

Coding in communication system

In the engineering sense, coding can be classified into four areas:

- Encryption: to encrypt information for security purpose.
- Data compression: to reduce space for the data stream.
- Data translation: to change the form of representation of the information so that it can be transmitted over a communication channel.
- Error control: to encode a signal so that error occurred can be detected and possibly corrected.

In this article some basics about two types of coding (**source coding** and **channel coding**) will be addressed.

1. Introduction

The main aim of any communication schemes is to provide error-free data transmission. In a communication system, information can be transmitted by analog or digital signals. For analog means, the amplitude of the signal reflects the information of the source, whereas for digital case, the information will first be translated into a stream of '0' and '1'. Then two different signals will be used to represent '0' and '1' respectively. As can be referred to the following illustration, the main advantage of using digital signal is that errors introduced by noise during the transmission can be detected and possibly corrected. For communication using cables, the random motion of charges in conducting (e.g. resistors), known as thermal noise, is the major source of noise. For wireless communication channels, noise can be introduced in various ways. In the case of mobile phones, noise includes the signals sent by other mobile phone users in the system.

Figure 1 shows the flow of a simple digital communication system:

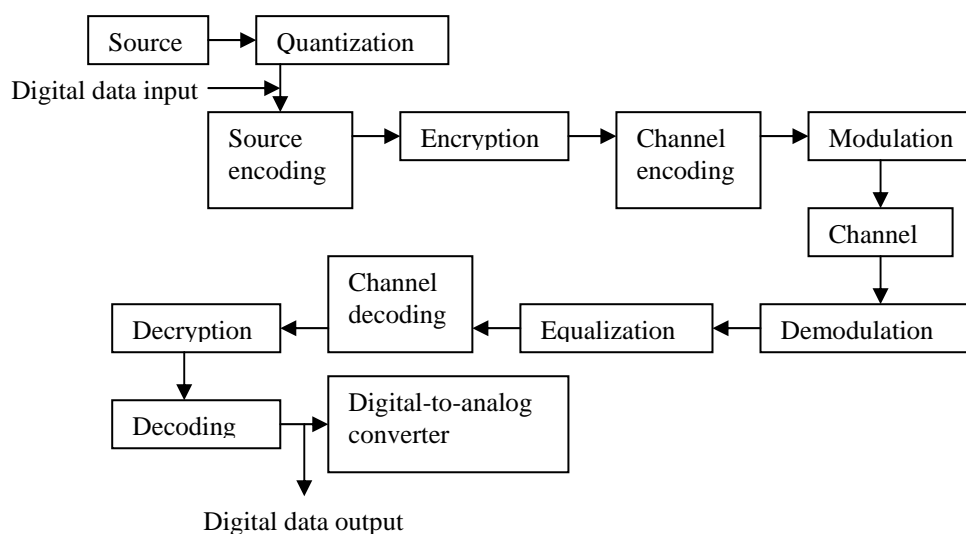


Figure 1: The flow of a simple digital communication system

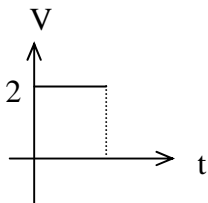
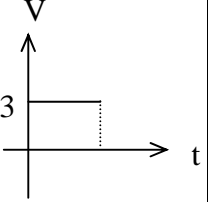
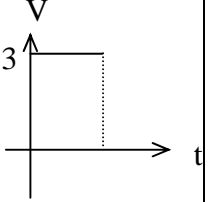
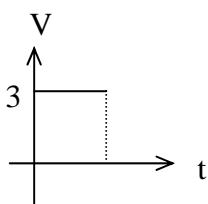
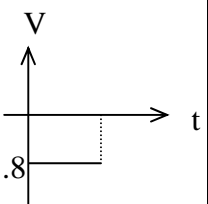
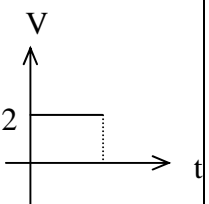
Type	Signal transmitted	Channel	Signal received	Information Detected
<p>Analog: All amplitudes are possible. The value represents the information (e.g. loudness of the voice)</p>	<p>Information: 2V</p> 	<p>Noise: 0.3V (The mean of noise is 0V)</p> 	<p>2.3V</p> 	2.3V
<p>Digital: Only two amplitudes ($\pm 3V$) will be transmitted to represent '1' and '0' respectively</p>	<p>Information: '2' which has been encoded as '1' and will be using 3V pass through the channel.</p> 	<p>Noise: -1.8V (The mean of noise is 0V)</p> 	<p>1.2V</p> 	<p>Since $1.2V > 0V$, so it is still detected as 3V. That means the code '1' is transmitted and therefore the information is '2'.</p>

Figure 2: The difference between signal detection for analog and digital signals

2. Source Coding

Suppose a word 'Zebra' is going to be sent out. Before this information can be transmitted to the channel, it is first translated into a stream of bits ('0' and '1'). The process is called source coding. There are many commonly used ways to translate that. For example, if ASCII code is used, each alphabet will be represented by 7-bit so-called the code word. The alphabets 'Z', 'e', 'b', 'r', 'a', will be encoded as

'1010101', '0110110', '0010110', '0010111', '0001110'

The **ASCII code** is an example of fixed-length code, because each of the code word is of the same length (7 bits). However, in the view of efficient communication, the occurrence of 'Z' is not as often as that of 'e' and 'a'. If there is a way of encoding information such that the alphabets with higher probability of occurrence are assigned with shorter code words, and longer for the other letters which seldom come out, then

on the whole it may be able to conserve the number of bits to be sent to the channel while sending the same information. This is what the variable length code can do. The following illustrates the **Huffman Codes**, which was developed by David Huffman in 1951.

Rules:

- Order the symbols ('Z', 'e', 'b', 'r', 'a') by decreasing probability and denote them as S_1 , to S_n ($n = 5$ for this case).
- Combine the two symbols (S_n, S_{n-1}) having the lowest probabilities. Assign '1' as the last symbol of S_{n-1} and '0' as the last symbol of S_n .
- Form a new source alphabet of $(n-1)$ symbols by combining S_{n-1} and S_n into a new symbol S'_{n-1} with probability $P'_{n-1} = P_{n-1} + P_n$.
- Repeat the above steps until the final source alphabet has only one symbol with probability equals to 1.

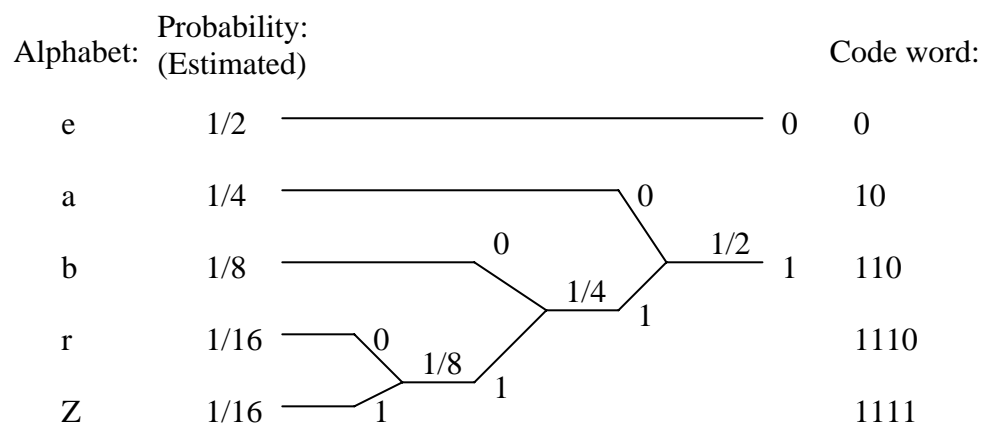


Figure 3: The process of coding the word 'Zebra' by Huffman Code

3. Channel Coding (Error Control Coding)

Error control coding is a method to detect and possibly correct errors by introducing redundancy to the stream of bits to be sent to the channel. The Channel Encoder will add bits to the message bits to be transmitted systematically. After passing through the channel, the Channel decoder will detect and correct the errors. A simple example is to send '000' ('111' correspondingly) instead of sending only one '0' ('1' correspondingly) to the channel. Due to noise in the channel, the received bits may become '001'. But since either '000' or '111' could have been sent. By majority-logic decoding scheme, it will be decoded as '000' and therefore the message has been a '0'.

In general the channel encoder will divide the input message bits into blocks of k message bits and replaces each k message bits block with a n -bit code word by introducing $(n-k)$ check bits to each message block. Some major codes include the **Block Codes** and **Convolutional Codes**.

Block Codes

Denoted by (n, k) a block code is a collection of code words each with length n , k information bits and $r = n - k$ check bits. It is linear if it is closed under addition mod 2. A **Generator Matrix** G (of order $k \times n$) is used to generate the code.

$$(3.1) \quad G = [I_k \ P]_{k \times n}$$

where I_k is the $k \times k$ identity matrix and P is a $k \times (n - k)$ matrix selected to give desirable properties to the code produced.

For example, denote D to be the message, G to be the generator matrix, C to be code word. For

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

we get the collection of code words in figure 4:

Messages (D)	Code words (C)
0 0 0	0 0 0 0 0 0
0 0 1	0 0 1 1 1 0
0 1 0	0 1 0 1 0 1
0 1 1	0 1 1 0 1 1
1 0 0	1 0 0 0 1 1
1 0 1	1 0 1 1 0 1
1 1 0	1 1 0 1 1 0
1 1 1	1 1 1 0 0 0

Figure 4: Codeword generated by G

In particular when $D = [0 \ 1 \ 1]$,

$$C = DG = \begin{bmatrix} 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 1 \end{bmatrix}.$$

Now define the **Parity Check Matrix** to be

$$(3.2) \quad H = \begin{bmatrix} P^T & I_{n-k} \end{bmatrix}_{(n-k) \times n}.$$

As long as the code word C is generated by G , the product $CH^T = 0$.

Denote the **received code word** after passing through the channel be R . It is made up of the original code word C and error bits E from the channel. Further define the **Error Syndrome** to be

$$(3.3) \quad S = RH^T$$

Then,

$$(3.4) \quad R = C + E$$

$$(3.5) \quad S = RH^T = (C + E) H^T = CH^T + EH^T = EH^T$$

If $S = 0$, $R \equiv C$ and D is the first k bits of R . If $S \neq 0$ and S is the j th row of H^T , then it implies an error occurs in the j th position of R .

To illustrate, suppose after passing through the channel, the received code word is $R = [0 \ 1 \ 1 \ 1 \ 1 \ 0]$.

Thus $RH^T = [0 \ 1 \ 1 \ 1 \ 1 \ 0] H^T = [1 \ 0 \ 1]$ which is the second row of H^T . Thus it implies there exists an error in the second bit of the received code word. So we can correct it and detect that $[0 \ 0 \ 1 \ 1 \ 1 \ 0]$ should have been sent.