $N^{\alpha_M}$  as the interleaver size  $N$  approaches  $\infty$ , for SNRs above the threshold. This illustrates how significantly the value of  $\alpha_M$  matters.

Interestingly, the value of  $\alpha_M$  can be calculated analytically. For example, it has been derived in [16] for serial concatenations of CCs. More generally, the authors in [16] provide, as a byproduct, a systematic method to derive  $\alpha_M$  for all types of concatenation. Assuming this method holds with VLCs, it has been used notably in Chapter 3 and in [9] to derive the interleaving gains on the  $\text{SER}_L$  and on the FER for several turbo systems based on VLCs. For example, the value of  $\alpha_M$  for the double serial concatenation between a VLC of free distance  $d_f^{\text{vlc}}$ , a regular rate- $\frac{1}{d_f^{\text{rc}}}$  repetition code (RC) and an inner recursive non-catastrophic CC encoder is given by

$$\alpha_M = 1 - \left\lfloor (d_f^{\text{rc}} + 1) / 2 \right\rfloor \mathbb{I}\{d_f^{\text{rc}} \geq 2\}, \quad (5.69)$$

as a direct application of [16, eq. (30)], where  $d_f^o = d_f^{\text{vlc}} d_f^{\text{rc}}$  is the free distance of the outer code formed by the VLC-RC.

Let us now show that the systematic method of [16] for the derivation of  $\alpha_M$  holds indeed with VLCs. Based on the results from the previous sections, this is straightforward. Recall the upper spectra  $A_{s_L, h}^{F_s, \text{app}}$ ,  $A_{s_L, h}^{F_b, \text{upp}}$ , given in (5.42), (5.55), for the framing rules  $F_s$ ,  $F_b$ , respectively. These upper spectra can be further upper bounded by

$$\begin{aligned} A_{s_L, h}^{F_s, \text{app}} &\leq \sum_n \binom{N_s}{n} \sum_{\substack{l_s, l_b, \\ l_s \leq N_s}} T_{s_L, h, l_s, l_b, n}^{\text{vlc}} \\ &\leq \sum_n \binom{N}{n} \sum_{l_s, l_b \leq N} T_{s_L, h, l_s, l_b, n'}^{\text{vlc}} \end{aligned} \quad (5.70)$$

$$\begin{aligned} A_{s_L, h}^{F_b, \text{upp}} &\leq \frac{l_{\max}^*}{l_{\text{gcd}}^*} \sum_n \binom{N_b^{\max}}{n} \sum_{\substack{l_s, l_b \\ l_b \leq N_b^{\max}}} T_{s_L, h, l_s, l_b, n}^{\text{vlc}} \\ &\leq \frac{l_{\max}^*}{l_{\text{gcd}}^*} \sum_n \binom{N}{n} \sum_{l_s, l_b \leq N} T_{s_L, h, l_s, l_b, n'}^{\text{vlc}} \end{aligned} \quad (5.71)$$

which correspond almost exactly to [16, eq. (15)]. The rest of the derivation proposed in [16] then holds straightforwardly.

Unfortunately, the value of  $\alpha_M$  is somehow pointless if the IGE Conjecture 5.53 does not hold. And though intuitive, this conjecture is not straightforward to prove. So far and to the best of our knowledge, it has been proved only in [92] for multiple parallel and serial concatenations of CCs, i.e., for multiple parallel and serial turbo codes. In the next subsection, we will prove it for serial concatenations with a VLC as outer code. But before, it is worth commenting intuitively an important assumption we are going to make on the VLC.

A closer look at the derivation of  $\alpha_M$  proposed in [16] reveals notably that this derivation holds with VLCs only if (necessary condition) the dependence in  $N$  in (5.70)–(5.71) is due only to the binomial coefficient  $\binom{N}{n}$ . In other words, the summation  $\sum_{l_s, l_b \leq N} T_{s_L, h, l_s, l_b, n}^{\text{vlc}}$  must necessarily be independent of  $N$  as a first approximation, for fixed  $n$  and large  $N$ , i.e., we must have  $\sum_{l_s, l_b \leq N} T_{s_L, h, l_s, l_b, n}^{\text{vlc}} \approx \sum_{l_s, l_b} T_{s_L, h, l_s, l_b, n}^{\text{vlc}}$ . This necessary condition is satisfied in particular when the VLC spectrum is bounded, as a corollary of Lemma 5.64

given in Appendix 5.C.3.2. Assuming a bounded VLC spectrum appears then to be relevant; the next subsections will show it is indeed a sufficient condition to ensure the interleaving gains.

### 5.7.6 Bounded spectrum and guaranteed interleaving gains

The main result of this subsection is summarized in the next Theorem. It states essentially that the interleaving gain (5.69) of a serial concatenation holds under Assumption 5.5 if the VLC spectrum is bounded (Definition 5.13). Since the “boundness” of the VLC spectrum can be tested efficiently with Algorithm 5.49, we are given a practical tool to test the VLC in order to guarantee the interleaving gains. However, it is worth noting that the Theorem states only the (theoretical) existence and the values of the interleaving gains; in particular, it does not provide any piece of information about the minimum interleaver length and the minimum channel SNR required to observe these gains in practice.

**Theorem 5.54.** *Consider the frame-based system in Fig. 5.2 with a recursive non-catastrophic CC encoder as ECC and with either the framing rule  $F_s$  or  $F_b$ . With  $F_s$ , consider besides  $N_s \geq kN$  for some constant  $k$  — recall we have already the constraint  $N \geq N_s l_{\max}^*$  by definition of  $F_s$ . Then, under Assumption 5.5, if  $d_f^{\text{vlc}} \geq 2$  and if the VLC satisfies one of the equivalences in Theorem 5.40, there exists some threshold independent of  $N$  such that, for any fixed SNR<sup>14</sup> above that threshold,*

$$\text{BER} = \mathcal{O}\left(N^{-\lfloor \frac{d_f^{\text{vlc}}+1}{2} \rfloor + \epsilon}\right), \quad (5.72)$$

$$\text{SER}_L = \mathcal{O}\left(N^{-\lfloor \frac{d_f^{\text{vlc}}+1}{2} \rfloor + \epsilon}\right), \quad (5.73)$$

$$\text{FER} = \mathcal{O}\left(N^{-\lfloor \frac{d_f^{\text{vlc}}-1}{2} \rfloor + \epsilon}\right), \quad (5.74)$$

for arbitrary  $\epsilon > 0$ . See proof in Appendix 5.D.6. □

Though Theorem 5.54 deals only with the serial concatenation, the developments used in the proof extend easily to other concatenations as well. For

<sup>14</sup>If we use instead the Bhattacharyya noise parameter  $\gamma$  of the channel and the noise exponent defined as  $\alpha = -\log(\gamma)$ , see details in [92], then the threshold is independent of the type of channel. More precisely, there exists a positive threshold  $c_0$  such that the theorem holds for any binary-input memoryless channel whose noise exponent  $\alpha$  satisfies  $\alpha > c_0$ .

example, a double serial concatenation of a VLC with two CCs was considered in Theorem 3.7 and a hybrid concatenation of a VLC serially concatenated with a multiple parallel turbo code was considered in Theorem 3.3. The proof of Theorem 5.54 extends easily to these systems. It extends also easily to the particular interleaving gains obtained with reversible VLCs in Theorem 3.5 if  $d_f^{\text{rc}} \geq 3$  and in Theorem 3.7 if  $d_f^{\text{cc,m}} \geq 3$  — but the extension for  $d_f^{\text{rc}} = 2$  and  $d_f^{\text{cc,m}} = 2$  is less straightforward and requires further investigation.

Regarding the SER, without any additional assumption (on the VLC), we can only state that the SER is upper bounded by the FER, hence the following corollary.

**Corollary 5.55.** *Under the assumptions of Theorem 5.54,*

$$\text{SER} = \mathcal{O}\left(N^{-\lfloor \frac{d_f^{\text{vlc}} - 1}{2} \rfloor + \epsilon}\right) \quad (5.75)$$

for arbitrary  $\epsilon > 0$ . ■

Note that particular interleaving gains on the SER have been proved in Chapter 3 for reversible VLCs of free distance  $d_f^{\text{vlc}} = 1$ .

So Theorem 5.54 states the VLC spectrum must be bounded — or one of the equivalences from Theorem 5.40 — in order to guarantee the interleaving gains for a serial concatenation. And this conclusion extends easily to other concatenations. Then, what happens when the VLC spectrum is not bounded? Before answering this question, it is worth noting that VLCs with unbounded spectrum are rare. For example, it is proved in [110] that “almost all” complete VLCs have a synchronizing sequence, which under Assumption 5.6 is equivalent to having a bounded spectrum by Corollary 5.41. So this should not be a real issue in practice and we can safely consider that Theorem 5.54 guarantees the interleaving gains for almost all practical VLCs. Nevertheless, when the VLC spectrum is not bounded, note that we have always at least the interleaving gains offered by the ECC. So if the ECC in Fig. 5.2 provides an interleaving gain on the FER, then we have at least this interleaving gain on the  $\text{SER}_{\perp}$  and on the FER of the global code. Note already that simulation-based results in Section 5.9 will reveal some additional and unexpected properties of VLCs with unbounded spectrum when the frame-MAP decoder is used.

## 5.8 Discussion and further work

This section is a collection of remarks without proof, of thoughts that arose during the developments, and of possible extensions for further work.

### 5.8.1 Other framing rules

In Section 5.6, we envisaged two framing rules, namely  $F_s$  and  $F_b$ . Note that a third one was also considered shortly in Section 5.6.8. The results obtained so far are extensible to other framing rules, sometimes straightforwardly.

#### 5.8.1.1 Framing rule $F_b$ with side-information

As a first example, let us focus on the framing rule  $F_b$  and particularly on the bits  $U_{N_b+1:N}$ . In the definition of  $F_b$ , we have set these bits to zero. What happens if, instead, we use them to send side-information? More generally, let us consider that we replace (5.45) by

$$\begin{aligned} N_b &= I(S_{1:N_s}) \leq N - k, \\ N_b + I(S_{N_s+1}) &> N - k \end{aligned} \tag{5.76}$$

for some constant  $k$  independent of the interleaver length  $N$ , and that the bits  $U_{N_b+1:N}$  are identically and uniformly distributed. Then, compared to the original definition of  $F_b$ , the level of performance on the VLC bits  $U_{1:N_b}$  is of course degraded. However, the interleaving gains on  $U_{1:N_b}$  — we do not care about errors in  $U_{N_b+1:N}$  — remain the same. This can be shown by straightforward extension of previous results.

#### 5.8.1.2 Framing irrespectively of codeword boundaries

As a second example, let us frame the VLC stream into VLC blocks of exactly  $N$  bits each, irrespectively of codeword boundaries. VLC blocks may then start/finish in the middle of a codeword. At the decoder, this is an additional source of uncertainty, which makes this framing rule less robust than  $F_b$ . However, if the VLC spectrum is bounded, the VLC decoder rapidly resynchronizes and bit errors tend to have only a local impact in the VLC block. Thus, if the VLC spectrum is bounded, the decrease in robustness is limited

to the beginning and to the end of the VLC block — at the end of the VLC block, the impact is somehow similar to a CC with a non-terminated trellis. Therefore, this framing rule offers BER and  $\text{SER}_L$  performance close to  $F_b$ , though always less attractive, for asymptotically large  $N$ , and the same interleaving gains. This conclusion extends also to the FER if we define it as  $\text{FER} = \mathbb{I}\{U_{1:N} \neq \check{U}_{1:N}\}$ .

But such a definition of the FER is simplistic; for example, it does not take the correct segmentation of the codeword boundaries into account. Similarly, defining the SER poses a few problems. A better definition of the FER and of the SER would require to focus on a given application. For example, we could frame the VLC stream into groups of  $m$  VLC blocks, the current group being made up of the next  $N_s$  symbols in the stream subject to (this is similar to a frame with  $F_b$ )

$$\begin{aligned} N' = I(S_{1:N_s}) &\leq mN, \\ N' + I(S_{N_s+1}) &> mN, \end{aligned} \quad (5.77)$$

completed with  $mN - N'$  zeros to get a total of  $mN$  bits, where the value of  $N'$  is assumed to be known at the decoder. The current group is then itself framed into  $m$  blocks of length  $N$ , irrespectively of codeword boundaries. With such a framing rule, only the first VLC block of each group is guaranteed to start with a codeword and only the last VLC block is guaranteed to finish with a codeword at bit position  $N' - (m - 1)N$ . As discussed in the previous paragraph, the interleaving gains on the BER,  $\text{SER}_L$  and FER are identical to  $F_b$  if the VLC spectrum is bounded. We can then easily define a “group error rate”, which makes more sense than the FER. It is trivially upper bounded by  $m$  times the FER and has therefore the same interleaving gain as the FER if  $m$  is a constant independent of  $N$ . At last, we can also define the SER inside each group by using the technique discussed later on in Section 5.8.2.

### 5.8.1.3 Framing and multiplexing VLCs

As a third and last example, let us frame and multiplex two VLC streams into a multiplexed VLC block of length  $N$ . Consider two VLC streams, say  $V^1$  and  $V^2$ , with bounded spectra and respective free distances  $d_f^1$  and  $d_f^2$ . Each stream  $V^i = \text{vlc}(S_1^i S_2^i S_3^i \dots)$  is framed into a sub-block of length  $N^i$  according

to the framing rule  $F_b$ , except that the constraint (5.45) is replaced by

$$\begin{aligned} N_b^i &= l(S_{1:N_b^i}^i) \leq N^i, \\ N_b^i + l(S_{N_b^i+1}^i) &> N^i, \end{aligned} \quad (5.78)$$

where the values of  $N^i$  grow linearly with  $N$ , are subject to  $N^1 + N^2 = N$  and are known at the decoder. Given  $U_{1:N_b^i}^i = \text{vlc}(S_{1:N_b^i}^i)$ , we then append  $N^i - N_b^i$  zeros to form the sub-block  $U_{1:N^i}^i$  and concatenate the sub-blocks as  $U_{1:N} = U_{1:N^1}^1 U_{1:N^2}^2$  to form the VLC block.

Multiplexing the streams may of course alter the robustness of each stream; the interesting question that arises is how much. We can *partially* answer this question in terms of interleaving gains, by extending previous results. Only final results are summarized hereafter without proof, for the concatenation with a multiple parallel turbo code. Let us focus on the impact of  $V_2$  on  $V_1$ , i.e., how the interleaving gain on  $V_1$  is altered when  $V_1$  is multiplexed with  $V_2$ . Straightforwardly, if  $d_f^2 \geq d_f^1$ , the interleaving gain is unchanged. So let us focus on  $d_f^2 < d_f^1$ .

Consider a multiplexed VLC block serially concatenated with  $J \geq 1$  parallel concatenated recursive non-catastrophic CC encoders. Note that  $J = 1$  is the serial concatenation considered in Theorem 5.54,  $J = 2$  is a parallel turbo code and  $J \geq 3$  is a multiple parallel turbo code. By extension of the proposed results, we can show that the interleaving gains on  $V_1$  are unaltered when  $d_f^2 \geq 3$ , or when  $d_f^2 \geq 2$  and  $J \geq 2$ , or when  $d_f^1$  is even,  $d_f^2 = 1$  and  $J \geq 3$ . They are degraded by one unit when  $d_f^1$  is odd,  $d_f^2 = 1$  and  $J \geq 3$ . And it is self-evident that they are completely degraded when  $d_f^2 = 1$  and  $J = 1$ . So it only remains to discuss the cases ( $d_f^2 = 1, J = 2$ ) and ( $d_f^2 = 2, J = 1$ ). Unfortunately, this is not as straightforward. For example, when  $d_f^2 = 1$  and  $J = 2$ , we can show that the interleaving gain is degraded by at least one unit when  $d_f^1$  is odd. But more precision requires additional developments beyond the scope of this discussion.

Further work about this issue is worth considering since most applications in practice involve multiplexing different sources of data. We have provided only a partial answer to the question, focused on interleaving gains. Besides, some cases considered above remain unanswered at this stage. These cases

concern VLCs with small free distances, which are unfortunately the most common in practice since the primary purpose of VLCs is compression.

### 5.8.2 SER for $N_s$ unknown

As mentioned in Section 5.5.2.2, when the number of symbols  $N_s$  is unknown at the receiver, the decoded sequence  $\check{S}_{1:\check{N}_s}$  may have a different number of symbols than the transmitted sequence  $S_{1:N_s}$ ,  $N_s \neq \check{N}_s$ . Because of that, there is some ambiguity in the meaning of  $S_i \neq \check{S}_i$  and of  $d_S(S, \check{S})$  in order to measure the SER.

There is one notable exception: FLCs. With FLCs, the upper bounds on the  $\text{SER}_L$  in Theorem 5.18 and in Theorem 5.32 are straightforward upper bounds on the SER.

Let us focus therefore on VLCs that are not FLCs, and more precisely on VLC blocks for a certain framing rule  $F$ . Note already that the SER is always upper bounded by the FER, for which we have an upper bound in Theorem 5.32.

To resolve the ambiguity in  $d_S(S, \check{S})$ , let us consider a new alphabet  $\mathcal{A}' = \mathcal{A} \cup \{\varphi\}$  and use the ' prime notation to indicate variables based on this new alphabet, such that  $S'_i = \varphi$  means that no symbol is produced by the source at symbol time  $i$ . Then, there exists straightforwardly a constant  $N'_s$  such that each realization  $s$  of the VLC block has a unique representation  $s'_{1:N'_s}$  in  $\mathcal{A}'^{N'_s}$ , with  $s'_{1:l_S(s)} = s$  and  $s'_i = \varphi$  for  $i > l_S(s)$ . Given these representations, there is an error at symbol time  $i$  if  $\check{s}'_i \neq s'_i$ , which is properly defined, and  $d_S(s', \check{s}') = \sum_{i=1}^{N'_s} \mathbb{I}\{s'_i \neq \check{s}'_i\}$ . Note  $d_S(s', \check{s}') = d_S(s_{1:l_S^{\min}}, \check{s}'_{1:l_S^{\min}}) + |l_S(s) - l_S(\check{s}')|$ , where  $l_S^{\min} = \min\{l_S(s), l_S(\check{s}')\}$ . To calculate the SER, we divide the symbol distance by the number of symbols,

$$\begin{aligned} \text{SER} &\triangleq \mathbf{E} \left\{ \frac{d_S(S', \check{S}')}{l_S(S)} \right\} \\ &= \mathbf{E} \left\{ \frac{d_S(S_{1:L_S^{\min}}, \check{S}'_{1:L_S^{\min}}) + |l_S(S) - l_S(\check{S}')|}{l_S(S)} \right\}, \end{aligned} \quad (5.79)$$

where the expectation is taken over all realizations  $s$  of the VLC block and over all channel realizations (assuming  $\check{S}$  depends deterministically on  $S$  and

on the channel realization). Note that the fraction in (5.79) can be larger than 1 for some realizations, as with the  $\text{SER}_L$ .

Extending the upper bounds on the  $\text{SER}_L$  to the SER is not straightforward. This is because the sensitivity to desynchronization has an additional dimension with the SER. In Section 5.5.1, we characterized the synchronization with  $\rho_{k,k'}(S, \check{S})$  and the decoder was considered synchronized in  $S_k$  if  $\rho_{k,k'}(S, \check{S})$  for some  $k'$ . For the SER,  $k'$  must be equal to  $k$ , otherwise the decoder still produces symbol errors by definition of  $d_S(S', \check{S}')$ , even if  $S'_{k+i} = \check{S}'_{k+i}$  for all  $i \geq 0$ . So, for the SER, the decoder is considered synchronized in  $S_k$  if  $\rho_{k,k}(S', \check{S}')$ . Integrating this into the definition (5.20) of the error event is possible but it leads to arbitrarily long error events in practice and makes the computation of the corresponding spectrum intractable.

Instead, we can *loosely* upper bound the SER of a non-concatenated VLC block, under Assumption 5.5, with a simple extension of the spectrum given in Definition 5.10. Let

$$A_{s_L, s, d, h, l_s, l_b}^{\text{vlc}} \triangleq \sum_{s_{1:l_s} \in \mathcal{A}^{l_s}, l(s_{1:l_s})=l_b} P(s_{1:l_s}) A_{s_L, s, d, h | s_{1:l_s}}^{\text{vlc}}, \quad (5.80)$$

where  $A_{s_L, s, d, h | s}^{\text{vlc}}$  is the conditional spectrum

$$A_{s_L, s, d, h | s}^{\text{vlc}} \triangleq \left| \left\{ \check{s} : \begin{array}{l} s \text{ and } \check{s} \text{ form an error event,} \\ d_H(s, \check{s})=h, d_{S_L}(s, \check{s})=s_L, \\ d_S(s, \check{s})=s, l_S(\check{s})-l_S(s)=d \end{array} \right\} \right|.$$

Notice that the parameter  $d$  is the difference in number of symbols between the decoded and transmitted sequences. Therefore, we can get a loose upper bound by considering that the decoder produces symbol errors until the end of the frame when it encounters an error event with  $d \neq 0$ , i.e., it produces a maximum of  $N_s^{\max}$  symbol errors, hence

$$\text{SER} \leq \sum_{h, s_L, s \geq 1} P_h \left( s A_{s_L, s, d=0, h}^{\text{vlc}} + N_s^{\max} \sum_{d \neq 0} A_{s_L, s, d, h}^{\text{vlc}} \right) \quad (5.81)$$

by extension of Theorem 5.18, where  $N_s^{\max}$  is the maximum admissible value of  $N_s$ , for the given framing rule  $F$ .

We can get a tighter performance estimation by replacing the maximum value  $N_s^{\max}$  with the average value  $\bar{N}_s$ , if we are not interested in an ‘‘upper’’ bound. We can then tighten this estimation further by taking the position of

the error event into account. For example, an error event with  $d \neq 0$  occurring at the beginning of the frame produces nearly  $\bar{N}_s$  symbol errors in average while one occurring at the end produces 1 symbol error. Averaging over all positions, we can replace  $N_s^{\max}$  by  $\bar{N}_s/2$  in (5.81).

These results can be extended to upper bound the SER of VLC blocks concatenated with linear codes, by extensions of Theorem 5.26, Theorem 5.31 and Theorem 5.32 with the spectrum  $A_{s_L, s, d, h, l_s, l_b}^{\text{vlc}}$ .

At last, note that we have considered here above the SER inside each VLC block, averaged over all VLC blocks. For some applications, it is interesting to consider the SER inside a group of  $k$  consecutive VLC blocks for some constant  $k$ , in order to take into account the possible propagation of desynchronization across consecutive VLC blocks. For example, when the decoder finishes a VLC block with a wrong number of symbols, then it starts the next block desynchronized and produces symbol errors until some bit errors haz- ardously resynchronize the symbol clock of the decoder. Of course, if  $k$  approaches  $+\infty$ , the SER will tend to a value close to 1 on noisy channels. This illustrates the well-known result that it becomes primordial, when  $k$  is large, to have some technique to synchronize once in a while the number of symbols, i.e., the symbol counter or the symbol clock, between the transmitter and the decoder. This shows also that the SER does not make sense with an infinite VLC stream, without a proper symbol clock synchronization technique.

### 5.8.3 Bounds for $N_s$ known at the decoder

Knowing the number of symbols  $N_s$  at the decoder generally increases the resilience at the expense of a higher decoding complexity. Extending the results to such a situation is not straightforward. Indeed, the ML-decoder is to be considered synchronized in symbol  $S_k$  if and only if  $\rho_{k,k}(S_:, \check{S}_:)$ , just as with the SER in Section 5.8.2. Integrating this new concept of synchronization into the definition (5.20) of the error event is possible but it leads, again, to arbitrarily long error events in practice and makes the computation of the corresponding spectrum intractable.

Instead, let us only extend slightly the VLC spectrum. Since the decoder knows the value of  $N_s$ , the transmitted and decoded VLC blocks,  $u_:$  and  $\check{u}_:$ ,

have the same number of symbols,  $l_S(u_\cdot) = l_S(\check{u}_\cdot)$ . Thus, compared to Section 5.8.2, no error event with  $d \neq 0$  can occur alone. More precisely, given an error event  $e = (u'_\cdot, \check{u}'_\cdot)$ , let  $d_L(e) = l_S(\check{u}'_\cdot) - l_S(u'_\cdot)$  — this corresponds to the parameter  $d$  in (5.80). Then, the error events in the set  $\mathcal{E}(u_\cdot, \check{u}_\cdot)$  (see Section 5.5.1) always satisfy  $\sum_{e \in \mathcal{E}(u_\cdot, \check{u}_\cdot)} d_L(e) = 0$  for any VLC block realizations  $u_\cdot$  and  $\check{u}_\cdot$ , since  $l_S(u_\cdot) = l_S(\check{u}_\cdot)$ .

This property can be used to calculate the VLC block spectrum from the VLC stream spectrum  $A_{s_L, d, h, l_s, l_b}^{\text{vlc}}$  given in (5.80) — note that we have discarded the unnecessary subscript  $s$ . Indeed, the transformation (5.43) can be straightforwardly extended to calculate  $T_{s_L, d, h, l_s, l_b, n}^{\text{vlc}}$  (obvious extension of  $T_{s_L, h, l_s, l_b, n}^{\text{vlc}}$ ) from  $A_{s_L, d, h, l_s, l_b}^{\text{vlc}}$ . Next, we obtain the VLC block spectrum by extensions of (5.42), (5.54), if we restrict the summation to  $d = 0$  (since  $\sum_{e \in \mathcal{E}(u_\cdot, \check{u}_\cdot)} d_L(e) = 0$ ). At last, the upper bounds on the BER, the SER<sub>L</sub> and the FER follow from (5.57) and from Theorem 5.32.

Upper bounding the SER is possible as well, but requires modifications that are less straightforward. To make things simpler, let us find a loose upper bound. Let  $A_{s_L, s, d, e, h, l_s, l_b}^{\text{vlc}}$  be equal to  $A_{s_L, s, d, h, l_s, l_b}^{\text{vlc}}$  when  $e = |d|$  and to 0 otherwise —  $A_{s_L, s, d, h, l_s, l_b}^{\text{vlc}}$  is given in (5.80). Again, we can straightforwardly extend the transformation (5.43) in order to calculate  $T_{s_L, s, d, e, h, l_s, l_b, n}^{\text{vlc}}$  from  $A_{s_L, s, d, e, h, l_s, l_b}^{\text{vlc}}$ . We can then calculate the VLC block spectrum  $A_{s_L, s, e, h}^F$  and the global code spectrum  $A_{s_L, s, e, w, h}^{\text{gc}}$  by extensions of (5.42), (5.54), (5.57), if we restrict the summation to  $d = 0$ . When  $e > 0$ , notice that all codeword pairs enumerated by  $A_{s_L, s, e, w, h}^{\text{gc}}$  form at least one error event with  $d \neq 0$ . Since  $d \neq 0$ , this error event desynchronizes the decoder and, at the worst (loose upper bound), all symbols in the VLC block are erroneously decoded. Therefore, as an extension of (5.59), we can upper bound the SER finally as

$$\text{SER} \leq \sum_{h, s_L, s, w} P_h \left( \frac{s}{N_s^{\min}} A_{s_L, s, e=0, w, h}^{\text{gc}} + \sum_{e > 0} A_{s_L, s, e, w, h}^{\text{gc}} \right).$$

At last, note that there exist VLCs that lead to a larger free distance of the VLC block when  $N_s$  is known at the decoder than when  $N_s$  is unknown. Such examples of VLC are rare, however; in particular, this can be proved never to happen with VLCs that have two codewords of the same length at a Hamming distance equal to the free distance  $d_f^{\text{vlc}}$  defined in Definition 5.12.

Besides, when this increase in free distance happens, the benefit performance-wise should diminish when  $N_s$  becomes large, though further investigations would be necessary to confirm this.

**Example 5.56.** Consider the source alphabet  $\mathcal{A} = \{a, b\}$  and the VLC  $\{0, 10\}$ , i.e.,  $\text{vlc}(a) = 0$  and  $\text{vlc}(b) = 10$ . When  $N_s$  is unknown, the free distance is  $d_f^{\text{vlc}} = 1$ , e.g.,  $d_H(aa, b) = d_H(00, 10) = 1$ . When  $N_s$  is known, the free distance is 2. Indeed, recall the constraint  $\sum_{e \in \mathcal{E}(u, \check{u})} d_L(e) = 0$ , which requires either (i) one error event  $e$  with  $d_L(e) = 0$ , or (ii) at least two error events. The case (ii) implies trivially  $\sum_{e \in \mathcal{E}(u, \check{u})} d_H(e) \geq 2$ . The case (i) implies  $d_H(e) \geq 2$  since the only error events  $e$  with  $d_H(e) = 1$  are  $e = (aa, b)$  and  $e = (b, aa)$  and they do not satisfy  $d_L(e) = 0$ . One example for case (i) is  $e = (ab, ba) = (010, 100)$  with  $d_H(e) = 2$ .  $\square$

#### 5.8.4 Extension to the MAP decoder

We have focused so far on the ML decoder, for reasons discussed in Section 5.3.3. Extensions to the MAP decoder are now briefly addressed.

##### 5.8.4.1 ML versus MAP discussion

It is self-evident and well known that the MAP detection (5.14) and the ML detection (5.16) are equivalent when the frames  $s_i$  are equally probable from the decoder standpoint. Under Assumption 5.5, this occurs when the probability distribution of the source symbols is given by

$$P_{\text{ml}}(s) = 2^{-Kl(s)}, \quad \text{with } K \text{ subject to } \sum_{s \in \mathcal{A}} 2^{-Kl(s)} = 1. \quad (5.82)$$

##### 5.8.4.2 Extension of the pairwise error probability for the AWGN

To upper bound the performance of the ML decoder, we have used notably the ML pairwise error probability  $P_h$ . This is the conditional probability that a given sequence  $\check{s}_i$  has a larger likelihood metric than the transmitted sequence  $s_i$ . Given a binary, symmetric, memoryless, time invariant channel, it depends only on the Hamming distance  $h$  between the coded sequences associated with  $s_i$  and  $\check{s}_i$ .

To extend the performance bounds to the MAP decoder, we especially need to extend the pairwise error probability. With the sequence-MAP decoder, it is the conditional probability that a given sequence  $\check{s}_i$  has a larger a

posteriori metric than the transmitted sequence  $s_i$ . Therefore, it depends not only on the Hamming distance but also on the a priori probabilities of  $s_i$  and  $\check{s}_i$ , and more precisely on the *a priori distance* between  $s_i$  and  $\check{s}_i$ . Let us illustrate this with the AWGN channel.

**Definition 5.57.** The a priori distance<sup>15</sup> between two sequences of symbols  $s_i, \check{s}_i \in \mathcal{A}^+$  is given by

$$d_A(s_i, \check{s}_i) \triangleq \log(P(s_i)) - \log(P(\check{s}_i)) \quad (5.83)$$

where  $P(s_i)$  and  $P(\check{s}_i)$  are the respective probabilities of  $s_i$  and  $\check{s}_i$ .  $\square$

The received discrete-time signal from the AWGN channel can be normalized and written as

$$Y = C + N \quad (5.84)$$

where  $C \in \{+1, -1\}$  is a BPSK-modulated coded bit and  $N$  is a Gaussian random variable  $\mathcal{N}(0, \sigma^2)$  with  $\sigma^2 = N_0/(2R_c E_b)$ , where  $R_c$  is the global code rate,  $N_0/2$  is the double-sided noise spectral density and  $E_b$  the energy per bit of entropy. Let  $s_i$  be the transmitted symbol sequence and  $\check{s}_i$  be the decoded symbol sequence. Let  $c_i$  and  $\check{c}_i$  be the corresponding coded sequences. Let

$$Z_i = c_i + N_i \quad (5.85)$$

be the received signal given that  $c_i$  was sent, by (5.84). Let  $h = d_H(c_i, \check{c}_i)$  and  $a = d_A(s_i, \check{s}_i) = d_A(c_i, \check{c}_i)$ . Let then  $P_{h,a}$  be the MAP pairwise error probability, which by definition is the probability to satisfy the inequality

$$P(C_i = c_i | Y_i = Z_i) \leq P(C_i = \check{c}_i | Y_i = Z_i), \quad (5.86)$$

which is equivalent to

$$\frac{p(Y_i = Z_i | C_i = c_i)}{p(Y_i = Z_i | C_i = \check{c}_i)} \leq \frac{P(C_i = \check{c}_i)}{P(C_i = c_i)}. \quad (5.87)$$

Note  $Z_i$  is random while  $c_i$  and  $\check{c}_i$  are realizations. Let  $\mathcal{I}$  be the set of positions  $i$  such that  $c_i \neq \check{c}_i$ . Then, by taking the logarithm on either side, (5.87) is equivalent to

$$\sum_{i \in \mathcal{I}} G_i \leq -a \quad (5.88)$$

<sup>15</sup>Note that  $d_A(s_i, \check{s}_i)/\log(2)$  is equal to  $I(\check{s}_i) - I(s_i)$ , i.e., the difference in self-information between  $\check{s}_i$  and  $s_i$ .

where  $G_i = \log(p(Y_i = Z_i|C_i = c_i)/p(Y_i = Z_i|C_i = \check{c}_i))$ . If we denote  $G = \sum_{i \in \mathcal{I}} G_i$ , the MAP pairwise error probability is given by  $P_{h,a} = P(G + a \leq 0)$ . With a few simplifications, it follows from the expression of  $Y_i$  in (5.84) that  $G_i = 2Z_i c_i / \sigma^2$ , where  $Z_i \sim \mathcal{N}(c_i, \sigma^2)$  by (5.85), thus  $G_i \sim \mathcal{N}(2/\sigma^2, 4/\sigma^2)$  and finally  $G + a \sim \mathcal{N}(2h/\sigma^2 + a, 4h/\sigma^2)$ . At last, since  $P_{h,a} = P(G + a \leq 0)$  and since  $\sigma^2 = N_0 / (2R_c E_b)$ ,

$$P_{h,a} = \frac{1}{2} \operatorname{erfc} \left( \sqrt{hR_c E_b / N_0} + \frac{a}{4\sqrt{hR_c E_b / N_0}} \right) \quad (5.89)$$

where  $h = d_H(c, \check{c})$  is the Hamming distance and  $a = d_A(c, \check{c})$  is the a priori distance, see (5.83).

#### 5.8.4.3 Extension of the bounds

This expression shows that extending the bounds to the MAP decoder for VLC streams is as simple as adding one dimension to the VLC spectrum (Definition 5.10), namely the a priori distance  $a = d_A(s, \check{s})$ . For example, the upper bound (5.29) on the  $\text{SER}_L$  becomes

$$\text{SER}_L \leq \sum_{h \geq 1} \sum_a P_{h,a} \sum_{s_L \geq 1} s_L A_{s_L, h, a}^{\text{vlc}} \quad (5.90)$$

where  $A_{s_L, h, a}^{\text{vlc}} = \sum_{l_s, l_b} A_{s_L, h, a, l_s, l_b}^{\text{vlc}}$  and

$$A_{s_L, h, a, l_s, l_b}^{\text{vlc}} \triangleq \sum_{\substack{s_{1:l_s} \in \mathcal{A}^{l_s} \\ \text{s.t. } l(s_{1:l_s}) = l_b}} P(s_{1:l_s}) A_{s_L, h, a | s_{1:l_s}}^{\text{vlc}} \quad (5.91)$$

$$A_{s_L, h, a | s}^{\text{vlc}} \triangleq \left| \left\{ \check{s} \in \mathcal{A}^+ : \begin{array}{l} s, \text{ and } \check{s} \text{ form an error event,} \\ d_{s_L}(s, \check{s}) = s_L, d_H(s, \check{s}) = h, \\ d_A(s, \check{s}) = a \end{array} \right\} \right|. \quad (5.92)$$

Notice the MAP upper bound (5.90) converges to the ML upper bound (5.29) at high  $E_b/N_0$  ratios under Assumption 5.6, as expected, because  $P_{h,a}$  in (5.89) converges to  $P_h$ .

In practice, we need some simplifications, though. Indeed, the a priori distance is a real number and thus the triplet  $(s_L, h, a)$  takes too many different values for practical computer storage of the spectrum  $A_{s_L, h, a}^{\text{vlc}}$ . This can be solved easily by quantizing the a priori distance and limiting its range of values, for example with  $q(d_A(s, \check{s}))$  for some function  $q(\cdot)$ . It is important,

however, that the upper bounds remain upper bounds despite the quantization/limitation. We can show that any odd function  $q(\cdot)$ ,  $q(d) = -q(-d)$ , fulfills these requirements. For example, we can use

$$d_A^{\delta,\Delta}(s_:, \check{s}_:) = q^{\delta,\Delta}(d_A(s_:, \check{s}_:)), \quad (5.93)$$

$$q^{\delta,\Delta}(d) \triangleq \begin{cases} \delta \lfloor |d|/\delta + 1/2 \rfloor \text{sign}(d), & \text{if } |d| < \Delta, \\ \Delta \text{sign}(d), & \text{otherwise,} \end{cases} \quad (5.94)$$

which rounds the a priori information to the nearest multiple of  $\delta$  and limits the values to the interval  $[-\Delta, \Delta]$ . Since  $q^{\delta,\Delta}(\cdot)$  is odd, we can use  $d_A^{\delta,\Delta}(s_:, \check{s}_:)$  instead of  $d_A(s_:, \check{s}_:)$  in (5.92) and still keep an inequality in (5.90). Notice in particular that we find back the ML performance bounds when  $\Delta = 0$ .

For concatenated VLC blocks, extending the bounds to the MAP decoder is very similar. We simply have to incorporate the a priori distance in the spectra and in the bounds, with obvious extensions of (5.43), (5.42), (5.54) and (5.57). Note that the extension of (5.43), in particular, is possible because the a priori distance  $d_A(s_:, \check{s}_:)$  and the quantized a priori distance  $d_A^{\delta,\Delta}(s_:, \check{s}_:)$  are additive, i.e.,

$$d_A(\check{s}_:, s_:) = \sum_{e \in \mathcal{E}(\check{s}_:, s_)} d_A(e), \quad (5.95)$$

where  $d_A(e)$  is the a priori distance of the error event  $e$ , i.e., if  $e$  is formed by the sub-sequences  $s_{m:n}$  and  $\check{s}_{m':n'}$ , then  $d_A(e) = d_A(s_{m:n}, \check{s}_{m':n'})$ . In other words, the a priori distance between two sequences  $s_:, \check{s}_:$  is the sum of the a priori distances of the errors events formed by  $s_:$  and  $\check{s}_:$ .

#### 5.8.4.4 Extension of the pairwise error probability for the BSC

At last, for the binary symmetric channel with error probability  $p$ , the MAP pairwise error probability is given by

$$P_{h,a} = \sum_{j=\max\{1+\lfloor j^* \rfloor, 0\}}^h \binom{h}{j} p^j (1-p)^{h-j} + \frac{1}{2} \mathbb{I}\{j^* \in \mathbb{Z}, 0 \leq j^* \leq h\} \binom{h}{j^*} p^{j^*} (1-p)^{h-j^*} \quad (5.96)$$

where  $j^* = (a / \log(\frac{1-p}{p}) + h) / 2$ .

### 5.8.5 Extension to sources with memory

Extending the results to sources with memory, i.e., with  $P(S_k|S_{k-1}) \neq P(S_k)$ , is possible in theory. But it can be difficult in practice. The evaluation of the spectrum has indeed a computational complexity that increases with the memory of the source/VLC Markov chain, as mentioned in Section 5.5.3. Thus the computation of the spectrum increases with the source memory. This can be however counterbalanced if the source memory is associated with a strong increase in error correcting capabilities, which tends to decrease the computational complexity.

As for the extension itself, there are two important issues that are worth mentioning. Let us consider a Markov source of symbols, with  $P(S_k|S_{k-1}, S_{k-2}, \dots) = P(S_k|S_{k-1})$ .

Firstly, note that many proofs in the appendices are based on the probability to finish some codeword at a given bit position (probability of the root state in the Balakirsky trellis, see Section 5.2.2). Due to the source memory, this must be extended to the probability to finish a given codeword at a given bit position, for all possible codewords. Similarly, without memory, an error event ends at a given bit position if both sequences finish a codeword at that position. For sources with memory, an error event ends at a given bit position if both sequences finish the *same* codeword at that position.

Secondly, and perhaps more importantly, the extension of the bounds to a source with memory does not make sense without considering the extension to the MAP decoder discussed in Section 5.8.4. Indeed, the source memory, which is so beneficial with a MAP decoder, is not taken into account by the ML bounds. More precisely, the ML bounds depend on the source statistics only through the average of the conditional spectrum over all possible sequence realizations, see Definition 5.10. By contrast, the MAP bounds depend on the source statistics both through this average and through the a priori distance  $d_A(\cdot, \cdot)$  (with the modified pairwise error probability  $P_{h,\alpha}$ ). In particular, the MAP conditional spectrum (5.92) is much influenced by the source statistics.

## 5.9 Simulation results

In this Section, we are interested mainly in illustrating the validity of the results and the tightness of the estimations with simulation results. Note that design examples illustrating the interleaving gains have already been considered in Section 3.7 and in [9].

### 5.9.1 Tightness of the estimations

The tightness of the estimations at moderate and high SNRs is illustrated in Fig. 5.3, where the estimation (5.59) of the  $SER_L$ , with the modification suggested in Corollary 5.33, is compared with simulation results. The system considered consists in the source and the VLCs listed in Table 5.2, framed according to  $F_b$  (Section 5.6.3), serially concatenated with a uniform interleaver of size  $N = 384$  and with a recursive systematic punctured CC encoder with generators  $(07, 05)_8$ , where  $07_8$  is the feedback. The parity bits of the CC are punctured to get a channel code rate of  $2/3$ . The simulation results are based on the iterative suboptimal MAP decoder described in Section 5.3.2 and are averaged over 5000 error frames.

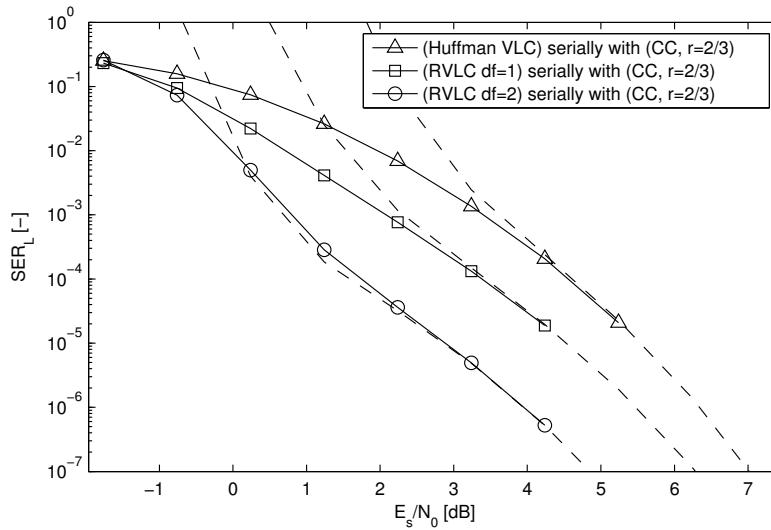
This system is based on relatively weak components and deserves some explanation. On the one hand, illustrating the tightness of the bounds requires simulation results with a high level of statistical accuracy. On the other hand, such a level of accuracy requires lengthy Monte-Carlo simulations — to measure a sufficient number of representative error samples — and is therefore difficult. By choosing a system based on weak components, we have actually lowered the robustness of the system and thus the time needed for the Monte-Carlo simulations since errors occur more often.

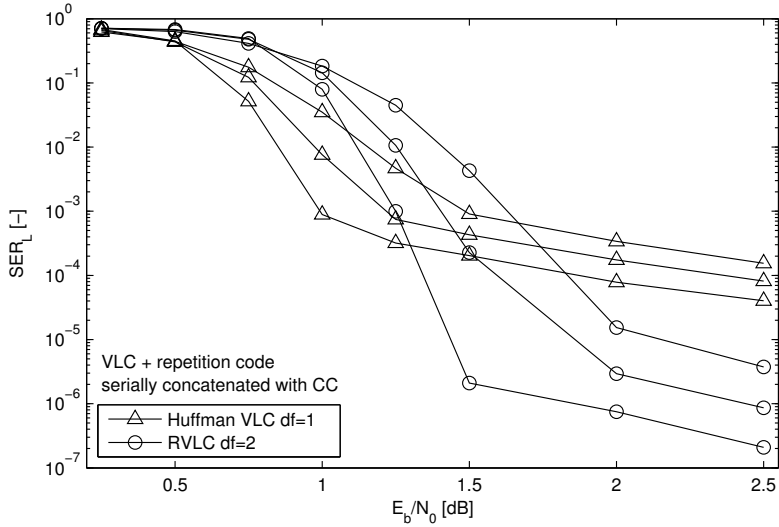
### 5.9.2 Interleaving gains

To illustrate the impact of the interleaving gains, let us consider the following more robust system. Consider a source/VLC, framed according to  $F_b$ , directly followed by a rate- $1/2$  regular repetition code, which repeats each VLC bit twice. The repetition code is then serially concatenated with a uniform interleaver of size  $N$  and with a recursive non-systematic punctured CC encoder with generators  $(037, 021)_8$ , where  $037_8$  is the feedback. The global code

**Table 5.2** Symbols probabilities and VLCs used in Fig. 5.3.

	Prob.	VLC	RVLC <sub>df=1</sub>	RVLC <sub>df=2</sub>
	0.33	11	00	10
	0.30	10	01	01
	0.18	00	10	000
	0.10	011	111	111
	0.09	010	11011	1100
Bounded spectrum	-	yes	yes	yes
Entropy	2.139	-	-	-
Average length $\bar{l}$	-	2.19	2.37	2.46
Free distance $d_f^{\text{vlc}}$	-	1	1	2

**Figure 5.3** Simulation results (solid lines) and union bounds (dashed lines) on the  $\text{SER}_L$  for different VLCs listed in Table 5.2 and concatenated with a punctured recursive systematic CC encoder.



**Figure 5.4** Illustration of the interleaving gains on the  $SER_L$  for two VLCs and different interleaver sizes  $N \in \{1000, 2000, 4000\}$ .

rate is fixed to  $R_c = 1/2$ , which is obtained by adapting the puncturing rate of the CC coded bits given the code rate of the chosen VLC. Finally, the VLC bits and the CC coded bits are multiplexed and sent across the channel. The repetition code repeats each bit twice; thus the output of the repetition code can be viewed as the output of a new VLC with a free distance twice as large. Therefore, the interleaving gain (5.73) on the  $SER_L$ , for example, becomes

$$SER_L = \mathcal{O}\left(N^{-\lfloor \frac{2d_f^{vlc}+1}{2} \rfloor + \epsilon}\right) = \mathcal{O}\left(N^{-d_f^{vlc} + \epsilon}\right). \tag{5.97}$$

This interleaving gain on the  $SER_L$  is illustrated in Fig. 5.4, with the English alphabet source (see [49] and references therein), followed by a Huffman VLC with  $d_f^{vlc} = 1$  or a reversible VLC with  $d_f^{vlc} = 2$ . As we can see, the  $SER_L$  decreases as expected as  $N^{-1}$  in Fig. 5.4 when the Huffman VLC is used, and as  $N^{-2}$  when the reversible VLC is used.

### 5.9.3 Bounded spectrum and interleaving gains

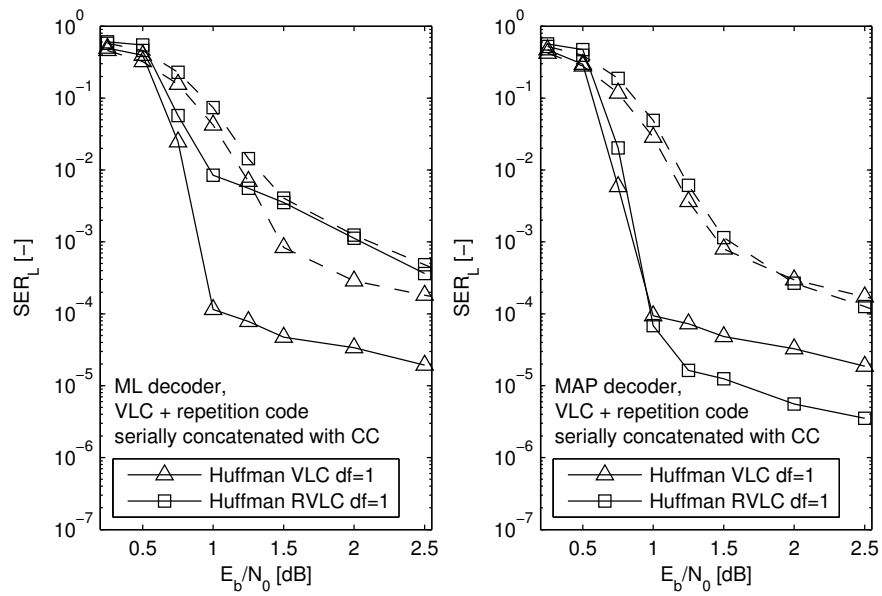
At last, let us examine how significant it is to assume a bounded VLC spectrum — or one of the equivalences from Theorem 5.40 — in order to ensure

**Table 5.3** Symbols probabilities and VLCs used in Fig. 5.5.

	Prob.	VLC	RVLC	Prob. $P_{\text{ml}}(s)$
	0.30	10	01	0.25
	0.13	110	000	0.125
	0.11	010	100	0.125
	0.09	011	110	0.125
	0.09	001	111	0.125
	0.09	1110	0010	0.0625
	0.08	1111	0011	0.0625
	0.06	0001	1010	0.0625
	0.05	0000	1011	0.0625
Bounded spectrum	-	yes	no	-
Entropy	2.943	-	-	3.000
Average length $\bar{l}$	-	2.980	2.980	-
Free distance $d_f^{\text{vlc}}$	-	1	1	-

the interleaving gains in Theorem 5.54. Let us consider again the system from Section 5.9.2 (with a rate-1/2 regular repetition code), but this time with the source and the two VLCs listed in the first three columns in Table 5.3. Both VLCs are Huffman VLCs. The former has no particularity. The latter is a complete reversible VLC (RVLC); its spectrum is thus not bounded, by Proposition 5.45. This reversible VLC comes from Example 5.46; together with the repetition code, it is equivalent to the reversible VLC in Example 5.47. For this system, recall that the interleaving gain on the  $\text{SER}_L$  is given by (5.97) when the VLC spectrum is bounded. The  $\text{SER}_L$  is shown in Fig. 5.5, with the frame-ML decoder on the left and with the frame-MAP decoder on the right.

When the ML decoder is used, the interleaving gains are as expected. With the non-reversible VLC (bounded spectrum), there is an interleaving gain and the  $\text{SER}_L$  decreases as  $N^{-1}$ . With the reversible VLC (unbounded spectrum), there is no interleaving gain. So this example corroborates the significance of bounded VLC spectra with the ML decoder.



**Figure 5.5** Illustration of the interleaving gains on the  $SER_L$  with the ML decoder and with the MAP decoder, for two VLCs and two interleaver sizes  $N = 1000$  (dashed lines) and  $N = 10000$  (solid lines).

Interestingly, when the MAP decoder is used, things are different. There is an interleaving gain with both VLCs, even though the spectrum of the reversible VLC is not bounded. As a consequence, this interleaving gain gives rise to a huge difference in  $\text{SER}_L$  between the ML decoder and the MAP decoder when the reversible VLC is used. Such a difference is quite surprising at first sight, for two reasons. Firstly, the difference between the ML and MAP decoders is only due to the residual redundancy in the VLC stream, which is very small in this case (close to 1%). Secondly, and by contrast, there is almost no difference in  $\text{SER}_L$  between the ML and MAP decoders when the non-reversible VLC is used.

It is interesting to elucidate what lies behind this difference. To guarantee the interleaving gain, the key requirement that emerges from the developments underlying Theorem 5.52 and Theorem 5.54 is essentially that the probability of long error events must be “small”. When the VLC spectrum is bounded, this requirement is trivially satisfied. When the VLC spectrum is not bounded, it is not necessarily satisfied but it might well be, though, if the frame-MAP decoder is used (instead of the frame-ML decoder) *and* if the symbols are not distributed as in (5.82).

To understand this, let us consider the error event formed by the pair  $(S_{1:n}, \check{S}_{1:\check{n}})$  subject to  $d_H(S_{1:n}, \check{S}_{1:\check{n}}) = h$ . If the a priori distance  $d_A(S_{1:n}, \check{S}_{1:\check{n}}) = \log(P(S_{1:n})) - \log(P(\check{S}_{1:\check{n}}))$  is sufficiently large compared to the reliability of the channel, then the MAP decoder will most likely discard  $\check{S}_{1:\check{n}}$ , unlike the ML decoder, and thus the probability of this error event is small when the MAP decoder is used.

Let us investigate therefore whether the a priori distance  $d_A(S_{1:n}, \check{S}_{1:\check{n}})$  is large when  $n$  is large. By definition, the a priori distance is the difference between two metrics, that is,  $\log(P(S_{1:n}))$  and  $\log(P(\check{S}_{1:\check{n}}))$ . As  $n$  becomes large, the metric  $\log(P(S_{1:n})) = \sum_{m=1}^n \log(P(s_m))$  is relatively close to

$$n \sum_{s \in \mathcal{A}} P(s) \log(P(s)) = -n \log(2) H(S)$$

with probability 1, by ergodicity and by the law of large numbers, where  $H(S)$  is the source entropy. Similarly, the metric  $\log(P(\check{S}_{1:\check{n}})) = \sum_{m=1}^{\check{n}} \log(P(\check{s}_m))$  is relatively close to

$$\check{n} \sum_{s \in \mathcal{A}} P^*(s) \log(P(s))$$

where  $P^*(s)$  is the distribution of the decoded symbols  $\check{S}_{1:\check{n}}$  given that  $S_{1:n}$  has been transmitted and given  $h$  bit errors. Since  $l(S_{1:n}) = l(\check{S}_{1:\check{n}})$  is close to  $n\bar{l}$  and to  $\check{n} \sum_{s \in \mathcal{A}} l(s)P^*(s)$  for large  $n$ , we have the approximation  $\check{n} \approx n\bar{l} / \sum_{s \in \mathcal{A}} l(s)P^*(s)$ . Consequently, the a priori distance  $d_A(S_{1:n}, \check{S}_{1:\check{n}})$  increases with  $n$  with probability 1, for large  $n$ , if

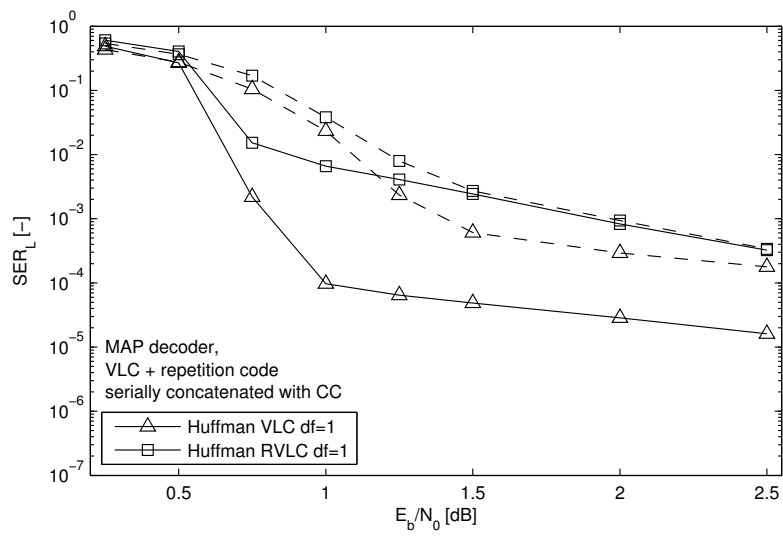
$$-\log(2)H(S) > \frac{\bar{l}}{\sum_{s \in \mathcal{A}} l(s)P^*(s)} \sum_{s \in \mathcal{A}} P^*(s) \log(P(s)), \quad (5.98)$$

which depends on the particular distribution  $P^*(s)$ . For the complete reversible VLC in Table 5.3, it seems reasonable to assume that the sequence  $\check{S}_{1:\check{n}}$  obtained from  $S_{1:n}$  given  $h$  bit errors is any sequence equally likely, thus to assume that  $P^*(s)$  is given by (5.82). Using this  $P^*(s)$  and the values from Table 5.3, we can check numerically that the inequality (5.98) is satisfied for this source and this reversible VLC.

This partially confirms what we wanted to show: The a priori distance  $d_A(S_{1:n}, \check{S}_{1:\check{n}})$  is most probably large for large  $n$  and thus the probability of long error events is small when the MAP decoder is used. Combined with the paragraphs above, this explains partially why there is an interleaving gain in Fig. 5.5 with the complete reversible VLC and the MAP decoder, while there is no interleaving gain with the ML decoder. Put differently, the paragraphs above tend to show that this complete reversible VLC is statistically synchronizable (Definition 5.34) with the source probabilities given in the first column in Table 5.3 when<sup>16</sup> the MAP decoder is used, while it is not synchronizable (by Proposition 5.45) when the ML decoder is used.

One important condition, though, is that the symbols must not be distributed as in (5.82). Otherwise, the MAP decoder is equivalent to the ML decoder, the a priori distance  $d_A(S_{1:n}, \check{S}_{1:\check{n}})$  is always equal to 0 (subject to  $l(S_{1:n}) = l(\check{S}_{1:\check{n}})$ ), the inequality (5.98) is not satisfied and thus there is no interleaving gain with the complete reversible VLC, as illustrated in Fig. 5.6 with the source distribution given in the last column in Table 5.3.

<sup>16</sup>This explains why, compared to [17], we made the concept of statistical synchronization depending on the considered decoder in Definition 5.34 in Section 5.7.1. By contrast, in [17], the concept of statistical synchronization is independent of the considered decoder because the results therein are implicitly restricted to decoders that do not use the source statistics, which excludes the MAP decoder.



**Figure 5.6** Illustration of the interleaving gains on the  $SER_L$  with the MAP decoder, for two VLCs and two interleaver sizes  $N = 1000$  (dashed lines) and  $N = 10000$  (solid lines), when the source distribution is given by (5.82). Same parameters as in Fig. 5.5, except the source distribution which is given in the last column of Table 5.3.

Actually, the closer the symbol distribution is to (5.82), the larger the interleaver size will have to be in order to notice the expected interleaving gain — because the difference between the left-hand side and the right-hand side in (5.98) will be smaller. Similarly, the more reliable the channel is, the larger the interleaver size will have to be so as to notice the gain — because the a priori distance  $d_A(S_{1:n}, \check{S}_{1:n})$  will have to be larger in order to “compete” with the higher channel reliability.

To conclude, simulation results corroborate the significance of bounded VLC spectra. When the VLC spectrum is bounded, the interleaving gains are guaranteed by Theorem 5.54, *independently* of the symbol distribution<sup>17</sup> — thus even when the symbol distribution is unknown at the decoder. When the VLC spectrum is not bounded, simulation results reveal that the MAP decoder might still be able to offer the expected interleaving gains if the symbol distribution is not given by (5.82).

## 5.10 Conclusions

Several theoretical results on the resilience of prefix VLCs concatenated with linear ECCs have been stated and proved, assuming an optimal maximum likelihood (ML) decoder. In a sense, this work pursues previous contributions [20, 72, 73] one step further, by developing more accurate results as well as new results, and by proving them rigorously.

The developed results include notably an approximate expression of the average distance spectrum of the global code (VLC+ECC), performance bounds as well as interleaving gains, and the novel concept of bounded VLC spectrum. Regarding the interleaving gains, we have investigated in particular whether the VLC can contribute to the interleaving gains, just as a convolutional code with non-catastrophic encoder would. We have proved that it does indeed contribute if its spectrum is bounded, even when the source statistics are unknown at the decoder (ML decoder). Besides, simulation results have

---

<sup>17</sup>Note though that, for some VLCs, the “boundness” of the spectrum depends on the source statistics, recall indeed Remark 5.11. In other words, we state here that the interleaving gains are guaranteed for all symbol distributions and source/VLC mappings that make the VLC spectrum bounded.

highlighted that it can contribute also when its spectrum is not bounded, depending on the source statistics, if a maximum a posteriori (MAP) decoder is used.

This last conclusion, not proved and based on simulation results, illustrates one limitation of the present work: The theoretical results herein are focused mainly on the ML decoder and on memoryless sources. Extensions to sources with memory and to the MAP decoder have been discussed in Section 5.8 but further work in this direction would be interesting. Note also that, for the sake of clarity, all theoretical results have been developed in the same framework of assumptions which is summarized in Section 5.4. But many results, taken apart, are certainly extensible to less restrictive assumptions.

Another interesting extension of the present contribution concerns the novel concept of bounded VLC spectrum. We have proved that the class of VLCs with bounded spectrum is closely related, under certain assumptions, to the class of statistically synchronizable VLCs [17], which itself relates to the statistical synchronization of the decoder on the Balakirsky trellis. It would be interesting to investigate how statistical synchronization on other trellises influence the distance spectra of VLCs and the interleaving gains. In particular, recent results have been developed in [60] about the synchronization properties of VLCs on the aggregated state model trellis, which is a general trellis that encompasses the Balakirsky trellis (see Section 2.5.4.2). Establishing the link between these synchronization properties, the distance spectrum and the interleaving gains of the global code is an interesting perspective for future works.

At last, the particular interleaving gains on the (non-Levenshtein) SER proved for reversible VLCs in Chapter 3 suggest to investigate other code properties that might improve the SER. In this context, statistical synchronization on the full symbol-clock trellis (see Section 2.5.1) is certainly one promising possibility, e.g., by means of markers that would ensure the synchronization of the symbol clock (at least statistically) — together with synchronizing sequences (of non-zero probability) which ensure the statistical synchronization of the VLC codewords.

## Appendix 5.A Root state probability, $P(\rho_i)$

The probability to start a codeword at a given bit position is an important quantity in the sequel. Therefore, given the transmitted and decoded streams  $U_{1:\infty} = \text{vlc}(S_{1:\infty})$  and  $\check{U}_{1:\infty} = \text{vlc}(\check{S}_{1:\infty})$ , we now introduce a shorter notation for  $\rho_i(U_{1:\infty}, U_{1:\infty})$ , see (5.18).

**Definition 5.58.** Let  $\rho_i$  be the event  $\rho_i \triangleq \rho_i(U_{1:\infty}, U_{1:\infty})$ . Let  $\bar{\rho}_i$  be the complement of  $\rho_i$ .  $\square$

$P(\rho_i)$  is the probability to start a codeword at a given bit position  $i$  and has the following straightforward properties:

$$P(\rho_i) = 1 - P(\bar{\rho}_i), \quad (5.99)$$

$$P(\rho_1) = 1, \quad (5.100)$$

$$P(\rho_i) = P(\rho_i|\rho_1), \quad (5.101)$$

and, under Assumption 5.4,

$$P(\rho_i) = P(X_{i-1} = R), \quad (5.102)$$

$$P(\rho_i|\rho_j) = P(\rho_{i-j+1}|\rho_1) = P(\rho_{i-j+1}), \quad (5.103)$$

for  $j < i$  and  $j \equiv 1 \pmod{l_{\text{gcd}}^*}$ , where  $X_{i-1}$  is defined in Section 5.2.2. Properties (5.99) and (5.100) follow from Definition 5.58. Property (5.101) follows from (5.100). Under Assumption 5.4, the source/VLC can be modeled by the Markov chain described in Section 5.2.2 and (5.102) follows from the definitions of  $\rho_i$  and  $X_{i-1}$ . At last, property (5.103) follows from the stationarity and periodicity of the Markov chain.

Property (5.102) above tells us that  $P(\rho_i)$  is equivalently the *root state probability* in the Balakirsky trellis (Section 5.2.2). Let us now have a look at its asymptotic behavior for large  $i$ .

**Lemma 5.59** (Root state probability,  $P(\rho_i)$ ). *Under Assumption 5.4, the root state probability  $P(\rho_i)$  has the following behavior:*

$$P(\rho_i) = \begin{cases} l_{\text{gcd}}^*/\bar{l} \pm \mathcal{O}(\theta^i), & \text{if } i \equiv 1 \pmod{l_{\text{gcd}}^*}, \\ 0, & \text{otherwise,} \end{cases} \quad (5.104)$$

where  $\theta = |\lambda_2|^{1/l_{\text{gcd}}^*} < 1$  and  $\lambda_2$  is the root of second highest absolute value of the polynomial

$$c(\lambda) = \lambda^{l_{\text{max}}^*/l_{\text{gcd}}^*} \left( 1 - \sum_{s \in \mathcal{A}} P(s) \lambda^{-l(s)/l_{\text{gcd}}^*} \right). \quad (5.105)$$

If  $l(s) = l_{\text{max}}^* = l_{\text{gcd}}^*$  for all  $s \in \mathcal{A}$ ,  $c(\lambda)$  has only one root and  $\lambda_2 = 0$ .  $\square$

To prove this result, the following lemma is useful and follows as a corollary of the Perron–Frobenius theorem applied to positive stochastic matrices.

**Lemma 5.60.** Consider the matrix

$$\mathbf{M} = \begin{bmatrix} p_1 & p_2 & \cdots & p_{n-1} & p_n \\ 1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \end{bmatrix}, \quad (5.106)$$

which has the characteristic polynomial

$$c(\lambda) = \lambda^n \left( 1 - \sum_{i=1}^n p_i \lambda^{-i} \right). \quad (5.107)$$

If  $\forall i, p_i \geq 0$ ,  $\sum_i p_i = 1$ ,  $\text{gcd}\{i : p_i > 0\} = 1$  and  $p_n > 0$ , then  $\lambda = 1$  is an eigenvalue of  $\mathbf{M}$  of algebraic multiplicity 1 and the all-one vector is an associated eigenvector. All other eigenvalues have an absolute value such that  $0 < |\lambda| < 1$ .  $\square$

**Proof.** Let us analyze the roots of  $c(\lambda)$ , assuming it is the characteristic polynomial of  $\mathbf{M}$ .

- If  $\lambda = 0$ ,  $c(\lambda) = -p_n \neq 0$ .
- If  $|\lambda| > 1$ , we have  $|\lambda^{-i}| < 1$ , hence  $|\sum_{i=1}^n p_i \lambda^{-i}| < 1$  since  $\forall i, p_i \geq 0$  and  $\sum_i p_i = 1$ . Consequently  $c(\lambda) \neq 0$ . At this stage, we have proved that all eigenvalues are such that  $0 < |\lambda| \leq 1$ .
- If  $\lambda = 1$ ,  $c(\lambda) = 0$ . So  $\lambda = 1$  is an eigenvalue of  $\mathbf{M}$ . Besides, given the expression of  $\mathbf{M}$ , the all-one vector is clearly an eigenvector for  $\lambda = 1$  since  $\sum_i p_i = 1$ .
- $\lambda = 1$  is of algebraic multiplicity 1. Indeed, let us factorize  $c(\lambda) = (\lambda - 1) q(\lambda)$  and let us prove that  $q(\lambda = 1) \neq 0$ . Since  $\frac{dc}{d\lambda}(\lambda = 1) =$

$q(\lambda = 1)$ , let us evaluate

$$\frac{dc}{d\lambda}(\lambda) = n\lambda^{n-1} - \sum_{i=1}^n p_i (n-i) \lambda^{n-i-1}. \quad (5.108)$$

Consequently,

$$q(\lambda = 1) = \frac{dc}{d\lambda}(\lambda = 1) = n - \sum_{i=1}^n p_i (n-i) = \sum_{i=1}^n p_i i \neq 0. \quad (5.109)$$

- $\lambda = 1$  is the only eigenvalue on the complex unit circle, i.e., if  $|\lambda| = 1$  and  $c(\lambda) = 0$ , then  $\lambda = 1$ . Indeed, let us write  $\lambda = e^{j\theta}$  for some  $\theta \in (0, 2\pi]$ . It follows from  $c(\lambda) = 0$  that  $\sum_i p_i \lambda^{-i} = 1$ , i.e.,  $\sum_i p_i \cos(i\theta) = 1$ . Let  $\mathcal{S} = \{i : p_i > 0\}$ . Since  $\sum_i p_i = 1$ ,  $\forall i, p_i \geq 0$  and  $\cos(i\theta) \leq 1$ , the equality  $\sum_i p_i \cos(i\theta) = 1$  is equivalent to

$$\forall i \in \mathcal{S}, \cos(i\theta) = 1 \quad (5.110)$$

$$\iff \forall i \in \mathcal{S}, \exists k_i \in \mathbb{N}_{>0}, \theta = 2\pi \frac{k_i}{i} \quad (5.111)$$

$$\iff \exists \theta_n, \theta_d \in \mathbb{N}_{>0}, \gcd\{\theta_n, \theta_d\} = 1$$

$$\forall i \in \mathcal{S}, \exists k_i \in \mathbb{N}_{>0}, \theta = 2\pi \frac{k_i}{i} = 2\pi \frac{\theta_n}{\theta_d} \quad (5.112)$$

$$\implies \forall i \in \mathcal{S}, i \frac{\theta_n}{\theta_d} \in \mathbb{N}_{>0} \quad (5.113)$$

$$\iff \forall i \in \mathcal{S}, \exists l_i \in \mathbb{N}_{>0}, i = \theta_d l_i. \quad (5.114)$$

Since  $\gcd(\mathcal{S}) = 1$ , we have  $\theta_d = 1$ . Consequently,  $\theta = 2\pi\theta_n$  and  $\lambda = e^{j\theta} = 1$ .

At last, the characteristic polynomial  $c(\lambda)$  can be evaluated trivially by recursive cofactor expansion. Here are the first few steps:

$$c(\lambda) = \det \begin{bmatrix} \lambda - p_1 & -p_2 & \cdots & -p_n \\ -1 & \lambda & & \\ & \ddots & \ddots & \\ & & -1 & \lambda \end{bmatrix} \quad (5.115)$$

$$= (\lambda - p_1) \det \begin{bmatrix} \lambda & & & \\ -1 & \lambda & & \\ & & \ddots & \ddots \\ & & & -1 & \lambda \end{bmatrix} \quad (5.116)$$

$$- (-1) \det \begin{bmatrix} -p_2 & -p_3 & \dots & -p_n \\ -1 & \lambda & & \\ & & \ddots & \ddots \\ & & & -1 & \lambda \end{bmatrix} \\ = (\lambda - p_1) \lambda^{n-1} \quad (5.117)$$

$$+ (-p_2) \det \begin{bmatrix} \lambda & & & \\ -1 & \lambda & & \\ & & \ddots & \ddots \\ & & & -1 & \lambda \end{bmatrix} \\ - (-1) \det \begin{bmatrix} -p_3 & -p_4 & \dots & -p_n \\ -1 & \lambda & & \\ & & \ddots & \ddots \\ & & & -1 & \lambda \end{bmatrix} \\ = (\lambda - p_1) \lambda^{n-1} - p_2 \lambda^{n-2} + \dots \quad (5.118)$$

■

**Proof of Lemma 5.59.** We proceed in two steps. Firstly, we assume that  $P(\rho_i)$  converges to a specific value and this value is computed. Then, the convergence is proved.

It follows from Proposition 5.1, Remark 5.8 and property (5.102) that  $P(\rho_i) = 0$  if  $i \not\equiv 1 \pmod{l_{\text{gcd}}^*}$ . Let us then have a look at  $P(\rho_i)$  for  $i \equiv 1 \pmod{l_{\text{gcd}}^*}$ .

Two main equations characterize  $P(\rho_i)$ . Firstly, the event  $\rho_i$  means a symbol  $s$  finished with the bit  $u_{i-1}$  (and started with  $u_{i-l(s)}$ , i.e.,  $\rho_{i-l(s)}$ ). By sum-

ming over all  $s$ , we find

$$P(\rho_i) = \sum_{s \in \mathcal{A}} P(\rho_i | s) P(s) = \sum_{s \in \mathcal{A}} P(\rho_{i-l(s)}) P(s). \quad (5.119)$$

This can be proved with the recursion

$$\begin{aligned} P(\rho_i) &= P(\rho_{i-1})P(\rho_i | \rho_{i-1}) \\ &\quad + P(\rho_{i-2})P(\rho_i, \overline{\rho_{i-1}} | \rho_{i-2}) \\ &\quad + P(\rho_{i-3})P(\rho_i, \overline{\rho_{i-1}}, \overline{\rho_{i-2}} | \rho_{i-3}) + \dots, \end{aligned} \quad (5.120)$$

where

$$P(\rho_i, \overline{\rho_{i-1}}, \dots, \overline{\rho_{i-l+1}} | \rho_{i-1}) = \sum_{s \in \mathcal{A}: l(s)=l} P(s). \quad (5.121)$$

Secondly, the event  $\overline{\rho_i}$  means a symbol  $s$  started in any possible position  $i - j$  subject to  $j < l(s)$  and subject to  $i - j \equiv 1 \pmod{l_{\text{gcd}}^*}$  — recall Remark 5.8. Therefore,

$$P(\overline{\rho_i}) = \sum_{s \in \mathcal{A}} \left( \sum_{k=1}^{l(s)/l_{\text{gcd}}^* - 1} P(\rho_{i-l_{\text{gcd}}^* k}) \right) P(s). \quad (5.122)$$

This can be proved with the recursion

$$\begin{aligned} P(\overline{\rho_i}) &= P(\rho_{i-1})P(\overline{\rho_i} | \rho_{i-1}) \\ &\quad + P(\rho_{i-2})P(\overline{\rho_i}, \overline{\rho_{i-1}} | \rho_{i-2}) \\ &\quad + P(\rho_{i-3})P(\overline{\rho_i}, \overline{\rho_{i-1}}, \overline{\rho_{i-2}} | \rho_{i-3}) + \dots, \end{aligned} \quad (5.123)$$

where

$$P(\overline{\rho_i}, \overline{\rho_{i-1}}, \dots, \overline{\rho_{i-l+1}} | \rho_{i-1}) = \sum_{s \in \mathcal{A}: l(s) > l} P(s). \quad (5.124)$$

The limit value of  $P(\rho_i)$  for large  $i$ , say  $P_\rho$ , can be deduced from (5.99) and (5.122) by assuming the limit exists:

$$1 - P_\rho = \sum_{s \in \mathcal{A}} \left( \frac{l(s)}{l_{\text{gcd}}^*} - 1 \right) P_\rho P(s) = P_\rho \left( \frac{\bar{l}}{l_{\text{gcd}}^*} - 1 \right), \quad (5.125)$$

hence  $P_\rho = l_{\text{gcd}}^* / \bar{l}$ .

A sketch of proof for the rate of convergence is the following. We want to demonstrate the convergence of the recurrence relation (5.119), with initial values  $P(\rho_1) = 1$  and  $P(\rho_j) = 0$  for  $j \leq 0$  — note that these values satisfy (5.122) for  $i = 1$ . Formulated as a matrix relation for  $i \equiv 1 \pmod{l_{\text{gcd}}^*}$ , the

recurrence (5.119) becomes

$$\mathbf{v}^i = \mathbf{M} \mathbf{v}^{i-l_{\text{gcd}}^*} \quad (5.126)$$

where

$$\mathbf{M} = \begin{bmatrix} P_{l_{\text{gcd}}^*} & P_{2l_{\text{gcd}}^*} & \dots & P_{l_{\text{max}}^* - l_{\text{gcd}}^*} & P_{l_{\text{max}}^*} \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix}, \quad (5.127)$$

$$\mathbf{v}^i = \begin{bmatrix} P(\rho_i) \\ P(\rho_{i-l_{\text{gcd}}^*}) \\ P(\rho_{i-2l_{\text{gcd}}^*}) \\ \vdots \\ P(\rho_{i-l_{\text{max}}^* + l_{\text{gcd}}^*}) \end{bmatrix}, \quad (5.128)$$

$$P_k = \sum_{\substack{s \in \mathcal{A} \\ l(s)=k}} P(s). \quad (5.129)$$

Note that  $P_{l_{\text{max}}^*} > 0$  by definition of  $l_{\text{max}}^*$ . The properties on the eigenvalues of  $\mathbf{M}$ , which are the roots of  $c(\lambda)$  in (5.105), follow from Lemma 5.60. The only eigenvector that is associated with the eigenvalue  $\lambda = 1$  and that satisfies (5.122) has all its elements equal to  $l_{\text{gcd}}^* / \bar{l}$ . Since all other eigenvalues have an absolute value  $|\lambda| < 1$ , the convergence is always guaranteed. Moreover, it is straightforward to show that the rate of convergence in (5.104) is characterized by  $\theta = |\lambda_2|^{1/l_{\text{gcd}}^*} < 1$ , where  $\lambda_2$  is the eigenvalue of second highest absolute value. ■

## Appendix 5.B Proofs related to VLC streams

### 5.B.1 Union bounds on Arbitrary Distortions

Let  $q(\cdot)$  be a distortion measure, e.g.,  $d_{S_L}(\cdot)$  or  $d_H(\cdot)$ . Given the transmitted and decoded streams,  $U_{1:\infty} = \text{vlc}(S_{1:\infty})$  and  $\check{U}_{1:\infty} = \text{vlc}(\check{S}_{1:\infty})$ , let  $\Delta_k^s, \Delta_l^b$  be

the random sets

$$\Delta_k^s = \{e \in \mathcal{E}(S_{1:\infty}, \check{S}_{1:\infty}) : e \text{ starts with } S_k\}, \quad (5.130)$$

$$\Delta_l^b = \{e \in \mathcal{E}(U_{1:\infty}, \check{U}_{1:\infty}) : e \text{ starts with } U_l\}, \quad (5.131)$$

and  $\delta_k^s, \delta_l^b$  the associated realizations. There is at most one element in these sets; if  $e \in \Delta_k^s \neq \emptyset$ , then we define  $q(\Delta_k^s) = q(e)$  and  $l_S(\Delta_k^s) = l_S(e)$ , otherwise  $q(\Delta_k^s) = 0$  and  $l_S(\Delta_k^s) = 0$ , and similarly for  $\Delta_l^b$ .

**Lemma 5.61** (Union Bounds on Arbitrary Distortions). *Under Assumption 5.5, for VLC streams, the quantities*

$$Q^s \triangleq \lim_{k \rightarrow +\infty} \mathbf{E}\{q(\Delta_k^s)\}, \quad (5.132)$$

$$Q^b \triangleq \lim_{i \rightarrow +\infty} \mathbf{E}\{q(\Delta_{1+l_{\text{gcd}}^*}^b i)\} \quad (5.133)$$

are upper bounded by

$$Q^s \leq \sum_{h \geq 1} P_h \sum_q q A_{q,h}^{\text{vlc}}, \quad (5.134)$$

$$Q^b \leq \frac{l^*}{l} \sum_{h \geq 1} P_h \sum_q q A_{q,h}^{\text{vlc}} \quad (5.135)$$

where

$$A_{q,h}^{\text{vlc}} = \sum_{s: \in \mathcal{A}^+} P(s:.) \left| \left\{ \check{s}: \in \mathcal{A}^+ : \begin{array}{l} s: \text{ and } \check{s}: \text{ form an error event,} \\ d_H(s:, \check{s}:) = h, q(s:, \check{s}:) = q \end{array} \right\} \right|.$$

□

Note that the limits in the definitions of  $Q^s$  and  $Q^b$  are used only to eliminate the non-stationary effect at the beginning of the semi-infinite streams (close to  $U_1, S_1$ ).

**Proof.** We start with the upper bound on  $Q^s$ . By Definition 5.7 of the error event,  $\Delta_k^s$  is empty, hence  $q(\Delta_k^s) = 0$ , if we have not  $\rho_{k,k'}(S:, \check{S}:)$  for some  $k'$ , which we will write  $\rho_k(S:, \check{S}:)$  for convenience. Consequently,

$$\mathbf{E}\{q(\Delta_k^s)\} = P(\rho_k(S:, \check{S}:)) \mathbf{E}\{q(\Delta_k^s) | \rho_k(S:, \check{S}:)\} \quad (5.136)$$

$$\leq \mathbf{E}\{q(\Delta_k^s) | \rho_k(S:, \check{S}:)\} \quad (5.137)$$

$$= \mathbf{E}\{q(\Delta_1^s)\}, \quad (5.138)$$

where (5.137) follows trivially from  $P(\rho_k(S:, \check{S}:)) \leq 1$  and (5.138) follows from the independence of  $\mathbf{E}\{q(\Delta_k^s) | \rho_k(S:, \check{S}:)\}$  w.r.t.  $k$  and from  $P(\rho_1(S:, \check{S}:)) = 1$ .

Note the inequality (5.137) is tight when the error event probability tends to 0. Indeed,  $P(\rho_k(S, \check{S}))$  is the probability that  $S$  and  $\check{S}$  are synchronized in  $S_k$  and  $S_{k'}$  for some  $k'$ . This probability obviously tends to 1 when the error event probability tends to 0, e.g., when we use VLCs with bounded spectrum at high signal to noise ratios.

Let us define a set more restrictive than  $\Delta_1^s$ :

$$\Delta_{1,l_s,q}^s = \{e \in \Delta_1^s : l_S(e) = l_s, q(e) = q\}. \quad (5.139)$$

By the union bound (see (5.10)), we have immediately

$$\begin{aligned} \mathbf{E}\{q(\Delta_1^s)\} &\leq \sum_{l_s,q} P(\Delta_{1,l_s,q}^s \neq \emptyset) q \\ &= \sum_{l_s,q,s_{1:l_s}} P(\Delta_{1,l_s,q}^s \neq \emptyset | s_{1:l_s}) P(s_{1:l_s}) q, \end{aligned} \quad (5.140)$$

The factor  $P(\Delta_{1,l_s,q}^s \neq \emptyset | s_{1:l_s})$  is the probability to select a wrong path  $\check{s}$  such that  $s_{1:l_s}$  and  $\check{s}$  form an error event of distortion  $q$ . Again, by the union bound,

$$P(\Delta_{1,l_s,q}^s \neq \emptyset | s_{1:l_s}) \leq \sum_{h \geq 1} P_h A_{q,h|s_{1:l_s}}^{\text{vlc}}, \quad (5.141)$$

where  $P_h$  is given in Section 5.1 and

$$A_{q,h|s_{1:l_s}}^{\text{vlc}} = \left| \left\{ \check{s} \in \mathcal{A}^+ : \begin{array}{l} s_{1:l_s} \text{ and } \check{s} \text{ form an error event,} \\ d_H(s_{1:l_s}, \check{s}) = h, q(s_{1:l_s}, \check{s}) = q \end{array} \right\} \right|.$$

Finally, (5.134) follows by gathering everything.

The bound on  $Q^b$  can be proved similarly. Let  $l = 1 + l_{\text{gcd}}^*$ . By Definition 5.7 of the error event,  $\Delta_l^b$  is empty, hence  $q(\Delta_l^b) = 0$ , if we have not  $\rho_l(U, \check{U})$ . Consequently,

$$\mathbf{E}\{q(\Delta_l^b)\} = P(\rho_l(U, \check{U})) \mathbf{E}\{q(\Delta_l^b) | \rho_l(U, \check{U})\} \quad (5.142)$$

$$= P(\rho_l) P(\rho_l(U, \check{U}) | \rho_l) \mathbf{E}\{q(\Delta_l^b) | \rho_l(U, \check{U})\} \quad (5.143)$$

$$\leq P(\rho_l) \mathbf{E}\{q(\Delta_l^b) | \rho_l(U, \check{U})\} \quad (5.144)$$

$$= P(\rho_l) \mathbf{E}\{q(\Delta_l^b)\}, \quad (5.145)$$

where (5.143) follows from the fact that we cannot have  $\rho_l(U, \check{U})$  without  $\rho_l$  — recall  $\rho_l \triangleq \rho_l(U, U)$ , Definition 5.58. The inequality (5.144) follows trivially from  $P(\rho_l(U, \check{U})) \leq 1$  and (5.145) follows from the independence

of  $\mathbf{E}\{q(\Delta_l^b)|\rho_l(U, \check{U})\}$  w.r.t.  $l$  (for  $l$  subject to  $l \equiv 1 \pmod{l_{\text{gcd}}^*}$ ) and from  $P(\rho_1(U, \check{U})) = 1$ . Note the inequality (5.144) is tight when the error event probability tends to 0, for the same reason as (5.137).

To finish the proof, note at last that we have trivially  $\mathbf{E}\{q(\Delta_1^b)\} = \mathbf{E}\{q(\Delta_1^s)\}$ , and by Lemma 5.59,  $P(\rho_l) = P(\rho_{1+l_{\text{gcd}}^*i}) \rightarrow l_{\text{gcd}}^*/\bar{L}$  when  $i \rightarrow \infty$ . ■

### 5.B.2 Proof of Theorem 5.18

**Proof.** By Lemma 5.61, it is sufficient to prove

$$\text{SER}_L \leq \lim_{k \rightarrow +\infty} \mathbf{E}\{d_{S_L}(\Delta_k^s)\}. \quad (5.146)$$

By straightforward extension of Proposition 5.19,

$$\frac{1}{l_s} d_{S_L}(S_{1:l_s}, \check{S}_{1:l_s}) \leq \frac{1}{l_s} \sum_{e \in \mathcal{E}^*(S_{1:l_s}, \check{S}_{1:l_s})} d_{S_L}(e) \quad (5.147)$$

where  $\mathcal{E}^*(S_{1:l_s}, \check{S}_{1:l_s}) = \{e \in \mathcal{E}(S_{1:\infty}, \check{S}_{1:\infty}) : e \text{ starts with } S_k, k \leq l_s\}$ . Note that there are two contributions that make this relation an inequality. The first one is related to the last error event in  $\mathcal{E}^*(S_{1:l_s}, \check{S}_{1:l_s})$ . This last error event might go beyond  $S_{l_s}$ , in which case it is accounted for totally on the right-hand side of (5.147) but only partially on the left-hand side (hence the inequality). This first contribution is however negligible for asymptotically large  $l_s$  because of the division by  $l_s$ . The second contribution comes from the inequality (5.30). For asymptotically large  $l_s$ , only this second contribution matters. Nevertheless, recall (5.30) is not an equality only with particular combinations of error events and thus (5.147) is quite tight in most cases for large  $l_s$ .

Next, recalling the definition of  $\Delta_k^s$  in (5.130),

$$\sum_{e \in \mathcal{E}^*(S_{1:l_s}, \check{S}_{1:l_s})} d_{S_L}(e) = \sum_{k=1}^{l_s} d_{S_L}(\Delta_k^s). \quad (5.148)$$

Therefore, by using the definition of the  $\text{SER}_L$ ,

$$\text{SER}_L \leq \lim_{l_s \rightarrow +\infty} E \left\{ \frac{1}{l_s} \sum_{k=1}^{l_s} d_{S_L}(\Delta_k^s) \right\} \quad (5.149)$$

$$= \lim_{l_s \rightarrow +\infty} \frac{1}{l_s} \sum_{k=1}^{l_s} E\{d_{S_L}(\Delta_k^s)\}, \quad (5.150)$$

which is equal to (5.146) by stationarity. This bound is quite tight; the inequality is only due to (5.30) — that is, the second contribution discussed above, since the first contribution converges to zero when  $l_s \rightarrow +\infty$ . ■

### 5.B.3 Proof of Theorem 5.21

**Proof.** By Lemma 5.61, it is sufficient to prove

$$\text{BER} = \frac{1}{l_{\text{gcd}}^*} \lim_{i \rightarrow +\infty} \mathbf{E}\{d_H(\Delta_{1+l_{\text{gcd}}^*}^b i)\}. \quad (5.151)$$

The proof of this equality is similar to the proof of (5.146):

$$\frac{d_H(U_{1:l_b}, \check{U}_{1:l_b})}{l_b} \leq \frac{1}{l_b} \sum_{e \in \mathcal{E}^*(U_{1:l_b}, \check{U}_{1:l_b})} d_H(e) \quad (5.152)$$

$$= \frac{1}{l_b} \sum_{l=1}^{l_b} d_H(\Delta_l^b) \quad (5.153)$$

$$= \frac{1}{l_{\text{gcd}}^*} \left( \frac{l_{\text{gcd}}^*}{l_b} \sum_{i=0}^{\lfloor (l_b-1)/l_{\text{gcd}}^* \rfloor} d_H(\Delta_{1+l_{\text{gcd}}^*}^b i) \right), \quad (5.154)$$

where  $\mathcal{E}^*(U_{1:l_b}, \check{U}_{1:l_b}) = \{e \in \mathcal{E}(U_{1:\infty}, \check{U}_{1:\infty}) : e \text{ starts with } U_l, l \leq l_b\}$ , and the set  $\Delta_l^b$  defined in (5.131) is empty (thus  $d_H(\Delta_l^b) = 0$ ) for all  $l \not\equiv 1 \pmod{l_{\text{gcd}}^*}$  if we restrict the enumeration to sub-sequences  $U_{1:l_b}$  of non-zero probabilities. Note the inequality (5.152) is only due to the last error event in  $\mathcal{E}^*(U_{1:l_b}, \check{U}_{1:l_b})$  — see the first contribution to the inequality (5.147) discussed here above — and converges therefore to an equality for asymptotically large  $l_b$ . The end of the proof is identical to the proof of Theorem 5.18. ■

Another definition of the BER, equivalent to (5.31), is

$$\text{BER} \triangleq \lim_{l \rightarrow +\infty} \mathbf{E} \left\{ \frac{1}{l_{\text{gcd}}^*} \sum_{i=0}^{l_{\text{gcd}}^*-1} \mathbb{I}\{\check{U}_{l+i} \neq U_{l+i}\} \right\}, \quad (5.155)$$

which simplifies into the intuitive expression

$$\text{BER} \triangleq \lim_{l \rightarrow +\infty} \mathbf{E}\{\mathbb{I}\{\check{U}_l \neq U_l\}\} \quad (5.156)$$

when  $l_{\text{gcd}}^* = 1$ . Note that such an intuitive expression for the  $\text{SER}_L$  is not possible because of Proposition 5.19.

We can prove the upper bound (5.32) on the BER, starting from definition (5.155). The proof is similar to the proof used for convolutional codes in [111, Sec. 11.1.5, “Bit error probability”, pp. 567–568].

**Proof of the upper bound (5.32) on the BER (5.155).** For clarity, let us consider  $l \equiv 1 \pmod{l_{\text{gcd}}^*}$ . The generalization to any  $l$  is straightforward.

We can restrict the proof to transmitted sequences made up of codewords of non-zero probability — other sequences are never transmitted and do not contribute therefore to the error rates. Having some bit errors  $\check{U}_{l+i} \neq U_{l+i}$  for  $0 \leq i < l_{\text{gcd}}^*$  means an error event started in the past, is still active in  $U_{l+i}$  for  $0 \leq i < l_{\text{gcd}}^*$  and leads to at least a bit error among these bits. Let us define

$$\Delta_{j,l_b,h}^b = \{e \in \Delta_j^b : l(e) = l_b, d_H(e) = h\}, \quad (5.157)$$

which is empty for all  $j \not\equiv 1 \pmod{l_{\text{gcd}}^*}$  and all  $l_b \not\equiv 0 \pmod{l_{\text{gcd}}^*}$ , by Definition 5.7 of an error event — recall we restrict the discussion to transmitted sequences made up of codewords of non-zero probability. Having an error event active in  $U_{l+i}$  for  $0 \leq i < l_{\text{gcd}}^*$  means  $\Delta_{j,l_b,h}^b \neq \emptyset$  for some  $h$ , some  $l_b \equiv 0 \pmod{l_{\text{gcd}}^*}$  and some  $j \in \mathcal{J}_{l,l_b} = \{l - l_b + l_{\text{gcd}}^*, l - l_b + 2l_{\text{gcd}}^*, \dots, l\}$ . Such a  $\Delta_{j,l_b,h}^b \neq \emptyset$  is associated with  $h$  bit errors, a fraction  $l_{\text{gcd}}^*/l_b$  of them being in average among the bits  $U_{l+i}$  for  $0 \leq i < l_{\text{gcd}}^*$ . Therefore, by the union bound,

$$\text{BER} \leq \frac{1}{l_{\text{gcd}}^*} \sum_{l_b,h} \sum_{j \in \mathcal{J}_{l,l_b}} P(\Delta_{j,l_b,h}^b \neq \emptyset) h \frac{l_{\text{gcd}}^*}{l_b} \quad (5.158)$$

$$\leq \frac{1}{l_{\text{gcd}}^*} \sum_{l_b,h} \sum_{j \in \mathcal{J}_{l,l_b}} P(\rho_j) P(\Delta_{j,l_b,h}^b \neq \emptyset | \rho_j(U, \check{U})) h \frac{l_{\text{gcd}}^*}{l_b} \quad (5.159)$$

$$= \frac{1}{l_{\text{gcd}}^*} \sum_{l_b,h} P(\Delta_{1,l_b,h}^b \neq \emptyset) h \left( \frac{l_{\text{gcd}}^*}{l_b} \sum_{j \in \mathcal{J}_{l,l_b}} P(\rho_j) \right), \quad (5.160)$$

where (5.159) follows from the same reason as (5.144), and (5.160) follows from the independence of  $P(\Delta_{j,l_b,h}^b | \rho_j(U, \check{U}))$  w.r.t.  $j$  (as long as  $j \equiv 1 \pmod{l_{\text{gcd}}^*}$ ) and from  $P(\rho_1(U, \check{U})) = 1$ . To finish the proof, note that the quantity between brackets converges toward  $l_{\text{gcd}}^*/\bar{l}$  when  $l \rightarrow +\infty$  by Lemma 5.59 and that the summation  $\sum_{l_b,h} P(\Delta_{1,l_b,h}^b \neq \emptyset) h$  can be upper bounded as (5.140). ■

### 5.B.4 Proof of Theorem 5.22

**Proof.** By Lemma 5.61, it is sufficient to prove

$$\text{EER}^s = \lim_{k \rightarrow +\infty} \mathbf{E}\{d_{EE}(\Delta_k^s)\}, \quad (5.161)$$

$$\text{EER}^b = \lim_{i \rightarrow +\infty} \mathbf{E}\{d_{EE}(\Delta_{1+l_{\text{gcd}}^*}^b)_i)\}, \quad (5.162)$$

where  $d_{EE}(\Delta_k^s) = \mathbb{I}\{\Delta_k^s \neq \emptyset\}$  and  $d_{EE}(\Delta_{1+l_{\text{gcd}}^*}^b)_i = \mathbb{I}\{\Delta_{1+l_{\text{gcd}}^*}^b_i \neq \emptyset\}$ . The proof of these equalities is similar to the proofs of (5.146) and (5.151). Note simply that the equations corresponding to (5.148) and (5.153) are

$$\frac{d_{EE}(S_{1:l_s}, \check{S}_{1:l_s})}{l_s} = \frac{1}{l_s} \sum_{k=1}^{l_s} d_{EE}(\Delta_k^s), \quad (5.163)$$

$$\frac{d_{EE}(U_{1:l_b}, \check{U}_{1:l_b})}{l_b/l_{\text{gcd}}^*} = \frac{l_{\text{gcd}}^*}{l_b} \sum_{l=1}^{l_b} d_{EE}(\Delta_l^b). \quad (5.164)$$

■

## Appendix 5.C Proofs related to VLC blocks

### 5.C.1 Proof of Theorem 5.26

**Proof.** Let us rewrite (5.37). The summation in (5.37) enumerates all possible  $s$ , and the conditional spectrum  $A_{s_L, h|s}^{\text{Bvlc}}$  involves all possible  $\check{s}$ , given  $s$ . Let us rearrange them such that the joint enumeration of  $s$  and  $\check{s}$  is sorted by the number  $n$  of events in  $\mathcal{E}(s, \check{s})$ , by the symbol lengths  $l_{1:n}$  of these events, by their starting positions  $j_{1:n}$ , by their Levenshtein symbol distances  $s_{L,1:n}$  and by their Hamming distances  $h_{1:n}$ . These quantities are constrained as follows. For  $l_{1:n}$ , let  $\mathcal{L}_n$  denote the set of possible lengths,

$$\mathcal{L}_n = \{l_{1:n} : \sum_m l_m \leq N_s\}. \quad (5.165)$$

For  $j_{1:n}$ , let  $\mathcal{J}_{l_{1:n}}$  be the set of possible positions,

$$\mathcal{J}_{l_{1:n}} = \{j_{1:n} : j_1 \geq 1, j_n \leq N_s - l_n + 1, j_m \geq j_{m-1} + l_{m-1} \text{ for } 2 \leq m \leq n\}. \quad (5.166)$$

For  $h_{1:n}$ ,  $\sum_m h_m = h = d_H(s, \check{s})$ . For  $s_{L,1:n}$ , unfortunately, we know from Proposition 5.19 that the link between  $s_{L,1:n}$  and  $s_L = d_{S_L}(s, \check{s})$  is not un-equivocal due to the inequality  $\sum_{m=1}^n s_{L,m} \geq s_L$ . Nevertheless, we make the

approximation in the following that it is an equality. This approximation is supported by two facts: (i) this inequality is an equality in most cases; (ii) because of the sense of this inequality, replacing it with an equality will result eventually in a spectrum that counts more Levenshtein symbol errors than the correct spectrum ( $\sum_{m=1}^n s_{L,m}$  instead of  $s_L$ ), which is compatible with our interest in upper performance bounds. So, eventually, we have the following constraints on  $h_{1:n}$  and  $s_{L,1:n}$ :

$$\sum_{m=1}^n h_m = h \quad \text{and} \quad \sum_{m=1}^n s_{L,m} = s_L. \quad (5.167)$$

By rearranging (5.37) as mentioned, the spectrum can be written approximately (the approximation comes from (5.167)) as

$$A_{s_L, h}^{F_s} \approx \sum_{s: \in \mathcal{A}^{N_s}} \sum_{n \geq 1} \sum_{l: \in \mathcal{L}_n} \sum_{j: \in \mathcal{J}_l} \sum_{s_{L,:}, h:} \text{s.t. (5.167)} P_{F_s}(B_{\text{vlc}} = s: | A_{s_L, h | s: s_{L,:}, h: j: j: l:}^{F_s}) \quad (5.168)$$

where  $A_{s_L, h | s: s_{L,:}, h: j: j: l:}^{F_s}$  counts the number of  $\mathcal{S}$ : that form  $n$  error events  $e_1, e_2, \dots, e_n$  with  $s$ : such that  $e_m$  starts with  $s_{j_m}$ ,  $l_m = l_S(e_m)$ ,  $h_m = d_H(e_m)$  and  $s_{L,m} = d_{S_L}(e_m)$ . Such sequences might not exist, so this number might be zero. It is equal to the product of the conditional spectra (5.24) of the sub-sequences  $s_{j_m:j_m+l_m-1}$  associated with each  $e_m$ ,

$$A_{s_L, h | s: s_{L,1:n}, h_{1:n}, j_{1:n}, l_{1:n}}^{F_s} = \prod_{m=1}^n A_{s_L=s_{L,m}, h=h_m | s_{y_m}}^{\text{vlc}} \quad (5.169)$$

where we use the notations  $\mathcal{Y}_m = j_m : j_m + l_m - 1$  and  $s_{y_m} = s_{j_m:j_m+l_m-1}$ . With  $s_{y:} = (s_{y_1}, s_{y_2}, \dots, s_{y_n})$ , eq. (5.168) can be written as

$$A_{s_L, h}^{F_b} \approx \sum_{s: \in \mathcal{A}^{N_s}} \sum_{n \geq 1} \sum_{l: \in \mathcal{L}_n} \sum_{j: \in \mathcal{J}_l} \sum_{s_{L,:}, h:} \text{s.t. (5.167)} P_{F_s}(s_{y:}) \left( \prod_{m=1}^n A_{s_L=s_{L,m}, h=h_m | s_{y_m}}^{\text{vlc}} \right) \times P_{F_s}(B_{\text{vlc}} = s: | s_{y:}) \quad (5.170)$$

$$\begin{aligned}
&= \sum_{n \geq 1} \sum_{l \in \mathcal{L}_n} \sum_{j \in \mathcal{J}_l} \sum_{\substack{s_{L,:}, h, \\ \text{s.t. (5.167)}}} \\
&\quad \sum_{s_{\mathcal{Y}}:} \left( \prod_{m=1}^n P_{F_s}(s_{\mathcal{Y}_m}) A_{s_L=s_{L,m}, h=h_m | s_{\mathcal{Y}_m}}^{\text{vlc}} \right) \\
&\quad \times \sum_{s' \in \mathcal{A}^{N_s}} P_{F_s}(B_{\text{vlc}} = s' | s_{\mathcal{Y}}). \tag{5.171}
\end{aligned}$$

We have immediately the following simplifications:

$$\sum_{s' \in \mathcal{A}^{N_s}} P_{F_s}(B_{\text{vlc}} = s' | s_{\mathcal{Y}}) = 1 \tag{5.172}$$

and

$$\begin{aligned}
\sum_{s_{\mathcal{Y}_m}} P_{F_s}(s_{\mathcal{Y}_m}) A_{s_L=s_{L,m}, h=h_m | s_{\mathcal{Y}_m}}^{\text{vlc}} &= \sum_{s_{1:l_m}} P(s_{1:l_m}) A_{s_L=s_{L,m}, h=h_m | s_{1:l_m}}^{\text{vlc}} \\
&= A_{s_L=s_{L,m}, h=h_m, l_s=l_m}^{\text{vlc}}. \tag{5.173}
\end{aligned}$$

Eventually, with all these simplifications, we can write

$$A_{s_{L,h}}^{F_s} \approx \sum_{n \geq 1} \sum_{l \in \mathcal{L}_n} \sum_{\substack{s_{L,:}, h, \\ \text{s.t. (5.167)}}} |\mathcal{J}_l| \prod_{m=1}^n A_{s_L=s_{L,m}, h=h_m, l_s=l_m}^{\text{vlc}} \tag{5.174}$$

where  $|\mathcal{J}_l|$  is the number of possible positions  $j$  subject to  $l$ . By combinatorics, it is straightforward to show that  $|\mathcal{J}_l| = \binom{N_s - \sum_{m=1}^n l_m + n}{n}$  — for details, see  $\mathcal{D}_{n, N=N_s - \sum_{m=1}^n l_m}$  in Lemma 5.63. After simplifications, (5.42) follows. ■

### 5.C.2 The framing rule $P_{F_b}$ , properties

**Proof of Property 5.27.** The knowledge of  $N_b^{(-1)}$  is a constraint on the first symbol, more precisely on the length of the first codeword,  $l(S_1) > N - N_b^{(-1)}$ . The subsequent symbols, from  $S_2$  to  $S_{N_s}$ , are not constrained. At last,  $S_{N_s}$  is the last symbol of the VLC block if the next symbol in the VLC stream cannot fit in the VLC block, i.e., if  $l(s_{N_s+1}) > N - l(s_{1:N_s})$ . Therefore, the probability of terminating the VLC block with  $S_{N_s}$  is  $P(l(S_{N_s+1}) > N - l(s_{1:N_s})) = P(l(S) >$

$N - l(s_{1:N_s})$ ). It follows that  $P_{F_b}(B_{\text{vlc}} = s: |N_b^{(-1)})$  can be expressed as

$$\begin{aligned} P_{F_b}(B_{\text{vlc}} = s: |N_b^{(-1)}) &= \mathbb{I}\{N - l_{\max}^* < l(s:) \leq N\} \\ &\times P(S_1 = s_1 | l(S_1) > N - N_b^{(-1)}) \\ &\times \left( \prod_{k=2}^{l_s(s:)} P(S_k = s_k) \right) P(l(S) > N - l(s:)) \end{aligned} \quad (5.175)$$

where

$$\begin{aligned} P(S_1 = s_1 | l(S_1) > N - N_b^{(-1)}) \\ &= P(S_1 = s_1) \frac{P(l(S_1) > N - N_b^{(-1)} | S_1 = s_1)}{P(l(S_1) > N - N_b^{(-1)})} \end{aligned} \quad (5.176)$$

$$= P(S = s_1) \frac{\mathbb{I}\{l(s_1) > N - N_b^{(-1)}\}}{P(l(S) > N - N_b^{(-1)})}. \quad (5.177)$$

After simplifications, this leads to (5.48). ■

The following lemma is useful to prove Property 5.28 and Property 5.29.

**Lemma 5.62** (Properties of framing rule  $F_b$ ). *Under Assumption 5.5, we have*

$$\begin{aligned} P_{F_b}(u_{j:i-1} | \rho_j, N_b^{(-1)}) &= P_{F_b}(u_{j:i-1} | \rho_j) \\ &= P(u_{j:i-1} | \rho_j) = P(s_{m:n}), \end{aligned} \quad (5.178)$$

for  $N \geq i > j > 1$  and  $u_{j:i-1} \equiv s_{m:n}$ , and therefore

$$\begin{aligned} P_{F_b}(\rho_i | \rho_j, N_b^{(-1)}) &= P_{F_b}(\rho_i | \rho_j) \\ &= P(\rho_i | \rho_j) = P(\rho_{i-j+1}). \end{aligned} \quad (5.179)$$

The asymptotic behavior of  $P_{F_b}(\rho_i | \rho_j)$  is thus determined by (5.104) for large  $(i - j)$  and  $j > 1$ . For  $j = 1$ , the asymptotic behavior of  $P_{F_b}(\rho_i | \rho_j, N_b^{(-1)}) = P_{F_b}(\rho_i | N_b^{(-1)})$  is given by

$$P_{F_b}(\rho_i | N_b^{(-1)}) = \begin{cases} l_{\text{gcd}}^* / \bar{l} \pm \mathcal{O}(\theta^i), & \text{if } i \equiv 1 \pmod{l_{\text{gcd}}^*}, \\ 0, & \text{otherwise,} \end{cases} \quad (5.180)$$

where  $\theta$  is given in Lemma 5.59. □

**Proof.** Eq. (5.178) follows from (5.48). Note that (5.178) does not hold necessarily for  $j = 1$  because the probability  $P_{F_b}(u_{j:i-1}|\rho_j)$  for  $j = 1$  involves the distribution of the first symbol, which is affected by the value of  $N_b^{(-1)}$  in (5.48). For  $j > 1$ , the first symbol is never involved. Eq. (5.179) follows trivially from (5.178). At last, for  $j = 1$ , (5.180) follows from (5.179) and Lemma 5.59. Indeed, for  $i > l_{\max}^*$  and  $i \equiv 1 \pmod{l_{\gcd}^*}$ , we have  $P_{F_b}(\rho_i|N_b^{(-1)}) = \sum_{s \in \mathcal{A}} P_{F_b}(S_1 = s|N_b^{(-1)})P_{F_b}(\rho_i|\rho_{1+l(s)})$ , where  $P_{F_b}(\rho_i|\rho_{1+l(s)}) = P(\rho_i|\rho_{1+l(s)}) = l_{\gcd}^*/\bar{l} \pm \mathcal{O}(\theta^{i-l(s)}) = l_{\gcd}^*/\bar{l} \pm \mathcal{O}(\theta^i)$ . ■

Though (5.104) and (5.180) are equal, note that  $P_{F_b}(\rho_i|N_b^{(-1)})$ ,  $P_{F_b}(\rho_i)$  and  $P(\rho_i)$  are not necessarily equal, because of the residual term  $\mathcal{O}(\theta^i)$ .

We can now prove Property 5.28 and Property 5.29 based on Property 5.27 and Lemma 5.62.

**Proof of Property 5.28.** The probability  $P_{F_b}(n_b|N_b^{(-1)})$  can be determined approximately for large  $N$  as follows. It is the joint probability of having  $\rho_{1+n_b}$  and having the symbol  $S_{N_s+1}$  longer than  $N - n_b$ , which are independent events. Therefore

$$P_{F_b}(n_b|N_b^{(-1)}) = P_{F_b}(\rho_{1+n_b}|N_b^{(-1)}) P(l(S) > N - n_b) \times \mathbb{I}\{N - l_{\max}^* < n_b \leq N\}. \quad (5.181)$$

For large  $N$ , it follows from Lemma 5.62 that

$$P_{F_b}(\rho_{1+n_b}|N_b^{(-1)}) = \left( \frac{l_{\gcd}^*}{\bar{l}} \pm \mathcal{O}(\theta^{n_b}) \right) \mathbb{I}\{n_b \equiv 0 \pmod{l_{\gcd}^*}\},$$

which proves the property. ■

**Proof of Property 5.29.** To calculate approximately

$$P_{F_b}(B_{\text{vlc}} = s) = \sum_{n_b^{(-1)}} P_{F_b}(B_{\text{vlc}} = s|n_b^{(-1)})P_{F_b}(n_b^{(-1)})$$

for large  $N$ , we can use Property 5.28 for  $P_{F_b}(n_b^{(-1)})$ . After simplifications, this leads to (5.51). ■

**Proof of Property 5.30.** Let us prove (5.52). By using the approximate expression of  $P_{E_b}(n_b)$  from Property 5.28,

$$\bar{N}_b \approx N_b^{\max} - \sum_{n_b \text{ s.t. (5.47)}} (N_b^{\max} - n_b) \frac{l_{\text{gcd}}^*}{\bar{l}} P(l(S) > N_b^{\max} - n_b) \quad (5.182)$$

$$= N_b^{\max} - \frac{l_{\text{gcd}}^*}{\bar{l}} \sum_{\substack{l \geq 0 \\ l \equiv 0 \pmod{l_{\text{gcd}}^*}} \substack{l_{\text{max}} - l_{\text{gcd}}^*}} l P(l(S) > l) \quad (5.183)$$

$$= N_b^{\max} - \frac{l_{\text{gcd}}^*}{\bar{l}} \sum_{s \in \mathcal{A}} P(s) f(s), \quad (5.184)$$

where  $f(s)$  is given by

$$\begin{aligned} f(s) &= 0 + l_{\text{gcd}}^* + 2l_{\text{gcd}}^* + 3l_{\text{gcd}}^* + \cdots + (l(s) - l_{\text{gcd}}^*) \\ &= \frac{l(s)^2 - l_{\text{gcd}}^* l(s)}{2l_{\text{gcd}}^*}. \end{aligned}$$

After simplifications, (5.52) follows.

Eq. (5.53) follows almost exactly from renewal theory [112, Th. 3.3.5]. The proof is here reproduced for the sake of completeness, with only a few modifications to handle the minor differences. Eq. (5.53) is equivalent to

$$P\left(\frac{N_s - \bar{N}_b/\bar{l}}{\sigma_l \sqrt{\bar{N}_b/\bar{l}^3}} < y\right) \rightarrow \frac{1}{2\pi} \int_{-\infty}^y e^{-x^2/2} dx \quad (5.185)$$

as  $N \rightarrow +\infty$ . Let

$$r_N = \bar{N}_b/\bar{l} + y \sigma_l \sqrt{\bar{N}_b/\bar{l}^3}.$$

Then

$$P\left(\frac{N_s - \bar{N}_b/\bar{l}}{\sigma_l \sqrt{\bar{N}_b/\bar{l}^3}} < y\right) = P(N_s < r_N) = P(N_s < \lceil r_N \rceil). \quad (5.186)$$

Note that by (5.45),

$$N_s < m \iff l(S_{1:m}) > N. \quad (5.187)$$

Therefore

$$\begin{aligned} P(N_s < \lceil r_N \rceil) &= P(l(S_{1:\lceil r_N \rceil}) > N) \\ &= P\left(\frac{l(S_{1:\lceil r_N \rceil}) - \lceil r_N \rceil \bar{l}}{\sigma_l \sqrt{\lceil r_N \rceil}} > \frac{N - \lceil r_N \rceil \bar{l}}{\sigma_l \sqrt{\lceil r_N \rceil}}\right). \end{aligned} \quad (5.188)$$

Firstly, by the central limit theorem,

$$\frac{l(S_{1:\lceil r_N \rceil}) - \lceil r_N \rceil \bar{l}}{\sigma_l \sqrt{\lceil r_N \rceil}}$$

converges to a Gaussian random variable  $\mathcal{N}(0, 1)$  as  $N$  (and thus  $\bar{N}_b$  and  $\lceil r_N \rceil$ ) approaches  $+\infty$ . Secondly, let

$$z_N = \frac{N - \lceil r_N \rceil \bar{l}}{\sigma_l \sqrt{\lceil r_N \rceil}}$$

and let us assume  $z_N \rightarrow -y$  as  $N \rightarrow +\infty$ . Then

$$P\left(\frac{N_s - \bar{N}_b/\bar{l}}{\sigma_l \sqrt{\bar{N}_b/\bar{l}^3}} < y\right) \rightarrow \frac{1}{2\pi} \int_{-y}^{\infty} e^{-x^2/2} dx, \quad (5.189)$$

which proves (5.185).

The convergence of  $z_N \rightarrow -y$  as  $N \rightarrow +\infty$  remains to be proved. This can be trivially shown by lower and upper bounding  $z_N$  with bounds that converge toward  $-y$  as  $N \rightarrow +\infty$ . Here is a lower bound:

$$z_N \geq \frac{N - (r_N + 1) \bar{l}}{\sigma_l \sqrt{r_N + 1}} \quad (5.190)$$

$$= \frac{N - \bar{N}_b - y \sigma_l \bar{l} \sqrt{\bar{N}_b/\bar{l}^3} - \bar{l}}{\sigma_l \sqrt{\bar{N}_b/\bar{l} + y \sigma_l \sqrt{\bar{N}_b/\bar{l}^3} + 1}} \quad (5.191)$$

$$\geq \frac{-y \sqrt{\bar{N}_b/\bar{l}} - \bar{l}/\sigma_l}{\sqrt{\bar{N}_b/\bar{l} + y \sigma_l \sqrt{\bar{N}_b/\bar{l}^3} + 1}}, \quad (5.192)$$

which converges to  $-y$  as  $N$  (and thus  $\bar{N}_b$ ) approaches  $+\infty$ . Note the second inequality follows from  $N - \bar{N}_b \geq 0$ . Here is an upper bound:

$$z_N \leq \frac{N - r_N \bar{l}}{\sigma_l \sqrt{r_N}} \quad (5.193)$$

$$= \frac{N - \bar{N}_b - y \sigma_l \bar{l} \sqrt{\bar{N}_b/\bar{l}^3}}{\sigma_l \sqrt{\bar{N}_b/\bar{l} + y \sigma_l \sqrt{\bar{N}_b/\bar{l}^3}}} \quad (5.194)$$

$$\leq \frac{-y \sqrt{\bar{N}_b/\bar{l}} + l_{\max}^*/\sigma_l}{\sqrt{\bar{N}_b/\bar{l} + y \sigma_l \sqrt{\bar{N}_b/\bar{l}^3}}}, \quad (5.195)$$

which converges to  $-y$  as  $N \rightarrow +\infty$ . Note the second inequality follows from  $N - \overline{N}_b \leq l_{\max}^*$ . ■

### 5.C.3 Proof of Theorem 5.31

A few difficulties characterize the proof of Theorem 5.31 proposed hereafter, compared to the proof of Theorem 5.26. To circumvent these difficulties in the first place, we depict a rather intuitive sketch of proof in Section 5.C.3.1. Then, a few lemmas are given in Section 5.C.3.2 and a full proof in Section 5.C.3.3.

#### 5.C.3.1 Introductory sketch of proof of Theorem 5.31

Let us consider a simplified case of Theorem 5.31 by considering that  $l_{\text{gcd}}^* = 1$  and that  $N_b$  is fixed to  $n_b$ . This simplified case can be proved similarly as Theorem 5.26 if we replace  $N_s$  by  $N_b$ . Let us just highlight informally the major differences.

In (5.168), the two factors

$$P_{F_s}(B_{\text{vlc}} = s) A_{s_L, h | s, s_L, h, j, l}^{F_s}$$

have trivial equivalents with bit sequences, namely

$$P_{F_b}(B_{\text{vlc}} = u_{1:n_b}) A_{s_L, h | u_{1:n_b}, s_L, h, j, l}^{F_b}$$

The main difference with Theorem 5.26 lies in the fact that to have  $n$  error events starting in  $j_{1:n}$ , the sequence  $u_{1:n_b}$  must necessarily have a codeword starting with each  $u_{j_m}$ , i.e., we must have  $\rho_{j_m}$  for  $m = 1, \dots, n$ , or more simply  $\rho_j$ , otherwise  $A_{s_L, h | u_{1:n_b}, s_L, h, j, l}^{F_b} = 0$ . Therefore

$$\begin{aligned} P_{F_b}(B_{\text{vlc}} = u_{1:n_b}) A_{s_L, h | u_{1:n_b}, s_L, h, j, l}^{F_b} \\ = P_{F_b}(B_{\text{vlc}} = u_{1:n_b}, \rho_j) A_{s_L, h | u_{1:n_b}, s_L, h, j, l}^{F_b} \end{aligned} \quad (5.196)$$

By factorization,

$$P_{F_b}(B_{\text{vlc}} = u_{1:n_b}, \rho_j) = P_{F_b}(B_{\text{vlc}} = u_{1:n_b} | u_{\mathcal{Y}}, \rho_j) P_{F_b}(\rho_j, u_{\mathcal{Y}}) \quad (5.197)$$

where  $P_{F_b}(B_{\text{vlc}} = u_{1:n_b} | u_{\mathcal{Y}}, \rho_j)$  corresponds roughly to  $P_{F_s}(B_{\text{vlc}} = s | s_{\mathcal{Y}})$  in (5.170) and  $P_{F_b}(\rho_j, u_{\mathcal{Y}})$  to  $P_{F_s}(s_{\mathcal{Y}})$ .

The next major difference with the proof of Theorem 5.26 is the factorization of  $P_{F_b}(\rho_j, u_{\mathcal{Y}})$ . From (5.170) to (5.171), we used the factorization

$$P_{F_s}(s_{\mathcal{Y}}) = \prod_{m=1}^n P_{F_s}(s_{\mathcal{Y}_m}), \quad (5.198)$$

which is possible since the symbols  $S_m$  are independent. Such a factorization is however not possible with  $P_{F_b}(\rho_j, u_{\mathcal{Y}})$  due to the dependence between the pairs  $(\rho_{j_m}, u_{\mathcal{Y}_m})$  for different values of  $m$ . Instead,

$$\begin{aligned} P_{F_b}(\rho_j, u_{\mathcal{Y}}) &= P_{F_b}(\rho_{j_1}) P_{F_b}(u_{\mathcal{Y}_1} | \rho_{j_1}) \\ &\times \prod_{m=2}^n \underbrace{P_{F_b}(\rho_{j_m} | \rho_{j_{m-1}}, u_{\mathcal{Y}_{m-1}})}_{=P_{F_b}(\rho_{j_m} | \rho_{j_{m-1}+l_{m-1}})} P_{F_b}(u_{\mathcal{Y}_m} | \rho_{j_m}). \end{aligned} \quad (5.199)$$

Still, if the positions  $j$  are sufficiently far away from each other, the factorization is possible because  $P_{F_b}(\rho_{j_m} | \rho_{j_{m-1}+l_{m-1}}) \approx P_{F_b}(\rho_{j_m})$ , see second case below.

Let  $l_{\text{sum}} = \sum_{m=1}^n l_m$ , let  $l_{\text{sum}}^{\text{th}}$  be some threshold, let

$$\mathcal{J}_{1:n}^{\Delta} = \{j_{1:n} : j_1 \geq 1+\Delta, j_n \leq n_b - l_n - \Delta + 1, j_m \geq j_{m-1} + l_{m-1} + \Delta \text{ for } 2 \leq m \leq n\} \quad (5.200)$$

be an extension of (5.166), where  $\Delta$  characterizes the distance between the error events, and lastly let  $\mathcal{J}_{1:n} = \mathcal{J}_{1:n}^{\Delta=0}$ . We can focus on small values of  $n$  in the following because: (i)  $h$  is assumed small in Theorem 5.31; (ii)  $h_m \geq d_f^{\text{vlc}}$  by Definition 5.12 of  $d_f^{\text{vlc}}$  and (iii)  $h = \sum_{m=1}^n h_m$ , hence  $h \geq n d_f^{\text{vlc}} \geq n$ . Let us now consider three cases.

*First case:*  $l_{\text{sum}} \geq l_{\text{sum}}^{\text{th}}$ . Since the VLC spectrum is bounded by assumption, the average of the quantity  $A_{s_L, h | u_{1:n_b}, s_L, j, j, l}^{F_b}$  over all  $u_{1:n_b}$  in (5.196) is also bounded — recall it is the average of a product of  $n$  VLC conditional spectra in (5.169). Therefore, for fixed  $n$ , there exists some threshold  $l_{\text{sum}}^{\text{th}}$  on  $l_{\text{sum}}$  above which the average of  $A_{s_L, h | u_{1:n_b}, s_L, j, j, l}^{F_b}$  is negligible. Consequently, the first case has a negligible impact on the spectrum of the VLC block and we can focus on the other cases.

*Second case:*  $l_{\text{sum}} < l_{\text{sum}}^{\text{th}}$  and  $j_{1:n} \in \mathcal{J}_{1:n}^{\Delta=\log(n_b)}$ . Roughly speaking, if the positions  $j$  are sufficiently far away from each other,  $\rho_{j_m}$  is approximately independent of  $\rho_{j_{m-1}+l_{m-1}}$  and  $P_{F_b}(\rho_{j_m} | \rho_{j_{m-1}+l_{m-1}}) \approx P_{F_b}(\rho_{j_m}) \approx 1/\bar{I}$ . Formally

speaking, for  $j_{1:n} \in \mathcal{J}_{l_{1:n}}^{\Delta=\log(n_b)}$ , it follows from (5.103) and Lemma 5.62 that the factors  $P_{F_b}(\rho_{j_1})$  and  $P_{F_b}(\rho_{j_m}|\rho_{j_{m-1}+l_{m-1}})$  in (5.199) approach  $1/\bar{l}$  as  $n_b \rightarrow +\infty$ . Therefore we have the factorization

$$P_{F_b}(\rho_{j_{1:n}}, u_{y_{1:n}}) \rightarrow \left(\frac{1}{\bar{l}}\right)^n \prod_{m=1}^n P_{F_b}(u_{y_m}|\rho_{j_m}) \quad (5.201)$$

for asymptotically large  $n_b$  when  $j_{1:n} \in \mathcal{J}_{l_{1:n}}^{\Delta=\log(n_b)}$ . We can thus finish the sketch of proof for this second case as the proof of Theorem 5.26. Note simply that  $P_{F_b}(u_{y_m}|\rho_{j_m})$  corresponds to  $P_{F_s}(s_{y_m})$  in (5.171) and that  $(1/\bar{l})^n$  is an additional factor (compared to Theorem 5.26) which we find back in (5.54).

*Third case:*  $l_{\text{sum}} < l_{\text{sum}}^{\text{th}}$  and  $j_{1:n} \in \mathcal{J}_{l_{1:n}} \setminus \mathcal{J}_{l_{1:n}}^{\Delta=\log(n_b)}$ . This case has a negligible impact on the spectrum because the number of involved positions is negligible, compared to the second case. Indeed, for fixed  $l_{\text{sum}}^{\text{th}}$  and fixed  $n$ , it is straightforward to show that  $|\mathcal{J}_{l_{1:n}}^{\Delta=\log(n_b)}| \rightarrow |\mathcal{J}_{l_{1:n}}|$  as  $n_b \rightarrow +\infty$ .

To summarize, the main difference with the proof of Theorem 5.26 is the fact that to have  $n$  error events starting in positions  $j_{1:n}$ , the transmitted sequence  $u$  must necessarily have a codeword starting with each  $u_{j_m}$  for  $m = 1, \dots, n$ , which happens with probability  $(l_{\text{gcd}}^*/\bar{l})^n$  when the error events are sufficiently far away from each other (second case above). This explains the additional factor  $(l_{\text{gcd}}^*/\bar{l})^n$  in Theorem 5.31 compared to Theorem 5.26.

### 5.C.3.2 Lemmas

The following lemmas help to tackle the proof of Theorem 5.31.

**Lemma 5.63.** *Given the set  $\mathcal{D}_{n,N}$  and the variable  $K_{n,N}$ ,*

$$\mathcal{D}_{n,N} \triangleq \left\{ (d_1, \dots, d_{n+1}) \in \mathbb{N}_{\geq 0}^{n+1} : \sum_{m=1}^{n+1} d_m = N \right\}, \quad (5.202)$$

$$K_{n,N} \triangleq \frac{1}{|\mathcal{D}_{n,N}|} \sum_{d \in \mathcal{D}_{n,N}} \prod_{m=1}^{n+1} P(\rho_{1+d_m l_{\text{gcd}}^*}), \quad (5.203)$$

where  $|\mathcal{D}_{n,N}| = \binom{N+n}{n}$ , we have for fixed  $n$

$$K_{n,N} \leq 1, \quad (5.204)$$

$$K_{n,N} = \left(\frac{l_{\text{gcd}}^*}{\bar{l}}\right)^{n+1} \pm \mathcal{O}\left(\frac{n^2 \log^2(N)}{N}\right), \quad (5.205)$$

The same result holds if we replace some  $P(\rho_{1+d_m l_{\text{gcd}}^*})$  with  $P_{F_b}(\rho_{1+d_m l_{\text{gcd}}^*})$ .  $\square$

It is certainly possible to develop a tighter expression of  $K_{n,N}$ , i.e., to develop an expression with a residue smaller than  $\mathcal{O}(n^2 \log^2(N)/N)$ . This is however beyond the scope of this study which focuses on asymptotically large values of  $N$ .

**Proof.** Given the definition of  $\mathcal{D}_{n,N}$ , its cardinality is clearly equal to the number of ways we can arrange  $n$  red balls among  $N$  black balls, which is  $\binom{N+n}{n}$  — let  $d_1$  be the number of black balls before the first red ball,  $d_2$  the number of black balls between the first and the second red balls, etc.

Eq. (5.204) follows trivially from  $P(\rho_{1+d_m l_{\text{gcd}}^*}) \leq 1$ .

To prove (5.205), let us define a subset of  $\mathcal{D}_{n,N}$  whose elements  $d_{1:n+1}$  are larger than  $\Delta = \lfloor \log^2(N) \rfloor$ :

$$\mathcal{D}_{n,N}^\Delta \triangleq \{d_{1:n+1} \in \mathcal{D}_{n,N} : d_m \geq \Delta \text{ for } 1 \leq m \leq n+1\}. \quad (5.206)$$

Then,

$$K_{n,N} = \frac{1}{|\mathcal{D}_{n,N}|} \left( \sum_{d: \in \mathcal{D}_{n,N}^\Delta} \prod_{m=1}^{n+1} P(\rho_{1+d_m l_{\text{gcd}}^*}) + \sum_{d: \in \mathcal{D}_{n,N} \setminus \mathcal{D}_{n,N}^\Delta} \prod_{m=1}^{n+1} P(\rho_{1+d_m l_{\text{gcd}}^*}) \right). \quad (5.207)$$

In the first summation, the elements  $d:$  are all larger than  $\Delta$ , thus  $P(\rho_{1+d_m l_{\text{gcd}}^*}) = l_{\text{gcd}}^* / \bar{l} \pm \mathcal{O}(|\lambda_2|^\Delta)$  by Lemma 5.59. In the second summation, let us upper bound all factors  $P(\rho_{1+d_m l_{\text{gcd}}^*})$  by 1. Then,

$$K_{n,N} = \frac{|\mathcal{D}_{n,N}^\Delta|}{|\mathcal{D}_{n,N}|} \left( l_{\text{gcd}}^* / \bar{l} \pm \mathcal{O}(|\lambda_2|^\Delta) \right)^{n+1} + \mathcal{O} \left( 1 - \frac{|\mathcal{D}_{n,N}^\Delta|}{|\mathcal{D}_{n,N}|} \right). \quad (5.208)$$

Straightforwardly, the cardinality of the set  $\mathcal{D}_{n,N}^\Delta$  is given by  $|\mathcal{D}_{n,N}^\Delta| = \binom{N-(n+1)\Delta+n}{n}$ . Besides, for fixed  $n$  and  $\Delta = \lfloor \log^2(N) \rfloor$ ,

$$\begin{aligned} 1 &\geq \frac{|\mathcal{D}_{n,N}^\Delta|}{|\mathcal{D}_{n,N}|} = \frac{N!}{(N+n)!} \frac{(N-(n+1)\Delta+n)!}{(N-(n+1)\Delta)!} \\ &\geq \frac{(N-(n+1)\Delta)^n}{(N+n)^n} \\ &= 1 - \mathcal{O} \left( \frac{n^2 \Delta}{N+n} \right) \end{aligned}$$

Substituting this into (5.208) proves (5.205).

At last, note these developments hold also if some of the factors  $P(\rho_{1+d_m l_{\text{gcd}}^*})$  in (5.203) are replaced by  $P_{F_b}(\rho_{1+d_m l_{\text{gcd}}^*})$ , by invoking Lemma 5.62 here above instead of Lemma 5.59. ■

**Lemma 5.64.** *If the VLC spectrum  $A_{h,l_b}^{\text{vlc}}$  is bounded (Definition 5.13), then the spectrum  $T_{h,l_b,n}^{\text{vlc}}$  defined in Theorem 5.31 is also bounded, i.e.,  $\exists c < 1, \forall h \in \mathbb{N}_{>0}, T_{h,l_b,n}^{\text{vlc}} = \mathcal{O}(c^{l_b})$ .* □

**Proof.** Let us consider a fixed value of  $h$ . By definition,

$$T_{h,l_b,n}^{\text{vlc}} = \sum_{\substack{h_1, h_2, \dots, h_n \geq 1 \\ \sum_m h_m = h}} \sum_{\substack{l_1, l_2, \dots, l_n \geq 1 \\ \sum_m l_m = l_b}} \prod_{m=1}^n A_{h=h_m, l_b=l_m}^{\text{vlc}}. \quad (5.209)$$

Since  $A_{h=h_m, l_b=l_m}^{\text{vlc}}$  is bounded,  $\exists c' < 1, \forall h_m, \exists l'_{h_m}, \forall l_m \geq l'_{h_m}, A_{h=h_m, l_b=l_m}^{\text{vlc}} \leq c'^{l_m}$ . Let  $l'_h = \max_{h_m \leq h} \{l'_{h_m}\}$  and  $K_h = \max_{h_m \leq h, l_m \leq l'_h} \{1, A_{h=h_m, l_b=l_m}^{\text{vlc}} / c'^{l_m}\}$ . Then, for  $h_m \leq h$ , we have  $A_{h=h_m, l_b=l_m}^{\text{vlc}} \leq K_h c'^{l_m}$  and

$$\prod_{m=1}^n A_{h=h_m, l_b=l_m}^{\text{vlc}} \leq K_h^n c'^{l_b}. \quad (5.210)$$

Since the first summation in (5.209) involves at most  $\binom{h-1}{n-1}$  terms and the second summation at most  $\binom{l_b-1}{n-1}$  terms,

$$T_{h,l_b,n}^{\text{vlc}} \leq \binom{h-1}{n-1} \binom{l_b-1}{n-1} K_h^n c'^{l_b} \quad (5.211)$$

$$\leq (h-1)^{n-1} (l_b-1)^{n-1} K_h^n c'^{l_b} \quad (5.212)$$

where we used  $\binom{n}{k} \leq n^k$ . At last, since  $h = \sum_{m=1}^n h_m \geq n d_f^{\text{vlc}} \geq n$  by Definition 5.12 of  $d_f^{\text{vlc}}$ ,

$$T_{h,l_b,n}^{\text{vlc}} \leq (h-1)^{h-1} (l_b-1)^{h-1} K_h^h c'^{l_b}. \quad (5.213)$$

Consequently, for any fixed  $h$ ,  $T_{h,l_b,n}^{\text{vlc}} = \mathcal{O}(c^{l_b})$  for any  $c$  subject to  $c' < c < 1$ , which proves the lemma. ■

### 5.C.3.3 Proof of Theorem 5.31

We address now the proof of Theorem 5.31. Recall that an intuitive but less rigorous proof of Theorem 5.31 is given in Appendix 5.C.3.1.

**Proof of Theorem 5.31.** The spectrum (5.37) can be written with bit sequences  $u$ : instead of symbol sequences  $s$ :

$$A_{s_L, h}^{F_b} = \sum_{n_b} \sum_{u_{1:n_b} \in \mathcal{Y}^+} P_{F_b}(B_{\text{vlc}} = u_{1:n_b}) A_{s_L, h | u_{1:n_b}}^{F_b}. \quad (5.214)$$

The second summation enumerates in (5.214) all possible  $u_{1:n_b}$  and the conditional spectrum  $A_{s_L, h | u_{1:n_b}}^{F_b}$  involves all possible  $\check{u}_{1:n_b}$  given  $u_{1:n_b}$ . Let us rearrange them such that the joint enumeration of  $u_{1:n_b}$  and  $\check{u}_{1:n_b}$  is sorted by the number  $n$  of error events,  $n = \mathcal{E}(\check{u}_{1:n_b}, u_{1:n_b})$ , by the bit lengths  $l_{1:n}$  of these events, by their starting bit positions  $j_{1:n}$ , by their Levenshtein symbol distances  $s_{L,1:n}$  and by their Hamming distances  $h_{1:n}$ . If we restrict the enumeration to realizations  $u_{1:n_b}$  of non-zero probability (which are the only realizations that contribute to the spectrum in (5.214)), these quantities are constrained as follows. For  $l_{1:n}$ , let  $\mathcal{L}_{n, n_b}$  denote the set of possible lengths,

$$\mathcal{L}_{n, n_b} = \{l_{1:n} : \forall m, l_m \equiv 0 \pmod{l_{\text{gcd}}^*}, \sum_m l_m \leq n_b\}. \quad (5.215)$$

For  $j_{1:n}$ , let  $\mathcal{J}_{l_{1:n}, n_b}$  be the set of possible positions,

$$\mathcal{J}_{l_{1:n}, n_b} = \left\{ j_{1:n} : \begin{array}{l} j_1 \geq 1, j_n \leq n_b - l_n + 1, \\ j_{m+1} \geq j_m + l_m \text{ for } 1 \leq m \leq n-1, \\ j_m \equiv 1 \pmod{l_{\text{gcd}}^*} \forall m \end{array} \right\}. \quad (5.216)$$

where the constraint  $j_m \equiv 1 \pmod{l_{\text{gcd}}^*}$  follows from Remark 5.8, i.e., symbol codewords, hence error events, can start only at bit positions satisfying this constraint. For  $h_{1:n}$ ,  $\sum_m h_m = h = d_H(u_{1:n_b}, \check{u}_{1:n_b})$ . For  $s_{L,1:n}$ , unfortunately, we know from Proposition 5.19 that the link between  $s_{L,1:n}$  and  $s_L = d_{S_L}(u_{1:n_b}, \check{u}_{1:n_b})$  is not unequivocal due to the inequality  $\sum_{m=1}^n s_{L,m} \geq s_L$ . Nevertheless, we make the approximation in the following that it is an equality. This approximation is supported by two facts: (i) this inequality is an equality in most cases; (ii) because of the sense of this inequality, replacing it with an equality will result eventually in a spectrum that counts more Levenshtein symbol errors than the correct spectrum ( $\sum_{m=1}^n s_{L,m}$  instead of  $s_L$ ), which is compatible with our interest in upper performance bounds. So, eventually, we have the following constraints on  $h_{1:n}$  and  $s_{L,1:n}$ :

$$\sum_{m=1}^n h_m = h, \quad \text{and} \quad \sum_{m=1}^n s_{L,m} = s_L. \quad (5.217)$$

Since  $h_m \geq d_f^{\text{vlc}}$ , it follows immediately that

$$h \geq n d_f^{\text{vlc}}. \quad (5.218)$$

By rearranging (5.214) as mentioned, the spectrum can be written approximately (the approximation is due to (5.217)) as

$$A_{s_L, h}^{F_b} \approx \sum_{n_b} \sum_{u_{1:n_b} \in \mathcal{V}^+} \sum_{n \geq 1} \sum_{l: \in \mathcal{L}_{n, n_b}} \sum_{j: \in \mathcal{J}_{l, n_b}} \sum_{\substack{s_{L, m}: h: \\ \text{s.t. (5.217)}}} A_{s_L, h | u_{1:n_b}, s_{L, m}, h: j: l}^{F_b} P_{F_b}(n_b, \rho_j, B_{\text{vlc}} = u_{1:n_b}) \quad (5.219)$$

where  $A_{s_L, h | u_{1:n_b}, s_{L, m}, h: j: l}^{F_b}$  counts the number of sequences that form  $n$  error events  $e_1, e_2, \dots, e_n$  with  $u_{1:n_b}$  such that  $e_m$  starts with  $u_{j_m}$ ,  $l_m = l(e_m)$ ,  $h_m = d_H(e_m)$  and  $s_{L, m} = d_{S_L}(e_m)$ . Such sequences might not exist, so this number might be zero. It is equal to the product of the conditional spectra (5.24) of the sub-sequences  $u_{j_m: j_m + l_m - 1}$  associated with each  $e_m$ :

$$A_{s_L, h | u_{1:n_b}, s_{L, 1: n}, h_{1: n}, j_{1: n}, l_{1: n}}^{F_b} = \prod_{m=1}^n A_{s_L = s_{L, m}, h = h_m | u_{\mathcal{Y}_m}}^{\text{vlc}} \quad (5.220)$$

where we use the notations  $\mathcal{Y}_m = j_m : j_m + l_m - 1$  and  $u_{\mathcal{Y}_m} = u_{j_m: j_m + l_m - 1}$ . Note that  $\rho_j$  in (5.219) (see Definition 5.58) accounts for the fact that to have error events starting in positions  $j$ , the transmitted sequence  $u_{1:n_b}$  must necessarily have codewords starting at those positions. With  $u_{\mathcal{Y}} = (u_{\mathcal{Y}_1}, u_{\mathcal{Y}_2}, \dots, u_{\mathcal{Y}_n})$ , the expression (5.219) can be written as

$$A_{s_L, h}^{F_b} \approx \sum_{n_b} \sum_{u_{1:n_b} \in \mathcal{V}^+} \sum_{n \geq 1} \sum_{l: \in \mathcal{L}_{n, n_b}} \sum_{j: \in \mathcal{J}_{l, n_b}} \sum_{\substack{s_{L, m}: h: \\ \text{s.t. (5.217)}}} P_{F_b}(\rho_j, u_{\mathcal{Y}}) P_{F_b}(n_b | \rho_j, u_{\mathcal{Y}}) \times \left( \prod_{m=1}^n A_{s_L = s_{L, m}, h = h_m | u_{\mathcal{Y}_m}}^{\text{vlc}} \right) \times P_{F_b}(B_{\text{vlc}} = u_{1:n_b} | \rho_j, n_b, u_{\mathcal{Y}}) \quad (5.221)$$

$$= \sum_{n_b} \sum_{n \geq 1} \sum_{l: \in \mathcal{L}_{n, n_b}} \sum_{j: \in \mathcal{J}_{l, n_b}} \sum_{\substack{s_{L, m}: h: \\ \text{s.t. (5.217)}}} \sum_{u_{\mathcal{Y}}} P_{F_b}(\rho_j, u_{\mathcal{Y}}) P_{F_b}(n_b | \rho_j, u_{\mathcal{Y}}) \times \left( \prod_{m=1}^n A_{s_L = s_{L, m}, h = h_m | u_{\mathcal{Y}_m}}^{\text{vlc}} \right) \times \sum_{u'_{1:n_b} \in \mathcal{V}^+} P_{F_b}(B_{\text{vlc}} = u'_{1:n_b} | \rho_j, n_b, u_{\mathcal{Y}}). \quad (5.222)$$

We have immediately the following simplifications:

$$\sum_{u'_{1:n_b} \in \mathcal{V}^+} P_{F_b} \left( B_{\text{vlc}} = u'_{1:n_b} | \rho_{j_1}, n_b, u_{\mathcal{Y}} \right) = 1 \quad (5.223)$$

and

$$\begin{aligned} P_{F_b}(n_b | \rho_{j_1}, u_{\mathcal{Y}}) &= P_{F_b}(n_b | \rho_{j_n}, u_{\mathcal{Y}_n}) = P_{F_b}(n_b | \rho_{j_n + l_n}) \\ &= P_{F_b}(n_b | \rho_{n_b+1}) P_{F_b}(\rho_{n_b+1} | \rho_{j_n + l_n}) \end{aligned} \quad (5.224)$$

and

$$\begin{aligned} P_{F_b}(\rho_{j_1}, u_{\mathcal{Y}}) &= P_{F_b}(\rho_{j_1}) P_{F_b}(u_{\mathcal{Y}_1} | \rho_{j_1}) \\ &\times \prod_{m=2}^n \underbrace{P_{F_b}(\rho_{j_m} | \rho_{j_{m-1}}, u_{\mathcal{Y}_{m-1}})}_{= P_{F_b}(\rho_{j_m} | \rho_{j_{m-1} + l_{m-1}})} P_{F_b}(u_{\mathcal{Y}_m} | \rho_{j_m}) \end{aligned} \quad (5.225)$$

where  $P_{F_b}(\rho_{n_b+1} | \rho_{j_n + l_n})$  and  $P_{F_b}(\rho_{j_m} | \rho_{j_{m-1} + l_{m-1}})$  can be simplified as

$$P_{F_b}(\rho_{j_m} | \rho_{j_{m-1} + l_{m-1}}) = P(\rho_{1+j_m - j_{m-1} - l_{m-1}}), \quad (5.226)$$

by Lemma 5.62. Furthermore, it follows from Lemma 5.62 that we have  $P_{F_b}(u_{\mathcal{Y}_m} | \rho_{j_m}) = P(u_{\mathcal{Y}_m} | \rho_{j_m})$  when  $j_m > 1$ , which enables the simplification

$$\begin{aligned} \sum_{u_{\mathcal{Y}_m}} P_{F_b}(u_{\mathcal{Y}_m} | \rho_{j_m}) A_{s_L=s_{L,m}, h=h_m | u_{\mathcal{Y}_m}}^{\text{vlc}} \\ = \sum_{u_{1:l_m}} P(u_{1:l_m}) A_{s_L=s_{L,m}, h=h_m | u_{1:l_m}}^{\text{vlc}} \end{aligned} \quad (5.227)$$

$$= A_{s_L=s_{L,m}, h=h_m, l_b=l_m}^{\text{vlc}}. \quad (5.228)$$

Since we have always  $j_m > 1$  for  $2 \leq m \leq n$  by (5.216), this simplification always holds for  $2 \leq m \leq n$ . For  $m = 1$ , however, it holds only when  $j_m = j_1 > 1$  but we make the approximation that it holds also when  $j_1 = 1$ ; this approximation will have only a negligible impact in the following because the number of  $j_1 \in \mathcal{J}_{l, n_b}$  subject to  $j_1 = 1$  is negligible, as explained hereafter.

Eventually, substituting all simplifications and approximations,

$$\begin{aligned}
 A_{s_L, h}^{F_b} &\approx \sum_{n_b} P_{F_b}(n_b | \rho_{n_b+1}) \sum_{n \geq 1} \sum_{l: \mathcal{L}_{n, n_b}} \sum_{\substack{s_L, h: \\ \text{s.t. (5.217)}}} \\
 &\left( \prod_{m=1}^n A_{s_L=s_{L,m}, h=h_m, l_b=l_m}^{\text{vlc}} \right) \sum_{j: \mathcal{J}_{l, n_b}} P_{F_b}(\rho_{j_1}) \\
 &\times P(\rho_{2+n_b-j_n-l_n}) \prod_{m=2}^n P(\rho_{1+j_m-j_{m-1}-l_{m-1}}). \quad (5.229)
 \end{aligned}$$

Let us simplify the summation over  $j: \in \mathcal{J}_{l, n_b}$ , by defining the new variables  $d_{1:n+1}$  as

$$d_m \triangleq \begin{cases} (j_1 - 1) / l_{\text{gcd}}^* & \text{if } m = 1, \\ (j_m - j_{m-1} - l_{m-1}) / l_{\text{gcd}}^* & \text{if } 2 \leq m \leq n, \\ (n_b + 1 - j_n - l_n) / l_{\text{gcd}}^* & \text{if } m = n + 1. \end{cases} \quad (5.230)$$

In words,  $d_1$  is the number of bits divided by  $l_{\text{gcd}}^*$  before the first error event,  $d_{2:n}$  between the error events, and  $d_{n+1}$  after the last error event. Given the set  $\mathcal{J}_{l, n_b}$  of possible  $j$ , the set of possible  $d_{1:n+1}$  is exactly  $\mathcal{D}_{n, (n_b - l_{\text{sum}}) / l_{\text{gcd}}^*}$  in Lemma 5.63, with  $l_{\text{sum}} = \sum_{m=1}^n l_m$ . Therefore, by invoking Lemma 5.63,

$$\begin{aligned}
 A_{s_L, h}^{F_b} &\approx \sum_{n_b} P_{F_b}(n_b | \rho_{n_b+1}) \sum_{n \geq 1} \sum_{l: \mathcal{L}_{n, n_b}} \sum_{\substack{s_L, h: \\ \text{s.t. (5.217)}}} \\
 &\left( \prod_{m=1}^n A_{s_L=s_{L,m}, h=h_m, l_b=l_m}^{\text{vlc}} \right) \\
 &\times \left( \binom{\frac{n_b - l_{\text{sum}}}{l_{\text{gcd}}^*} + n}{n} \right) \left( \left( \frac{l_{\text{gcd}}^*}{l} \right)^{n+1} \pm \mathcal{O} \left( \frac{n^2 \log^2(n_b - l_{\text{sum}})}{n_b - l_{\text{sum}}} \right) \right) \\
 &= \sum_{n_b} P_{F_b}(n_b | \rho_{n_b+1}) \sum_{n \geq 1} \sum_{l_{\text{sum}}=1}^{n_b} T_{s_L, h, l_b=l_{\text{sum}}, n}^{\text{vlc}} \\
 &\times \left( \binom{\frac{n_b - l_{\text{sum}}}{l_{\text{gcd}}^*} + n}{n} \right) \left( \left( \frac{l_{\text{gcd}}^*}{l} \right)^{n+1} \pm \mathcal{O} \left( \frac{n^2 \log^2(n_b - l_{\text{sum}})}{n_b - l_{\text{sum}}} \right) \right). \quad (5.232)
 \end{aligned}$$

where  $T_{s_L, h, l_b, n}^{\text{vlc}}$  is defined in Theorem 5.26.

To simplify this last expression further, we can capitalize on the largeness of  $N$ . For  $N$  large, the possible values of  $n_b$  are relatively close to  $N_b^{\text{max}} \approx N$

by Property 5.30. Because of that and if  $N$  is sufficiently large,  $T_{s_L, h, l_b = l_{\text{sum}}, n}^{\text{vlc}}$  becomes negligible when  $l_{\text{sum}} \rightarrow n_b$ , since  $T_{s_L, h, l_b, n}^{\text{vlc}}$  decreases exponentially with  $l_b$  by Lemma 5.64. This enables two simplifications: (i) the summation over  $l_{\text{sum}} \leq n_b$  can be replaced by a summation over  $l_{\text{sum}} \leq N_b^{\text{max}}$  without any impact; (ii) the term  $\mathcal{O}(n^2 \log^2(n_b - l_{\text{sum}})/(n_b - l_{\text{sum}}))$ , which becomes non-negligible when  $l_{\text{sum}} \rightarrow n_b$  but which is nevertheless always upper bounded by 1 by (5.204), is largely compensated by the exponential decrease of  $T_{s_L, h, l_b, n}^{\text{vlc}}$  when  $l_{\text{sum}} \rightarrow n_b$ . Also, note that for  $h$  small (as assumed in Theorem 5.31),  $n$  is small by (5.218) and therefore  $\binom{(n_b - l_{\text{sum}})/l_{\text{gcd}}^* + n}{n}$  is relatively close to  $\binom{(N_b^{\text{max}} - l_{\text{sum}})/l_{\text{gcd}}^* + n}{n}$  for  $N, n_b$  large. At last, recall that in order to legitimate (5.227) for  $j_1 = 1$ , we used the fact that the number of  $j: \in \mathcal{J}_{l, n_b}$  subject to  $j_1 = 1$  is negligible. This is indeed the case. Let  $\mathcal{J}_{l, n_b}^{j_1=1} \triangleq \{j: \in \mathcal{J}_{l, n_b} : j_1 = 1\}$ . As an extension of Lemma 5.63, we have trivially  $|\mathcal{J}_{l, n_b}^{j_1=1}| = |\mathcal{D}_{n-1, (n_b - l_{\text{sum}})/l_{\text{gcd}}^*}| = \binom{(n_b - l_{\text{sum}})/l_{\text{gcd}}^* + n - 1}{n - 1}$  and therefore

$$\frac{|\mathcal{J}_{l, n_b}^{j_1=1}|}{|\mathcal{J}_{l, n_b}|} = \frac{n}{\frac{n_b - l_{\text{sum}}}{l_{\text{gcd}}^*} + n} = \mathcal{O}\left(\frac{n}{n_b - l_{\text{sum}}}\right), \quad (5.233)$$

which can be neglected for the same reasons as the term  $\mathcal{O}(n^2 \log^2(n_b - l_{\text{sum}})/(n_b - l_{\text{sum}}))$  in (5.232). By combining all these approximations,

$$A_{s_L, h}^{F_b} \approx \sum_{n_b} P_{F_b}(n_b | \rho_{n_b+1}) \sum_{n \geq 1} \sum_{l_{\text{sum}}=1}^{N_b^{\text{max}}} \left(\frac{N_b^{\text{max}}}{l_{\text{gcd}}^*} - \frac{l_{\text{sum}}}{l_{\text{gcd}}^*} + n\right) \left(\frac{l_{\text{gcd}}^*}{l}\right)^{n+1} T_{s_L, h, l_b = l_{\text{sum}}, n}^{\text{vlc}}. \quad (5.234)$$

In this last expression, only the factor  $P_{F_b}(n_b | \rho_{n_b+1})$  depends on  $n_b$ . By (5.47),  $n_b$  can take values (of non-zero probability) only among the multiples of  $l_{\text{gcd}}^*$  subject to  $N - l_{\text{max}}^* < n_b \leq N$ . Given such a value of  $n_b$ , the factor  $P_{F_b}(n_b | \rho_{n_b+1})$  is the probability to have a symbol  $S$  that does not fit at the end of the current frame, i.e., whose length is strictly larger than  $N - n_b$ ,

$$P(l(S) > N - n_b) = P\left(\frac{l(S)}{l_{\text{gcd}}^*} > \left\lfloor \frac{N}{l_{\text{gcd}}^*} \right\rfloor - \frac{n_b}{l_{\text{gcd}}^*}\right), \quad (5.235)$$

where  $0 \leq \lfloor N/l_{\text{gcd}}^* \rfloor - n_b/l_{\text{gcd}}^* < l_{\text{max}}^*/l_{\text{gcd}}^*$  due to the restricted values of  $n_b$ . Therefore

$$\sum_{n_b} P_{F_b}(n_b | \rho_{n_b+1}) = \sum_{l=0}^{l_{\text{max}}^*/l_{\text{gcd}}^* - 1} P(l(S)/l_{\text{gcd}}^* > l) \quad (5.236)$$

$$= \sum_{l=0}^{l_{\text{max}}^*/l_{\text{gcd}}^* - 1} \sum_{\substack{s \in \mathcal{A} \\ l(s) > l_{\text{gcd}}^* l}} P(s) \quad (5.237)$$

$$= \sum_{s \in \mathcal{A}} P(s) l(s)/l_{\text{gcd}}^* = \bar{l}/l_{\text{gcd}}^*. \quad (5.238)$$

This result, together with (5.234), concludes the proof.  $\blacksquare$

For the sake of clarity, let us recall without comment the approximations made throughout the proof:

- We assumed the equality  $\sum_{m=1}^n s_{L,m} = s_L$  in (5.217).
- We made an approximation in (5.227) for  $j_1 = 1$ , which is compensated by the small value of (5.233).
- We used Lemma 5.63 in (5.231).
- At last, thanks to the largeness of  $N$ , we made several approximations between (5.232) and (5.234). Loosely speaking, recall that  $N$  (and thus  $n_b$ ) must be large enough so that  $T_{s_L, h, l_b = l_{\text{sum}}, n}^{\text{vlc}}$  is sufficiently small to compensate  $\mathcal{O}(n^2 \log^2(n_b - l_{\text{sum}})/(n_b - l_{\text{sum}}))$  when  $l_{\text{sum}} \rightarrow n_b$ .

Note that through these approximations, we have actually neglected the particular distribution  $P_{F_b}(S_1)$  of the first symbol — recall  $S_1$  depends on  $N_b^{(-1)}$  in (5.48). Indeed, the proof holds also if we replace  $P_{F_b}(\rho_{j_1}) P_{F_b}(u_{\gamma_1} | \rho_{j_1})$  in (5.225) with  $P_{F_b}(\rho_{j_1} | n_b^{(-1)}) P_{F_b}(u_{\gamma_1} | \rho_{j_1}, n_b^{(-1)})$  for any admissible  $n_b^{(-1)}$ , or with  $P(\rho_{j_1}) P(u_{\gamma_1} | \rho_{j_1})$ . This shows that (5.54) in Theorem 5.31 is also an approximate spectrum for the conditional spectrum

$$A_{s_L, h}^{F_b | N_b^{(-1)}} \triangleq \sum_{s:} P_{F_b}(B_{\text{vlc}} = s | N_b^{(-1)}) A_{s_L, h | s}^{\text{Bvlc}} \quad (5.239)$$

for any  $N_b^{(-1)}$ . In other words, this shows a result that is in agreement with the intuition: All these spectra (for different  $N_b^{(-1)}$ ) are roughly equal for large  $N$  and small  $h$ , and therefore (5.54) holds for all of them, independently of  $N_b^{(-1)}$ .

### 5.C.3.4 Upper spectrum of VLC block in (5.55)

Let us prove in two steps that  $A_{s_L, h}^{F_b, \text{upp}}$  in (5.55) is an upper spectrum, i.e., that it satisfies (5.39). Firstly, let us develop an upper spectrum for the conditional spectrum  $A_{s_L, h}^{F_b | N_b^{(-1)} = N_b^{\max}}$  given in (5.239). Next, let us extend this result to  $A_{s_L, h}^{F_b}$ .

Given  $N_b^{(-1)} = N_b^{\max}$ , the length of the first symbol  $S_1$  is not constrained any longer; the probability of  $S_1$  is thus the same as the subsequent symbols, see (5.177). Consequently, (5.227) holds even for  $j_1 = 1$ , i.e.,

$$\begin{aligned} \sum_{u_{\mathcal{Y}_m}} P_{F_b}(u_{\mathcal{Y}_m} | \rho_{j_m}, N_b^{(-1)} = N_b^{\max}) A_{s_L = s_{L,m}, h = h_m | u_{\mathcal{Y}_m}}^{\text{vlc}} \\ = A_{s_L = s_{L,m}, h = h_m, l_b = l_m}^{\text{vlc}}. \end{aligned} \quad (5.240)$$

Therefore, the only approximation used to get (5.229) is the equality  $\sum_{m=1}^n s_{L,m} = s_L$  in (5.217), which is compatible with (5.39). Next, we can upper bound (5.229) by replacing  $\mathcal{L}_{n, n_b}$  with  $\mathcal{L}_{n, N_b^{\max}}$  and  $\mathcal{J}_{l, n_b}$  with  $\mathcal{J}_{l, N_b^{\max}}$ . By reordering the summations differently, we get

$$\begin{aligned} A_{s_L, h}^{F_b | N_b^{(-1)} = N_b^{\max}} &\leq \sum_{n \geq 1} \sum_{l \in \mathcal{L}_{n, N_b^{\max}}} \sum_{\substack{s_L, h: \\ \text{s.t. (5.217)}}} \\ &\left( \prod_{m=1}^n A_{s_L = s_{L,m}, h = h_m, l_b = l_m}^{\text{vlc}} \right) \\ &\times \sum_{j \in \mathcal{J}_{l, N_b^{\max}}} P(\rho_{j_1}) \left( \prod_{m=2}^n P(\rho_{1+j_m - j_{m-1} - l_{m-1}}) \right) \\ &\times \sum_{n_b} \underbrace{P_{F_b}(n_b | \rho_{n_b+1}) P(\rho_{2+n_b - j_n - l_n})}_{= P_{F_b}(n_b | \rho_{j_n + l_n}), \text{ by (5.224), (5.226)}} \end{aligned} \quad (5.241)$$

where no approximation that violates (5.39) has been used. In this expression, the last summation trivially equals 1 and the probabilities  $P(\rho_{j_1})$ ,  $P(\rho_{1+j_m - j_{m-1} - l_{m-1}})$  are upper bounded by 1. After simplifications, we obtain

$$A_{s_L, h}^{F_b | N_b^{(-1)} = N_b^{\max}} \leq \sum_{\substack{l_s, l_b, n \\ l_b \leq N_b^{\max}}} \left( \frac{N_b^{\max}}{l_s^* \text{gcd}} - \frac{l_b}{l_s^* \text{gcd}} + n \right) T_{s_L, h, l_s, l_b, n}^{\text{vlc}}. \quad (5.242)$$

Let us now extend this result to  $A_{s_L, h}^{F_b}$ , by assuming temporarily

$$P_{F_b}(B_{\text{vlc}} = s:) \leq \frac{l_{\text{gcd}}^*}{l_{\text{gcd}}^*} P_{F_b}(B_{\text{vlc}} = s: | N_b^{(-1)} = N_b^{\text{max}}). \quad (5.243)$$

Then, by (5.37), (5.239), (5.242) and (5.243),

$$A_{s_L, h}^{F_b} \leq \frac{l_{\text{gcd}}^*}{l_{\text{gcd}}^*} A_{s_L, h}^{F_b | N_b^{(-1)} = N_b^{\text{max}}} \preceq A_{s_L, h}^{F_b, \text{upp}}, \quad (5.244)$$

which proves that  $A_{s_L, h}^{F_b, \text{upp}}$  in (5.55) is an upper spectrum. So we need only to prove (5.243). Straightforwardly,

$$P_{F_b}(B_{\text{vlc}} = s:) = \sum_{n_b^{(-1)}} P_{F_b}(B_{\text{vlc}} = s: | N_b^{(-1)} = n_b^{(-1)}) P_{F_b}(N_b^{(-1)} = n_b^{(-1)}), \quad (5.245)$$

where  $P_{F_b}(N_b^{(-1)} = n_b^{(-1)}) = P_{F_b}(N_b = n_b^{(-1)})$  and

$$P_{F_b}(B_{\text{vlc}} = s: | N_b^{(-1)} = n_b^{(-1)}) \leq \frac{P_{F_b}(B_{\text{vlc}} = s: | N_b^{(-1)} = N_b^{\text{max}})}{P(l(S) > N - n_b^{(-1)})} \quad (5.246)$$

by (5.48). Besides, by (5.181),  $P_{F_b}(N_b = n_b) \leq P(l(S) > N - n_b)$  for values of  $n_b$  satisfying (5.47). Eventually, by gathering everything, (5.243) follows from (5.245) since there are  $l_{\text{gcd}}^*/l_{\text{gcd}}^*$  values of  $n_b$  satisfying (5.47).

## Appendix 5.D Proofs on statistical synchronization and bounded spectrum

### 5.D.1 Proof of Lemma 5.37 and Lemma 5.39

Let us prove Lemma 5.39 only; Lemma 5.37 can be proved similarly.

**Proof of Lemma 5.39.** By Definition 5.35 and Definition 5.38, if the VLC has at least one synchronizing sequence of non-zero probability, then no anti-synchronizing sequence (w.r.t. sequences of non-zero probability) exists.

If no anti-synchronizing sequence w.r.t. sequences of non-zero probability exists, then straightforwardly there does not exist any prefix  $p \in \text{prefix}(\mathcal{V})$  that is anti-synchronizing w.r.t. sequences of non-zero probability.

At last, let us show that the VLC has a synchronizing sequence of non-zero probability if there does not exist any prefix  $p \in \text{prefix}(\mathcal{V})$  that is anti-synchronizing w.r.t. sequences of non-zero probability. For the sake of convenience, let  $\mathcal{X}_0 = \{p \in \{R\} \cup \text{prefix}(\mathcal{V}) : l(p) \equiv 0 \pmod{l_{\text{gcd}}}\}$  and let  $\mathcal{L} = \mathcal{V}^+ \cup (\mathcal{B}^+ \setminus \text{prefix}(\mathcal{V}^+))$ .

If there is no prefix  $p \in \text{prefix}(\mathcal{V})$  that is anti-synchronizing w.r.t. sequences of non-zero probability, then by Definition 5.38, for any prefix  $p \in \text{prefix}(\mathcal{V})$  with  $l(p) \equiv 0 \pmod{l_{\text{gcd}}}$ , there exists  $u \in \mathcal{V}^+$  with  $P(u) > 0$  and  $pu \in \mathcal{L}$ , i.e.,  $pu$  is either not admissible (does not belong to  $\text{prefix}(\mathcal{V}^+)$ ) or belongs to  $\mathcal{V}^+$ . Note that this conclusion holds also trivially if we replace  $p$  by the empty word  $R$  or by  $v.p$  for any sequence of codewords  $v \in \mathcal{V}^+$ . In other words, for any  $p \in \{R\} \cup \text{prefix}(\mathcal{V}^+)$  with  $l(p) \equiv 0 \pmod{l_{\text{gcd}}}$ , there exists  $u \in \mathcal{V}^+$  with  $P(u) > 0$  and  $pu \in \mathcal{L}$ .

Let  $p^1, p^2 \in \mathcal{X}_0$ . By the previous paragraph, there exists  $u^1 \in \mathcal{V}^+$  with  $P(u^1) > 0$  and  $p^1 u^1 \in \mathcal{L}$ . If  $p^2 u^1$  is admissible (belongs to  $\text{prefix}(\mathcal{V}^+)$ ), then by the previous paragraph there exists  $u^3 \in \mathcal{V}^+$  with  $P(u^3) > 0$  and  $p^2 u^1 u^3 \in \mathcal{L}$ . If  $p^2 u^1$  is not admissible, then  $p^2 u^1 u^3$  is not admissible either and thus  $p^2 u^1 u^3 \in \mathcal{L}$ . Note we have also  $p^1 u^1 u^3 \in \mathcal{L}$ . Consequently, for any pair  $p^1, p^2 \in \mathcal{X}_0$ , there exists  $u \in \mathcal{V}^+$  with  $P(u) > 0$ ,  $p^1 u \in \mathcal{L}$  and  $p^2 u \in \mathcal{L}$ . Therefore, by induction on  $\mathcal{X}_0$ , there exists  $u^s \in \mathcal{V}^+$  with  $P(u^s) > 0$  and  $pu^s \in \mathcal{L}$  for all  $p \in \mathcal{X}_0$ . This last conclusion holds also trivially if we replace  $p$  by a non-admissible sequence (in  $\mathcal{B}^+ \setminus \text{prefix}(\mathcal{V}^+)$ ) of length multiple of  $l_{\text{gcd}}$ , or by  $v.p$  for any sequence of codewords  $v \in \mathcal{V}^+$  (thus  $v.p \in \text{prefix}(\mathcal{V}^+)$ ). In other words, there exists  $u^s \in \mathcal{V}^+$  with  $P(u^s) > 0$  and  $\check{u}.u^s \in \mathcal{L}$  for all  $\check{u} \in \mathcal{B}^+$  subject to  $l(\check{u}) \equiv 0 \pmod{l_{\text{gcd}}}$ . The sequence  $u^s$  is thus synchronizing by Definition 5.35 and has a non-zero probability. ■

### 5.D.2 Proof of Theorem 5.40

Let us split the proof in three parts. We will successively show that propositions (b) to (d) are each equivalent to proposition (a).

**Notation 5.65** ( $\text{state}(w)$ ). For any  $w \in \{R\} \cup \text{prefix}(\mathcal{V}^+)$ , note that we can decompose  $w$  into a unique pair  $(v', v'')$  with  $w = v'v''$ ,  $v' \in \{R\} \cup \mathcal{V}^+$  and

$v'' \in \{R\} \cup \text{prefix}(\mathcal{V})$ . Then, let  $\text{state}(w)$  be the second element of this unique pair, i.e.,  $\text{state}(w) = v''$ .  $\square$

### 5.D.2.1 (b) if and only if (a)

**Proof of Theorem 5.40, (b) if and only if (a).** Though it is possible to show that this result follows almost as a particular case of [17, Corollary 1] by rewriting the VLC codewords with a new code alphabet  $X = \mathcal{B}^{l_{\text{gcd}}}$  instead of  $\mathcal{B}$ , we prefer the simpler proof hereafter which capitalizes on the prefix property of the VLC.

Under Assumption 5.5, if a VLC admits a synchronizing sequence  $u_i^s$  having a non-zero probability, then the probability of observing that sequence converges to 1 as the number of observed codewords approaches infinity. Given the value of  $i$  from Definition 5.34, let us consider for example  $U_{j:k} = u_i^s$  for some  $j \geq i$  and  $k = j + l(u_i^s) - 1$ , with  $U_{1:j-1} \in \mathcal{V}^+$ . Upon receiving  $u_i^s$ , the ML-sequence decoder is “forced” to resynchronize since the only admissible decoded sequences  $\check{u}_{1:k}$  subject to  $\check{u}_{j:k} = u_i^s$  belong to  $\mathcal{V}^+$ , by Definition 5.35 of the synchronizing sequence. Consequently, we have  $\rho_{k+1}(U_{1:\infty}, \check{U}_{1:\infty})$  and the VLC is statistically synchronizable.

Let us prove the converse. If the VLC is statistically synchronizable, then (5.66) is satisfied for any  $i$  and any  $u_{1:i-1} \in \text{prefix}(\mathcal{V}^+)$ . Let us consider values of  $i$  subject to  $i \equiv 1 \pmod{l_{\text{gcd}}^*}$  and  $u_{1:i-1} \in \mathcal{V}^+$  with  $P(u_{1:i-1}) > 0$ . By (5.66), the decoder resynchronizes with the correct sequence,  $\rho_t(U_{1:\infty}, \check{U}_{1:\infty})$  for some  $t$  subject to  $i \leq t \leq s$ , with probability 1 as the number of bits received correctly ( $\check{U}_{i:s-1} = U_{i:s-1}$ ) goes to infinity.

Let us prove that a synchronizing sequence necessarily exists and has a non-zero probability when (5.66) holds for the sequence-ML decoder. The sequence-ML decoder considers only admissible sequences and selects the closest one to the received signal. Here however, since (5.66) holds independently of channel errors, the ML-sequence decoder resynchronizes independently of the received signal, by relying solely on the constraint  $\check{U}_{i:s-1} = U_{i:s-1}$ . In other words, all admissible sequences  $\check{U}_{1:s-1}$  that satisfy  $\check{U}_{i:s-1} = U_{i:s-1}$  resynchronize with the correct sequence with probability 1 as  $s$  goes to infinity. It means that, for any potential and arbitrary (erroneous) candidate  $\check{u}_{1:i-1} \in \text{prefix}(\mathcal{V}^+)$ , there exist  $t$  and a sequence  $\omega = u_{i:t-1} \in \mathcal{V}^+$

(depending on  $\check{u}_{1:i-1}$ ) such that  $P(\omega) > 0$  and  $\check{u}_{1:i-1}\omega \in \mathcal{Z}$  with  $\mathcal{Z} = \mathcal{V}^+ \cup (\mathcal{B}^+ \setminus \text{prefix}(\mathcal{V}^+))$ , i.e., either  $\check{u}_{1:i-1}\omega$  is not admissible (does not belong to  $\text{prefix}(\mathcal{V}^+)$ ) and is thus discarded by the decoder (in favor of an admissible sequence), or  $\check{u}_{1:i-1}\omega$  is admissible, belongs to  $\mathcal{V}^+$  and thus resynchronizes the decoder ( $\rho_t(U_{1:\infty}, \check{U}_{1:\infty})$ ).

By the prefix property, note that the sequence  $\omega$  depends only on state( $\check{u}_{1:i-1}$ ). Besides, since the integer  $i$  and the candidate  $\check{u}_{1:i-1}$  are arbitrary subject to  $i \equiv 0 \pmod{l_{\text{gcd}}^*} \equiv 0 \pmod{l_{\text{gcd}}}$ , the value of state( $\check{u}_{1:i-1}$ ) can be any value in the set  $\mathcal{X}_0 = \{p \in \{R\} \cup \text{prefix}(\mathcal{V}) : l(p) \equiv 0 \pmod{l_{\text{gcd}}}\}$ . In other words, it follows from the previous paragraph that for any  $p \in \mathcal{X}_0$ , there exists  $\omega \in \mathcal{V}^+$  with  $P(\omega) > 0$  and  $p\omega \in \mathcal{Z}$ . The VLC has thus no prefix that is anti-synchronizing w.r.t. sequences of non-zero probability, by Definition 5.38. Consequently, the source/VLC has at least one synchronizing sequence of non-zero probability by Lemma 5.39. ■

### 5.D.2.2 (c) if and only if (a)

For the sake of clarity, we say in the following that a pair of sequences  $(u, \check{u})$  starts an error event if there exists a pair  $(u', \check{u}')$ ,  $l(u') = l(\check{u}') > 0$ , such that  $(u; u', \check{u}; \check{u}')$  forms an error event.

Let  $l_b^{\text{th}}$  be the threshold above which

- any integer  $l$  multiple of  $l_{\text{gcd}}^*$  can be written as a summation of lengths of codewords of non-zero probability, i.e.,  $\forall l \geq l_b^{\text{th}}$  with  $l \equiv 0 \pmod{l_{\text{gcd}}^*}$ , there exist  $k$  and  $s_{1:k} \in \mathcal{A}^k$  with  $P(s_{1:k}) > 0$  and  $l = l(s_{1:k})$ ;
- and any integer  $l$  multiple of  $l_{\text{gcd}}$  can be written as a summation of lengths of codewords, i.e.,  $\forall l \geq l_b^{\text{th}}$  with  $l \equiv 0 \pmod{l_{\text{gcd}}}$ , there exist  $k$  and  $s_{1:k} \in \mathcal{A}^k$  with  $l = l(s_{1:k})$ .

Such a threshold always exists. Let  $l_s^{\text{th}} = \lceil (l_b^{\text{th}} + 2l_{\text{max}}) / l_{\text{min}} \rceil$ .

**Lemma 5.66.** *Under Assumption 5.4, if a VLC has no synchronizing sequence of non-zero probability, then there exists a pair of sequences  $(u^1, \check{u}^1)$  that starts an error event such that  $u^1 \in \mathcal{V}^+$  with  $P(u^1) > 0$ ,  $\check{u}^1$  is anti-synchronizing w.r.t. sequences of non-zero probability,  $l(u^1) = l(\check{u}^1) \leq l_b^{\text{th}} + 2l_{\text{max}}$  and  $l_S(u^1) \leq l_s^{\text{th}}$ . □*

Recall Definition 5.38 of an anti-synchronizing sequence w.r.t. sequences of non-zero probability. This lemma states a trivial result, that the decoder can reach an anti-synchronizing sequence (w.r.t. sequences of non-zero probability) with a finite number of bit errors, actually with less than  $l_b^{\text{th}} + 2l_{\text{max}}$  bit errors. This is a loose bound; a tighter one is not needed.

**Proof.** By Lemma 5.39, there exists a prefix  $p \in \text{prefix}(\mathcal{V})$  that is anti-synchronizing w.r.t. sequences of non-zero probability. Note  $l(p) < l_{\text{max}}$  and  $l(p) \equiv 0 \pmod{l_{\text{gcd}}}$ . Let  $\check{u} \in \mathcal{V}^+$  be the shortest (in bit length) sequence of codewords with  $l(\check{u}) \geq l_b^{\text{th}}$  and  $l(\check{u};p) \equiv 0 \pmod{l_{\text{gcd}}^*}$ . Such a sequence always exists by definition of  $l_b^{\text{th}}$ . Besides, still by definition of  $l_b^{\text{th}}$ , there exists a sequence  $u \in \mathcal{V}^+$  of length  $l(u) = l(\check{u};p)$  with  $P(u) > 0$ . At last, let  $i$  be the largest integer with  $\rho_i(u, \check{u}; p)$  and let  $j = l(\check{u})$ . Note  $l(\check{u};p) \leq l_b^{\text{th}} + 2l_{\text{max}}$ . Therefore the pair formed by  $u^1 = u_{i:j+l(p)}$  and  $\check{u}^1 = \check{u}_{i:j} p$  proves this lemma. ■

**Lemma 5.67.** Under Assumption 5.4, given  $u^3 \in \mathcal{V}^+$  with  $l(u^3) \geq l_b^{\text{th}} + 2l_{\text{max}}$  and given a pair  $(u, \check{u})$  that starts an error event where  $u \in \mathcal{V}^+$ , there exist an integer  $l_b \leq l_b^{\text{th}} + 2l_{\text{max}}$  and a sequence  $\check{u}_{1:l_b}^3$  such that  $(u: u_{1:l_b}^3, \check{u}: \check{u}_{1:l_b}^3)$  forms an error event. □

This lemma states also a trivial result, that the decoder can always resynchronize with a finite number of bit errors, actually with less than  $l_b^{\text{th}} + 2l_{\text{max}}$  bit errors.

**Proof.** Let  $s \in \text{suffix}(\mathcal{V})$  be a suffix such that  $\check{u}; s \in \mathcal{V}^+$ . Such a suffix always exists. Note  $l(s) < l_{\text{max}}$  and  $l(s) \equiv 0 \pmod{l_{\text{gcd}}}$  since  $l(\check{u}) = l(u) \equiv 0 \pmod{l_{\text{gcd}}}$ . Let  $i$  be the smallest integer with  $i \geq l_b^{\text{th}} + l(s)$  and  $u_{1:i}^3 \in \mathcal{V}^+$ . By definition of  $l_b^{\text{th}}$ , there exists a sequence  $\check{u}' \in \mathcal{V}^+$  of length  $i - l(s)$ . At last, let  $\check{u}^3 = s \check{u}'$  and let  $l_b$  be the smallest integer with  $\rho_{1+l(u)+l_b}(u: u^3, \check{u}: \check{u}^3)$ . Note  $l_b \leq i \leq l_b^{\text{th}} + 2l_{\text{max}}$  and  $(u: u_{1:l_b}^3, \check{u}: \check{u}_{1:l_b}^3)$  forms an error event. ■

**Proof of Theorem 5.40, (c) if and only if (a).** If there exists a synchronizing sequence  $s_{1:k}^s$  of non-zero probability, then the VLC stream spectrum

$$A_{h,l_s}^{\text{vlc}} \triangleq \sum_{s_{1:l_s} \in \mathcal{S}^{l_s}} A_{h|s_{1:l_s}}^{\text{vlc}} P(s_{1:l_s})$$

can be upper bounded as follows. Let us restrict the enumeration to sequences  $s_{1:l_s}$  of non-zero probability (the others do not contribute to the spectrum).

Then, the sequence  $s_{1:l_s}$  has a maximum bit length of  $l_{\max}^* l_s$ , thus the conditional spectrum  $A_{h|s_{1:l_s}}^{\text{vlc}}$  is trivially upper bounded by  $\binom{l_{\max}^* l_s}{h}$ . Therefore

$$A_{h,l_s}^{\text{vlc}} \leq \binom{l_{\max}^* l_s}{h} P, \quad \text{where } P \triangleq \sum_{\substack{s_{1:l_s} \in \mathcal{A}^{l_s} \\ A_{h|s_{1:l_s}}^{\text{vlc}} > 0}} P(s_{1:l_s}).$$

Note  $P$  is the probability of sequences  $s_{1:l_s}$  such that  $A_{h|s_{1:l_s}}^{\text{vlc}} > 0$ . Let us now find a set which includes such sequences so that we can upper bound  $P$  by the probability of this set. To this end, let us split the sub-sequence  $s_{1:l_s-1}$  into  $\lfloor (l_s - 1)/k \rfloor$  parts of symbol length  $k$ , plus a possible residue of less than  $k$  symbols. Notice that the conditional spectrum  $A_{h|s_{1:l_s}}^{\text{vlc}} = 0$  when at least  $h + 1$  parts of  $s_{1:l_s-1}$  contain the synchronizing sequence  $s_{1:k}^s$ . Indeed, we have then  $h + 1$  copies of the synchronizing sequence; thus at least one of them is not affected by any of the  $h$  bit errors, which forces the resynchronization of the decoder and thus prevents any error event from going farther. Consequently, the set of sequences  $s'_{1:l_s}$  such that at most  $h$  parts of  $s'_{1:l_s-1}$  contain the synchronizing sequence  $s_{1:k}^s$  includes the set of sequences  $s_{1:l_s}$  subject to  $A_{h|s_{1:l_s}}^{\text{vlc}} > 0$ . Therefore,  $P$  can be upper bounded by

$$P \leq (1 - P(s_{1:k}^s))^{\lfloor (l_s-1)/k \rfloor - h}.$$

Let us define  $c'' = 1 - P(s_{1:k}^s)$ . Note  $c'' < 1$ . Then, for any fixed  $h$ ,

$$A_{h,l_s}^{\text{vlc}} \leq \binom{l_{\max}^* l_s}{h} c''^{\lfloor (l_s-1)/k \rfloor - h} \quad (5.247)$$

$$\leq \frac{(l_{\max}^* l_s)^h}{h!} c''^{(l_s-1)/k-1-h} \quad (5.248)$$

$$\leq \frac{(l_{\max}^* l_s)^h}{h!} c''^{-1/k-1-h} c''^{l_s/k} \quad (5.249)$$

$$= \mathcal{O}(c''^{l_s}), \quad (5.250)$$

for any  $c'$  such that  $c''^{1/k} < c' < 1$ . By Property 5.15, the VLC spectrum is thus bounded.

Let us now prove the converse, or rather the contrapositive of the converse, that the spectrum is not bounded if the VLC has no synchronizing sequence of non-zero probability. Let  $l \geq 2l_s^{\text{th}}$ . In the following, we are interested mainly

in the quantity

$$A_l \triangleq \sum_{h=1}^{2l_b^{\text{th}}+4l_{\text{max}}} \sum_{l_s=l-l_s^{\text{th}}+1}^l A_{h,l_s}^{\text{vlc}},$$

which can be rewritten as

$$A_l = \sum_{s_{1:l} \in \mathcal{S}^l} P(s_{1:l}) A_{l|s_{1:l}} \quad (5.251)$$

where

$$A_{l|s_{1:l}} \triangleq \sum_{h=1}^{2l_b^{\text{th}}+4l_{\text{max}}} \sum_{l_s=l-l_s^{\text{th}}+1}^l A_{h,l_s|s_{1:l_s}}^{\text{vlc}}. \quad (5.252)$$

Notice that  $A_l$  is bounded,  $A_l = \mathcal{O}(c^l)$  where  $c < 1$ , if and only if the VLC spectrum is bounded. We can therefore prove that the VLC spectrum is not bounded by proving that  $A_l$  is lower bounded by a constant independent of  $l$ , which we will do for  $l \geq 2l_s^{\text{th}}$ .

Let us prove  $A_l$  is lower bounded by a constant. The summation in (5.251) is taken over all sequences  $s_{1:l} \in \mathcal{S}^l$ . Let us restrict this summation to all  $s_{1:l}$  that start with the sub-sequence  $s_{:1}^1 = \text{vlc}^{-1}(u_{:1}^1)$ , where  $u_{:1}^1$  is given by Lemma 5.66 — note  $P(s_{:1}^1) > 0$ . Let  $l_1 = l_s(s_{:1}^1)$ . If we assume temporarily that  $A_{l|s_{1:l}} \geq 1$  for sequences  $s_{1:l}$  that start with  $s_{:1}^1$ , then  $A_l$  is trivially lower bounded by

$$A_l \geq \sum_{\substack{s_{1:l} \in \mathcal{S}^l \\ \text{s.t. } s_{1:l_1} = s_{:1}^1}} P(s_{1:l}) A_{l|s_{1:l}} \geq P(s_{:1}^1), \quad (5.253)$$

which is a strictly positive constant independent of  $l$ .

So the proof boils down eventually to showing that  $A_{l|s_{1:l}} \geq 1$  for sequences  $s_{1:l}$  that start with  $s_{:1}^1$ . To this end, let  $s_{1:l}$  be any sequence in  $\mathcal{S}^l$  with  $s_{1:l_1} = s_{:1}^1$  and  $P(s_{1:l}) > 0$ , and let us find/construct a sequence  $\mathfrak{s}_{:}$  that forms an error event with  $s_{1:l_s}$ , for some  $l_s$  subject to  $l - l_s^{\text{th}} + 1 \leq l_s \leq l$ , with less than  $2l_b^{\text{th}} + 4l_{\text{max}}$  bit errors — then  $A_{l|s_{1:l}} \geq 1$  by (5.252). Let  $l_2 = l - l_s^{\text{th}} + 1$ ; note  $l_2 > l_1$  since  $l_1 \leq l_s^{\text{th}}$  (by Lemma 5.66) and since we assumed  $l \geq 2l_s^{\text{th}}$ . In the following, the VLC encoding is assumed implicitly; in particular, we use  $u_{:s}$  as a simpler notation for the concatenation of  $u_{:}$  with  $\text{vlc}(s_{:})$ . By Lemma 5.66, there exists a pair  $(s_{:1}^1, \mathfrak{u}_{:1}^1)$  that starts an error event. This error

event can be made arbitrarily long by appending codewords of non-zero probability since  $\check{u}^1$  is anti-synchronizing w.r.t. sequences of non-zero probability; in particular, by appending  $s_{l_1+1:l_2-1}$ , the pair  $(s^1_{l_1+1:l_2-1}, \check{u}^1_{l_1+1:l_2-1})$  starts an error event. At last, given  $s_{l_2:l}$ , there exists by Lemma 5.67 some  $\check{u}^3$  — note  $l_S(s_{l_2:l}) = l_s^{\text{th}}$  and thus  $l(s_{l_2:l}) \geq l_b^{\text{th}} + 2l_{\max}$  — such that the pair  $(s^1_{l_1+1:l_2-1} s_{l_2:l_s}, \check{u}^1_{l_1+1:l_2-1} \check{u}^3_{l_2:l_s})$  forms an error event  $e$  for some  $l_s$  subject to  $l_2 = l - l_s^{\text{th}} + 1 \leq l_s \leq l$ . The hamming distance of this error event,  $d_H(e)$ , is smaller than  $l(\check{u}^1) + l(\check{u}^3)$ , thus  $d_H(e) \leq 2l_b^{\text{th}} + 4l_{\max}$  by Lemma 5.66 and Lemma 5.67. Consequently, for  $l \geq 2l_s^{\text{th}}$  and given any sequence  $s_{1:l}$  of non-zero probability that starts with  $s^1$ , we have found a sequence  $\check{s}$  that forms an error event with  $s_{1:l_s}$  for some  $l_s$  subject to  $l - l_s^{\text{th}} + 1 \leq l_s \leq l$ , with less than  $2l_b^{\text{th}} + 4l_{\max}$  bit errors. Therefore,  $A_{l|s_{1:l}}$  is larger than 1 by (5.252),  $A_l$  is lower bounded by a constant by (5.253) and the VLC spectrum is thus not bounded. ■

### 5.D.2.3 (d) if and only if (a)

**Proof of Theorem 5.40, (d) if and only if (a).** The VLC spectrum is trivially bounded if it is strongly bounded. Therefore, by the equivalence between (a) and (c), the VLC has at least one synchronizing sequence of non-zero probability if its spectrum is strongly bounded.

Let us now prove the converse, that the spectrum is strongly bounded if the VLC has a synchronizing sequence of non-zero probability. Let us replace  $l_s$  by  $\eta h$  in (5.249) for some  $\eta$ . Then

$$A_{h,l_s=\eta h}^{\text{vlc}} \leq \frac{(l_{\max}^* \eta h)^h}{h!} c''^{-1/k-1-h} c''^{\eta h/k} \quad (5.254)$$

where  $c'' < 1$ . Note  $h^h/h! \leq e^h$ . Let  $K$  be the constant  $K = e l_{\max}^* c''^{-1/k-2}$  and let  $c' = c''^{1/k}$ . Then

$$A_{h,l_s=\eta h}^{\text{vlc}} \leq K^h \eta^h c'^{\eta h} = (K \eta c'^{\eta})^h. \quad (5.255)$$

From this expression, it follows that there exists some  $c < 1$  and some  $\eta$  such that for all  $l_s \geq \eta h$ ,  $A_{h,l_s}^{\text{vlc}} \leq c^{l_s}$ , which proves the VLC spectrum is strongly bounded by Property 5.16. ■

### 5.D.3 Proof of Proposition 5.45

**Proof.** Let us assume there exists a statistically synchronizable complete reversible VLC that is not an FLC. Then, by Theorem 5.40, there exists a synchronizing sequence  $s^s \equiv u^s$  of non-zero probability. Let  $p \in \text{prefix}(\mathcal{V})$  be a prefix of length  $l(p) \equiv 0 \pmod{l_{\text{gcd}}}$ . Such a  $p$  always exists if the VLC is not an FLC. By Definition 5.35 of a synchronizing sequence, we have  $pu^s \in \mathcal{V}^+$  — note that we have not  $pu^s \notin \text{prefix}(\mathcal{V}^+)$  since the VLC is complete and thus  $\text{prefix}(\mathcal{V}^+) = \mathcal{B}^+$ . But since the VLC is reversible, no VLC codeword is the suffix of another VLC codeword and it follows from  $pu^s \in \mathcal{V}^+$  that  $p$  is a VLC codeword ( $p \in \mathcal{V}$ ), which contradicts the fact that  $p$  is a prefix. ■

### 5.D.4 Proof of Proposition 5.51

**Proof.** This result follows trivially from Definition 5.13 of a bounded spectrum. Let  $h$  be a finite number of bit errors; let  $c$  and  $l'_h$  be the values from Definition 5.13. Then, the average number of Levenshtein symbol errors is finite since

$$\sum_{s_L=1}^{\infty} s_L A_{s_L,h}^{\text{vlc}} \leq \sum_{l_b=1}^{\infty} l_b A_{l_b,h}^{\text{vlc}} \quad (5.256)$$

$$= \sum_{l_b=1}^{l'_h-1} l_b A_{l_b,h}^{\text{vlc}} + \sum_{l_b=l'_h}^{\infty} l_b A_{l_b,h}^{\text{vlc}} \quad (5.257)$$

where the first summation in (5.257) involves a finite number of terms and the second summation converges to a finite value since  $A_{l_b,h}^{\text{vlc}} \leq c^{l_b}$  for  $l_b \geq l'_h$ . ■

### 5.D.5 Proof of Proposition 5.52

The next two lemmas make the proof easier.

**Lemma 5.68.** *Under Assumption 5.5, there exists an integer  $\kappa$  such that, for all  $h$ , either  $A_{l_b,h}^{\text{vlc}} > 0$  for some  $l_b \leq \kappa h$ , or  $A_h^{\text{vlc}} = 0$ . □*

**Proof.** Let  $\kappa = (2^{l_{\text{max}}} + 1)l_{\text{max}}^*$ . If  $A_h^{\text{vlc}} > 0$ , then there exists at least one pair  $(u_{1:l_b}, \check{u}_{1:l_b})$ , for some integer  $l_b$ , that forms an error event such that  $d_H(u_{1:l_b}, \check{u}_{1:l_b}) = h$  and  $P(u_{1:l_b}) > 0$  (of non-zero probability). If  $l_b \leq \kappa h$ , the lemma holds trivially. If, on the contrary,  $l_b > \kappa h$ , then let us prove that

there exists an error event strictly shorter than  $l_b$ . This would prove the lemma by induction.

Since  $l_b > \kappa h = h(2^{l_{\max}} + 1)l_{\max}^*$  and since  $d_H(u_{1:l_b}, \check{u}_{1:l_b}) = h$ , there exist  $i, j$  such that  $j - i + 1 > 2^{l_{\max}}l_{\max}^*$  and  $u_{i:j} = \check{u}_{i:j}$ . We can show this as follows. Recall there is always a bit error during the first symbol of an error event. If we do not consider the first symbol, there are at most  $h - 1$  bit errors among at least  $l_b - l_{\max}^*$  bits (since the maximum length of a symbol of non-zero probability is  $l_{\max}^*$ ). Therefore, the longest error free gap between two consecutive bit errors, or between the last bit error and the last bit of the error event, is longer than  $j - i + 1 \geq \frac{l_b - l_{\max}^* - (h-1)}{h}$ . After simplifications,  $j - i + 1 > 2^{l_{\max}}l_{\max}^*$ .

Let  $\mathcal{S}$  be the set  $\mathcal{S} = \{s : i \leq s \leq j, \rho_s(u_{1:l_b}, u_{1:l_b})\}$ , or equivalently,  $\mathcal{S} = \{s : i \leq s \leq j, \text{state}(u_{1:s-1}) = R\}$  — recall Notation 5.65. Note that  $\text{state}(\check{u}_{1:s-1}) \neq R$  for all  $s \in \mathcal{S}$ . Otherwise, indeed, we would have  $\text{state}(u_{1:s-1}) = \text{state}(\check{u}_{1:s-1}) = R$ , i.e.,  $\rho_s(u_{1:l_b}, \check{u}_{1:l_b})$ , for some  $s \leq j \leq l_b$  and thus the pair  $(u_{1:l_b}, \check{u}_{1:l_b})$  would not form an error event by Definition 5.7. Therefore  $\text{state}(\check{u}_{1:s-1}) \in \text{prefix}(\mathcal{V})$  for all  $s \in \mathcal{S}$ . Note the cardinality of  $\mathcal{S}$  is larger than  $2^{l_{\max}}$  since  $j - i + 1 > 2^{l_{\max}}l_{\max}^*$  and the cardinality of  $\text{prefix}(\mathcal{V})$  is strictly smaller than  $2^{l_{\max}} - 1$ . Because of that,  $\text{state}(\check{u}_{1:s-1})$  takes at least two times the same value over  $s \in \mathcal{S}$ . More precisely, there exist  $m, n \in \mathcal{S}$ ,  $m \neq n$ , such that  $\text{state}(\check{u}_{1:m-1}) = \text{state}(\check{u}_{1:n-1})$ . To conclude the proof, notice that the pair  $(u_{1:m-1} u_{n:l_b}, \check{u}_{1:m-1} \check{u}_{n:l_b})$  forms an error event of length strictly smaller than  $l_b$  with  $P(u_{1:m-1} u_{n:l_b}) > 0$ . ■

The next lemma is a straightforward corollary.

**Lemma 5.69.** *Under Assumption 5.5, there exist some  $\epsilon < 1$  such that, for all  $h$ , either  $A_h^{\text{vlc}} = 0$  or  $A_h^{\text{vlc}} > \epsilon^h$ . □*

**Proof.** Let  $p$  be the smallest strictly positive symbol probability:  $p = \min\{P(S = s) > 0 : s \in \mathcal{A}\}$ . Note that if  $A_{l_b, h}^{\text{vlc}} > 0$ , then  $A_{l_b, h}^{\text{vlc}} > p^{l_b}$ . Indeed, if  $A_{l_b, h}^{\text{vlc}} > 0$ , then there exists at least one error event  $(u_{1:l_b}, \check{u}_{1:l_b})$  where the sequence  $u_{1:l_b}$  has a strictly positive probability. Since the sequence  $u_{1:l_b}$  contains at most  $l_b$  symbols, its probability is at least  $p^{l_b}$ . By Lemma 5.68, we can conclude the proof with  $\epsilon = p^k$ . ■

**Proof of Theorem 5.52.** When  $A_h^{\text{vlc}} = 0$ , (5.67) holds trivially. Let us thus prove (5.67) for  $A_h^{\text{vlc}} > 0$ . Let  $c$  and  $\eta$  be values from Definition 5.14 of a strongly bounded spectrum. Then, for  $\mu' \geq \eta$ ,

$$\sum_{s_L=1}^{\infty} s_L A_{s_L,h}^{\text{vlc}} \leq \sum_{l_b=1}^{\mu' h-1} l_b A_{l_b,h}^{\text{vlc}} + \sum_{l_b=\mu' h}^{\infty} l_b A_{l_b,h}^{\text{vlc}} \quad (5.258)$$

$$\leq \mu' h A_h^{\text{vlc}} + \sum_{l_b=\mu' h}^{\infty} l_b c^{l_b}. \quad (5.259)$$

Since  $c < 1$ , there exist  $c' < 1$  and  $\mu' \geq \eta$ , independent of  $h$ , such that  $\sum_{l_b=\mu' h}^{\infty} l_b c^{l_b} \leq c'^{\mu' h}$ . At last, given the value of  $\epsilon$  from Lemma 5.69, notice that (5.67) holds for any  $\mu$  subject to  $\mu - 1 \geq \mu'$  and to  $c'^{\mu-1} < \epsilon$ . Indeed, with such a value of  $\mu$ ,

$$\sum_{s_L=1}^{\infty} s_L A_{s_L,h}^{\text{vlc}} \leq (\mu - 1) h A_h^{\text{vlc}} + c'^{(\mu-1)h}, \quad (5.260)$$

where  $c'^{(\mu-1)h} < \epsilon^h < A_h^{\text{vlc}} \leq h A_h^{\text{vlc}}$ . ■

### 5.D.6 Proof of Theorem 5.54

**Lemma 5.70.** *If the VLC spectrum is strongly bounded, then under Assumption 5.5 the spectrum of the VLC block satisfies*

$$\sum_{s_L} s_L A_{s_L,h}^F \leq \mu h \theta^h \binom{N}{\lfloor h/d_f^{\text{vlc}} \rfloor} \quad (5.261)$$

where  $\mu, \theta'$  are constants independent of  $h$  and  $N$ , and  $F$  is the framing rule  $F_s$  or  $F_b$ . □

**Proof.** Recall the expression of  $T_{s_L, h, l_s, l_b, n}^{\text{vlc}}$  from (5.43). With a few manipulations, we get the upper bound

$$\begin{aligned} & \sum_{l_b, l_s, s_L \geq 1} s_L T_{s_L, h, l_s, l_b, n}^{\text{vlc}} \\ & \leq \sum_{\substack{s_{L,1}, s_{L,2}, \dots, s_{L,n} \geq 1, \\ h_1, h_2, \dots, h_n \geq 1 \\ \text{s.t. } h_1 + h_2 + \dots + h_n = h}} \left( \sum_{m=1}^n s_{L,m} \right) \prod_{m'=1}^n A_{s_{L,m'}, h_{m'}}^{\text{vlc}} \end{aligned}$$

$$\begin{aligned}
&= \sum_{m=1}^n \sum_{\substack{s_{L,1}, s_{L,2}, \dots, s_{L,n} \geq 1, \\ h_1, h_2, \dots, h_n \geq 1 \\ \text{s.t. } h_1 + h_2 + \dots + h_n = h}} s_{L,m} A_{s_{L,m}, h_m}^{\text{vlc}} \prod_{\substack{m'=1 \\ m' \neq m}}^n A_{s_{L,m'}, h_{m'}}^{\text{vlc}} \\
&\stackrel{(a)}{\leq} \sum_{m=1}^n \sum_{\substack{h_1, h_2, \dots, h_n \geq 1 \\ \text{s.t. } h_1 + h_2 + \dots + h_n = h}} \mu h_m A_{h_m}^{\text{vlc}} \prod_{\substack{m'=1 \\ m' \neq m}}^n A_{h_{m'}}^{\text{vlc}} \\
&= \sum_{\substack{h_1, h_2, \dots, h_n \geq 1 \\ \text{s.t. } h_1 + h_2 + \dots + h_n = h}} \left( \sum_{m=1}^n \mu h_m \right) \prod_{m'=1}^n A_{h_{m'}}^{\text{vlc}} \\
&= \mu h \sum_{\substack{h_1, h_2, \dots, h_n \geq 1 \\ \text{s.t. } h_1 + h_2 + \dots + h_n = h}} \prod_{m=1}^n A_{h_m}^{\text{vlc}} \tag{5.262}
\end{aligned}$$

where step (a) follows from Theorem 5.52. Let  $c$  and  $\eta$  be values from Definition 5.14 of a strongly bounded spectrum, and let  $K'' = \sum_{l_b \geq \eta} c^{l_b}$ ,  $K' = K'' + 2^\eta$  and  $K = 2K'$ . Then

$$A_h^{\text{vlc}} = \sum_{l_b=1}^{\infty} A_{h, l_b}^{\text{vlc}} \leq \sum_{l_b=1}^{\eta h-1} 2^{l_b} + \sum_{l_b=\eta h}^{\infty} c^{l_b} \leq 2^{\eta h} + K'' \leq K'^h,$$

where we used the inequality  $A_{h, l_b}^{\text{vlc}} \leq \binom{l_b}{h} \leq 2^{l_b}$ . Therefore, since the summation in (5.262) involves  $\binom{h-1}{n-1}$  terms and since  $\binom{h-1}{n-1} \leq 2^{h-1}$ ,

$$\sum_{l_b, l_s, s_L \geq 1} s_L T_{s_L, h, l_s, l_b, n}^{\text{vlc}} \leq \mu h K'^h \binom{h-1}{n-1} \leq \mu h K^h.$$

Recall the upper bounds (5.70) and (5.71) on the upper (see Definition 5.25) spectra  $A_{s_L, h}^{F_s, \text{app}}$  and  $A_{s_L, h}^{F_b, \text{upp}}$ . Note that  $T_{s_L, h, l_s, l_b, n}^{\text{vlc}} = 0$  if  $n > \lfloor h/d_f^{\text{vlc}} \rfloor$ , since  $h = \sum_{m=1}^n h_m$  with  $h_m \geq d_f^{\text{vlc}}$ . Therefore, the spectra of the VLC blocks for both framing rules ( $F_s$  and  $F_b$ ) satisfy

$$\sum_{s_L} s_L A_{s_L, h}^F \leq \mu h K^h \frac{l_{\max}^*}{l_{\text{gcd}}^*} \sum_{n=1}^{\lfloor h/d_f^{\text{vlc}} \rfloor} \binom{N}{n}. \tag{5.263}$$

Let us consider two cases. Firstly, if  $\lfloor h/d_f^{\text{vlc}} \rfloor \geq N/2$ , then  $h \geq d_f^{\text{vlc}} N/2 \geq N/2$  and the lemma holds with  $\theta' = 4K l_{\max}^* / l_{\text{gcd}}^*$  since  $\sum_{n=0}^N \binom{N}{n} = 2^N = 4^{N/2}$ . Secondly, if  $\lfloor h/d_f^{\text{vlc}} \rfloor < N/2$ , then the largest binomial coefficient in (5.263) is  $\binom{N}{\lfloor h/d_f^{\text{vlc}} \rfloor}$ . Since the summation over  $n$  in (5.263) has at most  $h$

terms,  $\sum_{s_L} s_L A_{s_L, h}^F \leq \mu h K^h (I_{\max}^*/I_{\text{gcd}}^*) h^{\binom{N}{\lfloor h/d_f^{\text{vlc}} \rfloor}}$  and the lemma holds again with  $\theta' = 4 K I_{\max}^*/I_{\text{gcd}}^*$ . ■

By defining a new constant  $\theta = \max\{3\mu\theta'; 3\theta'\}$ , the next lemma is a straightforward corollary.

**Lemma 5.71.** *If the VLC spectrum is strongly bounded, then under Assumption 5.5 the spectrum of the VLC block satisfies*

$$A_h^F \leq \theta^h \binom{N}{\lfloor h/d_f^{\text{vlc}} \rfloor}, \quad (5.264)$$

$$h A_h^F \leq \theta^h \binom{N}{\lfloor h/d_f^{\text{vlc}} \rfloor}, \quad (5.265)$$

$$\sum_{s_L} s_L A_{s_L, h}^F \leq \theta^h \binom{N}{\lfloor h/d_f^{\text{vlc}} \rfloor}, \quad (5.266)$$

where  $\theta$  is a constant independent of  $h$  and  $N$ , and  $F$  is the framing rule  $F_s$  or  $F_b$ . ■

Note (5.264) and (5.265) are straightforward when  $d_f^{\text{vlc}} = 1$  since  $A_h^F \leq \binom{N}{h}$  and thus  $h A_h^F \leq h \binom{N}{h} \leq 2^h \binom{N}{h}$ .

**Lemma 5.72.** *The interleaving gains (5.72)–(5.74) on the BER,  $\text{SER}_L$  and FER, given in Theorem 5.54, hold for  $d_f^{\text{vlc}} \geq 3$ . □*

Lemma 5.72 is an extension of [92, Th. 8.4] and the proof hereafter relies heavily on the results from [92]. To make things clearer, the same notations are used in the proof and the reader is invited to consult [92] for details.

**Proof.** Let  $n$  denote the length of the global (concatenated) code in Fig. 5.2,  $n = N/r_{cc}$  where  $r_{cc}$  is the rate of the CC. Let  $\bar{P}_e^{(n)}$  be the error rate we would like to upper bound, either BER,  $\text{SER}_L$  or FER. By Theorem 5.32 and by (5.57), this error rate  $\bar{P}_e^{(n)}$  can be upper bounded as

$$\bar{P}_e^{(n)} \leq k^{(n)} \sum_{h \geq 1} \bar{B}_h^{(n)} P_h \quad (5.267)$$

if we define

$$\bar{B}_h^{(n)} = \sum_{d=1}^N \frac{B_d^{[1]} A_{d, h}^{[2]}}{\binom{N}{d}} \quad (5.268)$$

where  $A_{d,h}^{[2]}$  is the spectrum of the block-equivalent CC, and  $B_d^{[1]}$  and  $k^{(n)}$  depend on the chosen error rate:

$$\begin{aligned} B_d^{[1]} &= d A_{h=d}^F, & k^{(n)} &= \frac{k^{-1}}{N - l_{\max}^*}, & \text{for the BER,} \\ B_d^{[1]} &= \sum_{s_L \geq 1} s_L A_{s_L, h=d}^F, & k^{(n)} &= \frac{l_{\max}^* k^{-1}}{N - l_{\max}^*}, & \text{for the SER}_L, \\ B_d^{[1]} &= A_{h=d}^F, & k^{(n)} &= 1, & \text{for the FER.} \end{aligned}$$

Note these values of  $k^{(n)}$  are upper bounds on  $(N_b^{\min})^{-1}$  and  $(N_s^{\min})^{-1}$  from Theorem 5.32 and hold for both framing rules.

Let us develop  $\bar{B}_h^{(n)}$  further. Let  $\bar{B}_{\leq h}^{(n)}$  denote  $\sum_{d=1}^h \bar{B}_{h=d}^{(n)}$ , let  $L_2$  be the trellis length of the CC and note that  $A_{d,h}^{[2]} \neq 0$  implies  $d \leq \mu h$  by [92, Th. A.1] for some constant  $\mu$ . Then, by applying Lemma 5.71 on  $B_d^{[1]}$  and by applying [92, Th. A.3] on  $A_{d,h}^{[2]}$ ,

$$\begin{aligned} \bar{B}_{\leq h}^{(n)} &= \sum_{d=1}^{\mu h} \frac{B_d^{[1]} A_{d,\leq h}^{[2]}}{\binom{N}{d}} \\ &\leq \sum_{d=1}^{\mu h} \theta^d \frac{\binom{N}{\lfloor d/d_f^{\text{vlc}} \rfloor}}{\binom{N}{d}} \sum_{j=0}^{\lfloor d/2 \rfloor} \binom{L_2}{j} \binom{\eta h}{d-j}, \end{aligned} \quad (5.269)$$

for some constants  $\mu$ ,  $\theta$  and  $\eta$  independent of  $n$  and  $h$ . This expression corresponds exactly to [92, eq. (7.3)]. The rest of the developments in [92, Section VII] then holds straightforwardly.

This enables to draw a few conclusions. Firstly, the ensemble threshold  $c_0$  (defined in [92]) is finite for  $d_f^{\text{vlc}} \geq 2$ , by [92, Lemma 8.5]. Secondly, it follows straightforwardly from [92, Corollary 5.2] and [92, Lemma 8.6] that

$$\sum_{h \geq 1} \bar{B}_h^{(n)} P_h = \mathcal{O}\left(N^{-\lfloor \frac{d_f^{\text{vlc}} - 1}{2} \rfloor + \epsilon}\right) \quad (5.270)$$

for  $d_f^{\text{vlc}} \geq 3$ . To conclude the proof, note the interleaving gain (5.74) on the FER follows from (5.267) and (5.270) since  $k^{(n)} = 1$ ; the interleaving gains (5.72) on the BER and (5.73) on the  $\text{SER}_L$  follow since  $k^{(n)} = \mathcal{O}(N^{-1})$ . ■

**Proof of Theorem 5.54**<sup>18</sup>. The interleaving gain (5.74) on the FER follows from Lemma 5.72 for  $d_f^{\text{vlc}} \geq 3$  and from the inequality  $\text{FER} \leq 1$  for  $d_f^{\text{vlc}} = 2$ . Let us therefore focus on the interleaving gains (5.72) on the BER and (5.73) on the  $\text{SER}_L$ . Though we only need to prove them for  $d_f^{\text{vlc}} = 2$  thanks to Lemma 5.72, the proof hereafter holds for  $d_f^{\text{vlc}} \geq 2$ .

Recall the notations and the intermediate results from the proof of Lemma 5.72. Besides, let  $H$  be the number of bit errors among the coded bits and let  $D_n$  be, as in [92], a fixed sequence of integers satisfying  $D_n/n^\epsilon \rightarrow 0$  and  $\log(n)/D_n \rightarrow 0$  as  $n \rightarrow +\infty$ , for all  $\epsilon > 0$ . For example,  $D_n = \log^2(n)$ .

Let us first develop an upper bound on the FER. This bound will then be used to tackle the BER and the  $\text{SER}_L$ . By definition, the FER is equal to  $P(H \geq 1) = P(1 \leq H \leq D_n) + P(H > D_n)$ , where the term  $P(H > D_n)$  can be upper bounded by the union bound  $\sum_{h>D_n} \bar{B}_h^{(n)} P_h$ . Since the ensemble threshold  $c_0$  is finite for  $d_f^{\text{vlc}} \geq 2$  (see proof of Lemma 5.72), it follows from the proof of [92, Theorem 5.1] that there exist an integer  $n_0$  and constants  $K, \delta > 0$  such that  $\sum_{h>D_n} \bar{B}_h^{(n)} P_h \leq K e^{-\delta D_n}$  for  $n > n_0$ , and thus

$$\text{FER}^{(n)} \leq P(1 \leq H \leq D_n) + K e^{-\delta D_n} \quad (5.271)$$

where “ $^{(n)}$ ” recalls the dependency on  $n = N/r_{cc}$ .

Let  $W$  be the number of bit errors among the VLC bits. Note that  $A_{w,h}^{[2]} \neq 0$  implies  $w \leq \mu h$  by [92, Th. A.1] for some constant  $\mu$ . Therefore,  $H \leq D_n$  implies  $W \leq \mu D_n$  and

$$\begin{aligned} \text{BER}^{(n)} &\leq (N_b^{\min})^{-1} \mu D_n P(1 \leq H \leq D_n) + K e^{-\delta D_n} \\ &\leq k^{(n)} \mu D_n \text{FER}^{(n)} + K e^{-\delta D_n} \end{aligned} \quad (5.272)$$

<sup>18</sup>Though Theorem 5.54 seems to follow from (5.270), this is not the case — at least, this is not straightforward — because (5.270) has been proved in [92] only for  $d_f^{\text{vlc}} \geq 3$ . More precisely, in the proof of [92, Lemma 8.6], the expression

$$\sum_{d=d_f^{\text{vlc}}}^{\mu D_n} \Theta^d n^{\lfloor d/d_f^{\text{vlc}} \rfloor - \lceil d/2 \rceil} D_n^{d + \lceil d/2 \rceil} = \mathcal{O}(n^{-\lfloor (d_f^{\text{vlc}} - 1)/2 \rfloor + \epsilon})$$

does not hold for  $d_f^{\text{vlc}} = 2$ . Indeed, when  $d_f^{\text{vlc}} = 2$ , the exponent of  $n$  equals 0 on the left-hand side for even values of  $d$ . Thus the largest term of the summation is the term with  $d = 2 \lfloor \mu D_n / 2 \rfloor$ . Therefore, the whole expression is  $\mathcal{O}(D_n^{3\mu D_n / 2})$ , instead of  $\mathcal{O}(n^\epsilon)$ .

where  $(N_b^{\min})^{-1} \leq k^{(n)} = \mathcal{O}(N^{-1})$  and  $D_n = \mathcal{O}(n^\epsilon) = \mathcal{O}(N^\epsilon)$  for any  $\epsilon > 0$ . The interleaving gain (5.72) on the BER then follows from the interleaving gain (5.74) on the FER, which we have already proved for  $d_f^{\text{vlc}} \geq 2$ . Note that  $e^{-\delta D_n} = \mathcal{O}(n^{-k})$  for any  $k > 0$  since  $n^{-k} = e^{-k \log(n)}$  and  $\log(n)/D_n \rightarrow 0$ .

Proving the interleaving gain (5.73) on the  $\text{SER}_L$  is a bit trickier but can be done similarly. Let  $S_L$  be the number of symbol errors with the Levenshtein distance. Then,

$$\begin{aligned} \text{SER}_L^{(n)} &\leq \left( (N_s^{\min})^{-1} s^{(n)} P(S_L \leq s^{(n)} | W \leq \mu D_n) \right. \\ &\quad \left. + P(S_L > s^{(n)} | W \leq \mu D_n) \right) \\ &\quad \times P(1 \leq H \leq D_n) + K e^{-\delta D_n} \end{aligned} \quad (5.273)$$

for any parameter  $s^{(n)}$ , since  $H \leq D_n$  implies  $W \leq \mu D_n$ . To simplify this expression, note that  $P(S_L \leq s^{(n)} | W \leq \mu D_n) \leq 1$ ,  $P(1 \leq H \leq D_n) \leq \text{FER}^{(n)}$  and  $(N_s^{\min})^{-1} \leq k^{(n)}$ . To simplify it further, assume temporarily that, for the sequence  $s^{(n)}$  defined as

$$s^{(n)} \triangleq (1 + D_n) \mu D_n k \frac{l_{\max}}{l_{\min}} = \mathcal{O}(D_n^2)$$

where  $k$  is a constant, the probability  $P(S_L > s^{(n)} | W \leq \mu D_n)$  decreases as

$$P(S_L > s^{(n)} | W \leq \mu D_n) = \mathcal{O}(e^{-\delta' D_n}) \quad (5.274)$$

for some constant  $\delta' > 0$ . Then, the upper bound (5.273) becomes

$$\text{SER}_L^{(n)} \leq (k^{(n)} \mathcal{O}(D_n^2) + \mathcal{O}(e^{-\delta' D_n})) \text{FER}^{(n)} + K e^{-\delta D_n}$$

where, again,  $k^{(n)} = \mathcal{O}(N^{-1})$ ,  $D_n = \mathcal{O}(n^\epsilon)$  for any  $\epsilon > 0$ , and  $e^{-\delta D_n} = \mathcal{O}(N^{-k})$  for any  $k$ . This upper bound proves the interleaving gain (5.73) on the  $\text{SER}_L$  since the term between brackets is  $\mathcal{O}(N^{-1+2\epsilon})$  for any  $\epsilon > 0$ .

So the only thing left to prove is (5.274). To that end, let  $s_{1:k}^s$  be a synchronizing sequence, which always exists by assumption. Let us define an  $s^s$ -free sequence as a sequence that does not contain  $s_{1:k}^s$ ; note that an  $s^s$ -free sequence can contain a sub-sequence of  $s_{1:k}^s$  but it cannot contain an occurrence of  $s_{1:k}^s$  entirely. Let then  $\mathcal{P}^{(n)}$  be the set of VLC block realizations that contain at least one  $s^s$ -free sequence of at least  $D_n k$  symbols long. At last, let  $\overline{\mathcal{P}}^{(n)}$  be the complement of  $\mathcal{P}^{(n)}$ .

Let us first investigate the set  $\overline{\mathcal{P}}^{(n)}$ . By construction, the longest  $s_{1:k}^s$ -free sequence has a symbol length of at most  $D_n k - 1$ . Thus  $\overline{\mathcal{P}}^{(n)}$  is the set of VLC block realizations such that the occurrences of the synchronizing sequence  $s_{1:k}^s$  are separated by a gap of at most  $D_n k - 1 - 2(k - 1) = D_n k - k + (1 - k) \leq D_n k - k$  symbols — consider the longest  $s_{1:k}^s$ -free sequence (of length  $D_n k - 1$ ) starts right after the first symbol of an occurrence of  $s_{1:k}^s$  and ends right before the last symbol of the next occurrence. Let us consider one such VLC block realization and  $W \leq \mu D_n$  bit errors. These bit errors generate one or several error events. But the total length of these error events cannot be arbitrarily large. Indeed, on the one hand, the decoder is forced to resynchronize each time it receives one occurrence of  $s_{1:k}^s$  error-free. On the other hand, the VLC block realization contains many error-free occurrences (for sufficiently large  $N$ ) since at most  $W \leq \mu D_n$  occurrences (among  $\mathcal{O}(N/(kD_n))$  occurrences) are affected by bit errors. Straightforwardly, the worst-case scenario is as follows: the  $W$  bit errors generate  $t = \lfloor W/d_f^{\text{vlc}} \rfloor$  error events  $e_1, e_2, \dots, e_t$ ; the event  $e_i$  is made up of  $m_i$  consecutive occurrences of  $s_{1:k}^s$  (separated by a gap of at most  $D_n k - k$  symbols), each occurrence being affected by exactly one bit error, subject to  $\sum_{i=1}^t m_i = W$ ; each event ends with one error-free occurrence  $s_{1:k}^s$  (to resynchronize the decoder). Since the error event  $e_i$  involves  $m_i + 1$  consecutive occurrences of  $s_{1:k}^s$  and since the occurrences of  $s_{1:k}^s$  are separated by at most  $D_n k - k$  symbols, the symbol length of  $e_i$  is upper bounded by  $(m_i + 1)k + m_i(D_n k - k) = m_i D_n k + k$  and thus the total symbol length of the  $t$  error events by  $W D_n k + t k \leq (1 + D_n) W k$ . Since  $W \leq \mu D_n$  and since the Levenshtein distance satisfies  $d_{S_L}(a, b) \leq \max\{l_S(a), l_S(b)\} \leq \frac{l_{\max}}{l_{\min}} \min\{l_S(a), l_S(b)\}$  when  $l(a) = l(b)$ , the number of Levenshtein symbol errors is upper bounded by  $S_L \leq (1 + D_n) \mu D_n k \frac{l_{\max}}{l_{\min}} = s^{(n)}$ .

This shows that, for  $n = N/r_{cc}$  large enough, no sequence in  $\overline{\mathcal{P}}^{(n)}$  can lead to  $S_L > s^{(n)}$  given at most  $W \leq \mu D_n$  bit errors. In other words, all sequences that can lead to  $S_L > s^{(n)}$  given  $W \leq \mu D_n$  are included in the set  $\mathcal{P}^{(n)}$ . Therefore,

$$P(S_L > s^{(n)} | W \leq \mu D_n) \leq P(\mathcal{P}^{(n)}). \quad (5.275)$$

To conclude the proof of (5.274), let us show that  $P(\mathcal{P}^{(n)}) = \mathcal{O}(e^{-\delta' D_n})$  for some constant  $\delta'$ . Recall that all realizations in  $\mathcal{P}^{(n)}$  contain an  $s_{1:k}^s$ -free sequence of at least  $D_n k$  symbols. On the one hand, the probability of starting,

at a given symbol position, an  $s^s$ -free sequence of at least  $D_n k$  symbols is trivially upper bounded by  $(1 - P(s_{1:k}^s))^{D_n}$ . On the other hand, for the framing rule  $F_s$ , there are of course at most  $N_s$  possible starting positions (this is a loose bound). Therefore, since  $N_s \leq N$ ,

$$P(\mathcal{P}^{(n)}) \leq N(1 - P(s_{1:k}^s))^{D_n}. \quad (5.276)$$

We can get straightforwardly the same upper bound with the framing rule  $F_b$  for  $N$  large enough. This shows that  $P(\mathcal{P}^{(n)}) = \mathcal{O}(c^{D_n})$  for some constant  $c < 1$ , i.e.,  $P(\mathcal{P}^{(n)}) = \mathcal{O}(e^{-\delta' D_n})$  for  $\delta' = -\log(c) > 0$ . This, together with (5.275), proves (5.274) and thus concludes the proof. ■



# Conclusions

# 6

The tandem approach has been prevailing so far in the design and the implementation of state-of-the-art communication systems. This approach separates completely the source layer and the channel layer in the communication protocol, which is supported and validated by Shannon's source and channel coding separation theorem. Unfortunately, this theorem holds only under asymptotic conditions that are rarely satisfied with today's multimedia content and mobile channels. It is therefore usually wise in practice to drop the separation and to allow at least some cross-layer cooperation between the source and channel layers, i.e., to consider joint source-channel techniques.

In this context, joint source-channel techniques based on the turbo principle look quite promising. These techniques are indeed able to improve the end-to-end distortions by effectively combining at the receiver the redundancy left, or intentionally introduced, by the source code with the non-perfect error correcting capability of the channel code. This efficiency comes besides with the possibility of keeping a good level of separation between the source layer and the channel layer, since only an iterative interface of real numbers is necessary at the receiver.

This thesis provides several significant contributions in this field for applications based on variable length codes, with notably insightful theoretical results on the robustness of these codes in turbo systems.

## Contributions

*A first contribution* is a thorough overview of the literature in the field of joint source-channel turbo techniques. Much attention has been especially devoted to make this overview unified. Based on the factor graph framework, the turbo principle has been introduced and detailed in the joint source-channel context, with different well-known source codes and several optimal soft source decoding algorithms. Some source codes raise specific difficulties, such as high decoding complexity or decoder desynchronization, for which suboptimal decoding solutions and ways to improve the robustness have been presented.

*Publication related to this contribution: [8].*

*A second contribution* is the proposal and the analysis of a transmission system that generalizes certain previous systems from the literature, with the simple addition of a repetition code. That system provides significant advantages over previous systems. For example, it can adapt easily to different levels of redundancy present in the bit stream produced by the variable length code. That kind of flexibility is especially interesting when dealing with heterogeneous sources of data, e.g., with multimedia content. Besides, when the level of redundancy is small, which is a case not extensively studied in the literature, the proposed system provides better error rates and, at the same time, a lower decoding complexity. These advantages have been successfully analyzed through interleaving gains, EXIT charts and complexity analysis. One notable result of this analysis is the discovery and the proof of better interleaving gains on the symbol error rate when reversible variable length codes are used.

*Publications related to this contribution: [9–11].*

*A third contribution* is the extension of the EXIT charts to non-uniform binary sources, so as to be able to assess the convergence of turbo receivers in the presence of such sources. Numerous applications deal indeed with non-uniform binary sources in practice and, unfortunately, a naive application of the original EXIT charts with such sources leads often to incorrect results, as it was illustrated with a basic system example. The proposed extension consists in two methods that enable to compute the EXIT charts when the bits are not uniform. Though equivalent for the prediction of convergence under

certain assumptions, these methods have different advantages and disadvantages that make them definitely complementary in practice.

*Publication related to this contribution: [13].*

*A fourth contribution* is the proof of several insightful theoretical results on the resilience of variable length codes concatenated with linear error correcting codes, assuming an optimal maximum likelihood (ML) decoder. The developed results include notably an approximate expression of the average distance spectrum of the concatenated code, performance bounds, interleaving gains, and the novel concept of bounded spectrum.

As regards the interleaving gains, it has been investigated in particular whether the variable length code can contribute to the interleaving gains, just as a convolutional code with non-catastrophic encoder would. It has been proved that it does indeed contribute if its spectrum is bounded, even when the source statistics are unknown at the decoder (ML decoder). Besides, simulation results have highlighted that it can contribute also when its spectrum is not bounded, depending on the source statistics, if a maximum a posteriori (MAP) decoder is used. At last, the concept of bounded spectrum has been proved to be closely related, under certain assumptions, to the well known concept of statistically synchronizable variable length codes, which makes it possible to reuse previously known results from the literature.

*Publications related to this contribution: [18–21].*

To sum up and conclude, this work has developed both theoretical and practical results that illustrate how significantly joint source-channel turbo techniques can improve the level of performance of applications based on variable length codes.

## Discussion and open issues

There are many interesting open issues that deserve further investigation. In particular, the theoretical results developed in Chapter 5, notably the performance bounds and the conclusions regarding the interleaving gains, are focused mainly on the ML decoder, for reasons discussed in Section 5.3.3, and on memoryless sources. Extensions to sources with memory and to the MAP decoder have been discussed in Section 5.8 but further work in this direction

is really needed to fully grasp the possibilities of joint source-channel turbo techniques with variable length codes.

It would be interesting also to extend the theoretical results to other source codes, such as arithmetic codes or quasi-arithmetic codes. Theoretically, the amount of redundancy in the bit stream generated by an arithmetic code is almost zero, making it useless to apply the turbo principle. Recently, however, it has been suggested to use a forbidden symbol in the alphabet to make the bit stream resistant to residual bit errors. This forbidden symbol introduces indeed error correcting capabilities in the bit stream that can be exploited by turbo decoders. In terms of bounded spectrum and interleaving gains, it could be possible that this forbidden symbol plays some similar role as the synchronizing sequence does with variable length codes. An interesting perspective in this context would be to extend Algorithm 5.49 to arithmetic or quasi-arithmetic codes in order to test at least whether a synchronizing sequence exists.

Another extension regarding the performance bounds concerns the union bound. The union bound is indeed at the heart of the proposed performance bounds. Unfortunately, it is well known that the union bound is tight only at moderate and high signal to noise ratios and diverges a lot at low signal to noise ratios. This was notably observed in Chapter 5 when we compared the proposed bounds with the simulation results. An interesting improvement would be to consider the Tangential Sphere Bound [113, 114] instead of the union bound to make the bounds tighter at lower signal to noise ratios.

Let us conclude this thesis with two open issues that are not specifically related to the present work but rather to the general field of joint source-channel turbo techniques.

One issue with joint source-channel turbo techniques is related to cryptography. In any communication protocol, the layer of cryptography lies between the source layer and the channel layer, and is a significant barrier to joint source-channel turbo techniques, for applications that require cryptography. Indeed, the most widely used and robust cyphers so far, such as RSA [115] and AES [116], have a very large memory, which prevents the application of the turbo principle and prevents the exchange of soft values between the source layer and the channel layer. As long as no improvement is made in that field,

the success of joint source-channel turbo techniques will be limited to applications that do not require cryptography.

Another issue is related to the *uniform error property* [111]: Joint source-channel turbo techniques do not possess generally this property — linear error correcting codes do —, i.e., the source sequences are not all protected equally. There are mainly two reasons explaining this: the MAP decoder, which favors the most probable sequences (when several sequences are equally close to the received signal), and the source code when it is not linear. So far, all contributions in the literature, including this work, have been interested in average performance results. However, it is important in practice to know also how the system performs for the worst-case scenario. This is further supported by the fact that if the communication system encounters and fails to transmit a sequence that is a worst-case scenario, retransmitting might well be useless since the retransmitted sequence is the same sequence and remains thus a worst-case scenario. Unfortunately, worst-case scenario analysis does not seem straightforward at first sight, except in trivial cases.

One such trivial case is the serial concatenation of a non-uniform memoryless binary source with a linear error correcting code. In this case, the fact that the uniform error property is not satisfied is only due to the MAP decoder and it is thus self-evident that the worst-case scenario is associated with the least probable sequences of bits. In other words, in this case, the worst-case scenario is the least probable scenario and thus almost never happens in probability.

Nevertheless, though the worst-case scenario might be the least probable scenario, what shall we do if it happens? It is certainly inadequate to neglect it just because it does not affect average performance. Besides, it is worth noting that the least probable scenario is also the most informative one (by taking the logarithm of the inverse of the probability); in other words, the information it carries can be of the utter most importance. Worst-case analysis is thus important, cannot be neglected in practice, and escape procedures must exist in the communication protocol to deal with worst-case scenarios when they happen — at least, if they are associated with dramatic performance.



# List of Publications

## Journal papers

- X. Jaspar, C. Guillemot, and L. Vandendorpe, "Joint source-channel turbo techniques for discrete-valued sources: from theory to practice," *Proc. IEEE*, vol. 95, no. 6, pp. 1345–1361, June 2007.
- X. Jaspar and L. Vandendorpe, "Joint source-channel codes based on irregular turbo codes and variable length codes," *IEEE Trans. Commun.*, accepted for publication.
- X. Jaspar and L. Vandendorpe, "EXIT charts with non-uniform binary sources," *Journal Paper*, to be submitted.
- X. Jaspar and L. Vandendorpe, "Performance bounds and distance spectra of variable length codes in turbo/concatenated systems," *Journal Paper*, to be submitted.
- X. Jaspar and L. Vandendorpe, "Impact of variable length codes on the interleaving gain of turbo systems: The concept of bounded spectrum," *Journal Paper*, to be submitted.

## Conference papers

- H. Sneessens, X. Jaspar, C. Herzet, and L. Vandendorpe, "Predictions on turbo-decoding performance for adaptive channel coding," in *Proc. Asilomar Conference on Signals, Systems, and Computers*, Monterey, USA, Oct. 2008.

- D. Van Renterghem, X. Jaspar, B. Macq, and L. Vandendorpe, "Distributed source coding with optimized irregular turbo codes," in *Proc. IEEE ICC*, Glasgow, Scotland, June 2007.
- H. Sneessens, X. Jaspar, C. Herzet, and L. Vandendorpe, "Calculating turbo-decoding performance characteristics for adaptive channel coding," in *Proc. IEEE SPAWC*, Helsinki, Finland, June 2007.
- X. Jaspar and L. Vandendorpe, "Turbo techniques for joint source-channel decoding of multimedia content," in *Proc. MOBIMEDIA-MSAN*, Alghero, Italy, Sept. 2006.
- X. Jaspar and L. Vandendorpe, "Variable length codes in turbo schemes," in *Proc. Joint Conference on Coding and Communications*, Sölden, Austria, Mar. 2006.
- X. Jaspar and L. Vandendorpe, "Design and performance analysis of joint source-channel turbo schemes with variable length codes," in *Proc. IEEE ICC*, vol. 1, Seoul, Korea, May 2005, pp. 526–530.
- A. Dejonghe, X. Jaspar, X. Wautelet, and L. Vandendorpe, "Turbo-equalization considering bit-interleaved turbo-coded modulation: Performance bounds," in *Proc. IEEE ICC*, Seoul, Korea, May 2005.
- X. Jaspar and L. Vandendorpe, "Performance and convergence analysis of joint source-channel turbo schemes with variable length codes," in *Proc. IEEE ICASSP*, vol. 3, Philadelphia, PA, USA, Mar. 2005, pp. 485–488.
- A. Dejonghe, X. Jaspar, X. Wautelet, and L. Vandendorpe, "Assessing the performance of turbo-equalized bit-interleaved turbo-coded modulation," in *Proc. IEEE SCVT*, Gent, Belgium, Nov. 2004.
- X. Jaspar and L. Vandendorpe, "New iterative decoding of variable length codes with turbo codes," in *Proc. IEEE ICC*, vol. 5, Paris, France, June 2004, pp. 2606–2610.
- X. Jaspar and L. Vandendorpe, "Three SISO modules joint source-channel turbo-decoding of variable length coded images," in *Proc. ITG SCC*, Erlangen, Germany, Jan. 2004, pp. 279–286.

# Bibliography

- [1] C. E. Shannon, "A mathematical theory of communication," *Bell System Tech. Journal*, vol. 27, pp. 379–423/623–656, July/Oct. 1948.
- [2] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo-codes," *IEEE Trans. Commun.*, vol. 44, no. 10, pp. 1261–1271, Oct. 1996.
- [3] R. Bauer and J. Hagenauer, "Symbol-by-symbol MAP decoding of variable length codes," in *Proc. ITG SCC*, Munich, Germany, Jan. 2000, pp. 111–116.
- [4] G. D. Forney, "Codes on graphs: Normal realizations," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 520–548, Feb. 2001.
- [5] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.
- [6] H.-A. Loeliger, "An introduction to factor graphs," *IEEE Signal Processing Mag.*, vol. 21, no. 1, pp. 28–41, Jan. 2004.
- [7] H.-A. Loeliger, J. Dauwels, J. Hu, S. Korl, L. Ping, and F. R. Kschischang, "The factor graph approach to model-based signal processing," *Proc. IEEE*, vol. 95, no. 6, pp. 1295–1322, June 2007.
- [8] X. Jaspard, C. Guillemot, and L. Vandendorpe, "Joint source-channel turbo techniques for discrete-valued sources: from theory to practice," *Proc. IEEE*, vol. 95, no. 6, pp. 1345–1361, June 2007.

- [9] X. Jaspar and L. Vandendorpe, "Joint source-channel codes based on irregular turbo codes and variable length codes," *IEEE Trans. Commun.*, accepted for publication.
- [10] —, "Three SISO modules joint source-channel turbo-decoding of variable length coded images," in *Proc. ITG SCC*, Erlangen, Germany, Jan. 2004, pp. 279–286.
- [11] —, "New iterative decoding of variable length codes with turbo codes," in *Proc. IEEE ICC*, vol. 5, Paris, France, June 2004, pp. 2606–2610.
- [12] S. ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Trans. Commun.*, vol. 49, no. 10, pp. 1727–1737, Oct. 2001.
- [13] X. Jaspar and L. Vandendorpe, "EXIT charts with non-uniform binary sources," *Journal Paper*, to be submitted.
- [14] V. Buttigieg, "Variable-length error-correcting codes," Ph.D. dissertation, Department of Electrical Engineering, University of Manchester, England, 1995.
- [15] S. Benedetto and G. Montorsi, "Unveiling turbo codes: some results on parallel concatenated coding schemes," *IEEE Trans. Inform. Theory*, vol. 42, no. 2, pp. 409–428, Mar. 1996.
- [16] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Serial concatenation of interleaved codes: Performance analysis, design and iterative decoding," *IEEE Trans. Inform. Theory*, vol. 44, no. 3, pp. 909–926, May 1998.
- [17] R. M. Capocelli, L. Gargano, and U. Vaccaro, "On the characterization of statistically synchronizable variable-length codes," *IEEE Trans. Inform. Theory*, vol. 34, no. 4, pp. 817–825, July 1988.
- [18] X. Jaspar and L. Vandendorpe, "Performance bounds and distance spectra of variable length codes in turbo/concatenated systems," *Journal Paper*, to be submitted.

- [19] —, “Impact of variable length codes on the interleaving gain of turbo systems: The concept of bounded spectrum,” *Journal Paper*, to be submitted.
- [20] —, “Design and performance analysis of joint source-channel turbo schemes with variable length codes,” in *Proc. IEEE ICC*, vol. 1, Seoul, Korea, May 2005, pp. 526–530.
- [21] —, “Performance and convergence analysis of joint source-channel turbo schemes with variable length codes,” in *Proc. IEEE ICASSP*, vol. 3, Philadelphia, PA, USA, Mar. 2005, pp. 485–488.
- [22] R. Perkert, M. Kaindl, and T. Hindelang, “Iterative source and channel decoding for GSM,” in *Proc. IEEE ICASSP*, Salt Lake City, UT, USA, May 2001.
- [23] J. Hagenauer and R. Bauer, “The turbo principle in joint source channel decoding of variable length codes,” in *Proc. IEEE ITW*, Cairns, Australia, Sept. 2001, pp. 128–130.
- [24] L. Guivarch, J.-C. Carlach, and P. Siohan, “Joint source-channel soft-decoding of Huffman codes with turbo-codes,” in *Proc. IEEE DCC*, Snowbird, USA, Mar. 2000, pp. 88–92.
- [25] Z. Peng, Y.-F. Huang, and D. J. Costello, “Turbo codes for image transmission - a joint channel and source decoding approach,” *IEEE J. Select. Areas Commun.*, vol. 6, pp. 868–879, June 2000.
- [26] A. Guyader, E. Fabre, C. Guillemot, and M. Robert, “Joint source-channel turbo decoding of entropy-coded sources,” *IEEE J. Select. Areas Commun.*, vol. 19, pp. 1680–1696, Sept. 2001.
- [27] N. Görtz, “On the iterative approximation of optimal joint source-channel decoding,” *IEEE J. Select. Areas Commun.*, vol. 19, pp. 1662–1670, Sept. 2001.
- [28] J. Garcia-Frias and J. D. Villasenor, “Joint turbo decoding and estimation of hidden Markov sources,” *IEEE J. Select. Areas Commun.*, vol. 19, pp. 1671–1679, Sept. 2001.

- [29] A. Hedayat and A. Nosratinia, "List decoding of variable length codes with application in joint source channel coding," in *Asilomar Conf. on Sign., Syst. and Comp.*, Nov. 2002.
- [30] K. Laković and J. Villasenor, "Combining variable length codes and turbo codes," in *Proc. IEEE VTC*, Birmingham, USA, May 2002, pp. 1719–1723.
- [31] R. Thobaben and J. Kliewer, "On iterative source-channel decoding for variable-length encoded markov sources using a bit-level trellis," in *Proc. IEEE SPAWC*, Rome, Italy, June 2003, pp. 50–54.
- [32] N. Görtz, "Optimization of bit mappings for iterative source-channel decoding," in *Proc. Int. Symp. on Turbo Codes and Related Topics*, Brest, France, Sept. 2003.
- [33] J. Kliewer and R. Thobaben, "Parallel concatenated joint source-channel coding," *IEE Elect. Letters*, vol. 39, no. 23, pp. 1664–1666, Nov. 2003.
- [34] T. Guionnet and C. Guillemot, "Soft decoding and synchronization of arithmetic codes for image transmission over error-prone channels," *IEEE Trans. Image Processing*, vol. 12, no. 12, pp. 1599–1609, Dec. 2003.
- [35] —, "Soft and joint source-channel decoding of quasi-arithmetic codes," *Eurasip Journal on Applied Signal Processing*, Mar. 2004.
- [36] M. Grangetto, B. Scanavino, and G. Olmo, "Joint source-channel iterative decoding of arithmetic codes," in *Proc. IEEE ICC*, vol. 2, Paris, France, June 2004, pp. 886–890.
- [37] G.-C. Zhu, F. Alajaji, J. Bajcsy, and P. Mitran, "Transmission of nonuniform memoryless sources via nonsystematic turbo codes," *IEEE Trans. Commun.*, vol. 8, no. 52, pp. 1344–1354, Aug. 2004.
- [38] H. Nguyen and P. Duhamel, "Iterative joint source-channel decoding of variable length encoded video sequences exploiting source semantics," in *Proc. IEEE ICIP*, Singapor, Oct. 2004, pp. 3221–3224.
- [39] M. Adrat and P. Vary, "Iterative source-channel decoding: Improved system design using EXIT charts," *Eurasip Journal on Applied Signal Processing*, no. 6, pp. 928–947, May 2005.

- [40] C. Poulliat, D. Declercq, C. Lamy-Bergot, and I. Fijalkow, "Analysis and optimization of irregular LDPC codes for joint source-channel decoding," *IEEE Commun. Lett.*, vol. 9, pp. 1064–1066, Dec. 2005.
- [41] G.-C. Zhu and F. Alajaji, "Joint source-channel turbo coding for binary markov sources," *IEEE Trans. Wireless Commun.*, vol. 5, no. 5, pp. 1065–1075, May 2006.
- [42] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. 20, pp. 284–287, Mar. 1974.
- [43] K. Sayood and J. Borkenhagen, "Use of residual redundancy in the design of joint source/channel coders," *IEEE Trans. Commun.*, vol. 39, no. 6, pp. 838–846, June 1991.
- [44] D. J. Miller and M. Park, "A sequence-based approximate MMSE decoder for source coding over noisy channels using discrete hidden Markov models," *IEEE Trans. Commun.*, vol. 46, no. 2, pp. 222–231, Feb. 1998.
- [45] A. H. Murad and T. E. Fuja, "Robust transmission of variable-length encoded sources," in *Proc. IEEE WCNC*, New Orleans, L.A., USA, Sept. 1999.
- [46] K. Laković, J. Villasenor, and R. Wesel, "Robust joint huffman and convolutional decoding," in *Proc. IEEE VTC*, Amsterdam, The Netherlands, Sept. 1999, pp. 2551–2555.
- [47] C. Boyd, J. G. Cleary, S. A. Irvine, I. Rinsma-Melchert, and I. H. Witten, "Integrating error detection into arithmetic coding," *IEEE Trans. Commun.*, vol. 45, no. 1, pp. 1–3, Jan. 1997.
- [48] Y. Takishima, M. Wada, and H. Murakami, "Reversible variable length codes," *IEEE Trans. Commun.*, vol. 43, no. 2/3/4, pp. 158–162, Feb.-Apr. 1995.
- [49] K. Laković and J. Villasenor, "On design of error-correcting reversible variable length codes," *IEEE Commun. Lett.*, vol. 6, no. 8, pp. 337–339, Aug. 2002.

- [50] H. Jégou and C. Guillemot, "Robust multiplexed codes for compression of heterogeneous data," *IEEE Trans. Inform. Theory*, vol. 51, no. 4, pp. 1393–1403, Apr. 2005.
- [51] —, "Error-resilient first-order multiplexed source codes: performance bounds, design and decoding algorithms," *IEEE Trans. Signal Processing*, vol. 54, no. 4, pp. 1483–1493, Apr. 2006.
- [52] D. Forney, G., "The viterbi algorithm," *Proc. IEEE*, vol. 61, pp. 268–278, Mar. 1973.
- [53] A. Ashikhmin, G. Kramer, and S. ten Brink, "Extrinsic information transfer functions: Model and erasure channel properties," *IEEE Trans. Inform. Theory*, vol. 50, no. 11, pp. 2657–2673, Nov. 2004.
- [54] D. A. Huffman, "A method for the construction of minimum-redundancy codes," in *Proc. IRE*, vol. 40, 1952, pp. 1098–1101.
- [55] V. B. Balakirsky, "Joint source-channel coding with variable length codes," in *Proc. IEEE ISIT*, Ulm, Germany, July 1997, p. 491.
- [56] J. J. Rissanen, "Generalized kraft inequality and arithmetic coding," *IBM J. Res. Develop.*, vol. 20, pp. 198–203, May 1976.
- [57] P. G. Howard and J. S. Vitter, *Image and Text Compression*. Kluwer Academic Publisher, 1992, pp. 85–112.
- [58] —, "Design and analysis of fast text compression based on quasi-arithmetic coding," in *Proc. IEEE DCC*, Snowbird, Utah, Mar. 1993, pp. 98–107.
- [59] S. K. M. Bystrom and A. Kopansky, "Soft source decoding with applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 10, pp. 1108–1120, Oct. 2001.
- [60] H. Jégou, S. Malinowski, and C. Guillemot, "Trellis state aggregation for soft decoding of variable length codes," in *Proc. IEEE workshop on signal processing systems, SIPS*, Nov. 2005.
- [61] G. Mohammad-Khani, C.-M. Lee, M. Kieffer, and P. Duhamel, "Simplification of VLC tables with application to ML and MAP decoding algorithms," *IEEE Trans. Commun.*, vol. 54, pp. 1835–1844, Oct. 2006.

- [62] J. Anderson and S. Mohan, "Sequential coding algorithms: A survey and cost analysis," *IEEE Trans. Commun.*, vol. 32, pp. 169–176, Feb. 1984.
- [63] S. J. Simmons, "Breadth-first trellis decoding with adaptive effort," *IEEE Trans. Commun.*, vol. 38, pp. 3–12, Jan. 1990.
- [64] F. Jelinek, "Fast sequential decoding algorithm using a stack," *IBM Journal Research and Devel.*, vol. 13, pp. 675–685, Nov. 1969.
- [65] R. M. Fano, "A heuristic discussion of probabilistic decoding," *IEEE Trans. Inform. Theory*, vol. 9, pp. 64–74, Apr. 1963.
- [66] J. Hagenauer and C. Kuhn, "The List-Sequential (LISS) algorithm and its application," *IEEE Trans. Commun.*, vol. 55, pp. 918–928, May 2007.
- [67] B. Pettijohn, M. Hoffman, and K. Sayood, "Joint source/channel coding using arithmetic codes," *IEEE Trans. Commun.*, vol. 49, no. 5, May 2001.
- [68] S. Ben-Jamaa, C. Weidmann, and M. Kieffer, "Asymptotic error-correcting performance of joint source-channel schemes based on arithmetic coding," in *Proc. IEEE MMSP*, Victoria, Canada, Oct. 2006.
- [69] J. Maxted and J. Robinson, "Error recovery for variable length codes," *IEEE Trans. Inform. Theory*, vol. IT-31, no. 6, pp. 794–801, Nov. 1985.
- [70] P. F. Swaszek and P. DiCicco, "More on the error recovery for variable length codes," *IEEE Trans. Inform. Theory*, vol. IT-41, no. 6, pp. 2064–2071, Nov. 1995.
- [71] S. Malinowski, H. Jégou, and C. Guillemot, "On the link between the synchronization recovery and soft decoding of variable length codes," in *Proc. Int. Symp. on Turbo Codes and Related Topics*, Nov. 2006.
- [72] A. Hedayat and A. Nosratinia, "Performance analysis and design criteria for finite-alphabet source-channel codes," *IEEE Trans. Commun.*, vol. 52, no. 11, pp. 1672–1879, Nov. 2004.
- [73] —, "Iterative list decoding of concatenated source-channel codes," *Eurasip Journal on Applied Signal Processing*, no. 6, pp. 954–960, 2005.

- [74] *Information technology – Generic coding of moving pictures and associated audio information – Part 7: Advanced Audio Coding (AAC)*, ISO/IEC 13818-7.
- [75] *Information technology – Coding of audio-visual objects*, ISO/IEC 14496.
- [76] C. Weidmann and P. Siohan, “Décodage conjoint source-canal avec estimation en ligne de la source,” in *Proc. CORESA*, Lyon, France, Jan. 2003, in french.
- [77] T. Guionnet, C. Guillemot, and E. Fabre, “Soft decoding of multiple descriptions,” in *IEEE ICME*, Lausanne, Switzerland, Aug. 2002.
- [78] J. Barros, J. Hagenauer, and N. Görtz, “Turbo cross decoding of multiple descriptions,” in *IEEE ICC*, New York, US, Apr. 2002.
- [79] G. Caire, S. Shamai, and S. Verdú, “Noiseless data compression with low-density parity-check codes,” in *Advances in Network Information Theory*, ser. DIMACS Series in Discrete Mathematics and Theoretical Computer Science, G. K. P. Gupta and A. J. van Wijngaarden, Eds. American Mathematical Society, 2004, vol. 66, pp. 263–284.
- [80] N. Dütsch, G. Sebastian, J. Garcia-Frias, and J. Hagenauer, “Source model aided lossless turbo source coding,” in *Proc. Int. Symp. on Turbo Codes and Related Topics*, Munich, Germany, Apr. 2006.
- [81] K. Subbalakshmi and Q. Chen, “Joint source-channel decoding for MPEG-4 coded video over wireless channels,” in *Proc. IASTED Wireless and Optical Communications*, July 2002, pp. 617–622.
- [82] L. Perros-Meilhac and C. Lamy, “Huffman tree based metric derivation for a low complexity sequential soft VLC decoding,” in *Proc. IEEE ICC*, May 2002, pp. 783–787.
- [83] A. Kopansky and M. Bystrom, “Sequential decoding of MPEG-4 coded bitstreams for error resilience,” in *Proc. Information Sciences and Systems*, Mar. 1999.
- [84] M. Jeanne, C. Guillemot, T. Guionnet, and F. Pauchet, “Error resilient decoding of context-based adaptive binary arithmetic codes,” *Signal, Image and Video Processing*, vol. 1, no. 1, pp. 77–87, Apr. 2007.

- [85] S. Khayam, S. Karande, H. Radha, and D. Loguinov, "Performance analysis and modeling of errors and losses over 802.11b lans for high-bitrate real-time multimedia," *Signal Processing: Image Communication*, vol. 18, no. 7, pp. 575–595, Aug. 2003.
- [86] R. Bauer and J. Hagenauer, "On variable length codes for iterative source/channel decoding," in *Proc. IEEE DCC*, Snowbird, USA, Mar. 2001, pp. 273–282.
- [87] J. Wang, L.-L. Yang, and L. Hanzo, "Iterative construction of reversible variable-length codes and variable-length error-correcting codes," *IEEE Commun. Lett.*, vol. 8, no. 11, pp. 671–673, Nov. 2004.
- [88] M. Jeanne, J.-C. Carlach, and P. Siohan, "Joint source-channel decoding of variable-length codes for convolutional codes and turbo codes," *IEEE Trans. Commun.*, vol. 53, pp. 10–15, Jan. 2005.
- [89] B. J. Frey and D. J. C. MacKay, "Irregular turbocodes," in *Proc. IEEE ISIT*, Sorrento, Italy, June 2000.
- [90] D. Divsalar, S. Dolinar, and C. Jones, "Construction of protograph LDPC codes with linear minimum distance," in *Proc. IEEE ISIT*, Seattle, USA, July 2006, pp. 664–668.
- [91] D. Divsalar, H. Jin, and R. McEliece, "Coding theorems for "Turbo-Like" codes," in *Proc. Allerton Conf. Communications, Control and Computing*, Monticello, USA, Sept. 1998, pp. 201–210.
- [92] H. Jin and R. McEliece, "Coding theorems for turbo code ensembles," *IEEE Trans. Inform. Theory*, vol. 48, no. 6, pp. 1451–1461, June 2002.
- [93] M. Tüchler, "Convergence prediction for iterative decoding of threefold concatenated systems," in *Proc. IEEE GLOBECOM*, vol. 2, Taipei, Taiwan, Nov. 2002, pp. 1358–1362.
- [94] —, "Design of serially concatenated systems depending on the block length," *IEEE Trans. Commun.*, vol. 52, pp. 209–218, Feb. 2004.
- [95] E. Y. Lam and J. W. Goodman, "A mathematical analysis of the DCT coefficients distributions for images," *IEEE Trans. Image Processing*, vol. 9, no. 10, pp. 1661–1666, Oct. 2000.

- [96] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Analysis, design, and iterative decoding of double serially concatenated codes with interleavers," *IEEE J. Select. Areas Commun.*, vol. 16, no. 2, pp. 231–244, Feb. 1998.
- [97] N. Dütsch, "Code optimisation for lossless compression of binary memoryless sources based on fec codes," *European Transactions on Telecommunications*, vol. 17, no. 2, pp. 219–229, Mar./Apr. 2006.
- [98] M. Tüchler and J. Hagenauer, "EXIT charts of irregular codes," in *Proc. Conf. on Information Sciences and Systems*, Mar. 2002.
- [99] I. Land, P. A. Hoeher, and S. Gligorević, "Computation of symbol-wise mutual information in transmission systems with LogAPP decoders and applications to EXIT charts," in *Proc. ITG SCC*, Erlangen, Germany, Jan. 2004, pp. 195–202.
- [100] P. A. Hoeher, I. Land, and U. Sorger, "Log-likelihood values and Monte Carlo simulation - some fundamental results," in *Proc. Int. Symp. on Turbo Codes and Related Topics*, Brest, France, Sept. 2000.
- [101] J. Hagenauer, "The EXIT chart - introduction to extrinsic information transfer in iterative processing," in *Proc. EUSIPCO*, Vienna, Austria, Sept. 2004.
- [102] S. ten Brink, G. Kramer, and A. Ashikhmin, "Design of low-density parity-check codes for modulation and detection," *IEEE Trans. Commun.*, vol. 52, no. 4, pp. 670–678, Apr. 2004.
- [103] M. Lentmaier, D. V. Truhachev, K. S. Zigangirov, and D. J. Costello, "An analysis of the block error probability performance of iterative decoding," *IEEE Trans. Inform. Theory*, vol. 51, pp. 3834–3855, Nov. 2005.
- [104] D. E. Knuth, "Big Omicron and big Omega and big Theta," *ACM SIGACT News*, vol. 8, no. 2, pp. 18–24, 1976.
- [105] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Soviet Physics Doklady*, vol. 10, pp. 707–710, 1966.

- [106] N. Kahale and R. Urbanke, "On the minimum distance of parallel and serially concatenated codes," *IEEE Trans. Inform. Theory*, Submitted for publication.
- [107] F. J. Damerau, "A technique for computer detection and correction of spelling errors," *Communications of the ACM*, vol. 7, pp. 171–176, Mar. 1964.
- [108] V. Buttigieg, "Two algorithms to calculate the distance spectrum of VLEC codes," Department of Electrical Engineering, University of Manchester, England, Internal Report pp. 1–15, Apr. 1994.
- [109] X. Jaspar, "Vlc synchronizing sequence test algorithm," Laboratoire de Télécommunications et Télédetection, Université catholique de Louvain, Belgium, <http://www.tele.ucl.ac.be/digicom/jaspar/software/vlcsynchro/> (see also <http://staticurl.xjaspar.net/research/software/vlcsynchro/>).
- [110] C. F. Freiling, F. Théberge, and K. Zeger, "Almost all complete binary prefix codes have a self-synchronizing string," *IEEE Trans. Inform. Theory*, vol. 49, no. 9, pp. 2219–2225, Sept. 2003.
- [111] S. Benedetto and E. Biglieri, *Principles of Digital Transmission: With Wireless Applications*, ser. Information Technology: Transmission, Processing and Storage. Springer, 1999.
- [112] S. M. Ross, *Stochastic processes second edition*. John Wiley and sons, 1996.
- [113] G. Poltyrev, "Bounds on the decoding error probability of binary linear codes via their spectra," *IEEE Trans. Inform. Theory*, vol. 40, no. 4, pp. 1284–1292, July 1994.
- [114] I. Sason and S. Shamai, "Improved upper bounds on the ML decoding error probability of parallel and serial concatenated turbo codes via their ensemble distance spectrum," *IEEE Trans. Inform. Theory*, vol. 46, no. 1, pp. 24–47, Jan. 2000.
- [115] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978.

- [116] *Advanced encryption standard (AES) (FIPS PUB 197)*, National Institute of Standards and Technology (NIST) Std.