國立交通大學

電信工程研究所

碩士論文

針對非均勻訊源之固定長度整合訊源與通道編碼系統之設計

Fixed-Length Joint Source-Channel Coding System for Generally Non-uniform Sources

研究生：林鉑涵

指導教授：陳伯寧　教授

中華民國一〇二年七月

針對非均勻訊源之固定長度整合訊源與通道編碼系統之設計

Fixed-Length Joint Source-Channel Coding System for Generally Non-uniform Sources

研 究 生：林鉑涵　　Student：Po-Han Lin

指導教授：陳伯寧　　Advisor：Po-Ning Chen

國 立 交 通 大 學

電信工程研究所

碩 士 論 文

A Thesis

Submitted to Institute of Communication Engineering

College of Electrical and Computer Engineering

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Communication Engineering

July 2013

Hsinchu, Taiwan, Republic of China

中華民國一〇二年七月

針對非均勻訊源之固定長度整合訊源與通道編碼系統之設計

學生: 林鉑涵　　指導教授: 陳伯寧

國立交通大學電信工程研究所碩士班

摘要

固定長度整合訊源與通道編碼(FLEC)爲一個將訊源編碼與通道編碼結合的系統。在這篇碩士論文中，我們提出了兩個可用於非均勻訊源(non-uniform source)的FLEC的設計方法。第一個方法基於所推導的FLEC的聯集錯誤率上界(union bound)，設計使此上界相對較小的FLEC。由於第一個方法僅能適用於碼長較短的FLEC設計，我們因此再提出第二個方法。第二個方法基於渦輪碼(turbo code)的架構，重新設計可適用於非均勻訊源的渦輪碼的解碼量度。模擬結果顯示我們所提出的第一個方法，系統錯誤率會比傳統結合赫夫曼(Huffman)來源編碼與BCH碼的分離式設計低。而第二個方法的效能比結合赫夫曼(Huffman)來源編碼與渦輪碼的分離式設計好。

Fixed-Length Joint Source-Channel Coding System for Generally Non-uniform Sources

Student: Po-Han Lin Advisor: Po-Ning Chen

Institute of Communications Engineering

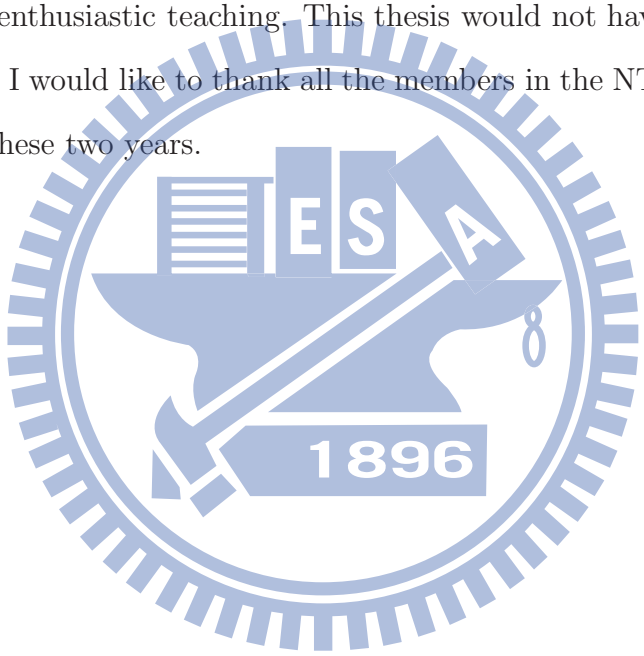
National Chiao Tung University

Abstract

In this thesis, the design of fixed-length joint source-channel error-correcting codes (FLEC) for generally non-uniform source statistics is considered as contrary to the usual variable-length joint source-channel error-correcting coding system. Such a system has the advantage that the receiver can identify easily the codeword margin via a length counter. Two different approaches are attempted. We first derive the union bounds of decoding errors of the FLECs for generally non-uniform sources, and then find the FLECs that have acceptably good union bound values. Since the first approach is only suitable for FLECs of short block length, the second approach assumes the turbo code structure and modifies the turbo decoding metrics to adapt to the tranceiving of non-uniform information. Simulations show that the first proposed approach outperforms the traditional tandem scheme that concatenates the Huffman source code with a BCH code, while the second proposed approach beats the concatenation of the Huffman source code with a turbo code of similar rate.

Acknowledgements

First of all, I would like to express my gratitude to my advisor, Professor Po-Ning Chen, for his patient guidance and support. Secondly, I would like to show my special gratitude to Dr. Ting-Yi Wu for his enthusiastic teaching. This thesis would not have been possible without these helps. Finally, I would like to thank all the members in the NTL lab and all those who have helped me in these two years.



Contents

List of Figures

Chapter 1

Introduction

The underlying aim of a communication system is to transmit data from the source to the destination over possibly noisy channels. From the aspect of engineering as well as research, the ultimate goal is to find approaches to transmit data more efficiently and meanwhile more reliably.

In leu of transmission efficacy, the procedure of *source coding* (more often be referred to as *data compression*) is employed in communication systems to reduce as much redundancy as possible. In a sense, the source coding shortened the information sequence (either distortionlessly or with an acceptable degree of distortion); hence, the efficiency is improved. In reality, there often exist noise and interference when transmitting data over a medium such that the receiver might not ensure the correctness of the received data. So in order to enhance transmission reliability, the channel coding is introduced. As a contrary, channel coding commonly adds redundancy to protect the data from corruption.

In 1948 , Shannon [1] has proved that there exist a source coding scheme and separately a channel coding scheme such that the data can be transmuted at a rate approaching the theoretical transmission limit. Ever since the communication system usually treats the source coding and channel coding separately for their design convenience. A typical digital

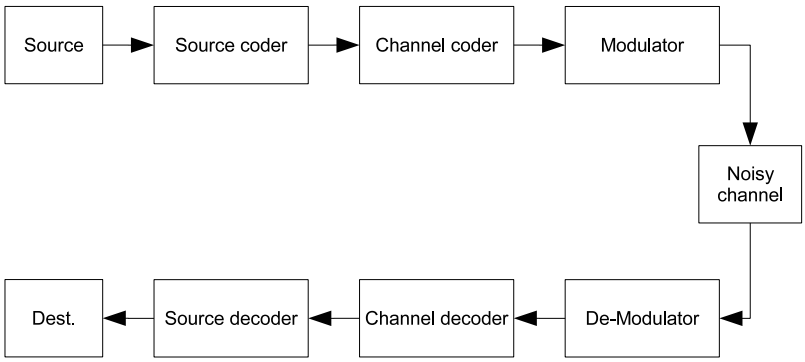


Figure 1.1: Block diagram of a typical digital communication system with separate source and channel coding.

communication system is depicted in Fig. 1.1.

One of the theoretical requirement for achieving the capacity by separate design is the the length of information sequence should approach infinity, which in a sense implies that the communication delay and decoding complexity might go to infinity. However, in reality, we cannot tolerate infinite delay and complexity, and hence under the practical constraints of finite delay and limited complexity, the joint source-channel coding system might outperform the traditional separate one [2].

Along this research direction, the authors in [3] derive the union bound for the so-called joint source-channel variable-length error-correcting (VLEC) codes , and propose a method to construct the VLEC code that possibly minimizes this bound. However, there are some possible drawbacks for a VLEC: e.g., *i*) its varying decoding delay, *ii*) error propagation, and *iii*) an extra mechanism may be needed for the receiver to identify the end of a codeword. For this reason, we will focus on the joint source-channel fixed-length error-correcting code (FLEC) design in this thesis.

Two different approaches to design FLECs will be discussed in this thesis. The first one

is to derive the union bound for the error rate of FLECs as parallel to [3], and determine the best FLEC that possibly minimizes this bound. The second approach is to presume the turbo coding structure and devises a joint source-channel turbo coding scheme that suits the transceiving of non-uniform sources.

There have been some publications taking the second approach to design FLECs. In [4] and [5], the authors provides encoding and decoding schemes for binary non-uniform sources and binary Markov sources assuming the turbo coding structure. In this thesis, we extend their results to non-binary non-uniform sources. Details will be given in Chapter 4.

The remainder of the thesis is organized as follows. Chapter 2 introduces the notations we use in this thesis and provides the necessary background for the *maximum a posteriori* (MAP) metric and turbo coding. Chapter 3 derives the union bound of the FLEC, followed by the construction of the FLEC that gives an acceptably low union bound low. Chapter 4 devises the modified turbo coder for n-ary non-uniform sources. Chapter 5 presents and remarks on numerical and simulation results, and Chapter 6 concludes the thesis.

Chapter 2

Preliminaries

2.1 Codeword and Codebook

In the joint source-channel coding system considered in this thesis, there are n symbols $\{s_1, s_2, \ldots, s_n\}$ in the source alphabet with probabilities of transmission $\{p_1, p_2, \ldots, p_n\}$, respectively. The ith source symbol s_i will be mapped to a codeword $\boldsymbol{c}_i = (c_{i,1}, c_{i,2}, \ldots, c_{i,\ell})$ of fixed length ℓ, where $c_{i,j} \in \{0, 1\}$ for every i, j. The code book $\mathcal{C}$ can therefore be represented as a matrix below:

$$\mathcal{C} \triangleq \begin{bmatrix} \boldsymbol{c}_1 \\ \boldsymbol{c}_2 \\ \vdots \\ \boldsymbol{c}_n \end{bmatrix} = \begin{bmatrix} c_{1,1} & c_{1,2} & \ldots & c_{\ell,l} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,\ell} \\ \vdots & \vdots & & \vdots \\ c_{n,1} & c_{n,2} & \ldots & c_{n,\ell} \end{bmatrix}.$$

2.2 MAP Decoding Criterion

Denoting by $\boldsymbol{r} = (r_1, r_2, \ldots, r_\ell)$ the received vector corresponding to the transmission of codewords through the additive white Gaussian noise (AWGN) channel, we present the respective MAP decoding criterion as follows.

2.2.1 Hard-Decision Decoding

Define the hard-decision sequence $\boldsymbol{y} = (y_1, y_2, \ldots, y_\ell)$ corresponding to received vector $\boldsymbol{r} = (r_1, r_2, \ldots, r_\ell)$ by

$$y_i = \begin{cases} 0, & \text{if } r_i > 0 \\ 1, & \text{otherwise} \end{cases}$$

Based upon $\boldsymbol{y}$, the *maximum a posteriori* (MAP) hard-decision decoding is to find the codeword $\boldsymbol{c}_m$ such that

$$\Pr(\boldsymbol{c}_m \mid \boldsymbol{y}) \geq \Pr(\boldsymbol{c}_i \mid \boldsymbol{y}) \quad \text{for all } 1 \leq i \leq n. \tag{2.1}$$

Equivalently, (2.1) can be transformed to:

$$\begin{aligned} &\frac{\Pr(\boldsymbol{y} \mid \boldsymbol{c}_m)\Pr(\boldsymbol{c}_m)}{\Pr(\boldsymbol{y})} \geq \frac{\Pr(\boldsymbol{y} \mid \boldsymbol{c}_i)\Pr(\boldsymbol{c}_i)}{\Pr(\boldsymbol{y})} \\ \Leftrightarrow\ &\Pr(\boldsymbol{y} \mid \boldsymbol{c}_m)\Pr(\boldsymbol{c}_m) \geq \Pr(\boldsymbol{y} \mid \boldsymbol{c}_i)\Pr(\boldsymbol{c}_i). \end{aligned} \tag{2.2}$$

Instead of using the above hard-decision decoding aspect, we can directly reduce the AWGN channel with hard-decision decoding to the binary symmetric channel (BSC) with crossover probability $b = Q\left(\sqrt{\frac{2E_bR}{N_0}}\right)$, where $Q(x) = \frac{1}{\sqrt{2\pi}}\int_x^\infty e^{-u^2/2}du$ is the Q-function, E_b is the energy per information bit, N_0 is the variance of additive Gaussian noise sample, and R is the code rate. Letting $h_i = d_{\mathrm{H}}(\boldsymbol{c}_i, \boldsymbol{y})$, where $d_{\mathrm{H}}(\cdot,\cdot)$ is the Hamming distance, we have

$$\Pr(\boldsymbol{y} \mid \boldsymbol{c}_i) = b^{h_i}(1-b)^{\ell - h_i}.$$

Criterion (2.2) can thus be re-expressed as that for all $1 \leq i \leq n$,

$$\begin{aligned} & b^{h_m}(1-b)^{\ell-h_m}\Pr(\boldsymbol{c}_m) \geq b^{h_i}(1-b)^{\ell-h_i}\Pr(\boldsymbol{c}_i) \\ \Leftrightarrow\ & \ln\left[b^{h_m}(1-b)^{\ell-h_m}\Pr(\boldsymbol{c}_m)\right] \geq \ln\left[b^{h_i}(1-b)^{\ell-h_i}\Pr(\boldsymbol{c}_i)\right] \\ \Leftrightarrow\ & h_m\ln(b) + (\ell-h_m)\ln(1-b) + \ln\Pr(\boldsymbol{c}_m) \geq h_i\ln(b) + (\ell-h_i)\ln(1-b) + \ln\Pr(\boldsymbol{c}_i) \\ \Leftrightarrow\ & h_m[\ln(b) - \ln(1-b)] + \ln\Pr(\boldsymbol{c}_m) \geq h_i[\ln(b) - \ln(1-b)] + \ln\Pr(\boldsymbol{c}_i). \end{aligned}$$

2.2.2 Soft-Decision Decoding

For a soft-decision decoding, the log-likelihood ratio ϕ_i in response to the reception of soft received value r_i is given by:

$$\phi_i = \ln \frac{\Pr(r_i \mid 0)}{\Pr(r_i \mid 1)} = \ln \frac{\frac{1}{\sqrt{\pi N_0}} e^{-(r_i-1)^2/N_0}}{\frac{1}{\sqrt{\pi N_0}} e^{-(r_i+1)^2/N_0}} = \ln e^{\left[-(r_i-1)^2+(r_i+1)^2\right]/N_0} = \frac{4}{N_0} r_i.$$

Thus, the soft maximum a posteriori decision must satisfy that for $1 \le i \le n$,

$$\begin{aligned}
&\Pr(\boldsymbol{c}_m \mid \boldsymbol{r}) \ge \Pr(\boldsymbol{c}_i \mid \boldsymbol{r}) \\
\Leftrightarrow\ & \frac{f(\boldsymbol{r} \mid \boldsymbol{c}_m)\Pr(\boldsymbol{c}_m)}{f(\boldsymbol{r})} \ge \frac{f(\boldsymbol{r} \mid \boldsymbol{c}_i)\Pr(\boldsymbol{c}_i)}{f(\boldsymbol{r})} \\
\Leftrightarrow\ & f(\boldsymbol{r} \mid \boldsymbol{c}_m)\Pr(\boldsymbol{c}_m) \ge f(\boldsymbol{r} \mid \boldsymbol{c}_i)\Pr(\boldsymbol{c}_i) \\
\Leftrightarrow\ & \prod_{j=1}^{\ell} f(r_j \mid c_{m,j})\Pr(\boldsymbol{c}_m) \ge \prod_{j=1}^{\ell} f(r_j \mid c_{i,j})\Pr(\boldsymbol{c}_i) \\
\Leftrightarrow\ & \sum_{j=1}^{\ell} \ln f(r_j \mid c_{m,j}) + \ln \Pr(\boldsymbol{c}_m) \ge \sum_{j=1}^{\ell} \ln f(r_j \mid c_{i,j}) + \ln \Pr(\boldsymbol{c}_i) \\
\Leftrightarrow\ & \sum_{j=1}^{\ell} \ln \frac{f(r_j \mid c_{m,j})}{f(r_j \mid c_{i,j})} + \ln \Pr(\boldsymbol{c}_m) \ge \ln \Pr(\boldsymbol{c}_i) \\
\Leftrightarrow\ & \sum_{j=1}^{\ell} \left[\ln \frac{f(r_j \mid c_{m,j})}{f(r_j \mid c_{i,j})} - \ln \frac{f(r_j \mid c_{i,j})}{f(r_j \mid c_{mj})} \right] + 2\ln \Pr(\boldsymbol{c}_m) \ge 2\ln \Pr(\boldsymbol{c}_i) \\
\Leftrightarrow\ & \sum_{j=1}^{\ell} \left[(-1)^{c_{m,j}} \phi_j - (-1)^{c_{i,j}} \phi_j \right] + 2\ln \Pr(\boldsymbol{c}_m) \ge 2\ln \Pr(\boldsymbol{c}_i) \\
\Leftrightarrow\ & \sum_{j=1}^{\ell} (-1)^{c_{m,j}} \phi_j + 2\ln \Pr(\boldsymbol{c}_m) \ge \sum_{j=1}^{\ell} (-1)^{c_{i,j}} \phi_j + 2\ln \Pr(\boldsymbol{c}_i)
\end{aligned}$$

where we use $f(\cdot)$ to denote the probability density function of respective random entity.

2.3 Turbo Encoder

The scheme of a turbo encoder consists of two recursive systematic convolutional encoders (RSC) and an interleaver. The component encoder used in our thesis is the (37,21) RSC as

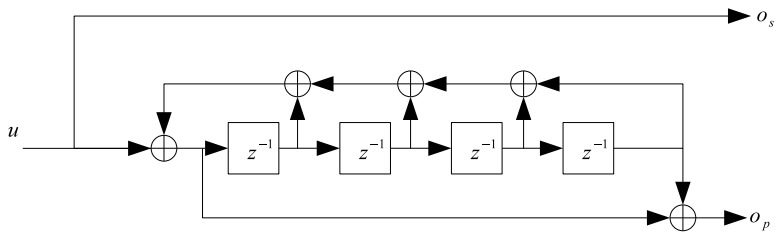


Figure 2.1: The (37,21) recursive systematic encoder .

shown in Figure 2.1. In addition, we adopt the the Berrou-Glavieux interleaver, which is described in the following.

Place the input sequence as an $s \times s$ matrix. For a bit located at row i and column j, the permuted or interleaved position at row i_r and column j_r satisfy

$$i_r = 129\,(i+j) \mod s$$

and

$$j_r = [p\,(\xi) \times (j+1) - 1] \mod s,$$

where $\xi = (i+j) \mod 8$ and the assignment of function $p(\cdot)$ is given by:

$$\begin{cases} p(0) = 17 \\ p(1) = 37 \\ p(2) = 19 \\ p(3) = 29 \\ p(4) = 41 \\ p(5) = 23 \\ p(6) = 13 \\ p(7) = 7 \end{cases}$$

2.4 Turbo Decoding

In this section, we derive the metric that will be used for turbo decoding.

Again, $\boldsymbol{r}$ is the received sequence of length ℓ. Denote by u_i is the ith bit in the decision sequence. Then, we derive:

$$\Pr\left(u_i = 0 \mid \boldsymbol{r}\right) = \sum_{\left(S_{\bar{w}}^{i-1}, S_w^i\right) \in T_0} \Pr\left(S_{\bar{w}}^{i-1}, S_w^i \mid \boldsymbol{r}\right) = \sum_{\left(S_{\bar{w}}^{i-1}, S_w^i\right) \in T_0} \frac{\Pr\left(S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{r}\right)}{\Pr\left(\boldsymbol{r}\right)}$$

where T_0 is the set of consecutive two states respectively located at trellis level $i-1$ and i, of which the connecting branch is labeled with code bit 0, and subscripts $\bar{w}$ and w denote the indices of the trellis states. In parallel, we also derive:

$$\Pr\left(u_i = 1 \mid \boldsymbol{r}\right) = \sum_{\left(S_{\bar{w}}^{i-1}, S_w^i\right) \in T_1} \Pr\left(S_{\bar{w}}^{i-1}, S_w^i \mid \boldsymbol{r}\right) = \sum_{\left(S_{\bar{w}}^{i-1}, S_w^i\right) \in T_1} \frac{\Pr\left(S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{r}\right)}{\Pr\left(\boldsymbol{r}\right)}$$

where T_1 is the set of consecutive two states respectively located at trellis level $i-1$ and i, of which the connecting branch is labeled with code bit 1.

Define the log-likelihood ratio (LLR) of the ith decision bit as follows:

$$\Lambda\left(i\right) = \log \frac{\Pr\left(u_i = 1 \mid \boldsymbol{r}\right)}{\Pr\left(u_i = 0 \mid \boldsymbol{r}\right)} = \log \frac{\sum_{\left(S_{\bar{w}}^{i-1}, S_w^i\right) \in T_1} \Pr\left(S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{r}\right)}{\sum_{\left(S_{\bar{w}}^{i-1}, S_w^i\right) \in T_0} \Pr\left(S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{r}\right)}.$$

Then, the ith bit is declared zero, i.e., $u_i = 0$, if $\Lambda\left(i\right) \leq 0$, and is declared one, otherwise. The term inside the summation, i.e., $\Pr\left(S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{r}\right)$ can be further decomposed as follows.

$$\begin{aligned}
\Pr\left(S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{r}\right) &= \Pr\left(S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{r}_1^{i-1}, \boldsymbol{r}_i, \boldsymbol{r}_{i+1}^{\ell}\right) \\
&= \Pr\left(\boldsymbol{r}_{i+1}^{\ell} \mid S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{r}_1^{i-1}, \boldsymbol{r}_i\right) \Pr\left(S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{r}_1^{i-1}, \boldsymbol{r}_i\right) \qquad (2.3) \\
&= \Pr\left(\boldsymbol{r}_{i+1}^{\ell} \mid S_w^i\right) \Pr\left(S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{r}_1^{i-1}, \boldsymbol{r}_i\right) \qquad (2.4) \\
&= \Pr\left(\boldsymbol{r}_{i+1}^{\ell} \mid S_w^i\right) \Pr\left(S_w^i, \boldsymbol{r}_i \mid S_{\bar{w}}^{i-1}, \boldsymbol{r}_1^{i-1}\right) \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{r}_1^{i-1}\right) \\
&= \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{r}_1^{i-1}\right) \Pr\left(S_w^i, \boldsymbol{r}_i \mid S_{\bar{w}}^{i-1}, \boldsymbol{r}_1^{i-1}\right) \Pr\left(\boldsymbol{r}_{i+1}^{\ell} \mid S_w^i\right) \\
&= \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{r}_1^{i-1}\right) \Pr\left(S_w^i, \boldsymbol{r}_i \mid S_{\bar{w}}^{i-1}\right) \Pr\left(\boldsymbol{r}_{i+1}^{\ell} \mid S_w^i\right)
\end{aligned}$$

Define

$$\begin{aligned}
\alpha\left(S_{\bar{w}}^{i-1}\right) &\triangleq \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{r}_1^{i-1}\right) \\
\beta\left(S_w^i\right) &\triangleq \Pr\left(\boldsymbol{r}_{i+1}^{\ell} \mid S_w^i\right) \\
\gamma\left(S_{\bar{w}}^{i-1}, S_w^i\right) &\triangleq \Pr\left(S_w^i, \boldsymbol{r}_i \mid S_{\bar{w}}^{i-1}\right)
\end{aligned}$$

Then

$$\Pr\left(S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{r}\right) = \alpha\left(S_{\bar{w}}^{i-1}\right)\gamma\left(S_{\bar{w}}^{i-1}, S_w^i\right)\beta\left(S_w^i\right)$$

Therefore, determination of the three functions can decide $\Lambda(i)$, which in turns decides the ith bit u_i.

Functions α and β can be recursively computed as follows.

$$\begin{aligned}
\alpha\left(S_w^i\right) &= \Pr\left(S_w^i, \boldsymbol{r}_1^i\right) \\
&= \sum_{\bar{w}=0}^{15} \Pr\left(S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{r}_1^i\right) \\
&= \sum_{\bar{w}=0}^{15} \Pr\left(S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{r}_1^{i-1}, \boldsymbol{r}_i\right) \\
&= \sum_{\bar{w}=0}^{15} \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{r}_1^{i-1}\right)\Pr\left(S_w^i, \boldsymbol{r}_i \mid S_{\bar{w}}^{i-1}, \boldsymbol{r}_1^{i-1}\right) \\
&= \sum_{\bar{w}=0}^{15} \alpha\left(S_{\bar{w}}^{i-1}\right)\gamma\left(S_{\bar{w}}^{i-1}, S_w^i\right),
\end{aligned}$$

$$
\begin{aligned}
\beta\left(S_{\bar{w}}^{i}\right) &= \Pr\left(\boldsymbol{r}_{i+1}^{\ell} \mid S_{\bar{w}}^{i}\right) \\
&= \frac{\Pr\left(\boldsymbol{r}_{i+1}^{\ell}, S_{\bar{w}}^{i}\right)}{\Pr\left(S_{\bar{w}}^{i}\right)} \\
&= \sum_{w=0}^{15} \frac{\Pr\left(\boldsymbol{r}_{i+1}^{\ell}, S_{\bar{w}}^{i}, S_{w}^{i+1}\right)}{\Pr\left(S_{\bar{w}}^{i}\right)} \\
&= \sum_{w=0}^{15} \frac{\Pr\left(\boldsymbol{r}_{i+1}, \boldsymbol{r}_{i+2}^{\ell}, S_{\bar{w}}^{i}, S_{w}^{i+1}\right)}{\Pr\left(S_{\bar{w}}^{i}\right)} \\
&= \sum_{w=0}^{15} \frac{\Pr\left(\boldsymbol{r}_{i+2}^{\ell} \mid \boldsymbol{r}_{i+1}, S_{\bar{w}}^{i}, S_{w}^{i+1}\right) \Pr\left(\boldsymbol{r}_{i+1}, S_{\bar{w}}^{i}, S_{w}^{i+1}\right)}{\Pr\left(S_{\bar{w}}^{i}\right)} \\
&= \sum_{w=0}^{15} \frac{\Pr\left(\boldsymbol{r}_{i+2}^{\ell} \mid S_{w}^{i+1}\right) \Pr\left(\boldsymbol{r}_{i+1}, S_{\bar{w}}^{i}, S_{w}^{i+1}\right)}{\Pr\left(S_{\bar{w}}^{i}\right)} \\
&= \sum_{w=0}^{15} \frac{\Pr\left(\boldsymbol{r}_{i+2}^{\ell} \mid S_{w}^{i+1}\right) \Pr\left(\boldsymbol{r}_{i+1}, S_{w}^{i+1} \mid S_{\bar{w}}^{i}\right) \Pr\left(S_{\bar{w}}^{i}\right)}{\Pr\left(S_{\bar{w}}^{i}\right)} \\
&= \sum_{w=0}^{15} \Pr\left(\boldsymbol{r}_{i+2}^{\ell} \mid S_{w}^{i+1}\right) \Pr\left(\boldsymbol{r}_{i+1}, S_{w}^{i+1} \mid S_{\bar{w}}^{i}\right) \\
&= \sum_{w=0}^{15} \beta\left(S_{w}^{i+1}\right) \gamma\left(S_{\bar{w}}^{i}, S_{w}^{i+1}\right),
\end{aligned}
$$

where for the additive white Gaussian noise channel, function γ is equal to:

$$
\begin{aligned}
\gamma\left(S_{\bar{w}}^{i-1}, S_{w}^{i}\right) &= \Pr\left(S_{w}^{i}, \boldsymbol{r}_{i} \mid S_{\bar{w}}^{i-1}\right) \\
&= \frac{\Pr\left(S_{\bar{w}}^{i-1}, S_{w}^{i}, \boldsymbol{r}_{i}\right)}{\Pr\left(S_{\bar{w}}^{i-1}\right)} \\
&= \frac{\Pr\left(\boldsymbol{r}_{i} \mid S_{\bar{w}}^{i-1}, S_{w}^{i}\right) \Pr\left(S_{\bar{w}}^{i-1}, S_{w}^{i}\right)}{\Pr\left(S_{\bar{w}}^{i-1}\right)} \\
&= \frac{\Pr\left(\boldsymbol{r}_{i} \mid \boldsymbol{x}_{i}\right) \Pr\left(S_{\bar{w}}^{i-1}, S_{w}^{i}\right)}{\Pr\left(S_{\bar{w}}^{i-1}\right)} \\
&= \frac{\Pr\left(\boldsymbol{r}_{i} \mid \boldsymbol{x}_{i}\right) \Pr\left(S_{w}^{i} \mid S_{\bar{w}}^{i-1}\right) \Pr\left(S_{\bar{w}}^{i-1}\right)}{\Pr\left(S_{\bar{w}}^{i-1}\right)} \\
&= \Pr\left(\boldsymbol{r}_{i} \mid \boldsymbol{x}_{i}\right) \Pr\left(S_{w}^{i} \mid S_{\bar{w}}^{i-1}\right) \\
&= \Pr\left(c_{i}\right) \Pr\left(\boldsymbol{r}_{i} \mid \boldsymbol{x}_{i}\right) \\
&= \Pr\left(c_{i}\right) \frac{1}{\sqrt{2\pi}\sigma} \exp\{-\frac{|| \boldsymbol{r}_{i} - \boldsymbol{x}_{i} ||^{2}}{2\sigma^{2}}\}.
\end{aligned}
$$

We now state the algorithmic procedures to obtain the three functions. Note that the RSC trellis we adopt has 16 states as shown in Figure 2.1.

Derivation of function α

1. Initialization: Set $\alpha\left(S_0^0\right) = 1$ and $\alpha\left(S_w^0\right) = 0$ for all $1 \leq w \leq 15$.

2. Recursion: For $1 \leq i \leq \ell$, compute $\alpha\left(S_w^i\right) = \sum_{\bar{w}=0}^{15} \alpha\left(S_{\bar{w}}^{i-1}\right) \gamma\left(S_{\bar{w}}^{i-1}, S_w^i\right)$.

Derivation of function β

1. Initialization: Set $\beta\left(S_0^\ell\right) = 1$ and $\beta\left(S_w^\ell\right) = 0$ for $1 \leq w \leq 15$.

2. Recursion: For $1 \leq i \leq \ell$, $\beta\left(S_{\bar{w}}^i\right) = \sum_{w=0}^{15} \beta\left(S_w^{i+1}\right) \gamma\left(S_{\bar{w}}^i, S_w^{i+1}\right)$.

Derivation of function γ

1. Initialization: Give that c_i is the information bit corresponding to the state transition from $S_{\bar{w}}^{i-1}$ to S_w^i. Let the corresponding output vector be $\boldsymbol{x}_i$.

2. Computation: $\gamma\left(S_{\bar{w}}^{i-1}, S_w^i\right) = \Pr\left(c_i\right) \frac{1}{\sqrt{2\pi}\sigma} \exp\{-\frac{||\boldsymbol{r}_i - \boldsymbol{x}_i||^2}{2\sigma^2}\}$.

Chapter 3

Joint Source-Channel Block Code

In this chapter, the proposed construction approach for the joint source-channel block codes for non-uniform source distributions over additive white Gaussian noise (AWGN) channels will be presented.

3.1 Union bound

In general, it is difficult to determine the close-form formula for the symbol error probability (SEP) corresponding to a codebook and a source distribution. When a criterion for the SEP is needed, researchers will mostly derive the union bound instead. As such, we will use the union bound as a design criterion for the proposed joint source-channel code.

For the derivation of the union bound, a pair-wise error probability should be devised first. Consider two source symbols s_i and s_j respectively with probabilities p_i and p_j, and assume that they are mapped respectively to codewords c_i and c_j, between which the Hamming distance is h. In such case, an error occurs when the receiver declares the reception of s_j but s_i is transmitted, or vice versa. Denote the previously mentioned *pair-wise error probability*

by $\Pr(s_i \to s_j)$. Then, when c_i is transmitted over the BSC,

$$\Pr(s_i \to s_j) = \begin{cases} \sum_{e=(h+1)/2}^{h} \binom{h}{e} b^e (1-b)^{h-e} & \text{for odd } h \\ \frac{1}{2}\binom{h}{h/2} b^{h/2}(1-b)^{h/2} + \sum_{e=h/2+1}^{h} \binom{h}{e} b^e (1-b)^{h-e} & \text{for even } h \end{cases}$$

Since the above $\Pr(s_i \to s_j)$ is only a function of h, we will denote it by K_h.

Next, we denote by $P_e(h)$ the probability that a transmitted codeword c_i is incorrectly decoded to another codeword at Hamming distance h from it. Then, $P_e(h)$ can be bounded above by:

$$P_e(h) \leq \sum_{i=1}^{n} p_i \sum_{j:d_H(c_i,c_j)=h} K_h,$$

where $d_H(c_i, c_j)$ is the Hamming distance between codewords c_i and c_j. The system error probability P_e must then satisfy:

$$\begin{aligned} P_e &\leq \sum_{h=1}^{\ell} P_e(h) \\ &\leq \sum_{h=1}^{\ell} \sum_{i=1}^{n} p_i \sum_{j:d_H(c_i,c_j)=h} K_h \\ &= \sum_{h=1}^{\ell} K_h \sum_{i=1}^{n} p_i \sum_{j:d_H(c_i,c_j)=h} \end{aligned}$$

where the last step follows since K_h is nothing to do with i and j. Define $A_h \triangleq \sum_{i=1}^{n} \sum_{j:d_H(i,j)=h} p_i$ to be the average number of codeword pairs with Hamming distance h . Finally, we have

$$P_e \leq \sum_{h=1}^{\ell} K_h A_h.$$

We then use the above upper bound as a design criterion for our joint source-channel coding design.

In summary, based on a given code rate $R = 1/4$ and system signal-to-noise ratio per information bit $E_b/N_0 = 10$ dB, we can compute K_h. Afterwards, we will attempt to find a good joint source-channel block code of length ℓ that hopefully minimizes $\sum_{h=1}^{\ell} K_h A_h$.

3.2 Construction of Joint Source-Channel Block Code

For a small codeword length such as $\ell = 8$, we can exhaustively search for all the possible code designs and find the one that minimizes $\sum_{h=1}^{\ell} K_h A_h$. However, such an exhaustive search approach may not be feasible for a larger codeword length. We therefore propose a sub-optimal but low-complexity algorithm to construct a joint source-channel code with a prohibitively small union bound value.

For given n source symbol $s_1, \ldots, s_n$ with probabilities of occurrence $p_1, \ldots, p_n$, and for a specified codeword length ℓ , we propose to construct a joint source-channel block code as follows.

Step 0. Initialize the length index as one, namely, $y = 1$. Set code matrices

$$C = \begin{bmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,\ell} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,\ell} \\ \vdots & \vdots & & \vdots \\ c_{n,1} & c_{n,2} & \cdots & c_{n,\ell} \end{bmatrix} = \mathbf{0} \quad \text{and} \quad C' = \begin{bmatrix} c'_{1,1} & c'_{1,2} & \cdots & c'_{1,\ell} \\ c'_{2,1} & c'_{2,2} & \cdots & c'_{2,\ell} \\ \vdots & \vdots & & \vdots \\ c'_{n,1} & c'_{n,2} & \cdots & c'_{n,\ell} \end{bmatrix} = \mathbf{0},$$

where $\mathbf{0}$ is the all-zero matrix of proper size.

Step 1. For $1 \le i, j \le n$, calculate $a_{i,j,y} = d_H\left(c_{i,1}c_{i,2}\ldots c_{i,y-1}, c_{j,1}c_{j,2}\ldots c_{j,y-1}\right)$.

Step 2. Compute the union bound contribution for each codeword, i.e., for $1 \le i \le n$,

$$g_i = \sum_{j=1}^{n} K_{a_{i,j,y}} \times p_i.$$

Step 3. Set the codeword index $q = 1$. Sort $\{g_i\}_{i=1}^{n}$ such that $g_{s_1} \ge g_{s_2} \ge \ldots \ge g_{s_n}$, where $\{s_i\}_{i=1}^{n}$ denotes the sequence of the sorted ordering.

Step 4. For $1 \le u < q$, calculate

$$\begin{cases} d_{0,u} = (0 \oplus c_{s_u,y}) + a_{s_q,s_u,y} \\ d_{1,u} = (1 \oplus c_{s_u,y}) + a_{s_q,s_u,y} \end{cases}$$

and

$$r_0 = \begin{cases} 0, & q = 1 \\ \sum_{u=1}^{q-1} K_{d_{0,u}} \times \left(p_{s_q} + p_{s_u}\right), & q > 1 \end{cases}, \quad r_1 = \begin{cases} 0, & q = 1 \\ \sum_{u=1}^{q-1} K_{d_{1,u}} \times \left(p_{s_q} + p_{s_u}\right), & q > 1 \end{cases}$$

Step 5. Compare r_0 and r_1 as follows.

If $r_0 > r_1$, choose $c_{s_q,y} = 1$; else choose $c_{s_q,y} = 0$.

Step 6. For $1 \le i, j \le n$, calculate $b_{i,j,y} = d_H\left(c'_{i,1}c'_{i,2}\ldots c'_{i,y-2}, c'_{j,1}c'_{j,2}\ldots c'_{j,y-2}\right)$.

Step 7. Compute the union bound contribution for each codeword, i.e., for $1 \le i \le n$,

$$g_i = \sum_{j=1}^{n} K_{b_{i,j,y}} \times p_i.$$

Step 8. Set the codeword index $q = 1$. Sort $\{g_i\}_{i=1}^n$ such that $g_{s_1} \ge g_{s_2} \ge \ldots \ge g_{s_n}$, where $\{s_i\}_{i=1}^n$ denotes the sequence of the sorted ordering.

Step 9. If $y \ge 2$, compute for $1 \le u < q$,

$$\begin{cases} d_{00,u} = (00 \oplus c'_{s_u,y-1}c'_{s_u,y}) + b_{s_q,s_u,y} \\ d_{01,u} = (01 \oplus c'_{s_u,y-1}c'_{s_u,y}) + b_{s_q,s_u,y} \\ d_{10,u} = (10 \oplus c'_{s_u,y-1}c'_{s_u,y}) + b_{s_q,s_u,y} \\ d_{11,u} = (11 \oplus c'_{s_u,y-1}c'_{s_u,y}) + b_{s_q,s_u,y} \end{cases}$$

and

$$\begin{cases} r_{00} = \begin{cases} 0, & q = 1 \\ \sum_{u=1}^{q-1} K_{d_{00,u}} \times \left(p_{s_q} + p_{s_u}\right), & q > 1 \end{cases} \\ r_{01} = \begin{cases} 0, & q = 1 \\ \sum_{u=1}^{q-1} K_{d_{01,u}} \times \left(p_{s_q} + p_{s_u}\right), & q > 1 \end{cases} \\ r_{10} = \begin{cases} 0, & q = 1 \\ \sum_{u=1}^{q-1} K_{d_{10,u}} \times \left(p_{s_q} + p_{s_u}\right), & q > 1 \end{cases} \\ r_{11} = \begin{cases} 0, & q = 1 \\ \sum_{u=1}^{q-1} K_{d_{11,u}} \times \left(p_{s_q} + p_{s_u}\right), & q > 1 \end{cases} \end{cases}.$$

Step 10. Among r_{00}, r_{01}, r_{10} *and* r_{11};

if r_{00} *is the minimum one, choose* $c'_{s_q,y-1}c'_{s_q,y} = 00$;

else if r_{01} *is the minimum one, choose* $c'_{s_q,y-1}c'_{s_q,y} = 01$;

else if r_{10} *is the minimum one, choose* $c'_{s_q,y-1}c'_{s_q,y} = 10$;

else choose $c'_{s_q,y-1}c'_{s_q,y} = 11$.

Step 11. Set $q = q + 1$ *and repeat Steps 4–8 until* $q = n$.

Step 12. Compare the union bound of C *and* C', *and let* C^* *be the one with a smaller union bound.*

Step 13. Set $y = y + 1$, $C' = C$ *and* $C = C^*$. *Then repeat Steps 1–10 until* $y = \ell$.

This algorithm will give us a codebook C^* with a satisfiable union bound value.

3.3 Decoding

For a joint source-channel block code, it has been known that the MAP decoder is optimal in the sense of minimizing the symbol error probability. The MAP decoder in general does not have an efficient implementation. Fortunately, it can be noted that the MAP decoding metric derived in Section 2.2 is strictly decreasing when being calculated in a bit by bit fashion; hence, we can use the priority-first sequential search algorithm as a vehicle to obtain the MAP decision. In this thesis, the priority-first sequential search decoding algorithms have two forms: one for hard-decision decoding and the other for soft-decision decoding.

3.3-A) Priority-first sequential search hard-decision decoding algorithm

Step 0. Construct the binary code tree corresponding to the given codebook, in which a path form the root node to a leaf node is a codeword. Each leaf node (equivalently,

the end node of a codeword path) is associated with a probability corresponding to the probability of occurrence for the source letter that is encoded to this codeword. The probability associated with a non-leaf node is the largest one among all probabilities associated with those leaf nodes that are offsprings of this non-leaf node.

Step 1. Calculate the hard-decision sequence $\boldsymbol{y}$*. Initialize the metric of the root node as 0, and push the root node into the stack.*

Step 2. Extract the node with the maximal metric value from the stack, which we denote it by e*. Then, for all child nodes of the extracted node, if there is any, perform the following procedure and then repeat Step 2.*

- *If the child node of the extracted node is a non-leaf node, set the metric of this child node to*

$$e + (c_{i,j} \oplus y_j) \ln\left(\frac{b}{1-b}\right),$$

where $c_{i,j}$ *is the code bit corresponding to the tree branch connecting the extracted node and the child node, and* j *is the tree level at which the extracted node is located. Push the child node into the stack.*

- *Else if the child node of the extracted node is a leaf node, set the metric of this child node to*

$$e + (c_{i,j} \oplus y_j) \ln\left(\frac{b}{1-b}\right) + \ln\left(p_i\right),$$

where p_i *is the probability associated with this leaf node. Push the child node into the stack.*

If the extracted node is a leaf node (which certainly has has no child nodes), output the codeword corresponding to the leaf node and stop the algorithm.

3.3-B) Priority-first sequential search soft-decision decoding algorithm

Step 0. The same as Step 0 in the algorithm in 3.3-A.

Step 1. Calculate the log-likelihood ratio ϕ_j for all $1 \leq j \leq \ell$. Initilize the metric of the root node as 0, and push the root node into the stack.

Step 2. Extract the node with the maximal metric value from the stack, which we denote it by e. Then, for all child nodes of the extracted node, if there is any, perform the following procedure, and then repeat Step 2.

- *If the child node of the extracted node is a non-leaf node, set the metric of this child node to*

$$e + (-1)^{c_{i,j}} \phi_i,$$

where $c_{i,j}$ is the code bit corresponds to the tree branch connecting the extracted node and the child node, and and j is the tree level at which the extracted node is located. Push the child node into the stack.

- *Else if the child node of the extracted node is a leaf node, set the metric of this child node to*

$$e + (-1)^{c_{i,j}} \phi_i + 2 \ln (p_i),$$

where p_i is the probability associated with this leaf node. Push the child node into the stack.

If the extracted node is a leaf node (which certainly has has no child nodes), output the codeword corresponding to the leaf node and stop the algorithm.

We remark that when the metric is non-increasing along all paths, the sequential search of the above two algorithms guarantee to find the optimal leaf path with the maximal metric.

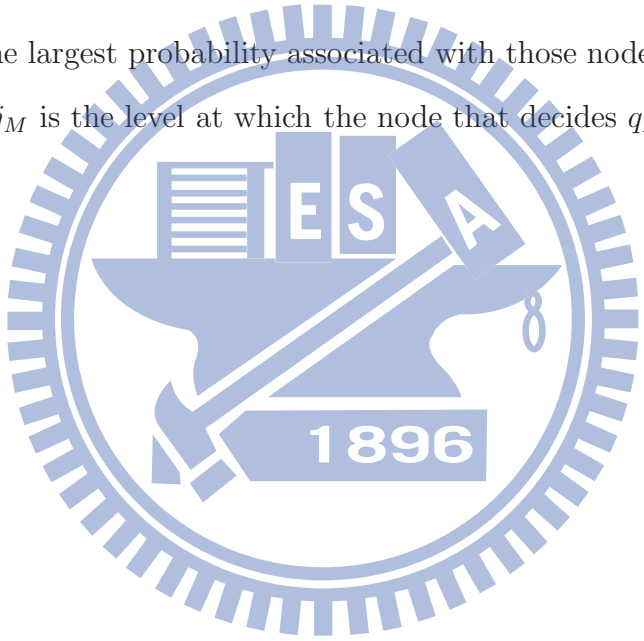
However, the metrics we adopt in the two algorithms are apparently not non-increasing; so the algorithms we propose may end up with a suboptimal codeword path.

Since at low SNR, the complexity of the second sequential search algorithm may be too large to be practical, an amendment refinement to remove some of the tree nodes during the decoding search is additional proposed below.

- When extracting a node at tree level j and associated with probability q, it will be directly discarded if

$$2\,(\ell - j_M) - \ln q_M \leq (j_M - j) - \ln q,$$

where q_M is the largest probability associated with those nodes that have been visited thus far, and j_M is the level at which the node that decides q_M is located.



Chapter 4

Modified Turbo Code

4.1 Background

In Chapter 3, we have proposed a novel approach to construct a joint source-channel block code for a non-uniform source. This approach however cannot be feasibly applied when the codeword length of interest is moderately large, and hence the attainable error rates are limited. As such, in this chapter, we turn to the modification of turbo codes that were previously designed by assuming uniform prior probabilities on codewords, and target a joint source-channel turbo coding system that can provide a practically acceptable performance for non-uniform sources.

In the literature, there have been publications working on turbo code design for non-uniform source. In [4], a turbo code has been proposed for binary independent and identically distributed (*i.i.d.*) source with non-uniform marginal distribution. In [5], the turbo code design has been extended to binary first-order Markov sources. Note that for binary *i.i.d.* sources, each information bit is statistically independent of all previous bits, which facilitates the derivation of the corresponding decoding metric for turbo decoder. When a first-order Markov source is considered, each bit only depends on the previous bit; nonetheless, the extension derivation of the turbo decoding metric can make use of this statistical

structure. In this thesis, we actually consider a source of different statistical nature, where each information bit is dependent on t previous bits with t being a known function of the bit location. Specifically, an English alphabet can be binary-indexed using five bits; so the first bit is surely dependent on the next four bits, and the second bit is statistically affected by the next three bits, etc., while the fifth bit is actually independent of the next bit if the stream of English letters is *i.i.d.* in nature. Details will be given in later sections.

4.2 Definitions and Notations

Assume there are n symbols $\{s_1, s_2, \ldots, s_n\}$ that are generated according to an independently and identically distributed distribution with marginal probabilities $\{p_1, p_2, \ldots, p_n\}$. Let $g = \lceil \log_2 n \rceil$. We can then binary-index each source symbol using g bits as $s_1 = (00\ldots00)$, $s_2 = (00\ldots01)$, $s_3 = (00\ldots10)$, etc. As a result, for $1 \leq a < g$, the $(mg + a)$th bit only depends on the previous $a - 1$ bits.

For notational convenience, we derive in the following by assuming the ith bit u_i is dependent on the previous t bits $u_{i-1}\ldots u_{i-t}$. By our setting in the previous paragraph, u_{i-1} should be dependent only on bits $u_{i-2}\ldots u_{i-t}$. Denote $\boldsymbol{U}_i = u_{i-1}\ldots u_{i-t} = u_{i-1}\boldsymbol{U}_{i-1}$.

Denote by S_w^i the state at level i, and by T_c the set of state pair $\left(S_{\bar{w}}^{i-1}, S_w^i\right)$ such that the input bit $u_i = c$ will make the trellis transition from state $S_{\bar{w}}^{i-1}$ to state S_w^i . Let $\boldsymbol{x}$ be the codeword sequence corresponding to the input information sequence $\{u_i\}_1^\ell$, and let $\boldsymbol{x}_i$ be the codeword vector portion corresponding to u_i. For convenience, we use $\boldsymbol{x}_i^j$ to denote $\boldsymbol{x}_i\boldsymbol{x}_{i+1}\ldots\boldsymbol{x}_j$. Similarly, denote by $\boldsymbol{r}_i$ the channel output due to input $\boldsymbol{x}_i$, and use $\boldsymbol{r}_i^j$ to denote $\boldsymbol{r}_i\boldsymbol{r}_{i+1}\ldots\boldsymbol{r}_j$. Also, abbreviate $\boldsymbol{r} = \boldsymbol{r}_1^\ell = \boldsymbol{r}_1\boldsymbol{r}_2\ldots\boldsymbol{r}_\ell$. Note that for 1/3-rate turbo coding system, $\boldsymbol{x}_i = \left(x_i^s, x_i^{1p}, x_i^{2p}\right)$ and $\boldsymbol{r}_i = \left(r_i^s, r_i^{1p}, r_i^{2p}\right)$, where superscripts "1$p$" and "2$p$" indicates the first and second parity-check codeword portions, respectively. We then illustrate the general scheme of a turbo decoder in Fig. 4.1, in which $\Lambda^{(j)}(i)$ is the log-likelihood ratio

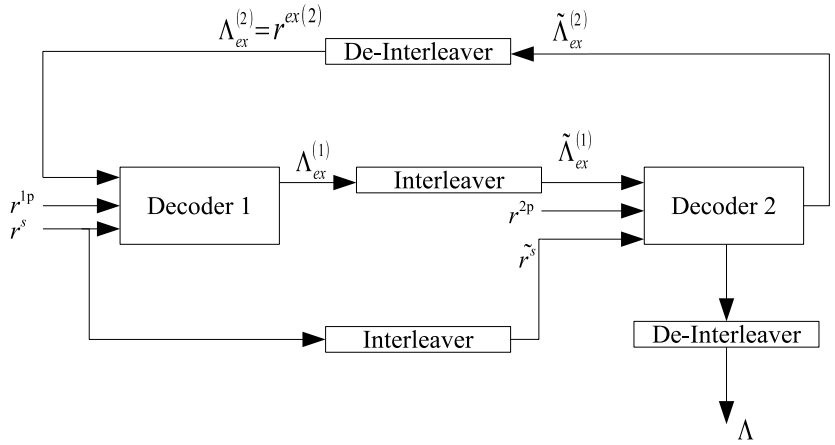


Figure 4.1: Diagram of a sample turbo decoder.

of the ith bit u_i computed by the jth component code decoder, and $\Lambda_{\text{ex}}^{(j)}(i)$ is the extrinsic information of the ith bit u_i obtained from the jth component code decoder.

4.3 Modified Decoding Metric

The derivation of the decoding metric for the turbo code in the previous chapter is based on the assumption that the source is uniform i.i.d. When the source is not uniformly distributed, the equality of (2.3) and (2.4) is no longer valid. Hence, an alternative decomposition of $\Pr(u_i = c \mid \boldsymbol{r})$ should be done.

Let $\boldsymbol{V}_t$ be the binary bit stream of length t , i.e., $\boldsymbol{V}_t = v_t v_{t-1} \ldots v_1$, where $v_i \in \{0, 1\}$ for every i, and denote $\boldsymbol{V}_t = v_t \boldsymbol{V}_{t-1}$. We then derive:

$$\begin{aligned} \Pr(u_i = c \mid \boldsymbol{r}) &= \frac{\Pr(u_i = c, \boldsymbol{r})}{\Pr(\boldsymbol{r})} \\ &= \sum_{(S_{\bar{w}}^{i-1}, S_w^i) \in T_c} \frac{\Pr\left(S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{r}\right)}{\Pr(\boldsymbol{r})}, \end{aligned}$$

where

$$
\begin{aligned}
&\sum_{\left(S_{\bar{w}}^{i-1},S_w^i\right)\in T_c} \Pr\left(S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{r}\right) \\
&= \sum_{\left(S_{\bar{w}}^{i-1},S_w^i\right)\in T_c} \sum_{\boldsymbol{V}_t\in\{0,1\}^t} \Pr\left(S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{U}_i = \boldsymbol{V}_t, \boldsymbol{r}\right) \\
&= \sum_{\left(S_{\bar{w}}^{i-1},S_w^i\right)\in T_c} \sum_{\boldsymbol{V}_t\in\{0,1\}^t} \Pr\left(S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{U}_i = \boldsymbol{V}_t, \boldsymbol{r}_1^{i-1}, \boldsymbol{r}_i, \boldsymbol{r}_{i+1}^{\ell}\right) \\
&= \sum_{\left(S_{\bar{w}}^{i-1},S_w^i\right)\in T_c} \sum_{\boldsymbol{V}_t\in\{0,1\}^t} \Pr\left(\boldsymbol{r}_1^{i-1}\right) \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t \mid \boldsymbol{r}_1^{i-1}\right) \Pr\left(S_w^i, \boldsymbol{r}_i, \boldsymbol{r}_{i+1}^{\ell} \mid S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t, \boldsymbol{r}_1^{i-1}\right)
\end{aligned}
\tag{4.1}
$$

Given state $S_{\bar{w}}^{i-1}$ and $\boldsymbol{U}_i$, the received bit stream $\boldsymbol{r}_1^{i-1}$ is independent of $\left(S_w^i, \boldsymbol{r}_i, \boldsymbol{r}_{i+1}^{\ell}\right)$. Hence, the derivation of (4.1) can be continued as follows.

$$
\begin{aligned}
&\sum_{\left(S_{\bar{w}}^{i-1},S_w^i\right)\in T_c} \sum_{\boldsymbol{V}_t\in\{0,1\}^t} \Pr\left(\boldsymbol{r}_1^{i-1}\right) \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t \mid \boldsymbol{r}_1^{i-1}\right) \Pr\left(S_w^i, \boldsymbol{r}_i, \boldsymbol{r}_{i+1}^{\ell} \mid S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t, \boldsymbol{r}_1^{i-1}\right) \\
&= \sum_{\left(S_{\bar{w}}^{i-1},S_w^i\right)\in T_c} \sum_{\boldsymbol{V}_t\in\{0,1\}^t} \Pr\left(\boldsymbol{r}_1^{i-1}\right) \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t \mid \boldsymbol{r}_1^{i-1}\right) \Pr\left(S_w^i, \boldsymbol{r}_i, \boldsymbol{r}_{i+1}^{\ell} \mid S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right) \\
&= \sum_{\left(S_{\bar{w}}^{i-1},S_w^i\right)\in T_c} \sum_{\boldsymbol{V}_t\in\{0,1\}^t} \left[\Pr\left(\boldsymbol{r}_1^{i-1}\right) \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t \mid \boldsymbol{r}_1^{i-1}\right) \frac{\Pr\left(S_w^i, \boldsymbol{r}_i, S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right)}{\Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right)}\right. \\
&\quad \left.\times \frac{\Pr\left(S_w^i, \boldsymbol{r}_i, S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t, \boldsymbol{r}_{i+1}^{\ell}\right)}{\Pr\left(S_w^i, \boldsymbol{r}_i, S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right)}\right] \\
&= \sum_{\left(S_{\bar{w}}^{i-1},S_w^i\right)\in T_c} \sum_{\boldsymbol{V}_t\in\{0,1\}^t} \left[\Pr\left(\boldsymbol{r}_1^{i-1}\right) \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t \mid \boldsymbol{r}_1^{i-1}\right) \Pr\left(S_w^i, \boldsymbol{r}_i \mid S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right)\right. \\
&\quad \left.\times \Pr\left(\boldsymbol{r}_{i+1}^{\ell} \mid \boldsymbol{r}_i, S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{U}_i = \boldsymbol{V}_t\right)\right].
\end{aligned}
$$

By noticing that knowing $\left(S_{\bar{w}}^{i-1}, S_w^i\right)$ is equivalent to knowing (u_i, S_w^i), and $\boldsymbol{r}_{i+1}^{\ell}$ is indepen-

dent of $\boldsymbol{r}_i$ when given $(u_i, \boldsymbol{U}_i)$, we can continued the derivation as:

$$\begin{aligned}
&\Pr\left(u_i = c, \boldsymbol{r}\right) \\
&= \sum_{\left(S_{\bar{w}}^{i-1}, S_w^i\right) \in T_c} \sum_{\boldsymbol{V}_t \in \{0,1\}^t} \Big[\Pr\left(\boldsymbol{r}_1^{i-1}\right) \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t \mid \boldsymbol{r}_1^{i-1}\right) \Pr\left(S_w^i, \boldsymbol{r}_i \mid S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right) \\
&\quad \times \Pr\left(\boldsymbol{r}_{i+1}^{\ell} \mid \boldsymbol{r}_i, S_{\bar{w}}^{i-1}, S_w^i, \boldsymbol{U}_i = \boldsymbol{V}_t\right)\Big] \\
&= \sum_{\left(S_{\bar{w}}^{i-1}, S_w^i\right) \in T_c} \sum_{\boldsymbol{V}_t \in \{0,1\}^t} \Big[\Pr\left(\boldsymbol{r}_1^{i-1}\right) \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t \mid \boldsymbol{r}_1^{i-1}\right) \Pr\left(S_w^i, \boldsymbol{r}_i \mid S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right) \\
&\quad \times \Pr\left(\boldsymbol{r}_{i+1}^{\ell} \mid u_i, \boldsymbol{U}_i = \boldsymbol{V}_t, S_w^i\right)\Big].
\end{aligned}$$

Define

$$\begin{aligned}
\alpha\left(\boldsymbol{V}_t, S_{\bar{w}}^{i-1}\right) &\triangleq \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t \mid \boldsymbol{r}_1^{i-1}\right) \\
\beta\left(c, \boldsymbol{V}_t, S_w^i\right) &\triangleq \Pr\left(\boldsymbol{r}_{i+1}^{\ell} \mid u_i = c, \boldsymbol{U}_i = \boldsymbol{V}_t, S_w^i\right) \\
\gamma\left(c, \boldsymbol{V}_t, S_{\bar{w}}^{i-1}, S_w^i\right) &\triangleq \Pr\left(u_i = c, S_w^i, \boldsymbol{r}_i \mid S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right) = \Pr\left(S_w^i, \boldsymbol{r}_i \mid S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right)
\end{aligned}$$

and observe that when $t = 0$, bit u_i is independent of any previous bits and the above functions are reduced to:

$$\begin{aligned}
\alpha\left(\boldsymbol{V}_0, S_{\bar{w}}^{i-1}\right) &\triangleq \Pr\left(S_{\bar{w}}^{i-1} \mid \boldsymbol{r}_1^{i-1}\right) \\
\beta\left(c, \boldsymbol{V}_0, S_w^i\right) &\triangleq \Pr\left(\boldsymbol{r}_{i+1}^{\ell} \mid u_i = c, S_w^i\right) \\
\gamma\left(c, \boldsymbol{V}_0, S_{\bar{w}}^{i-1}, S_w^i\right) &\triangleq \Pr\left(u_i = c, S_w^i, \boldsymbol{r}_i \mid S_{\bar{w}}^{i-1}\right) = \Pr\left(S_w^i, \boldsymbol{r}_i \mid S_{\bar{w}}^{i-1}\right)
\end{aligned}$$

Hence,

$$\begin{aligned}
\Lambda^{(1)}(i) &= \ln \frac{\Pr(u_i = 1 \mid \boldsymbol{r})}{\Pr(u_i = 0 \mid \boldsymbol{r})} \\
&= \ln \frac{\Pr(u_i = 1, \boldsymbol{r})}{\Pr(u_i = 0, \boldsymbol{r})} \\
&= \ln \frac{\sum_{\left(S_{\bar{w}}^{i-1}, S_w^i\right) \in T_1} \sum_{\boldsymbol{V}_t \in \{0,1\}^t} \Pr\left(\boldsymbol{r}_1^{i-1}\right) \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t \mid \boldsymbol{r}_1^{i-1}\right) \Pr\left(S_w^i, r_i \mid S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right) \Pr\left(\boldsymbol{r}_{i+1}^{\ell} \mid u_i, \boldsymbol{U}_i = \boldsymbol{V}_t, S_w^i\right)}{\sum_{\left(S_{\bar{w}}^{i-1}, S_w^i\right) \in T_0} \sum_{\boldsymbol{V}_t \in \{0,1\}^t} \Pr\left(\boldsymbol{r}_1^{i-1}\right) \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t \mid \boldsymbol{r}_1^{i-1}\right) \Pr\left(S_w^i, r_i \mid S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right) \Pr\left(\boldsymbol{r}_{i+1}^{\ell} \mid u_i, \boldsymbol{U}_i = \boldsymbol{V}_t, S_w^i\right)} \\
&= \ln \frac{\sum_{\left(S_{\bar{w}}^{i-1}, S_w^i\right) \in T_1} \sum_{\boldsymbol{V}_t \in \{0,1\}^t} \alpha\left(\boldsymbol{V}_t, S_{\bar{w}}^{i-1}\right) \gamma\left(1, \boldsymbol{V}_t, S_{\bar{w}}^{i-1}, S_w^i\right) \beta\left(1, \boldsymbol{V}_t, S_w^i\right)}{\sum_{\left(S_{\bar{w}}^{i-1}, S_w^i\right) \in T_0} \sum_{\boldsymbol{V}_t \in \{0,1\}^t} \alpha\left(\boldsymbol{V}_t, S_{\bar{w}}^{i-1}\right) \gamma\left(0, \boldsymbol{V}_t, S_{\bar{w}}^{i-1}, S_w^i\right) \beta\left(0, \boldsymbol{V}_t, S_w^i\right)}
\end{aligned}$$

For additive white Gaussian noise channels, function γ is given by

$$\begin{aligned}
&\gamma\left(c, \boldsymbol{V}_t, S_{\bar{w}}^{i-1}, S_w^i\right) \\
&= \Pr\left(u_i = c, S_w^i, \boldsymbol{r}_i \mid S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right) \\
&= \frac{\Pr\left(u_i = c, S_w^i, \boldsymbol{r}_i, S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right)}{\Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right)} \\
&= \frac{\Pr\left(\boldsymbol{r}_i \mid u_i = c, S_w^i, S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right) \Pr\left(u_i = c, S_w^i, S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right)}{\Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right)} \\
&= \Pr\left(\boldsymbol{r}_i \mid u_i = c, S_w^i, S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right) \Pr\left(u_i = c, S_w^i \mid S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t\right) \\
&= \Pr\left(\boldsymbol{r}_i \mid \boldsymbol{x}_i\right) \Pr\left(u_i = c \mid \boldsymbol{U}_i = \boldsymbol{V}_t\right)
\end{aligned}$$

The basic structure of the turbo decoding metric is now done.

Now after the execution of a certain rounds of turbo decoding, we denote by $r_i^{\mathrm{ex}(2)}$ the extrinsic information from the second component decoder after de-interleaving. This $r_i^{\mathrm{ex}(2)}$ will be the input to the first component decoder. The Gaussian assumption on $r_i^{\mathrm{ex}(2)}$ from [6] [7] then gives:

$$\Pr\left(\boldsymbol{r}_i \mid \boldsymbol{x}_i\right) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{\left(r_i^s - x_i^s\right)^2 + \left(r_i^{(1p)} - x_i^{(1p)}\right)^2}{2\sigma^2}\right\} \Pr\left(r_i^{\mathrm{ex}(2)} \middle| u_i = c\right).$$

Suppose $\bar{\sigma}^2$ and $\bar{M}$ are the estimated variance and mean of $r_i^{\mathrm{ex}(2)}$, respectively, which are defined as follows:

$$\bar{M} = \frac{1}{\ell}\sum_{i=1}^{\ell} \mid r_i^{\mathrm{ex}(2)} \mid$$

$$\bar{\sigma}^2 = \frac{1}{\ell - 1}\sum_{i=1}^{\ell} \left(\mid r_i^{\mathrm{ex}(2)} \mid - \bar{M} \right)^2$$

The probability $\Pr\left(r_i^{\mathrm{ex}(2)} \middle| u_i = c \right)$ can then be assigned as:

$$\Pr\left(r_i^{\mathrm{ex}(2)} \middle| u_i = c \right) = \begin{cases} 1, & \text{if } \bar{\sigma}^2 = 0 \\ \dfrac{1}{\sqrt{2\pi\bar{\sigma}^2}} \exp\left\{ -\dfrac{\left[r_i^{\mathrm{ex}(2)} - (2u_i - 1)\,\bar{M} \right]^2}{2\bar{\sigma}^2} \right\}, & \text{otherwise} \end{cases}$$

Hence, function γ can be given as:

$$\begin{aligned} &\gamma\left(c, \boldsymbol{V}_t, S_{\bar{w}}^{i-1}, S_w^i \right) \\ &= \Pr\left(u_i = c \mid \boldsymbol{U}_i = \boldsymbol{V}_t \right) \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ -\frac{(r_i^s - x_i^s)^2 + \left(r_i^{(1p)} - x_i^{(1p)} \right)^2}{2\sigma^2} \right\} \\ &\quad \times \Pr\left(r_i^{\mathrm{ex}(2)} \mid u_i = c \right). \end{aligned}$$

Similar to what has been introduced in the previous chapter, function α can be recursively computed through the following steps:

Case 1. $i \neq mg + 1$, where $m \in \mathbb{Z}^+$ and $g = \lceil \log_2(n) \rceil$. Knowing that bit u_i is dependent

on the previous t bits, we get:

$$\begin{aligned}
&\Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t, \boldsymbol{r}_1^{i-1}\right) \\
&= \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{U}_{i-1} = \boldsymbol{V}_{t-1}, u_{i-1} = v_1, \boldsymbol{r}_1^{i-2}, \boldsymbol{r}_{i-1}\right) \\
&= \sum_{w':\left(S_{w'}^{i-2}, S_{\bar{w}}^{i-1}\right)\in T_{v_1}} \Pr\left(S_{w'}^{i-2}, S_{\bar{w}}^{i-1}, \boldsymbol{U}_{i-1} = \boldsymbol{V}_{t-1}, \boldsymbol{r}_1^{i-2}, \boldsymbol{r}_{i-1}\right) \\
&= \sum_{w':\left(S_{w'}^{i-2}, S_{\bar{w}}^{i-1}\right)\in T_{v_1}} \left[\Pr\left(\boldsymbol{r}_1^{i-2}\right) \Pr\left(\boldsymbol{U}_{i-1} = \boldsymbol{V}_{t-1}, S_{w'}^{i-2} \mid \boldsymbol{r}_1^{i-2}\right)\right. \\
&\quad \left. \times \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{r}_{i-1} \mid S_{w'}^{i-2}, \boldsymbol{U}_{i-1} = \boldsymbol{V}_{t-1}, \boldsymbol{r}_1^{i-2}\right)\right] \\
&= \sum_{w':\left(S_{w'}^{i-2}, S_{\bar{w}}^{i-1}\right)\in T_{v_1}} \left[\Pr\left(\boldsymbol{r}_1^{i-2}\right) \Pr\left(\boldsymbol{U}_{i-1} = \boldsymbol{V}_{t-1}, S_{w'}^{i-2} \mid \boldsymbol{r}_1^{i-2}\right)\right. \\
&\quad \left. \times \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{r}_{i-1} \mid S_{w'}^{i-2}, \boldsymbol{U}_{i-1} = \boldsymbol{V}_{t-1}\right)\right] \\
&= \sum_{w':\left(S_{w'}^{i-2}, S_{\bar{w}}^{i-1}\right)\in T_{v_1}} \Pr\left(\boldsymbol{r}_1^{i-2}\right) \alpha\left(\boldsymbol{V}_{t-1}, S_{w'}^{i-2}\right) \gamma\left(v_1, \boldsymbol{V}_{t-1}, S_{w'}^{i-2}, S_{\bar{w}}^{i-1}\right),
\end{aligned}$$

$$\begin{aligned}
\Pr\left(\boldsymbol{r}_1^{i-1}\right) &= \sum_{\bar{w}=0}^{15} \sum_{\boldsymbol{V}_t\in\{0,1\}^t} \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t, \boldsymbol{r}_1^{i-1}\right) \\
&= \sum_{\bar{w}=0}^{15} \sum_{\boldsymbol{V}_t\in\{0,1\}^t} \sum_{w':\left(S_{w'}^{i-2}, S_{\bar{w}}^{i-1}\right)\in T_{v_1}} \Pr\left(\boldsymbol{r}_1^{i-2}\right) \alpha\left(\boldsymbol{V}_{t-1}, S_{w'}^{i-2}\right) \gamma\left(v_1, \boldsymbol{V}_{t-1}, S_{w'}^{i-2}, S_{\bar{w}}^{i-1}\right),
\end{aligned}$$

and

$$\begin{aligned}
\alpha\left(\boldsymbol{V}_t, S_{\bar{w}}^{i-1}\right) &= \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t \mid \boldsymbol{r}_1^{i-1}\right) \\
&= \frac{\Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{U}_i = \boldsymbol{V}_t, \boldsymbol{r}_1^{i-1}\right)}{\Pr\left(\boldsymbol{r}_1^{i-1}\right)} \\
&= \frac{\sum_{w':\left(S_{w'}^{i-2}, S_{\bar{w}}^{i-1}\right)\in T_{v_1}} \alpha\left(\boldsymbol{V}_{t-1}, S_{w'}^{i-2}\right) \gamma\left(v_1, \boldsymbol{V}_{t-1}, S_{w'}^{i-2}, S_{\bar{w}}^{i-1}\right)}{\sum_{\bar{w}=0}^{15} \sum_{\boldsymbol{V}_t\in\{0,1\}^t} \sum_{w':\left(S_{w'}^{i-2}, S_{\bar{w}}^{i-1}\right)\in T_{v_1}} \alpha\left(\boldsymbol{V}_{t-1}, S_{w'}^{i-2}\right) \gamma\left(v_1, \boldsymbol{V}_{t-1}, S_{w'}^{i-2}, S_{\bar{w}}^{i-1}\right)}.
\end{aligned}$$

Case 2. $i = mg + 1$, where $m \in \mathbb{Z}^+$ and $g = \lceil \log_2(n) \rceil$. In this case, bit u_i is independent

of any previous bits. So we can simply the above derivations to:

$$\begin{aligned}
\Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{r}_1^{i-1}\right) &= \sum_{c'=0}^{1} \sum_{w':\left(S_{w'}^{i-2}, S_{\bar{w}}^{i-1}\right)\in T_{c'}} \Pr\left(S_{w'}^{i-2}, S_{\bar{w}}^{i-1}, \boldsymbol{r}_1^{i-2}, \boldsymbol{r}_{i-1}\right) \\
&= \sum_{c'=0}^{1} \sum_{w':\left(S_{w'}^{i-2}, S_{\bar{w}}^{i-1}\right)\in T_{c'}} \Pr\left(\boldsymbol{r}_1^{i-2}\right) \Pr\left(S_{w'}^{i-2} \mid \boldsymbol{r}_1^{i-2}\right) \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{r}_{i-1} \mid S_{w'}^{i-2}, \boldsymbol{r}_1^{i-2}\right) \\
&= \sum_{c'=0}^{1} \sum_{w':\left(S_{w'}^{i-2}, S_{\bar{w}}^{i-1}\right)\in T_{c'}} \Pr\left(\boldsymbol{r}_1^{i-2}\right) \Pr\left(S_{w'}^{i-2} \mid \boldsymbol{r}_1^{i-2}\right) \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{r}_{i-1} \mid S_{w'}^{i-2}\right),
\end{aligned}$$

$$\begin{aligned}
\Pr\left(\boldsymbol{r}_1^{i-1}\right) &= \sum_{\bar{w}=0}^{15} \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{r}_1^{i-1}\right) \\
&= \sum_{\bar{w}=0}^{15} \sum_{c'=0}^{1} \sum_{w':\left(S_{w'}^{i-2}, S_{\bar{w}}^{i-1}\right)\in T_{c'}} \Pr\left(\boldsymbol{r}_1^{i-2}\right) \Pr\left(S_{w'}^{i-2} \mid \boldsymbol{r}_1^{i-2}\right) \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{r}_{i-1} \mid S_{w'}^{i-2}\right),
\end{aligned}$$

and

$$\begin{aligned}
\alpha\left(\boldsymbol{V}_0, S_{\bar{w}}^{i-1}\right) &= \Pr\left(S_{\bar{w}}^{i-1} \mid \boldsymbol{r}_1^{i-1}\right) \\
&= \frac{\Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{r}_1^{i-1}\right)}{\Pr\left(\boldsymbol{r}_1^{i-1}\right)} \\
&= \frac{\sum_{c'=0}^{1} \sum_{w':\left(S_{w'}^{i-2}, S_{\bar{w}}^{i-1}\right)\in T_{c'}} \Pr\left(S_{w'}^{i-2} \mid \boldsymbol{r}_1^{i-2}\right) \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{r}_{i-1} \mid S_{w'}^{i-2}\right)}{\sum_{\bar{w}=0}^{15} \sum_{c'=0}^{1} \sum_{w':\left(S_{w'}^{i-2}, S_{\bar{w}}^{i-1}\right)\in T_{c'}} \Pr\left(S_{w'}^{i-2} \mid \boldsymbol{r}_1^{i-2}\right) \Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{r}_{i-1} \mid S_{w'}^{i-2}\right)},
\end{aligned}$$

where

$$\begin{aligned}
\Pr\left(S_{w'}^{i-2} \mid \boldsymbol{r}_1^{i-2}\right) &= \sum_{\boldsymbol{V}_g \in \{0,1\}^g} \Pr\left(S_{w'}^{i-2}, \boldsymbol{U}_{i-1} = \boldsymbol{V}_g \mid \boldsymbol{r}_1^{i-2}\right) \\
&= \sum_{\boldsymbol{V}_g \in \{0,1\}^g} \alpha\left(\boldsymbol{V}_g S_{w'}^{i-2}\right),
\end{aligned}$$

and

$$\begin{aligned}
&\Pr\left(S_{\bar{w}}^{i-1}, \boldsymbol{r}_{i-1} \mid S_{w'}^{i-2}\right) = \Pr\left(u_{i-1} = c'\right) \Pr\left(\boldsymbol{r}_{i-1} \mid \boldsymbol{x}_{i-1}\right) \\
&= \Pr\left(u_{i-1} = c'\right) \frac{1}{\sqrt{2\pi\sigma^2}} \exp\{-\frac{\left(r_i^s - x_i^s\right)^2 + \left(r_i^{(1p)} - x_i^{(1p)}\right)^2}{2\sigma^2}\} \Pr\left(r_i^{\mathrm{ex}(2)} \mid u_i = c\right).
\end{aligned}$$

We next turn to the recursive computation of function β as follows.

Case 1. $i \neq mg$. In this case, the next bit u_{i+1} is dependent on the previous bit u_i. By defining $\boldsymbol{V}_{t+1} = \boldsymbol{V}_t c$, we derive:

$$
\begin{aligned}
&\Pr\left(\boldsymbol{r}_{i+1}^{\ell}, u_i = c, \boldsymbol{U}_i = \boldsymbol{V}_t, S_w^i\right) \\
&= \sum_{c'=0}^{1} \Pr\left(\boldsymbol{r}_{i+1}^{\ell}, u_i = c, \boldsymbol{U}_i = \boldsymbol{V}_t, u_{i+1} = c', S_w^i\right) \\
&= \sum_{c'=0}^{1} \Pr\left(\boldsymbol{r}_{i+1}, \boldsymbol{r}_{i+2}^{\ell}, \boldsymbol{U}_{i+1} = \boldsymbol{V}_{t+1}, u_{i+1} = c', S_w^i, S_{\bar{w}}^{i+1}\right) \\
&= \sum_{c'=0}^{1} \Big[\Pr\left(\boldsymbol{r}_{i+2}^{\ell} \mid u_{i+1} = c', \boldsymbol{U}_{i+1} = \boldsymbol{V}_{t+1}, S_{\bar{w}}^{i+1}\right) \\
&\quad \times \Pr\left(\boldsymbol{r}_{i+1} \mid \boldsymbol{r}_{i+2}^{\ell}, u_{i+1} = c', \boldsymbol{U}_{i+1} = \boldsymbol{V}_{t+1}, S_{\bar{w}}^{i+1}\right) \Pr\left(u_{i+1} = c', \boldsymbol{U}_{i+1} = \boldsymbol{V}_{t+1}, S_{\bar{w}}^{i+1}\right)\Big] \\
&= \sum_{c'=0}^{1} \Big[\beta\left(c', \boldsymbol{V}_{t+1}, S_{\bar{w}}^{i+1}\right) \Pr\left(\boldsymbol{r}_{i+1} \mid u_{i+1} = c', \boldsymbol{U}_{i+1} = \boldsymbol{V}_{t+1}, S_{\bar{w}}^{i+1}\right) \\
&\quad \times \Pr\left(u_{i+1} = c', \boldsymbol{U}_{i+1} = \boldsymbol{V}_{t+1}, S_{\bar{w}}^{i+1}\right)\Big] \\
&= \sum_{c'=0}^{1} \beta\left(c', \boldsymbol{V}_{t+1}, S_{\bar{w}}^{i+1}\right) \Pr\left(\boldsymbol{r}_{i+1}, u_{i+1} = c', \boldsymbol{U}_{i+1} = \boldsymbol{V}_{t+1}, S_{\bar{w}}^{i+1}\right) \\
&= \sum_{c'=0}^{1} \beta\left(c', \boldsymbol{V}_{t+1}, S_{\bar{w}}^{i+1}\right) \Pr\left(\boldsymbol{r}_{i+1}, u_{i+1} = c', \boldsymbol{U}_{i+1} = \boldsymbol{V}_{t+1}, S_{\bar{w}}^{i+1}, S_w^i\right) \\
&= \sum_{c'=0}^{1} \Big[\beta\left(c', \boldsymbol{V}_{t+1}, S_{\bar{w}}^{i+1}\right) \Pr\left(u_{i+1} = c', S_{\bar{w}}^{i+1}, \boldsymbol{r}_{i+1} \mid S_w^i, \boldsymbol{U}_{i+1} = \boldsymbol{V}_{t+1}\right) \\
&\quad \times \Pr\left(S_w^i, \boldsymbol{U}_{i+1} = \boldsymbol{V}_{t+1}\right)\Big] \\
&= \sum_{c'=0}^{1} \beta\left(c', \boldsymbol{V}_{t+1}, S_{\bar{w}}^{i+1}\right) \gamma\left(c', \boldsymbol{V}_{t+1}, S_w^i, S_{\bar{w}}^{i+1}\right) \Pr\left(S_w^i, \boldsymbol{U}_{i+1} = \boldsymbol{V}_{t+1}\right),
\end{aligned}
$$

where

$$\begin{aligned}
\beta\left(c,\, \boldsymbol{V}_t, S_w^i\right) &= \Pr\left(\boldsymbol{r}_{i+1}^{\ell} \mid u_i = c,\, \boldsymbol{U}_i = \boldsymbol{V}_t, S_w^i\right) \\
&= \frac{\Pr\left(\boldsymbol{r}_{i+1}^{\ell}, u_i = c,\, \boldsymbol{U}_i = \boldsymbol{V}_t, S_w^i\right)}{\Pr\left(u_i = c,\, \boldsymbol{U}_i = \boldsymbol{V}_t, S_w^i\right)} \\
&= \frac{\sum_{c'=0}^{1} \beta\left(c',\, \boldsymbol{V}_{t+1}, S_{\bar{w}}^{i+1}\right) \gamma\left(c',\, \boldsymbol{V}_{t+1}, S_w^i, S_{\bar{w}}^{i+1}\right) \Pr\left(S_w^i,\, \boldsymbol{U}_{i+1} = \boldsymbol{V}_{t+1}\right)}{\Pr\left(\boldsymbol{U}_{i+1} = \boldsymbol{V}_{t+1}, S_w^i\right)} \\
&= \sum_{c'=0}^{1} \beta\left(c',\, \boldsymbol{V}_{t+1}, S_{\bar{w}}^{i+1}\right) \gamma\left(c',\, \boldsymbol{V}_{t+1}, S_w^i, S_{\bar{w}}^{i+1}\right).
\end{aligned}$$

Case 2. $i = mg$. In this case, the next bit u_{i+1} is independent of the previous bit u_i. This simplifies our derivation to:

$$\begin{aligned}
&\Pr\left(\boldsymbol{r}_{i+1}^{\ell}, u_i = c,\, \boldsymbol{U}_i = \boldsymbol{V}_t, S_w^i\right) \\
&= \sum_{c'=0}^{1} \Pr\left(\boldsymbol{r}_{i+1}, \boldsymbol{r}_{i+2}^{\ell}, u_i = c,\, \boldsymbol{U}_i = \boldsymbol{V}_t, u_{i+1} = c', S_w^i, S_{\bar{w}}^{i+1}\right) \\
&= \sum_{c'=0}^{1} \left[\Pr\left(\boldsymbol{r}_{i+2}^{\ell} \mid u_{i+1} = c', S_{\bar{w}}^{i+1}\right) \Pr\left(\boldsymbol{r}_{i+1} \mid \boldsymbol{r}_{i+2}^{\ell}, u_{i+1} = c', u_i = c,\, \boldsymbol{U}_i = \boldsymbol{V}_t, S_{\bar{w}}^{i+1}\right)\right. \\
&\quad \left. \times \Pr\left(u_{i+1} = c', u_i = c,\, \boldsymbol{U}_i = \boldsymbol{V}_t, S_{\bar{w}}^{i+1}\right)\right] \\
&= \sum_{c'=0}^{1} \beta\left(c',\, \boldsymbol{V}_0, S_{\bar{w}}^{i+1}\right) \Pr\left(\boldsymbol{r}_{i+1}, u_{i+1} = c', u_i = c,\, \boldsymbol{U}_i = \boldsymbol{V}_t, S_{\bar{w}}^{i+1}\right) \\
&= \sum_{c'=0}^{1} \left[\beta\left(c',\, \boldsymbol{V}_0, S_{\bar{w}}^{i+1}\right) \Pr\left(u_{i+1} = c', S_{\bar{w}}^{i+1}, \boldsymbol{r}_{i+1} \mid S_w^i, u_i = c,\, \boldsymbol{U}_i = \boldsymbol{V}_t\right)\right. \\
&\quad \left. \times \Pr\left(S_w^i, u_i = c,\, \boldsymbol{U}_i = \boldsymbol{V}_t\right)\right] \\
&= \sum_{c'=0}^{1} \beta\left(c',\, \boldsymbol{V}_0, S_{\bar{w}}^{i+1}\right) \Pr\left(u_{i+1} = c', S_{\bar{w}}^{i+1}, \boldsymbol{r}_{i+1} \mid S_w^i\right) \Pr\left(S_w^i, u_i = c,\, \boldsymbol{U}_i = \boldsymbol{V}_t\right) \\
&= \sum_{c'=0}^{1} \beta\left(c',\, \boldsymbol{V}_0, S_{\bar{w}}^{i+1}\right) \gamma\left(c',\, \boldsymbol{V}_0, S_w^i, S_{\bar{w}}^{i+1}\right) \Pr\left(S_w^i, u_i = c,\, \boldsymbol{U}_i = \boldsymbol{V}_t\right),
\end{aligned}$$

where

$$\begin{aligned}\beta\left(c,\,\boldsymbol{V}_t,S_w^i\right) &= \frac{\Pr\left(\boldsymbol{r}_{i+1}^{\ell},u_i=c,\boldsymbol{U}_i=\boldsymbol{V}_t,S_w^i\right)}{\Pr\left(u_i=c,\boldsymbol{U}_i=\boldsymbol{V}_t,S_w^i\right)}\\ &= \sum_{c'=0}^{1}\beta\left(c',\,\boldsymbol{V}_0,S_{\bar{w}}^{i+1}\right)\gamma\left(c',\,\boldsymbol{V}_0,S_w^i,S_{\bar{w}}^{i+1}\right).\end{aligned}$$

The initial values of functions α and β are:

$$\alpha\left(\boldsymbol{V}_0,S_0^0\right)=1\ ,\ \alpha\left(\boldsymbol{V}_0,S_w^0\right)=0 \text{ for every } \ 0<w\le 16$$

$$\beta\left(0,\,\boldsymbol{V}_k,S_w^{\ell}\right)=\beta\left(1,\,\boldsymbol{V}_k,S_w^{\ell}\right)=\tfrac{1}{2} \text{ for every } 0\le w\le 16$$

where $k\equiv(\ell-1)\bmod g$.

It remains to discuss about the iterative decoding scheme based on our newly derived functions. A well-known iterative decoding equation is to iteratively exchange the so-called extrinsic information as follows.

$$\begin{aligned}&\Lambda^{(1)}(i)=\Lambda_{\text{ch}}^{(1)}(i)+\Lambda_{\text{ap}}^{(1)}(i)+\Lambda_{\text{ex}}^{(1)}(i)\\ \Longleftrightarrow\ &\Lambda_{\text{ex}}^{(1)}(i)=\Lambda^{(1)}(i)-\Lambda_{\text{ch}}^{(1)}(i)-\Lambda_{\text{ap}}^{(1)}(i)\\ &=\Lambda^{(1)}(i)-\ln\frac{\Pr\left(r_i^s\mid u_i=1\right)}{\Pr\left(r_i^s\mid u_i=0\right)}-\ln\frac{\Pr\left(r_i^{(ex)}\mid u_i=1\right)}{\Pr\left(r_i^{(ex)}\mid u_i=0\right)}\\ &=\begin{cases}\Lambda^{(1)}(i)-\frac{2}{\sigma^2}r_i^s & \text{if } \bar{\sigma}^2=0\\ \Lambda^{(1)}(i)-\frac{2}{\sigma^2}r_i^s-\frac{2\bar{M}}{\bar{\sigma}^2}r_i^{(ex)} & \text{otherwise}\end{cases},\end{aligned}$$

where $\Lambda_{\text{ch}}(i)$ and $\Lambda_{\text{ap}}(i)$ stand for the quantities due to channel and *a priori* information, respectively.

After updating the extrinsic information by the first component decoder, it is the turn of the second component decoder to modify the extrinsic information. Let $\{\tilde{u}\}_1^{\ell}$ be the input sequence after interleaving. Assume that $(\tilde{u})_1^{\ell}$ is i.i.d. in statistics; hence the traditional BCJR algorithm described in Section 2.4 can be applicable. Let $\Lambda^{(2)}(i)$ be the log-likelihood

ratio updated in Decoder 2. Then we have:

$$\begin{aligned}
&\tilde{\Lambda}^{(2)}(i) = \tilde{\Lambda}^{(2)}_{\text{ch}}(i) + \tilde{\Lambda}^{(2)}_{\text{ap}}(i) + \tilde{\Lambda}^{(2)}_{\text{ex}}(i) \\
\Longleftrightarrow\ &\tilde{\Lambda}^{(2)}_{\text{ex}}(i) = \tilde{\Lambda}^{(2)}(i) - \tilde{\Lambda}^{(2)}_{\text{ch}}(i) - \tilde{\Lambda}^{(2)}_{\text{ap}}(i) \\
&\qquad = \tilde{\Lambda}^{(2)}(i) - \ln \frac{\Pr\left(\tilde{r}^s_i \mid \tilde{u}_i = 1\right)}{\Pr\left(\tilde{r}^s_i \mid \tilde{u}_i = 0\right)} - \ln \frac{\Pr\left(\tilde{u}_i = 1\right)}{\Pr\left(\tilde{u}_i = 0\right)} \\
&\qquad = \tilde{\Lambda}^{(2)}(i) - \frac{2}{\sigma^2}\tilde{r}^s_i - \tilde{\Lambda}^{(1)}_{\text{ex}}(i)
\end{aligned}$$

where $\{\tilde{r}\}_1^\ell$ is the interleaved sequence for $\{r\}_1^\ell$, and $\{\tilde{\Lambda}^{(1)}_{\text{ex}}(i)\}_{i=1}^\ell$ is the interleaved sequence for $\{\Lambda^{(1)}_{\text{ex}}(i)\}_{i=1}^\ell$.

Based on [5], we slightly adjust the extrinsic information equation for Decoder 2 as follows. Again, let $\{\Lambda^{(2)}_{\text{ex}}(i)\}_{i=1}^\ell$ be the de-interleaved sequence for $\{\tilde{\Lambda}^{(2)}_{\text{ex}}(i)\}_{i=1}^\ell$. Then,

$$\begin{aligned}
&\Lambda^{(2)}_{\text{ex}}(i) \\
&= v\Lambda^{(2)}_{\text{ex}}(i) + (1-v)\ln\left[\frac{\sum_{\boldsymbol{V}_t\in\{0,1\}^t} \Pr\left(u_i = 1 \mid \boldsymbol{U}_i = \boldsymbol{V}_t\right)\Pr\left(\hat{u}_{i-t} = v_t\right)\ldots\Pr\left(\hat{u}_{i-1} = v_1\right)}{\sum_{\boldsymbol{V}_t\in\{0,1\}^t} \Pr\left(u_i = 0 \mid \boldsymbol{U}_i = \boldsymbol{V}_t\right)\Pr\left(\hat{u}_{i-t} = v_t\right)\ldots\Pr\left(\hat{u}_{i-1} = v_1\right)}\right]
\end{aligned}$$

where

$$\Pr(\hat{u}_i = 0) = \frac{1}{1 + e^{\Lambda^{(2)}_{\text{ex}}(i)}},\quad \Pr(\hat{u}_i = 1) = \frac{e^{\Lambda^{(2)}_{\text{ex}}(i)}}{1 + e^{\Lambda^{(2)}_{\text{ex}}(i)}},$$

and

$$v = \begin{cases} 1, & \text{if } (i-1) \bmod g = 0 \\ 0.85, & \text{otherwise} \end{cases}$$

Chapter 5

Simulation Results

In this chapter, simulation results are provided to demonstrate the performance of the FLEC that we derived in Chapters 3 and 4. Specifically, we examine what has been proposed in Chapter 3 in Section 5.1, and test the FLEC in Chapter 4 in Section 5.2. Remarks on our simulation results are given in Section 5.3.

5.1 Joint Source-Channel Block Code

Throughout this section, the code rate of FLECs examined is fixed as 1/4. The first-order Markov sources are the chosen source distributions. Three Markov source cases will be simulated.

Case 1. Transition distribution $\Pr(u_i = 0 \mid u_{i-1} = 0) = \Pr(u_i = 1 \mid u_{i-1} = 1) = 0.9$ with initial probability $\Pr(u_1 = 0) = 1 - \Pr(u_1 = 1) = 0.9$.

Case 2. Transition distribution $\Pr(u_i = 0 \mid u_{i-1} = 0) = \Pr(u_i = 1 \mid u_{i-1} = 1) = 0.95$ with initial probability $\Pr(u_1 = 0) = 1 - \Pr(u_1 = 1) = 0.5$.

Case 3. Transition distribution $\Pr(u_i = 0 \mid u_{i-1} = 0) = \Pr(u_i = 1 \mid u_{i-1} = 1) = 0.55$ with initial probability $\Pr(u_1 = 0) = 1 - \Pr(u_1 = 1) = 0.55$.

Below we illustrate the details of all figures in Section 5.1.

1. Figure 5.1 compares the SER performances between the JSC block codes constructed by exhaustive search and those constructed by our algorithm. The source under test follows Case 1.

2. The setting of Figure 5.2 is same as that in Figure 5.1 except that the source follows Case 2.

3. The setting of Figure 5.3 is same as that in Figure 5.1 except that the source follows Case 3.

4. Figure 5.4 shows the SER performances of JSC block codes constructed by the algorithm introduced in Section 3.2 and decoded by the sequential soft-decision decoding algorithm introduced in Section 3.3 for three different codeword lengths. The source follows Case 1.

5. The setting of Figure 5.5 is same as that of Figure 5.4 except that the source follows Case 2.

6. The setting of Figure 5.6 is same as that of Figure 5.4 except that the source follows Case 3.

7. Figure 5.7 compares the SER performances between the JSC block code we constructed and the tandem scheme. The hard-decision decoding scheme is presumed and the source under test follows Case 1. The codeword length of the JSC block code is 64. The tandem scheme selected for performance comparision is the concatenation of a Huffman code with an $(n, k, t) = (63, 16, 11)$ BCH code. Since the tandem scheme becomes variable length in nature, we use the Levenshtein distance [8] to account for the symbol errors.

8. The setting of Figure 5.8 is same as that of Figure 5.7 except the source follows Case 2.

9. The setting of Figure 5.9 is same as that of Figure 5.7 except the source follows Case 3.

Remarks and observations regarding these figures will be presented in Section 5.3.

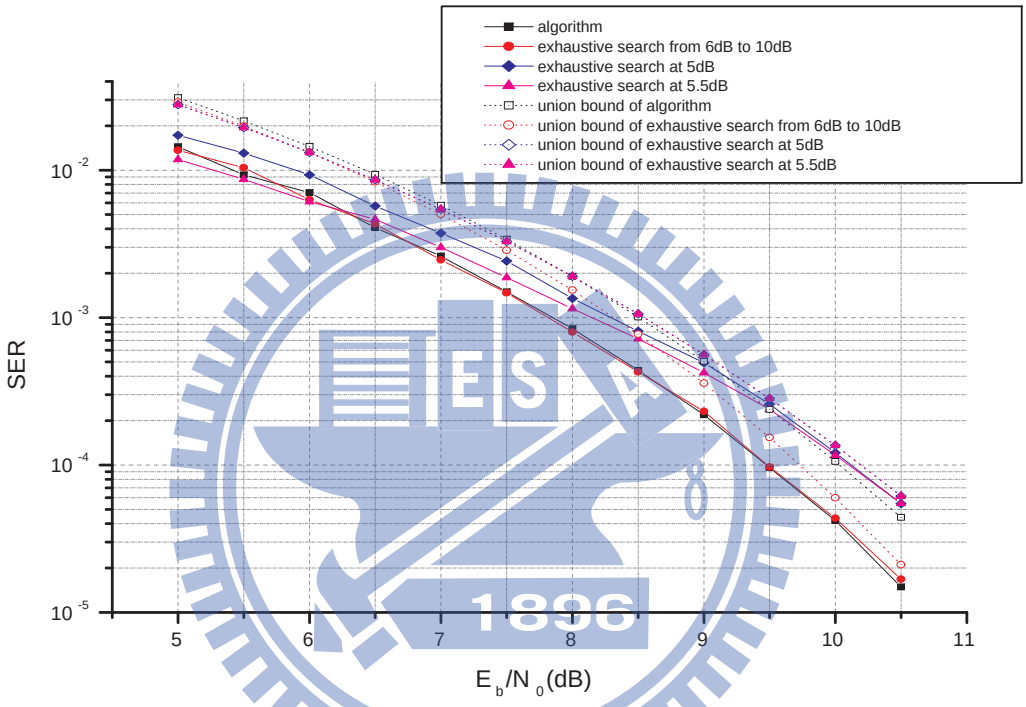


Figure 5.1: Performance comparison between the joint source-channel block code constructed by exhaustive search and that by our proposed algorithm with codeword length $\ell = 8$. The source follows Case 1 and hard-decision decoding is employed.

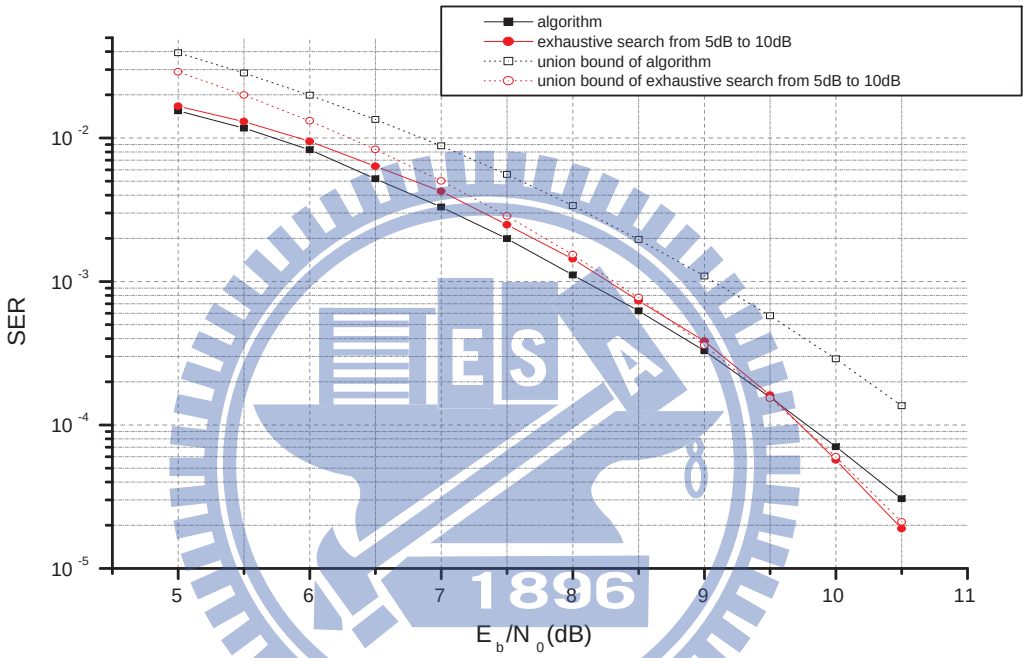


Figure 5.2: Performance comparison between the joint source-channel block code constructed by exhaustive search and that by our proposed algorithm with the codeword length $\ell = 8$. The source follows Case 2 and hard-decision decoding is employed.

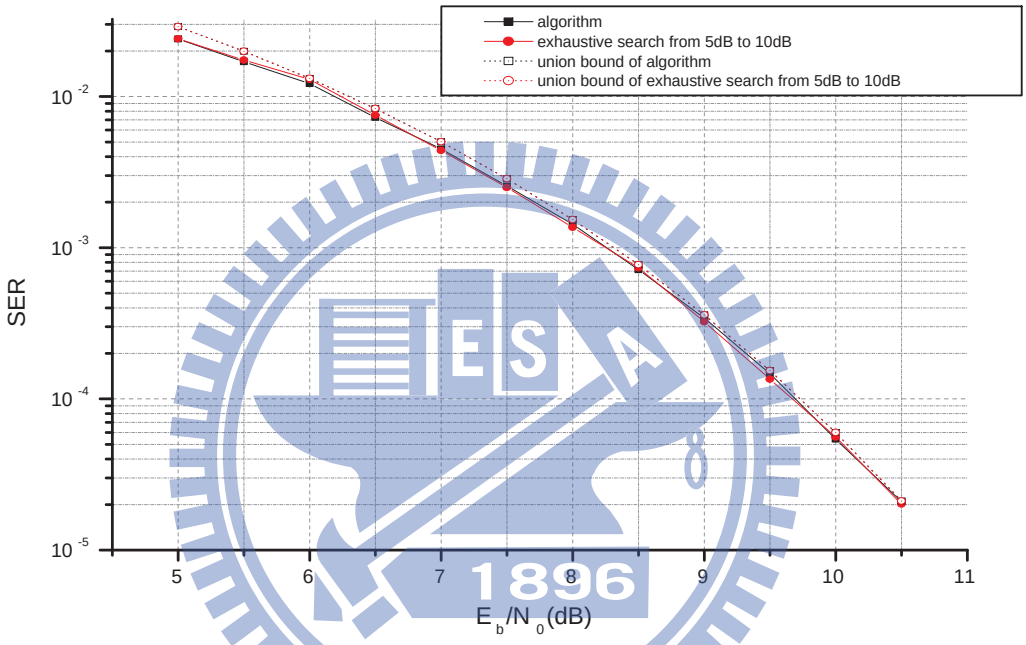


Figure 5.3: Performance comparison between the joint source-channel block code constructed by exhaustive search and that by our proposed algorithm with the codeword length $\ell = 8$. The source follows Case 3 and hard-decision decoding is employed.

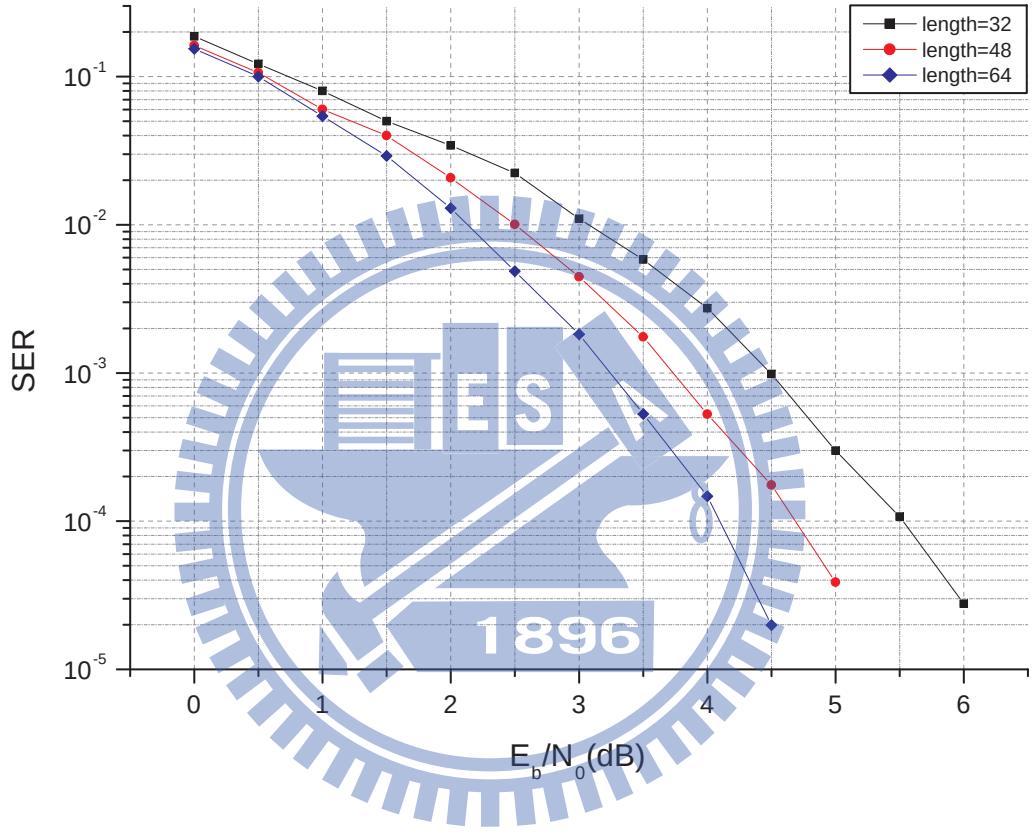


Figure 5.4: Performance comparison of the proposed joint source-channel block codes for different codeword length. The source follows Case 1 and soft-decision decoding is employed.

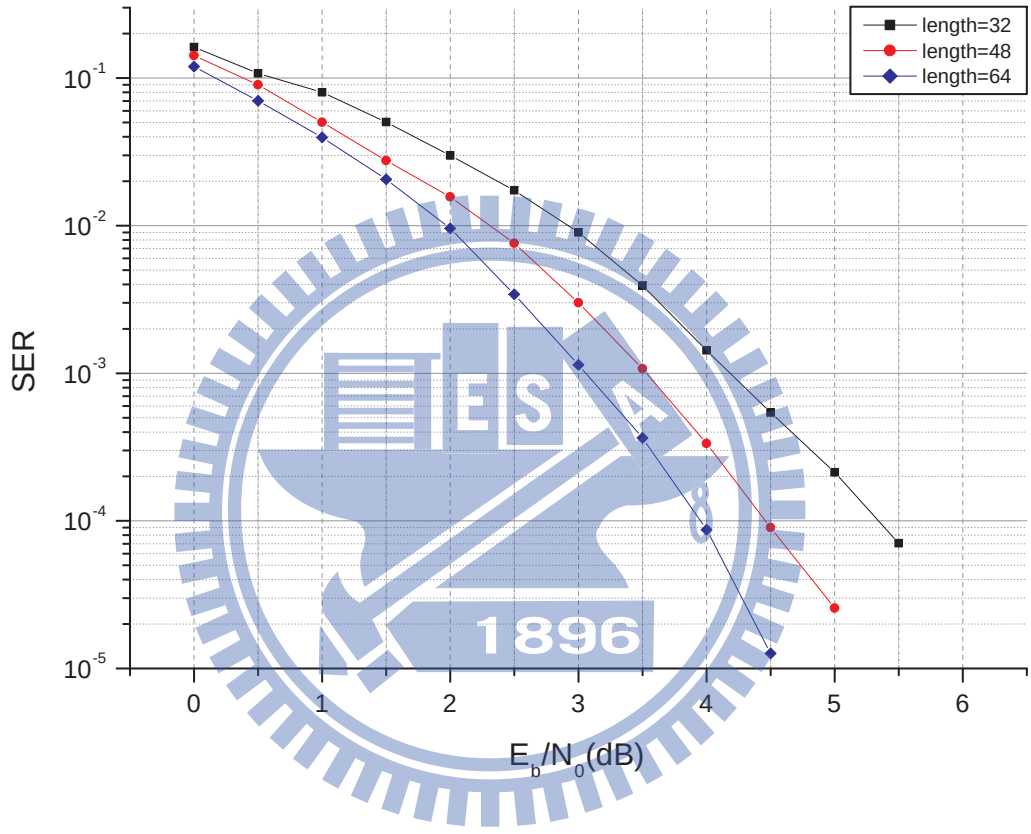


Figure 5.5: Performance comparison of the proposed joint source-channel block codes for different codeword length. The source follows Case 2 and soft-decision decoding is employed.

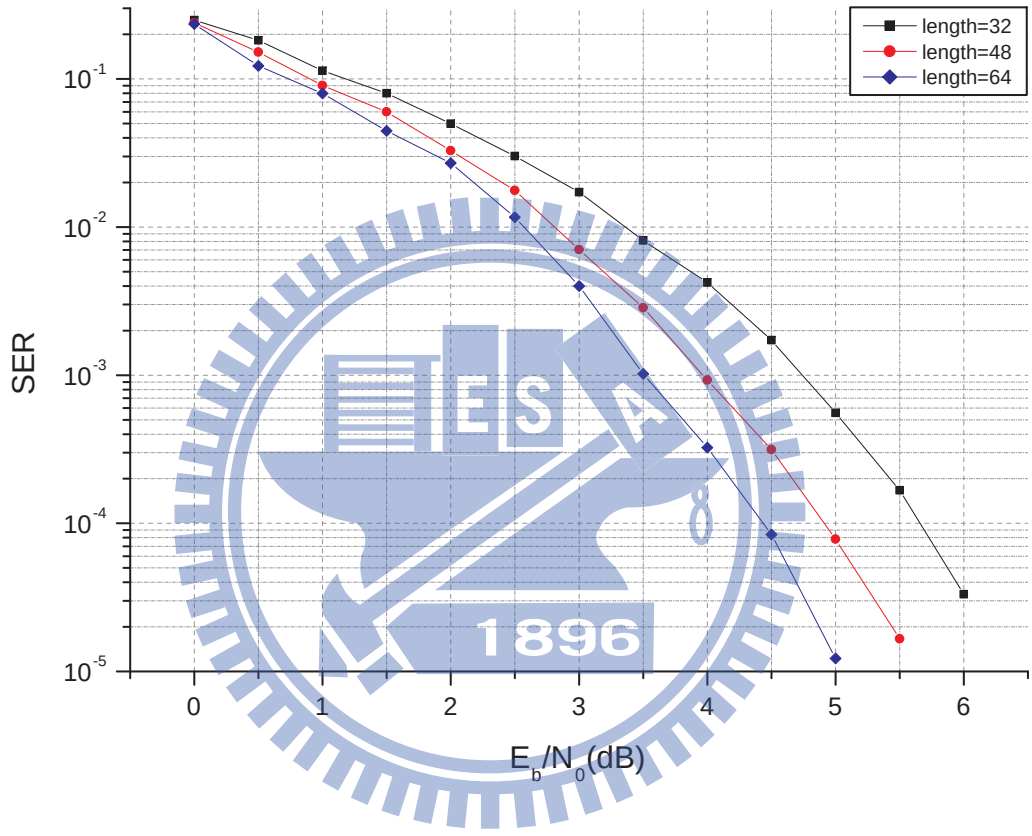


Figure 5.6: Performance comparison of the proposed joint source-channel block codes for different codeword length. The source follows Case 3 and soft-decision decoding is employed.

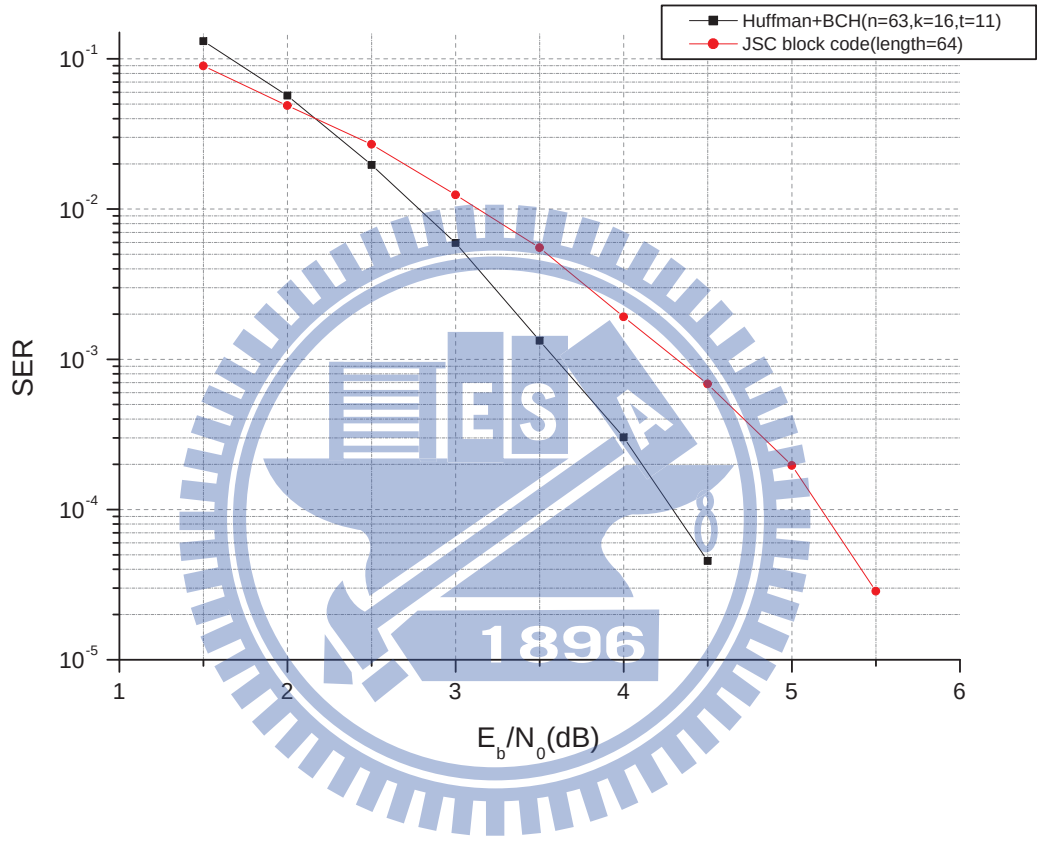


Figure 5.7: Performance comparison between the joint source-channel block code with codeword length 64 and the tandem scheme (i..e, the Huffman code + $(63, 16, 11)$ BCH code). The source follows Case 1 and hard-decision decoding is employed.

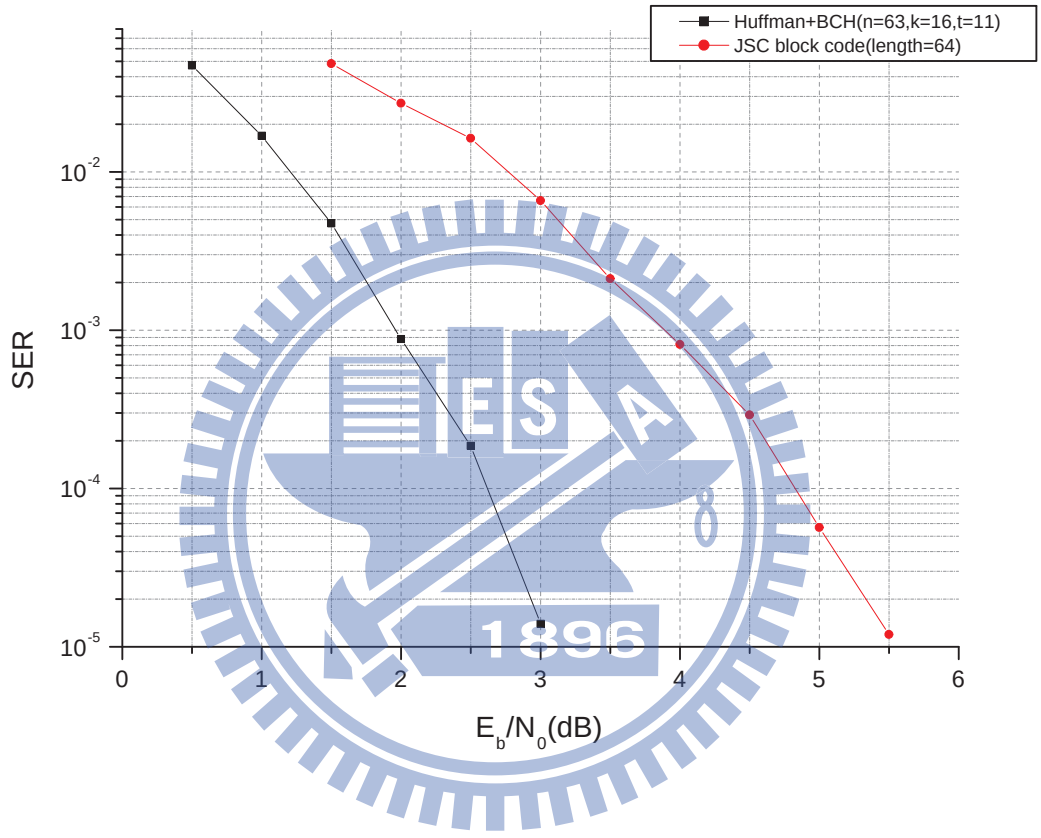


Figure 5.8: Performance comparison between the joint source-channel block code with codeword length 64 and the tandem scheme (i..e, the Huffman code + $(63, 16, 11)$ BCH code). The source follows Case 2 and hard-decision decoding is employed.

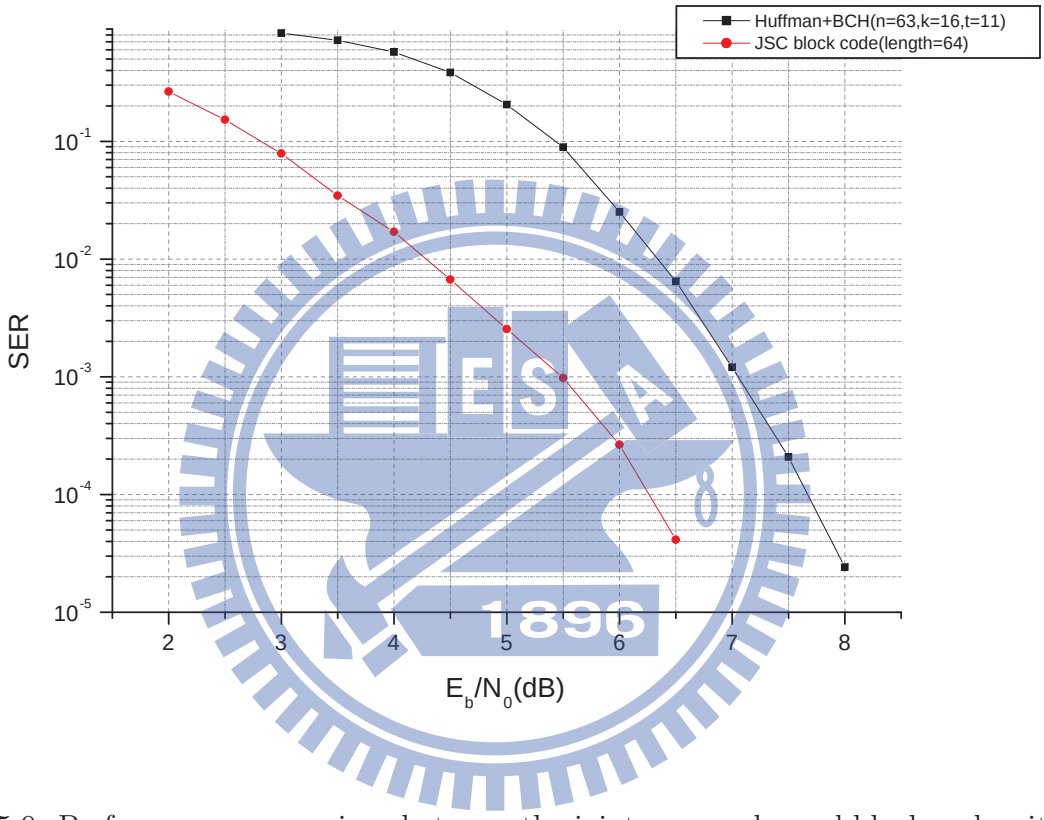


Figure 5.9: Performance comparison between the joint source-channel block code with codeword length 64 and the tandem scheme (i..e, the Huffman code + $(63, 16, 11)$ BCH code). The source follows Case 3 and hard-decision decoding is employed.

5.2 Modified Turbo Code

For all cases in Section 5.2, the number of iterations for turbo decoder is set as 18.

Below we illustrate the details of all figures in Section 5.2.

1. Figure 5.10 compares the SER performance between the modified JSC turbo code and a tandem scheme. The tandem scheme we choose for comparison in this section is the concatenation of the Huffman code with a traditional turbo code of interleaver size $\ell = 128 \times 128$. The number of source symbols is 16. The source symbol is resulted from a group of four bits, each of which is generated according to binary non-uniform source with $p_0 = 0.55$ and $p_1 = 0.45$. The resulting source entropy per symbol is 3.9711 bits.

2. The setting in Figure 5.11 is the same as that in Figure 5.10 except $p_0 = 0.58$. The resulting source entropy per symbol is 3.92582 bits.

3. The setting in Figure 5.12 is the same as that in Figure 5.10 except $p_0 = 0.634$. The resulting source entropy per symbol is 3.79021 bits.

4. The setting of Figure 5.13 is same as that in Figure 5.10 except $p_0 = 0.65$. The resulting source entropy per symbol is 3.73627 bits.

5. The setting of Figure 5.14 is same as that in Figure 5.10 except $p_0 = 0.686$. The resulting source entropy per symbol is 3.59095 bits.

6. The setting of Figure 5.15 is same as that in Figure 5.10 except $p_0 = 0.7$. The resulting source entropy per symbol is 3.52516 bits.

7. The setting of Figure 5.16 is same as that in Figure 5.10 except $p_0 = 0.75$. The resulting source entropy per symbol is 3.24512 bits.

8. Figure 5.17 compares the SER performance between two different source statistics of the same source entropy. The two different sources distributions are randomly generated.

9. Figure 5.18 compares the SER performances between the modified JSC turbo code and a tandem scheme. The tandem scheme chosen is the concatenation of Huffman code and a traditional turbo code with interleaver size $\ell = 128 \times 128$. The source is the 26 English alphabet, and its statistics is tabulated in Table 5.1.

10. The setting of Figure 5.19 is same as that in Figure 5.18 except the source alphabet now consists of only 16 most likely letters in Table 5.1. We normalize the distribution for 16-English-letter alphabet to make their probabilities sum to one (cf. Table 5.2).

11. The settings of Figures 5.20 and 5.21 are the same as that in Figure 5.19 except the interleaver sizes are changed to $\ell = 64 \times 64$ and 256×256, respectively.

12. Figure 5.22 compares the SER performances of the modified JSC turbo code under three different interleaver sizes. The source alphabet contains the normalized most probable 16 English letters.

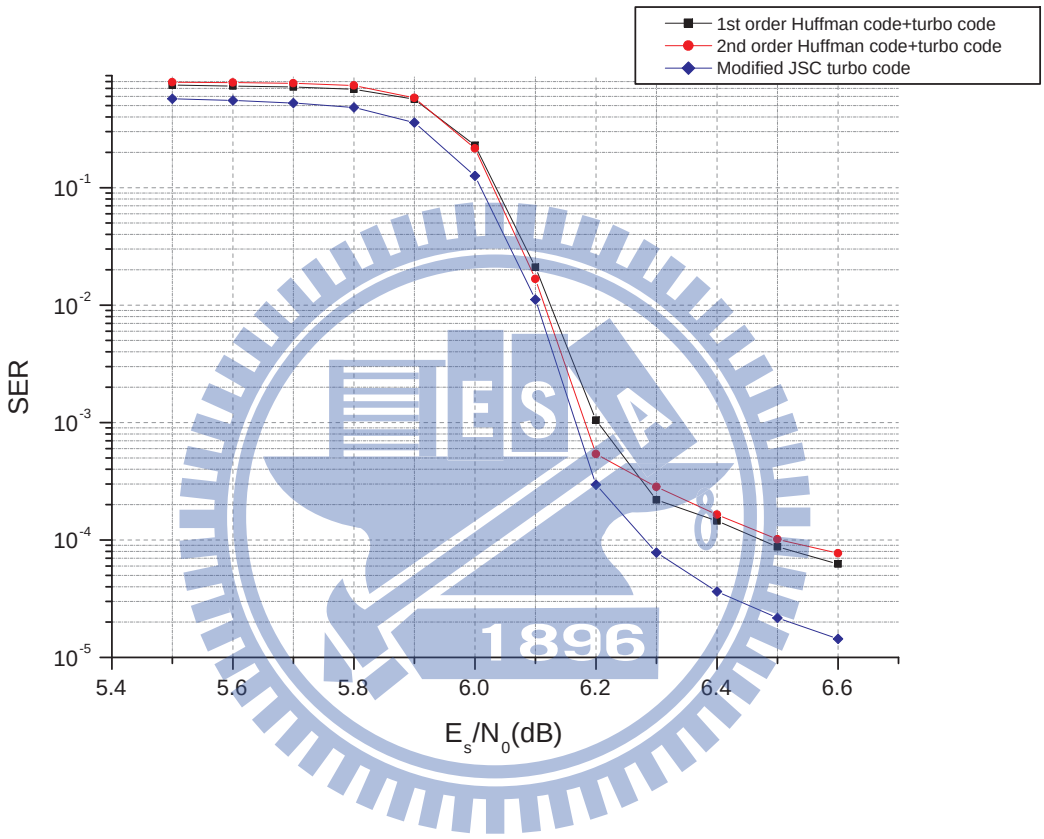


Figure 5.10: Performance comparison between the joint source-channel turbo code and the tandem scheme (i.e., the Huffman code + a traditional turbo code) under the source with per-letter source entropy 3.9711 bits. The interleaver size is $\ell = 128 \times 128$.

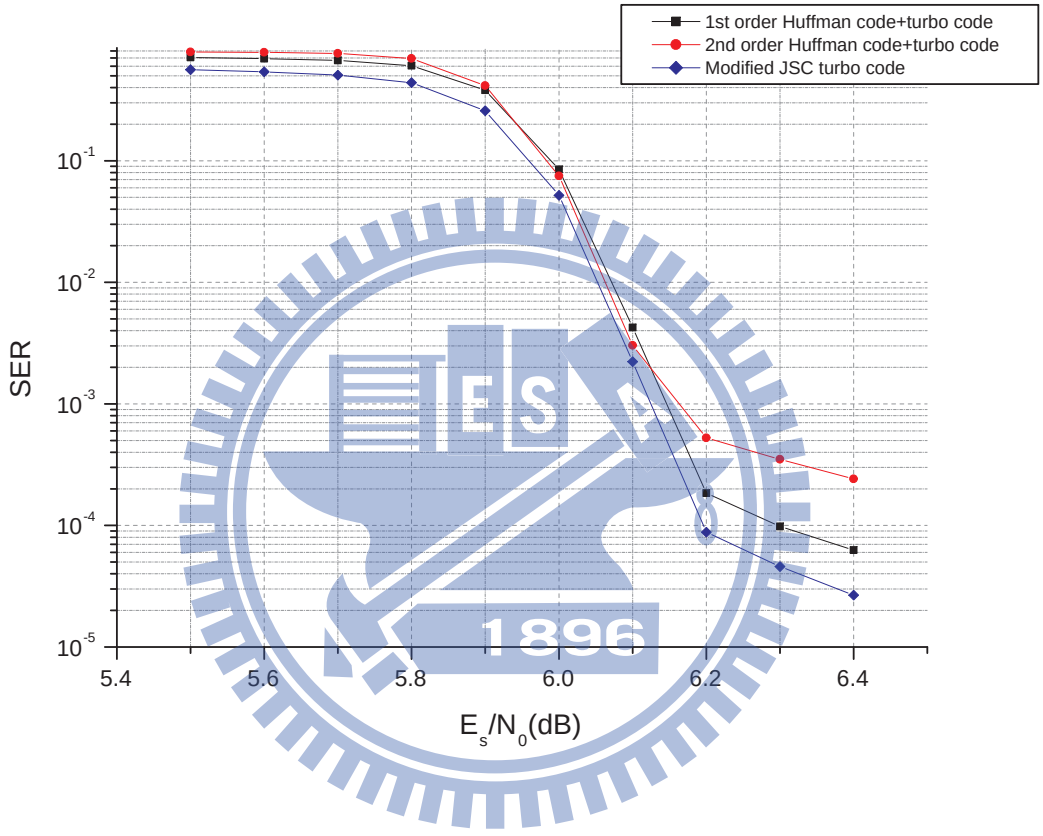


Figure 5.11: Performance comparison between the joint source-channel turbo code and the tandem scheme (i.e., the Huffman code + a traditional turbo code) under the source with per-letter source entropy 3.92582 bits. The interleaver size is $\ell = 128 \times 128$.

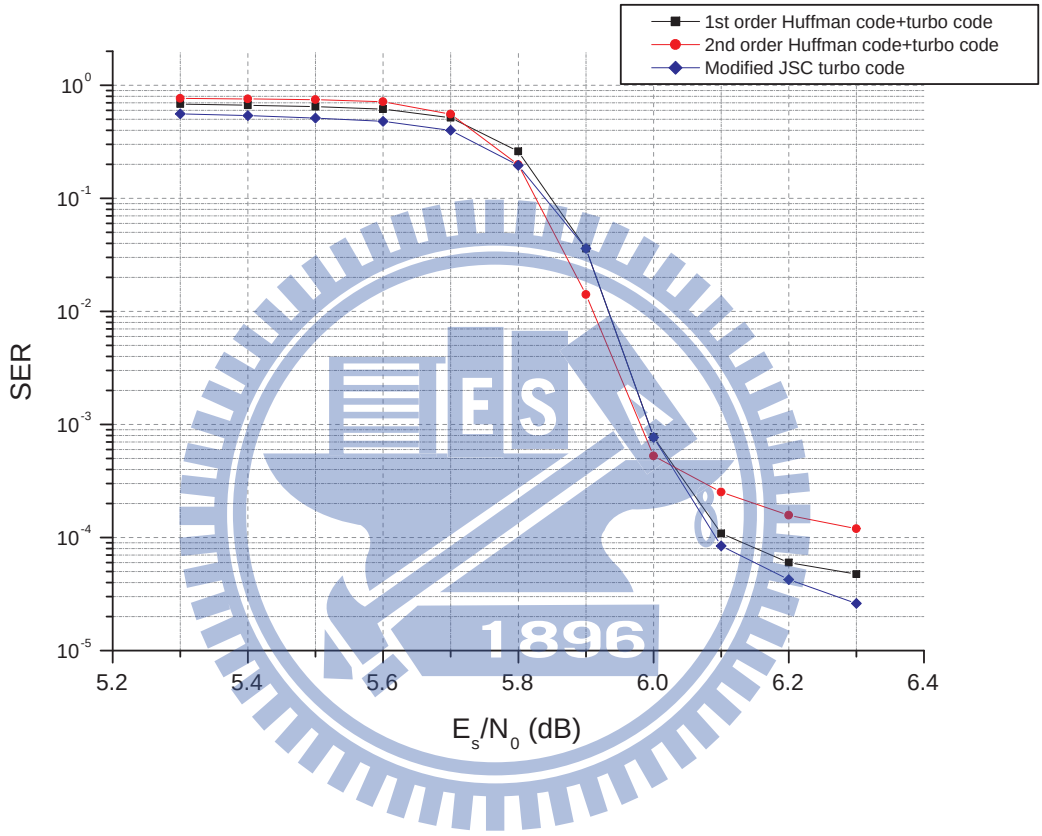


Figure 5.12: Performance comparison between the joint source-channel turbo code and the tandem scheme (i.e., the Huffman code + a traditional turbo code) under the source with per-letter source entropy 3.79021 bits. The interleaver size is $\ell = 128 \times 128$.

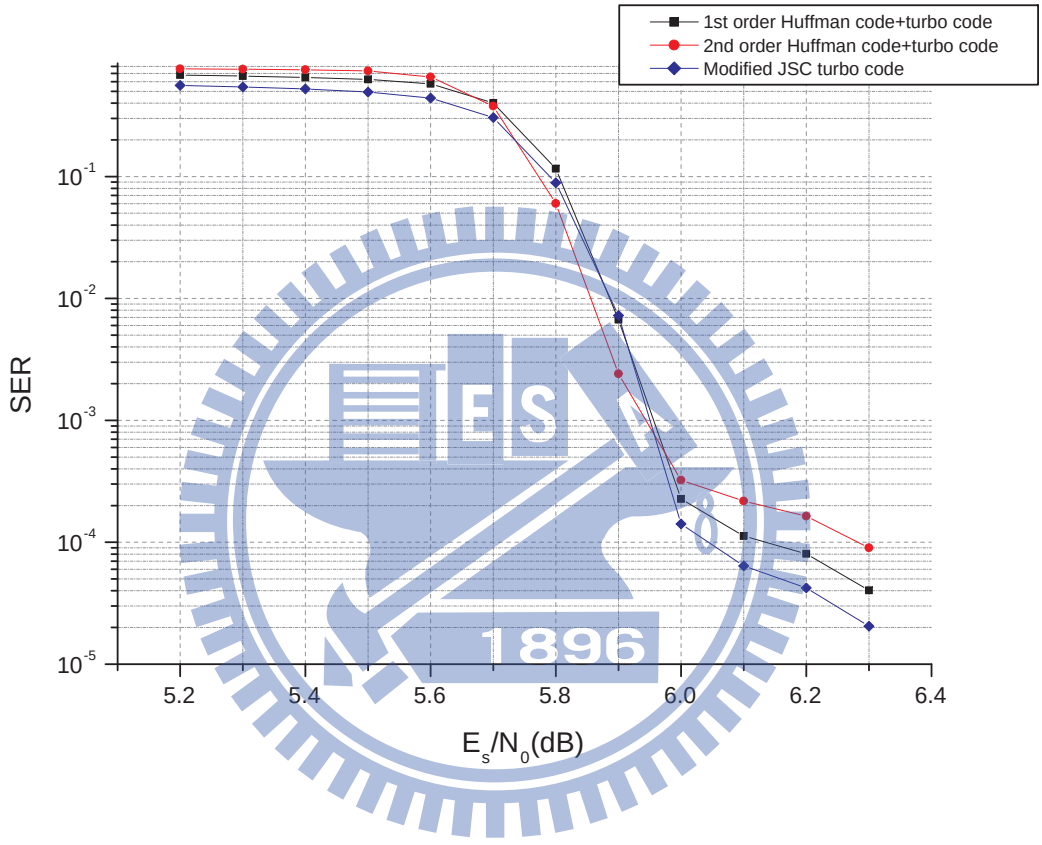


Figure 5.13: Performance comparison between the joint source-channel turbo code and the tandem scheme (i.e., the Huffman code + a traditional turbo code) under the source with per-letter source entropy 3.73627 bits. The interleaver size is $\ell = 128 \times 128$.

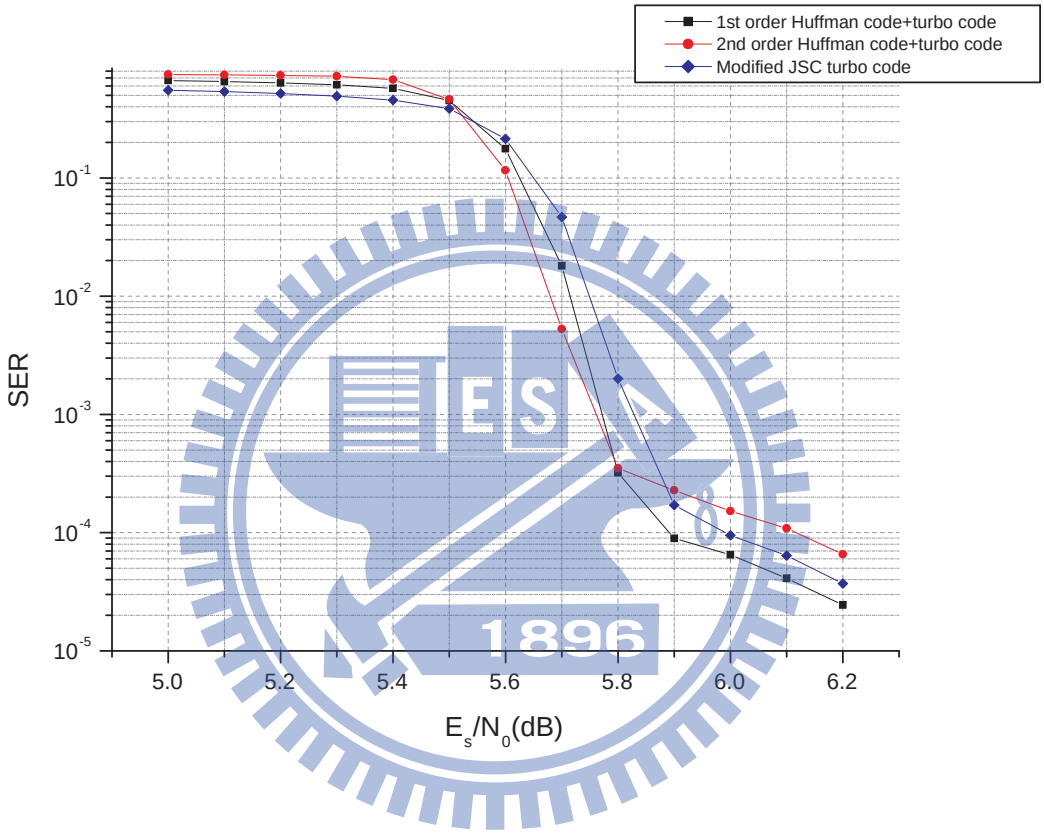


Figure 5.14: Performance comparison between the joint source-channel turbo code and the tandem scheme (i.e., the Huffman code + a traditional turbo code) under the source with per-letter source entropy 3.59095 bits. The interleaver size is $\ell = 128 \times 128$.

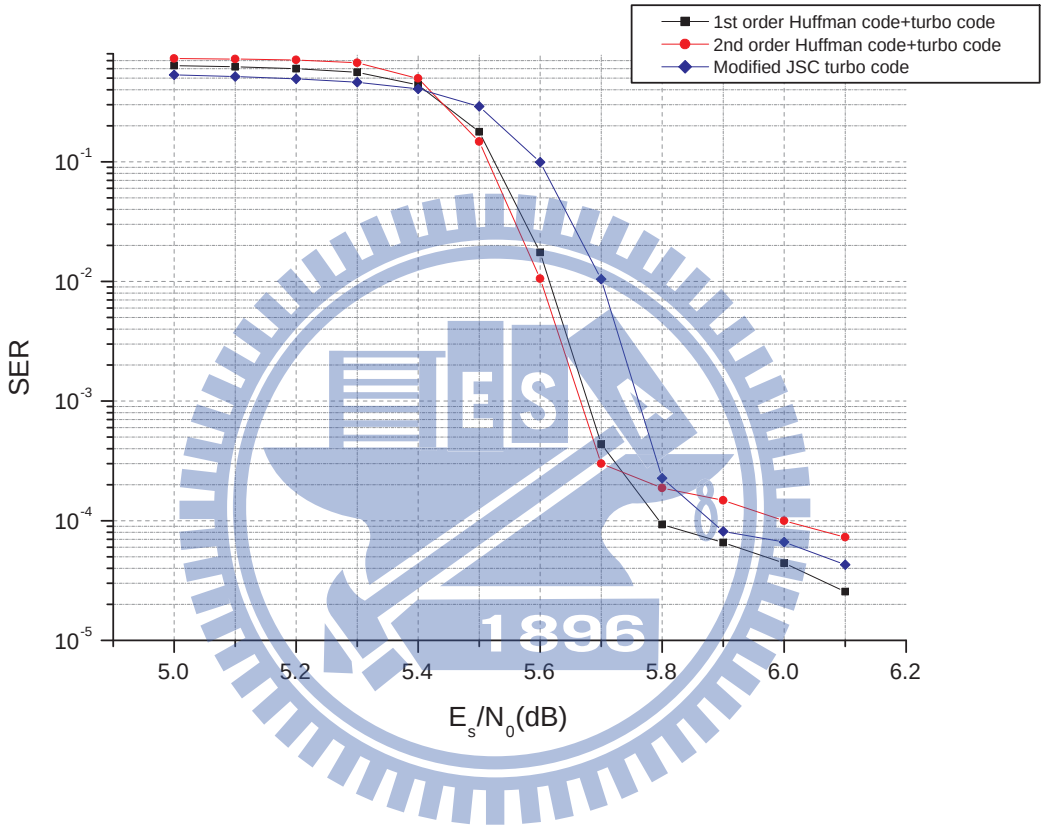


Figure 5.15: Performance comparison between the joint source-channel turbo code and the tandem scheme (i.e., the Huffman code + a traditional turbo code) under the source with per-letter source entropy 3.52516 bits. The interleaver size is $\ell = 128 \times 128$.

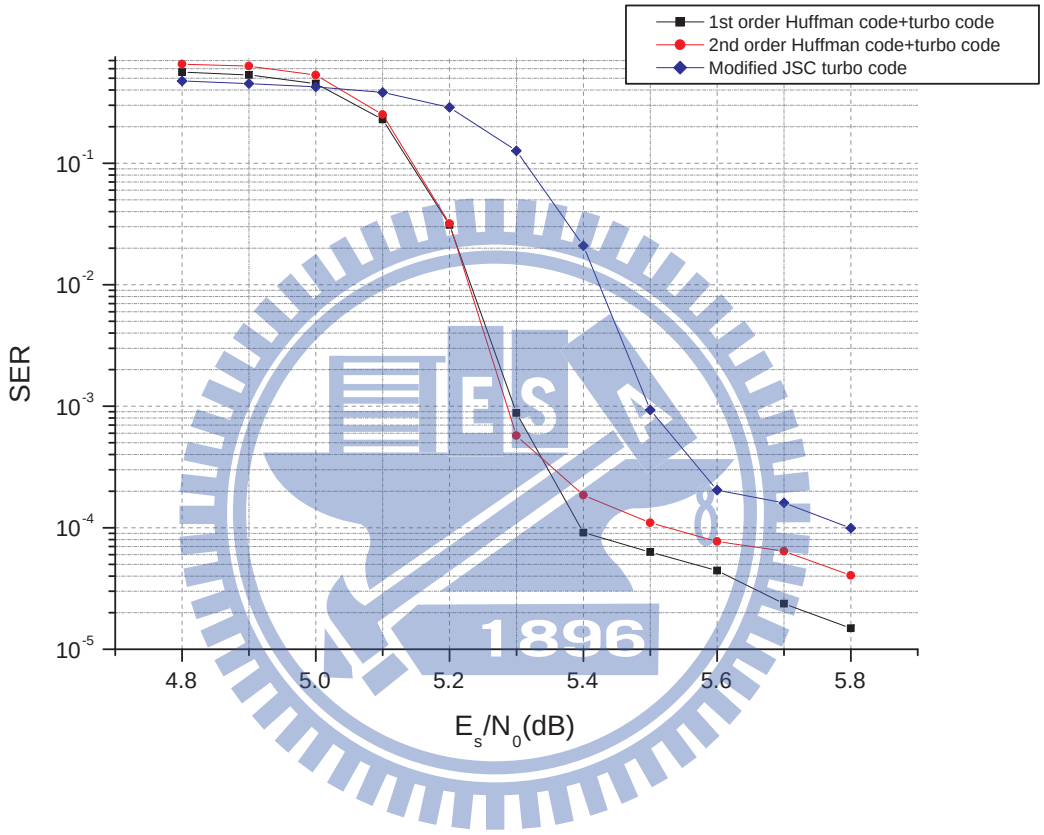


Figure 5.16: Performance comparison between the joint source-channel turbo code and the tandem scheme (i.e., the Huffman code + a traditional turbo code) under the source with per-letter source entropy 3.24512 bits. The interleaver size is $\ell = 128 \times 128$.

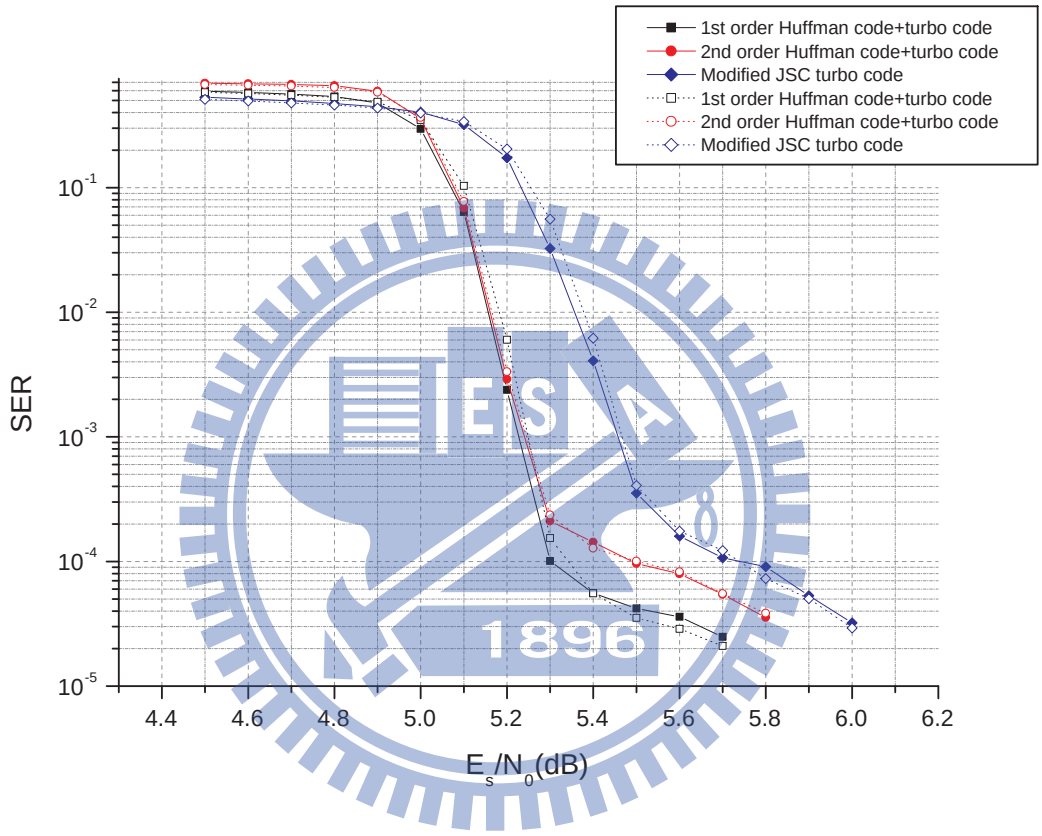


Figure 5.17: Performance comparison under two different source distributions with the same per-letter source entropy 3.20 bits. The interleaver size is $\ell = 128 \times 128$.

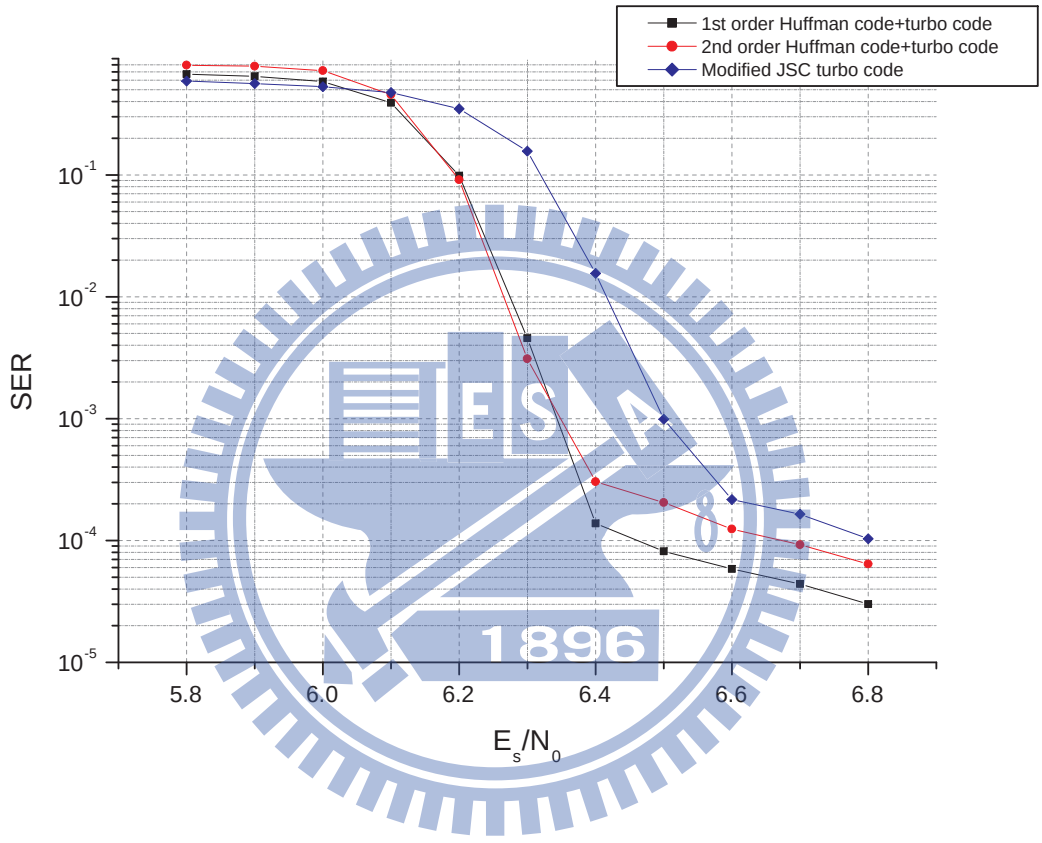


Figure 5.18: Performance comparison between the joint source-channel turbo code and the tandem scheme (i.e., the Huffman code + a traditional turbo code) under the 26-English-letter text source. The interleaver size is $\ell = 128 \times 128$.

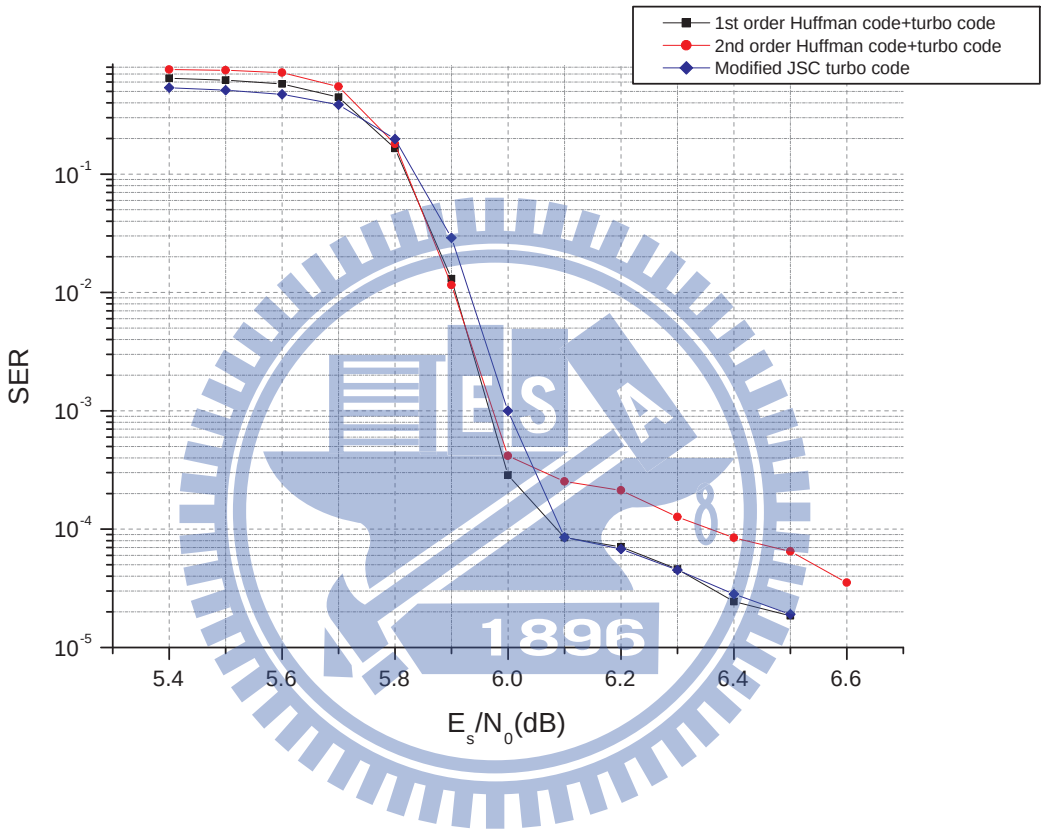


Figure 5.19: Performance comparison between the joint source-channel turbo code and the tandem scheme (i.e., the Huffman code + a traditional turbo code) under the normalized 16-English-letter text source. The interleaver size is $\ell = 128 \times 128$.

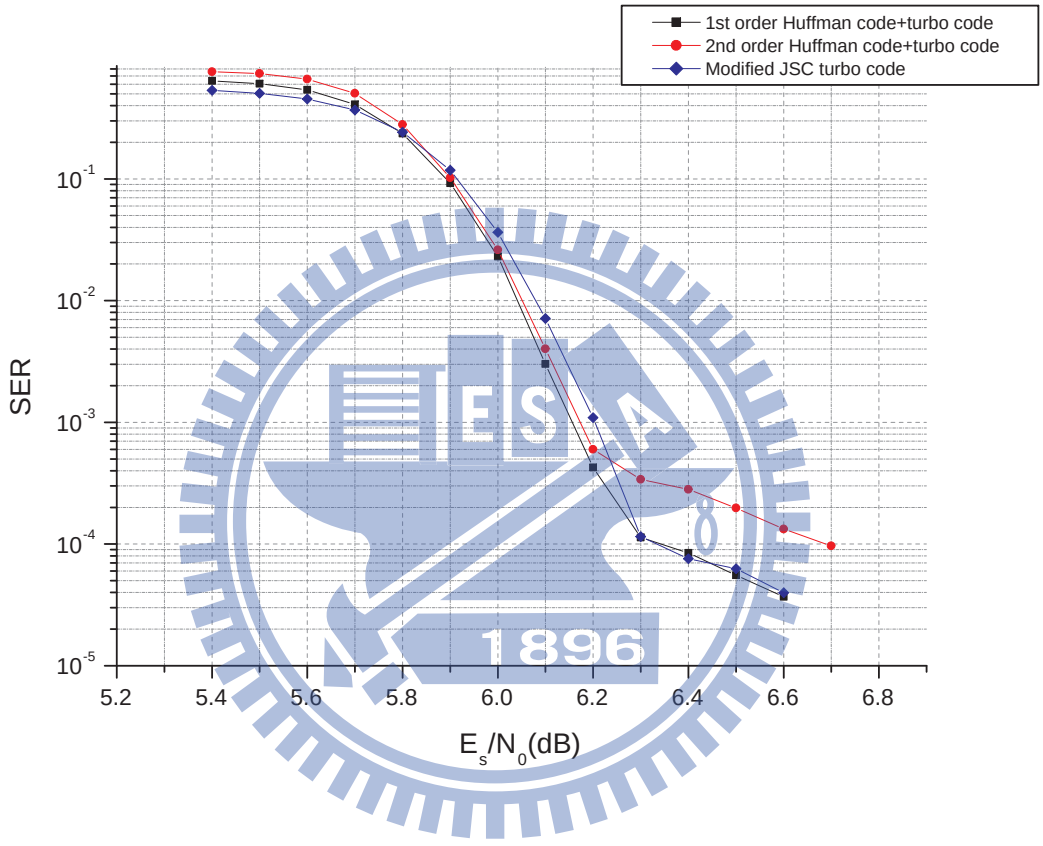


Figure 5.20: Performance comparison between the joint source-channel turbo code and the tandem scheme (i.e., the Huffman code + a traditional turbo code) under the normalized 16-English-letter text source. The interleaver size is $\ell = 64 \times 64$.

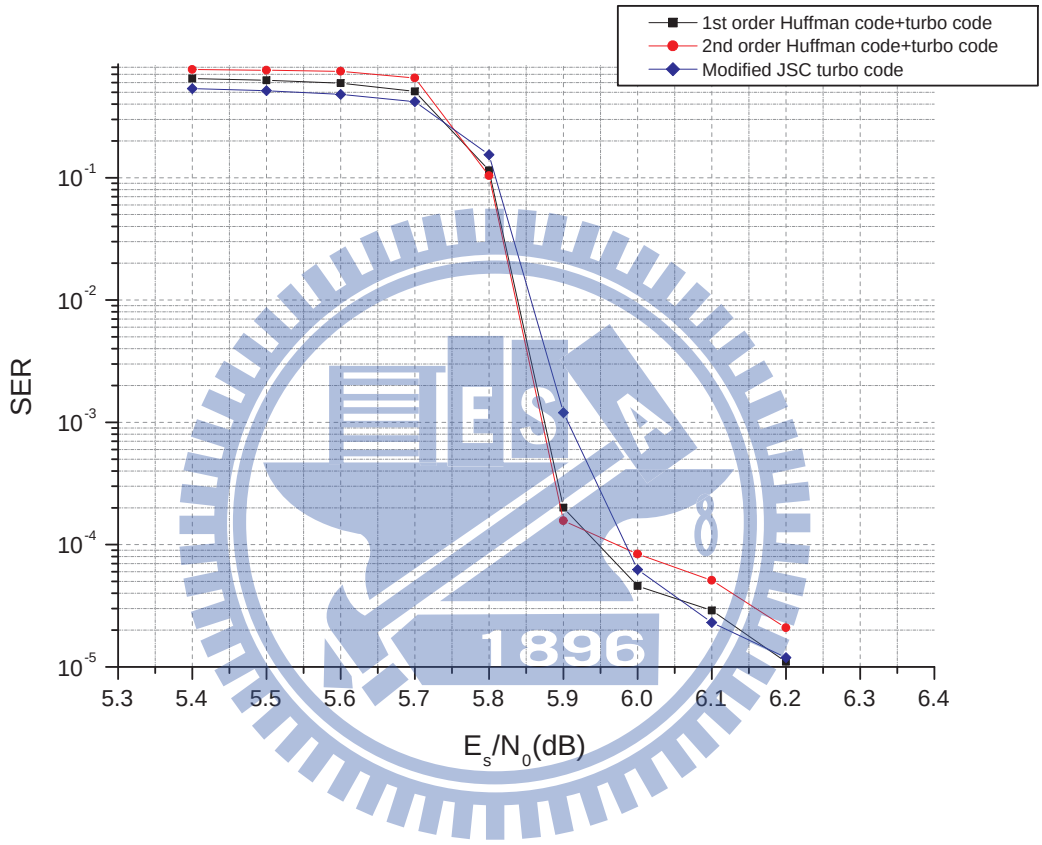


Figure 5.21: Performance comparison between the joint source-channel turbo code and the tandem scheme (i.e., the Huffman code + a traditional turbo code) under the normalized 16-English-letter text source. The interleaver size is $\ell = 256 \times 256$.

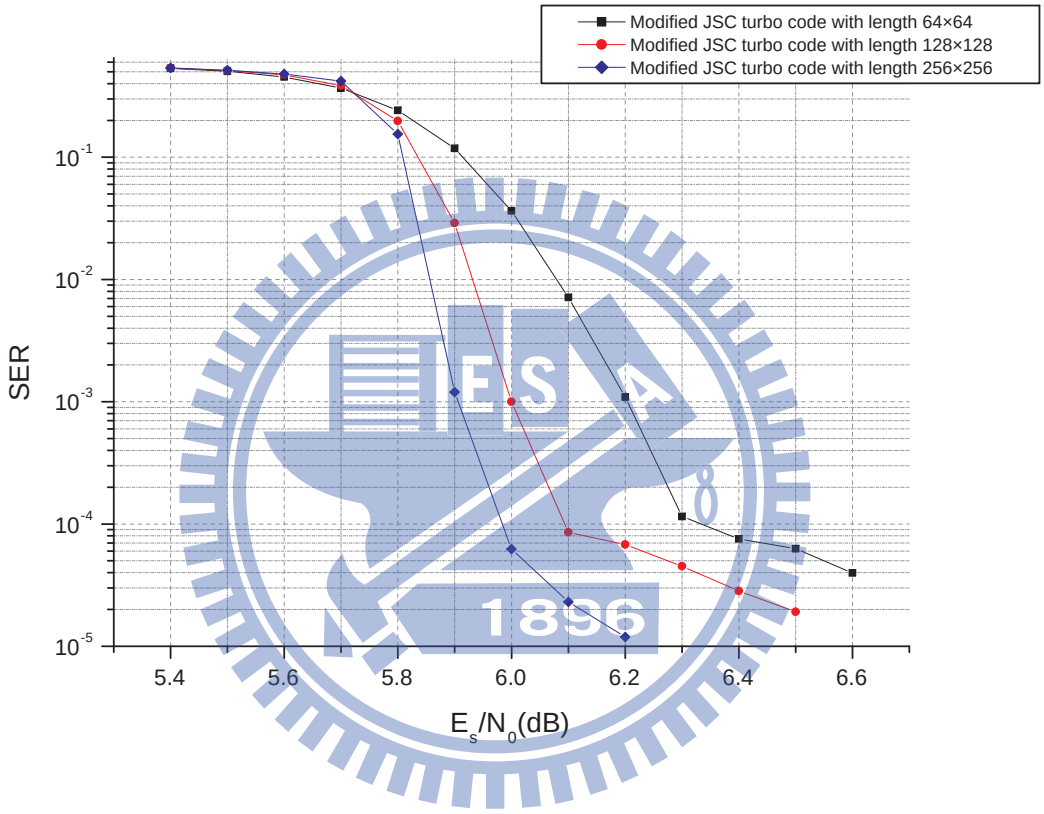


Figure 5.22: Performance comparison of the modified JSC turbo code by testing different interleaver sizes under the normalized 16-English-letter text source.

Table 5.1: Source distribution of 26 English alphabet symbols. The source entropy is 4.12091 bits.

English alphabet	probability	English alphabet	probability
E	0.14878610	T	0.09354149
A	0.08833733	O	0.07245769
R	0.06872164	N	0.06498532
H	0.05831331	I	0.05644515
S	0.05537763	D	0.04376834
L	0.04123298	U	0.02762209
P	0.02575393	F	0.02455297
M	0.02361889	C	0.02081665
W	0.01868161	G	0.01521216
Y	0.01521216	B	0.01267680
V	0.01160928	K	0.00867360
X	0.00146784	J	0.00080064
Q	0.00080064	Z	0.00053376

Table 5.2: Normalized source distribution of the 16 most probable English letters. The source entropy is 3.78611 bits.

English alphabet	probability	English alphabet	probability
E	0.1627270	T	0.1023060
A	0.0966141	O	0.0792466
R	0.0751605	N	0.0710741
H	0.0637770	I	0.0617338
S	0.0605662	D	0.0478692
L	0.0450963	U	0.0302101
P	0.0281670	F	0.0268535
M	0.0258319	C	0.0227671

5.3 Observations and remarks

In Figures 5.1 and 5.3, we can observe that the SER performance of the JSC block code constructed by our proposed algorithm in Section 3.2 is very close to that of the code constructed by exhaustive search under codeword length $\ell = 8$ for high E_b/N_0, where the exhaustive search approach finds the code that minimizes the SER union bound derived in Section 3.1 at a fixed E_b/N_0 (e.g., 5 dB) or at a per-E_b/N_0 basis (e.g., from 6 dB to 10 dB). Figures 5.1 and 5.2 even show that our code construction algorithm can sometimes produce a JSC block code that performs better than the one that minimizes the union bound when $E_b/N_0 \leq 6$ dB and 9.5 dB, respectively. This is because that the union bound is a rough bound; so its minimization does not implies the optimality in terms of the error probability.

Figures 5.4-5.6 indicate that the SERs of the constructed JSCs (or FLECs) decrease as the codeword length ℓ grows. Specifically, the FLEC of length $\ell = 64$ has around 1.4 dB gain over the FLEC of length $\ell = 32$ at SER $= 10^{-4}$ in both Figures 5.4 and 5.5. In Figure 5.6, this gain is reduced to 1.2 dB but still the trend of improvement by extending the codeword length remains evident. However, we fail to generate FLECs of length larger than 64 as the complexity of the code construction is infeasibly high.

We next turn to the performance comparison between the constructed JSC block code and a benchmark tandem scheme. The benchmark tandem scheme consists of a Huffman code concatenated with a BCH code. We group 16 bits as a block input to the Huffman code for compression. For a fair comparison, the codeword length and code rate of the codes to be compared are made similar. In Figures 5.7 and 5.8, we observe that the performance of our constructed JSC block code is far behind the benchmark tandem scheme; but in Figure 5.9, an opposite result is obtained that the tandem scheme performs worse than our constructed JSC block code. From these three figures, we conclude that due to its fixed-length nature,

our constructed JSC block code is specially suitable for sources with near-uniform statistics such as Case 3. As a quantitative index, by defining the compression efficiency of a source as

$$1 - \frac{\text{(average codeword length of Huffman compressed outputs)}}{\text{(codeword length of uncompressed symbols)}},$$

we obtain the compression efficiencies of the three sources considered are:

$$\begin{cases} \text{Case 1}: & 0.5283 \\ \text{Case 2}: & 0.6665 \\ \text{Case 3}: & 0.0054 \end{cases}$$

Hence, we may say that our constructed JSC block code is good for sources with low compression efficiencies (i.e., for sources with a more "uniform" statistics).

We now inspect the proposed modified turbo code for non-uniform sources. The results are summarized in Figures 5.10-5.21. We compare our modified turbo code with two tandem schemes: the concatenation of a first-order Huffman code with a conventional turbo code, and the concatenation of a second-order Huffman code with a conventional turbo code. For the two tandem schemes to be compared with, we note from our simulations that the waterfall region of the tandem scheme with a 2nd order Huffman code is in general improved in comparison with the one with a 1st order Huffman code, but at a price of a higher error floor.

In details, we observe from Figures 5.10-5.13 that the SER performances of the proposed modified JSC turbo code are better than both tandem schemes in the error floor region. However, in Figures 5.14 and 5.15, the error floor of the modified JSC turbo code becomes higher than the tandem scheme with a 1st order Huffman code. One possible cause is that the sources used in Figures 5.14 and 5.15 have lower source entropies and hence are more "non-uniform" in statistics. As expected, in Figures 5.16 and 5.17, the error floor of the modified JSC turbo code is above both tandem schemes when the source entropy rates further decrease.

In Figure 5.18, it can be seen that the two tandem schemes have about 0.2 dB advantage over the modified JSC turbo code in the waterfall region when the source is the 26-English text. Such an advantage is perhaps due to that in our design, we transform forcefully the 26-English letter into a 5-bit format, which inevitably introduce additional redundancy. In order to confirm the above interpretation, we modify the English text source by selecting only the 16 most probably letters and normalize their probabilities such that their probabilities sum to one. As such, we can use a 4-bit representation for these 16 letters. It can then be observed from Figure 5.19 that the error floor of the modified JSC turbo code is approaching that of the tandem scheme with 1 1st order Huffman code, and is below the other tandem scheme. This, to some extent, supports our interpretation for the result in Figure 5.18.

We also simulate the impact of different interleaver sizes on the performances of our modified JSC turbo code in Figures 5.20 and 5.21, and summarize the results in Figure 5.22. It can be observed that at SER $= 10^{-4}$, taking a larger interleaved size of $\ell = 256 \times 256$ has 0.35 dB gain over that of $\ell = 64 \times 64$. Hence, a larger interleaved size would help improving the performance.

We end this chapter by remarking that the entropy of the normalized 16 most probable English letter source is 3.78611 bit per letter, which is larger than the original 26-English alphabet source whose entropy rate is 3.73627 bits per letter. As the source entropy only increases $0.0576 = 10 \log_{10}(3.78611/3.73627)$ dB, the improvement of our modified JSC turbo code in performance, when switching from a source with a lower entropy to one with a higher entropy, is actually larger than this number in the error floor region. In fact, there does not seem to have an evident relation between the source entropy and error performance; but somehow they are vaguely correlated. Further study to identify their relation is required.

Chapter 6

Conclusion and Future Work

In this thesis, a code construction method for joint source-channel (JSC) block codes and the corresponding low-complexity sub-optimal decoding algorithm are proposed. Since the proposed code construction method cannot generate codes of long block length, we subsequently propose a modified turbo code (in particular the decoder) as a joint-source block code that can be used for non-uniform sources and long block length. For sources with low Huffman compression efficiency, both proposed codes outperform the tandem scheme formed by concatenating a Huffman code with a properly selected channel code. Since the proposed joint source-channel block code has fixed block length, it does not require an end-of-codeword detection or indication scheme at the receiver; hence, its practice is much easier than the traditional variable-length joint source-channel code.

At the current stage, the proposed code construction algorithm for joint source-channel block codes costs too much complexity as codeword length grows. Thus, it may be necessary and also interesting to reduce the complexity of the proposed code construction algorithm in the future. On the other hand, in the modified joint source-channel turbo coding system, the information of the source statistics is only used in the outer decoder, but not in the inner BCJR decoder (that still assumes that the information bit sequence is binary i.i.d. with uniform marginal distribution after interleaving). In general, the assumption made by the

inner BCJR algorithm is not valid; hence, how to incorporate the knowledge of the given source statistics into the inner decoder metrics is another future work of both practical and theoretical interest.

Bibliography

[1] C.E. Shannon, "A Mathematical Theory of Communication," *Bell Syst. Tech. J.*,vol. 27, pp. 379 - 423 and pp. 623 - 656, July and October 1948.

[2] Vembu S., Verdú,S. and Steinberg, Y., "The Source-Channel Separation Theorem Revisited," *IEEE Trans. Inform. Theory*, vol. 41, no. 1, pp. 44 - 54, January 1995.

[3] V.Buttigieg and P.G.Farrell, "Variable-length error-correcting codes," *IEE Proc. Commun.*,vol. 147, no. 4, pp. 211-215, August 2000.

[4] Guang-Chong Zhu and Fady Alajaji, "Turbo codes for nonuniform memoryless sources over noisy channels," *IEEE Commun. Letters*,vol. 6, no. 2, pp. 64 - 66, February 2002.

[5] Guang-Chong Zhu and Fady Alajaji, "Joint source-channel turbo coding for binary Markov sources," *IEEE Trans. Wireless Commun.*,vol. 5, no. 5, pp. 1065 - 1075, May 2006.

[6] Claude Berrou and Alain Glavieux, "Near optimum error correcting coding and decoding: turbo-codes," *IEEE Trans. Commun.*, vol. 44, no. 10, pp. 1261 - 1271, October 1996.

[7] Giulio Colavolpe, Gianluigi Ferrari and Riccardo Raheli, "Extrinsic information in turbo decoding: a unified view," *Proc of Globecom'99*, pp. 505 - 509, 1999.

[8] V.I. Levenshtein, "Binary codes capable of correcting deletions insertions and reversals," *Sov. Phys. Doklady*, vol. 10, no. 8, pp. 707 - 710, February 1966.