Chapter 6

Concatenated codes

The previous chapters presented the elementary laws of encoding like BCH, Reed-Solomon or CRSC codes. Most of these elementary codes are asymptotically good, in the sense that their minimum Hamming distances (MHD) can be made as large as we want, by sufficiently increasing the degree of the generator polynomials. The complexity of the decoders is unfortunately unacceptable for the degrees of polynomials that would guarantee the MHD required by practical applications.

A simple means of having codes with a large MHD and nevertheless easily decodable is to combine several reasonably-sized elementary codes, in such a way that the resulting global code has a high error correction capability. The decoding is performed in steps, each of them corresponding to one of the elementary encoding steps. The first composite encoding scheme was proposed by Forney during work on his thesis in 1965, called concatenated codes [6.4]. In this scheme, a first encoder, called the outer encoder, provides a codeword that is then re-encoded by a second encoder, called the inner encoder. If the two codes are systematic, the concatenated code is itself systematic. In the rest of this chapter, only systematic codes will be considered.

Figure 6.1(a) shows a concatenated code, as imagined by Forney, and the corresponding step decoder. The most judicious choice of constituent code is an algebraic code, typically a Reed-Solomon code, for the outer code, and a convolutional code for the inner code. The inner decoder is then the Viterbi decoder, which easily takes advantage of the soft values provided by the demodulator, and the outer decoder, which works on symbols with several bits (for example, 8 bits), can handle errors in bursts at the output of the first decoder. A permutation or interleaving function inserted between the two encoders, and its inverse function placed between the two decoders, can greatly increase the robustness of the concatenated code (Figure 6.1(b)). Such an encoding scheme has worked very successfully in applications as varied as deep space transmissions and digital, satellite and terrestrial television broadcasting. In particular, it is the encoding scheme adopted in many countries for digital terrestrial television [6.1].

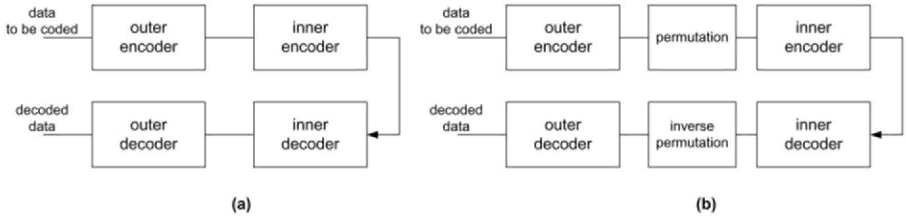

Figure 6.1 – serial concatenated code, (a) without and (b) with permutation. In both cases, the output of the outer encoder is entirely recoded by the inner encoder.

Nowadays, this first version of concatenated codes is called *serial concatenation* (SC). Its decoding, presented in Figures 6.1 is not optimal. Indeed, even if, locally, the two elementary decoders are optimal, the simple sequencing of these two decodings is not globally optimal as the inner decoder does not take any advantage of the redundancy produced by the outer code. It is this observation, that occurred fairly late in the history of information theory, that led to the development of new decoding principles, beginning with turbo decoding. We now know how to decode, quasi-optimally, all sorts of concatenated schemes, with the sole condition that the decoders of elementary codes are of the SISO (*soft-in/soft-out*) type. In this sense, we can note that the concept of concatenation has greatly evolved in the last few years, moving towards a wider notion of multi-dimensional encoding. Here, the *dimension of a code*, which should not be confused with the length (k) of the information message that we also call dimension, is the number of elementary codes used in the production of the final codeword.

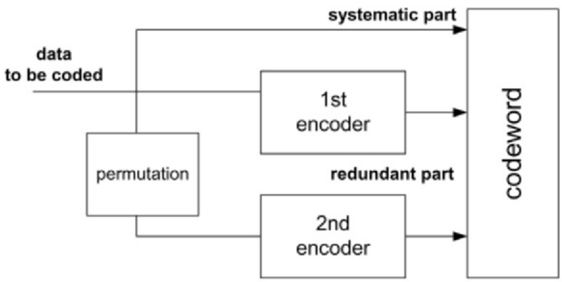

Figure 6.2 – Parallel concatenation of systematic encoders.

A new form of concatenation, called *parallel concatenation* (PC), was introduced at the beginning of the 1990s to elaborate turbo codes [6.3]. Figure 6.2 presents a PC with dimension 2, which is the classical dimension for turbo codes. In this scheme, the message is coded twice, in its natural order and in a permuted order. The redundant part of the codeword is formed by concatenating the redundant outputs of the two encoders. PCs differ from SCs in several ways, described in the next section.

6.1 Parallel concatenation and serial concatenation

Limiting ourselves to dimension 2, the PC, which associates two elementary codes with rates R_1 (code C_1) and R_2 (code C_2), has a global encoding rate:

$$R_p = \frac{R_1 R_2}{R_1 + R_2 - R_1 R_2} = \frac{R_1 R_2}{1 - (1 - R_1)(1 - R_2)} \tag{6.1}$$

This rate is higher than the global rate R_s of a serial concatenated code ($R_s = R_1 R_2$), for identical values of R_1 and R_2, and the lower the encoding rates the greater the difference. We can deduce from this that with the same error correction capability of component codes, parallel concatenation offers a better encoding rate, but this advantage diminishes when the rates considered tend towards 1. When the dimension of the composite code increases, the gap between R_p and R_s also increases. For example, three component codes of rate $1/2$ form a concatenated code with global rate $1/4$ for parallel, and $1/8$ for serial concatenation. That is the reason why it does not seem to be useful to increase the dimension of a serial concatenated code beyond 2, except for rates very close to unity.

However, with SC, the redundant part of a word processed by the outer decoder has benefited from the correction of the decoder(s) that precede(s) it. Therefore, at first sight, the correction capability of a serial concatenated code seems to be greater than that of a parallel concatenated code, in which the values representing the redundant part are never corrected. In other terms, the MHD of a serial concatenated code must normally be higher than that of a parallel concatenated code. We therefore find ourselves faced with the dilemma given in Chapter 1: PC performs better in the convergence zone (near the theoretical limit) since the encoding rate is more favourable, and the SC behaves better at low error rates thanks to a larger MHD. Encoding solutions based on the SC of convolutional codes have been studied [6.3], which can be an interesting alternative to classical turbo codes, when low error rates are required. Serial convolutional concatenated codes will not, however, be described in the rest of this book.

When the redundant parts of the inner and outer codewords both undergo supplementary encoding, the concatenation is said to be double serial concatenation. The most well-known example of this type of encoding structure is the product code, which implements BCH codes (see Chapters 4 and 8). Mixed structures, combining parallel and serial concatenations have also been proposed [6.6]. Moreover, elementary concatenated codes can be of a different nature, for example a convolutional code and a BCH code [6.2]. We then speak of hybrid concatenated codes. From the moment elementary decoders accept and produce weighted values, all sorts of mixed and/or hybrid schemes can be imagined.

Whilst SC can use systematic or non-systematic codes indifferently, parallel concatenation uses systematic codes. If they are convolutional codes, at least one of these codes must be recursive, for a fundamental reason to do with the minimum input weight w_{min}, which is only 1 for non-recursive codes but is 2 for recursive codes (see Chapter 5). To show this, see Figure 6.3 which presents two non-recursive systematic codes, concatenated in parallel. The input sequence is "all zero" (reference sequence) except in one position. This single "1" perturbs the output of the encoder C_1 for a short length of time, equal to the constraint length 4 of the encoder. The redundant information Y_1 is poor, in relation to this particular sequence, as it contains only 3 values different from 0. After permutation, of whatever type, the sequence is still "all zero", except in one single position. Again, this "1" perturbs the output of the encoder C_2 for a length of time equal to the constraint length, and redundancy Y_2 provided by the second code is as poor in information as redundancy Y_1. In fact, the minimum distance of this two-dimensional code is not higher than that of a single code, with the same rate as that of the concatenated code. If we replace at least one of the two non-recursive encoders by a recursive encoder, the "all zero" sequence except in one position is no longer a "Return to Zero" (RTZ, see Section 5.3.2) sequence for this recursive encoder, and the redundancy that it produces is thus of much higher weight.

What we have explained above about the PC of non-recursive convolutional codes suggests that the choice of elementary codes for the PC in general is limited. As another example, let us build a parallel concatenated code from the extended Hamming code defined by Figure 1.1 and the encoding Table 1.1. The information message contains 16 bits, arranged in a 4x4 square (Figure 6.4(a)). Each line and each column is encoded by the elementary Hamming code. The horizontal and vertical parity bits are denoted $r_{i,j}$ and $r'_{i,j}$, respectively. The global coding rate is 1/3. Decoding this type of code can be performed using the principles of turbo decoding (optimal local decoding according to the maximum likelihood and continuous exchanges of extrinsic information).

The MHD of the code is given by the pattern of errors of input weight 1 (Figure 6.4(b)). Whatever the position of the 1 in the information message, the weight is 7. The figure of merit Rd_{min} (see Section 1.5) is therefore equal to 7x(1/3), compared with the figure of merit of the elementary code 4x(1/2).

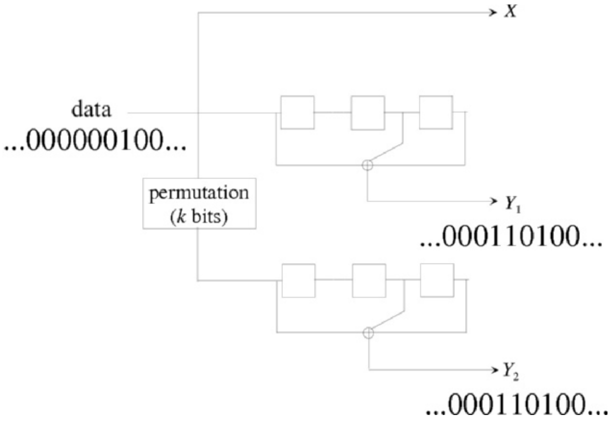

Figure 6.3 – The parallel concatenation of non-recursive systematic codes is a poor code concerning the information sequences of weight 1. In this example, the redundancy symbols Y_1 and Y_2 each contain only three 1.

$d_{0,0}$	$d_{0,1}$	$d_{0,2}$	$d_{0,3}$	$r_{0,0}$	$r_{0,1}$	$r_{0,2}$	$r_{0,3}$
$d_{1,0}$	$d_{1,1}$	$d_{1,2}$	$d_{1,3}$	$r_{1,0}$	$r_{1,1}$	$r_{1,2}$	$r_{1,3}$
$d_{2,0}$	$d_{2,1}$	$d_{2,2}$	$d_{2,3}$	$r_{2,0}$	$r_{2,1}$	$r_{2,2}$	$r_{2,3}$
$d_{3,0}$	$d_{3,1}$	$d_{3,2}$	$d_{3,3}$	$r_{3,0}$	$r_{3,1}$	$r_{3,2}$	$r_{3,3}$
$r'_{0,0}$	$r'_{0,1}$	$r'_{0,2}$	$r'_{0,3}$				
$r'_{1,0}$	$r'_{1,1}$	$r'_{1,2}$	$r'_{1,3}$				
$r'_{2,0}$	$r'_{2,1}$	$r'_{2,2}$	$r'_{2,3}$				
$r'_{3,0}$	$r'_{3,1}$	$r'_{3,2}$	$r'_{3,3}$				

(a)

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

(b)

Figure 6.4 – Parallel concatenation of extended Hamming codes (global rate: 1/3). On the right: a pattern of errors of input weight 1 and total weight 7.

The asymptotic gain has therefore not been extraordinarily increased by means of the concatenation (0,67 dB precisely), and a great reduction in the coding rate has occurred. If we wish to keep the same global rate of 1/2, a part of the redundancy must be punctured. We can choose, for example, not to transmit the 16 symbols present in the last two columns and the last two lines of the table of Figure 6.4(a). The MHD then drops to the value 3, that is, less than the MHD of the elementary code. The PC is therefore of no interest in this case.

Again from the extended Hamming code, a double serial concatenation can be elaborated in the form of a product code (Figure 6.5(a)). In this scheme, the redundant parts of the horizontal and vertical codewords are themselves re-encoded by elementary codes, which produce redundancy symbols denoted $w_{i,j}$. One useful algebraic property of this product code is the identity of the redundancy symbols coming from the second level of encoding, in the horizontal and vertical directions. The MHD of the code, which has a global rate 1/4, is again given by the patterns of errors of input weight 1 and is equal to 16, that is, the square of the MHD of the elementary code (Figure 6.5(b)). The figure of merit $Rd_{min} = 4$ has therefore been greatly increased compared to parallel concatenation. To attempt to increase the rate of this code by puncturing the redundancy symbols while keeping a good MHD is bound to fail.

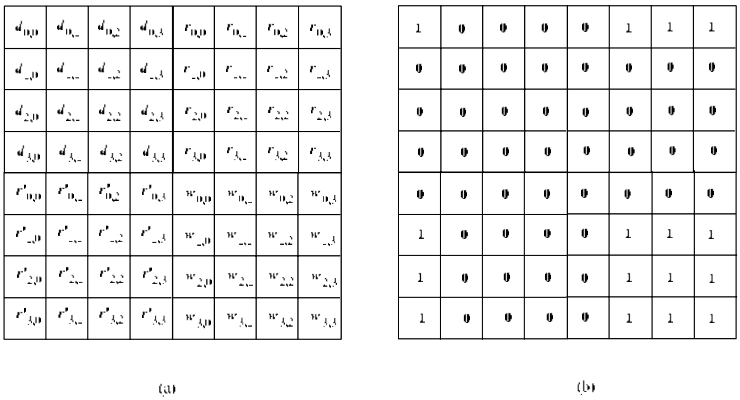

Figure 6.5 – Double serial concatenation (product code) of extended Hamming codes (global rate: 1/4). On the right: a pattern of errors of input weight 1 and total weight 16.

In conclusion, parallel concatenation cannot be used with just any elementary code. Today, only convolutional recursive systematic codes are used in this type of concatenation, with 2 dimensions. Serial concatenation can offer large MHD. The choice of codes is greater: convolutional codes, recursive or not, BCH codes or Reed-Solomon codes. However, with the same coding rates of elementary codes, serial concatenation has a lower global rate than parallel concatenation.

6.2 Parallel concatenation and LDPC codes

LDPC codes, which are described in Chapter 9, are codes where the lines and columns of the parity check matrix contain few 1s. LDPC codes can be seen as a multiple concatenation of $n - k$ parity relations containing few variables.

Here it is not a concatenation in the sense that we defined above, since the parity relations contain several redundancy variables and these variables appear in several relations. We cannot therefore assimilate LDPC codes to standard serial or parallel concatenation schemes. However, we can, like MacKay [6.5], observe that a turbo code is an LDPC code. An RSC code with generator polynomials $G_X(D)$ (recursivity) and $G_Y(D)$ (redundancy), whose input is X and redundant output Y, is characterized by the sliding parity relation:

$$G_Y(D)X(D) = G_X(D)Y(D) \tag{6.2}$$

Using the tail-biting technique (see CRSC, Section 5.5.1, the parity check matrix takes a very regular form, such as the one presented in Figure 6.6 for a coding rate 1/2, and choosing $G_X(D) = 1 + D + D^3$ and $G_Y(D) = 1 + D^2 + D^3$. A CRSC code is therefore an LDPC code since the check matrix is sparse. This is certainly not a good LDPC code, as the check matrix does not respect certain properties about the positions of the 1s. In particular, the 1s on a same line are very close to each other, which is not favourable to the belief propagation decoding method.

$$
\begin{array}{l}
\text{codeword} \quad X_0\ X_1\ X_2\ X_3 \ldots \qquad \ldots X_{k-3}\ X_{k-2}\ X_{k-1} \mid Y_0\ Y_1\ Y_2\ Y_3 \ldots \qquad \ldots Y_{k-3}\ Y_{k-2}\ Y_{k-1} \\
\text{control matrix} \quad
\left(\begin{array}{cccccccccccc|cccccccccccc}
1&0&1&1&0&0&\ldots&&\ldots&0&0&0&1&1&0&1&0&0&\ldots&&\ldots&0&0&0\\
0&1&0&1&1&0&\ldots&0&\ldots&0&0&0&0&1&1&0&1&0&\ldots&0&\ldots&0&0&0\\
0&0&1&0&1&1&\ldots&&\ldots&0&0&0&0&0&1&1&0&1&\ldots&&\ldots&0&0&0\\
\hdashline
1&0&0&0&0&0&\ldots&&\ldots&1&0&1&1&0&0&0&0&0&\ldots&&\ldots&1&1&0\\
1&1&0&0&0&0&\ldots&0&\ldots&0&1&0&0&1&0&0&0&0&\ldots&0&\ldots&0&1&1\\
0&1&1&0&0&0&\ldots&&\ldots&0&0&1&1&0&1&0&0&0&\ldots&&\ldots&0&0&1
\end{array}\right)
\end{array}
$$

Figure 6.6 – Check matrix of a tail-biting convolutional code. A convolutional code, particularly of the CRSC type, can be seen as an LDPC code since the check matrix is sparse.

A parallel concatenation of CRSC codes, that is a turbo code, is also an LDPC code since it associates elementary codes that are of the LDPC type. Of course, there are more degrees of freedom in the construction of an LDPC code, as each 1 of the check matrix can be positioned independently of the others. On the other hand, decoding a convolutional code, via an algorithm based on the trellis, does not encounter the problem of correlation between successive symbols that a belief propagation type of decoding would encounter, if it was applied to a simple convolutional code. A turbo code cannot therefore be decoded like an LDPC code.

6.3 Permutations

The functions of permutation or interleaving, used between elementary encoders in a concatenated scheme, have a twofold role. On the one hand, they ensure, at the output of each component decoder, a time spreading of the errors that can be produced by it in bursts. These packets of errors then become isolated errors for the following decoder, with far lower correlation effects. This technique for the spreading of errors is used in a wider context than that of channel coding. We can use it profitably, for example, to reduce the effects of more or less long attenuation in transmissions affected by fading, and more generally in situations where perturbations can alter consecutive symbols. On the other hand, in close liaison with the characteristics of constituent codes, the permutation is designed so that the MHD of the concatenated code is as large as possible. This is a problem of pure mathematics associating geometry, algebra and combinatory logic which, in most cases, has not yet found a definitive answer. Sections 7.3.2 and 9.1.6 develop the topic of permutation for turbo codes and graphs for LDPC codes, respectively.

6.4 Turbo crossword

To end this chapter, here is an example of parallel concatenation that is familiar to everyone: crosswords. The content of a grid has been altered during its retranscription, as can be seen in Figure 6.7. Fortunately, we have a correct clue for each line and for each column and we have at our disposal a dictionary of synonyms.

	1	2	3	4	5
I	**V**	**S**	**D**	**A**	**L**
II	**T**	**H**	**E**	**T**	**A**
III	**L**	**A**	**N**	**E**	**S**
IV	**A**	**P**	**O**	**N**	**G**
V	**P**	**I**	**N**	**T**	**O**

ACROSS

I. animate
II. Greek
III. Sticks
IV. Conflicts
V. slow

DOWN

1. oral
2. representation
3. projection
4. force
5. rope

Figure 6.7 – Crossword grid with wrong answers but correct clues.

To correct (or decode) this grid, we must operate iteratively by line and by column. The basic decoding rule is the following: "If there is a word in the dictionary, a synonym or an equivalent to the definition given that differs from the word in the grid by at most one letter, then this synonym is adopted".

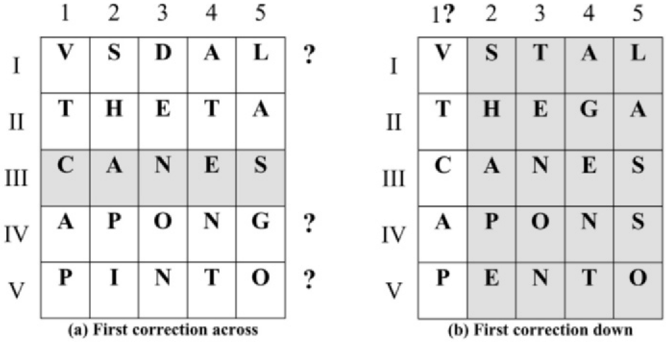

Figure 6.8 – First iteration of the line - column decoding process.

The horizontal definitions allow us to begin correcting the lines in the grid (Figure 6.8(a)):

I. There must be more than one wrong letter.

II. THETA is a Greek letter.

III. We can replace the L with a C, getting CANES (sticks).

IV. There must be more than one wrong letter.

V. There must be more than one wrong letter

After decoding this line, two words are correct or have been corrected, and three are still to be found. Using the vertical definitions, we can now decode the columns (Figure 6.8(b)):

1. There must be more than one wrong letter. No correction is possible.
2. Replacing the I with an E, we get SHAPE (representation).
3. A TENON is a projection (of wood).
4. Replacing the T with a G, we get AGENT (force).
5. A LASSO is a kind of rope.

After decoding the columns, there are still some unknown words and we have to perform a second iteration of the line - column decoding process (Figure 6.9). Line decoding leads to the following result (Figure 6.9(a)):

I. Replacing the S with an I, we get VITAL (animate).

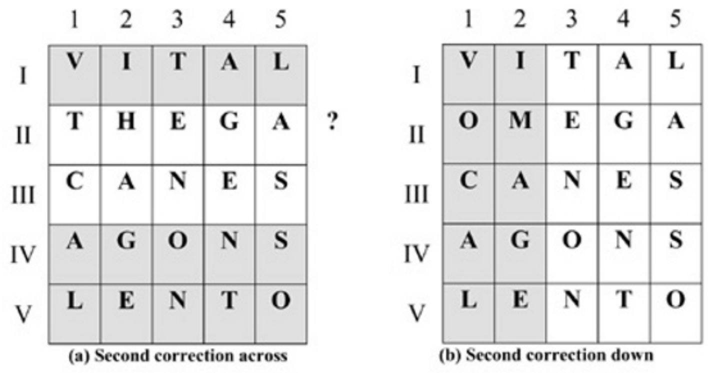

Figure 6.9 – Second iteration of the line - column decoding process.

II. There must be at least 2 wrong letters.

III. CANES is correct

IV. Replacing the P with a G, we get AGONS (conflicts)

V. We can replace the P with an L, getting LENTO (slow).

After this step, there is still one wrong line. It is possible to correct it by decoding the columns again (Figure 6.9(b)).

1. VOCAL is a synonym of oral.

2. IMAGE is also a kind of representation.

3. TENON is correct.

4. AGENT is correct.

5. LASSO is correct.

After this final decoding step, the wrong word on line II is identified: it is OMEGA, another Greek letter.

A certain number of remarks can be made after this experience of decoding a parallel concatenated code.

- To arrive at the right result, two iterations of the line and column decoding were necessary. It would have been a pity to stop after just one iteration. But that is what we do in the case of a classical concatenated code such as that of Figure 6.1.

- A word that was correct at the beginning (THETA is indeed a Greek letter) turned out to be wrong. Likewise, a correction made during the second step (SHAPE) turned out to be wrong. So the intermediate results must be considered with some caution and we must avoid any hasty decisions. In modern iterative decoders, this caution is measured by a probability that is never exactly 0 or 1.

Bibliography

[6.1] Dvb-terrestrial. ETSI EN 302 296 V1.1.1 (2004-04).

[6.2] P. Adde, R. Pyndiah, and C. Berrou. Performance of hybrid turbo codes. *Elect. Letters*, 32(24):2209–2210, Nov. 1996.

[6.3] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara. Serial concatenation of interleaved codes: performance analysis, design, and iterative decoding. *IEEE Trans. Info. Theory*, 44(3):909–926, May 1998.

[6.4] Jr. G. D. Forney. Performance of concatenated codes. In E. R. Berlekamp, editor, *Key papers in the development of coding theory*, pages 90–94. IEEE Press, 1974.

[6.5] D. J. C. MacKay. Good error-correcting codes based on very sparse matrices. *IEEE Transactions on Information Theory*, 45(2):399–431, March 1999.

[6.6] K. R. Narayanan and G. L. Stüber. Selective serial concatenation of turbo codes. *IEEE Comm. Letters*, 1(5):136–139, Sept. 1997.