CHAPTER IV *Encoding and Binary Digits*

A SOURCE OF INFORMATION may be English text, a man speaking, the sound of an orchestra, photographs, motion picture films, or scenes at which a television camera may be pointed. We have seen that in information theory such sources are regarded as having the properties of ergodic sources of letters, numbers, characters, or electrical signals. A chief aim of information theory is to study how such sequences of characters and such signals can be most effectively encoded for transmission, commonly by electrical means.

Everyone has heard of codes and the encoding of messages. Romantic spies use secret codes. Edgar Allan Poe popularized cryptography in *The Gold Bug*. The country is full of amateur cryptanalysts who delight in trying to read encoded messages that others have devised.

In this historical sense of cryptography or secret writing, codes are used to conceal the content of an important message from these for whom it is not intended. This may be done by substituting for the words of the message other words which are listed in a code book. Or, in a type of code called a cipher, letters or numbers may be substituted for the letters in the message according to some previously agreed upon secret scheme.

The idea of encoding, of the accurate representation of one thing by another, occurs in other contexts as well. Geneticists believe that the whole plan for a human body is written out in the

chromosomes of the germ cell. Some assert that the "text" consists of an orderly linear arrangement of four different units, or "bases," in the DNA (desoxyribonucleic acid) forming the chromosome. This text in turn produces an equivalent text in RNA (ribonucleic acid), and by means of this RNA text proteins made up of sequences of twenty amino acids are synthesized. Some cryptanalytic effort has been spent in an effort to determine how the four-character message of RNA is reencoded into the twenty-character code of the protein.

Actually, geneticists have been led to such considerations by the existence of information theory. The study of the transmission of information has brought about a new general understanding of the problems of encoding, an understanding which is important to any sort of encoding, whether it be the encoding of cryptography or the encoding of genetic information.

We have already noted in Chapter II that English text can be encoded into the symbols of Morse code and represented by short and long pulses of current separated by short and long spaces. This is one simple form of encoding. From the point of view of information theory, the electromagnetic waves which travel from an FM transmitter to the receiver in your home are an encoding of the music which is transmitted. The electric currents in telephone circuits are an encoding of speech. And the sound waves of speech are themselves an encoding of the motions of the vocal tract which produce them.

Nature has specified the encoding of the motions of the vocal tract into the sounds of speech. The communication engineer, however, can choose the form of encoding by means of which he will represent the sounds of speech by electric currents, just as he can choose the code of dots, dashes, and spaces by means of which he represents the letters of English text in telegraphy. He wants to perform this encoding well, not poorly. To do this he must have some standard which distinguishes good encoding from bad encoding, and he must have some insight into means for achieving good encoding. We learned something of these matters in Chapter II.

It is the study of this problem, a study that might in itself seem limited, which has provided through information theory new ideas important to all encoding, whether cryptographic or genetic. These

new ideas include a measure of amount of information, called *entropy,* and a unit of measurement, called the *bit.*

I would like to believe that at this point the reader is clamoring to know the meaning of "amount of information" as measured in bits, and if so I hope that this enthusiasm will carry him over a considerable amount of intervening material about the encoding of messages.

It seems to me that one can't understand and appreciate the solution to a problem unless he has some idea of what the problem is. You can't explain music meaningfully to a man who has never heard any. A story about your neighbor may be full of insight, but it would be wasted on a Hottentot. I think it is only by considering in some detail how a message can be encoded for transmission that we can come to appreciate the need for and the meaning of a measure of amount of information.

It is easiest to gain some understanding of the important problems of coding by considering simple and concrete examples. Of course, in doing this we want to learn something of broad value, and here we may foresee a difficulty.

Some important messages consist of sequences of discrete characters, such as the successive letters of English text or the successive digits of the output of an electronic computer. We have seen, however, that other messages seem inherently different.

Speech and music are variations with time of the pressure of air at the ear. This pressure we can accurately represent in telephony by the voltage of a signal traveling along a wire or by some other quantity. Such a variation of a signal with time is illustrated in *a* of Figure IV-1. Here we assume the signal to be a voltage which varies with time, as shown by the wavy line.

Information theory would be of limited value if it were not applicable to such *continuous* signals or messages as well as to discrete messages, such as English text.

In dealing with continuous signals, information theory first invokes a mathematical theorem called the *sampling theorem,* which we will use but not prove. This theorem states that a continuous signal can be represented completely by and reconstructed perfectly from a set of measurements or *samples* of its amplitude which are made at equally spaced times. The interval between such

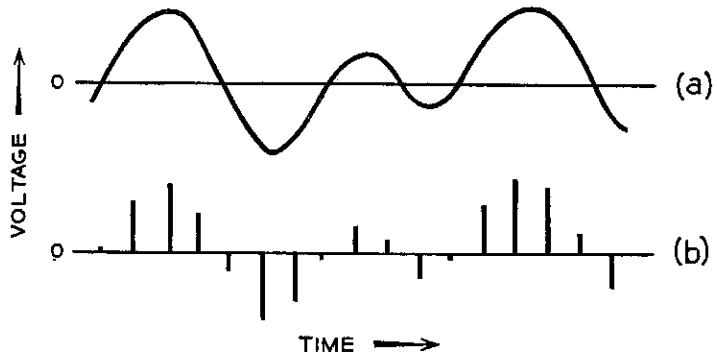

Fig. IV-1

samples must be equal to or less than one-half of the period of the highest frequency present in the signal. A set of such measurements or samples of the amplitude of the signal *a,* Figure IV-1, is represented by a sequence of vertical lines of various heights in *b* of Figure IV-1.

We should particularly note that for such samples of the signal to represent a signal perfectly they must be taken frequently enough. For a voice signal including frequencies from 0 to 4,000 cycles per second we must use 8,000 samples per second. For a television signal including frequencies from 0 to 4 million cycles per second we must use 8 million samples per second. In general, if the frequency range of the signal is f cycles per second we must use at least $2f$ samples per second in order to describe it perfectly.

Thus, the sampling theorem enables us to represent a smoothly varying signal by a sequence of samples which have different amplitudes one from another. This sequence of samples is, however, still inherently different from a sequence of letters or digits. There are only ten digits and there are only twenty-six letters, but a sample can have any of an infinite number of amplitudes. The amplitude of a sample can lie anywhere in a *continuous* range of values, while a character or a digit has only a limited number of *discrete* values.

The manner in which information theory copes with samples having a continuous range of amplitudes is a topic all in itself, to which we will return later. Here we will merely note that a signal

need not be described or reproduced perfectly. Indeed, with real physical apparatus a signal *cannot* be reproduced perfectly. In the transmission of speech, for instance, it is sufficient to represent the amplitude of a sample to an accuracy of about 1 per cent. Thus, we can, if we wish, restrict ourselves to the numbers 0 to 99 in describing the amplitudes of successive speech samples and represent the amplitude of a given sample by that one of these hundred integers which is closest to the actual amplitude. By so *quantizing* the signal samples, we achieve a representation comparable to the discrete case of English text.

We can, then, by sampling and quantizing, convert the problem of coding a continuous signal, such as speech, into the seemingly simpler problem of coding a sequence of discrete characters, such as the letters of English text.

We noted in Chapter II that English text can be sent, letter by letter, by means of the Morse code. In a similar manner, such messages can be sent by teletypewriter. Pressing a particular key on the transmitting machine sends a particular sequence of electrical pulses and spaces out on the circuit. When these pulses and spaces reach the receiving machine, they activate the corresponding type bar, and the machine prints out the character that was transmitted.

Patterns of pulses and spaces indeed form a particularly useful and general way of describing or encoding messages. Although Morse code and teletypewriter codes make use of pulses and spaces of different lengths, it is possible to transmit messages by means of a sequence of pulses and spaces of equal length, transmitted at perfectly regular intervals. Figure IV-2 shows how the electric current sent out on the line varies with time for two different patterns, each six intervals long, of such equal pulses and spaces. Sequence *a* is a pulse-space-space-pulse-space-pulse. Sequence *b* is pulse-pulse-pulse-space-pulse-pulse.

The presence of a pulse or a space in a given interval specifies one of two different possibilities. We could use any pair of symbols to represent such patterns of pulses or spaces as those of Figure IV-2: yes, no; +, −; 1, 0. Thus we could represent pattern *a* as follows:

pulse	*space*	*space*	*pulse*	*space*	*pulse*
Yes	*No*	*No*	*Yes*	*No*	*Yes*
+	−	−	+	−	+
1	0	0	1	0	1

The representation by 1 or 0 is particularly convenient and important. It can be used to relate patterns of pulses to numbers expressed in the *binary system* of notation.

When we write 315 we mean

$$\begin{aligned} &3 \times 10^2 + 1 \times 10^1 + 5 \times 1 \\ &= 3 \times 100 + 1 \times 10 + 5 \times 1 \\ &= 315 \end{aligned}$$

In this ordinary *decimal* system of representing numbers we make use of the ten different digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. In the binary system we use only two digits, 0 and 1. When we write 1 0 0 1 0 1 we mean

$$\begin{aligned} &1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2 + 1 \times 1 \\ &= 1 \times 32 + 0 \times 16 + 0 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1 \\ &= 37 \text{ in decimal notation} \end{aligned}$$

It is often convenient to let zeros precede a number; this does not change its value. Thus, in decimal notation we can say,

$$0016 = 16$$

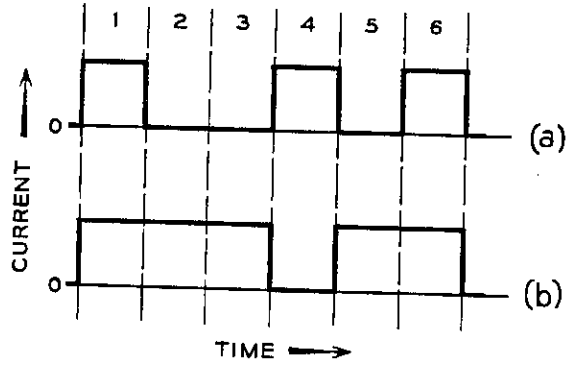

Fig. IV-2

Or in binary notation

$$001010 = 1010$$

In binary numbers, each 0 or 1 is a binary digit. To describe the pulses or spaces occurring in six successive intervals, we can use a sequence of six binary digits. As a pulse or space in one interval is equivalent to a binary digit, we can also refer to a pulse group of six binary digits, or we can refer to the pulse or space occurring in one interval as one binary digit.

Let us consider how many patterns of pulses and spaces there are which are three intervals long. In other words, how many three-digit binary numbers are there? These are all shown in Table VI.

TABLE VI

000	(0)
001	(1)
010	(2)
011	(3)
100	(4)
101	(5)
110	(6)
111	(7)

The decimal numbers corresponding to these sequences of 1's and 0's regarded as binary numbers are shown in parentheses to the right.

We see that there are 8 (0 and 1 through 7) three-digit binary numbers. We may note that 8 is 2^3. We can, in fact, regard an orderly listing of binary digits n intervals long as simply setting down 2^n successive binary numbers, starting with 0. As examples, in Table VII the numbers of different patterns corresponding to different numbers n of binary digits are tabulated.

We see that the number of different patterns increases very rapidly with the number of binary digits. This is because we double the number of possible patterns each time we add one digit. When we add one digit, we get all the old sequences preceded by a 0 plus all the old sequences preceded by a 1.

The binary system of notation is not the only alternative to the

TABLE VII

n (Number of Binary Digits)	*Number of Patterns (2^n)*
1	2
2	4
3	8
4	16
5	32
10	1,024
20	1,048,576

decimal system. The octal system is very important to people who use computers. We can regard the octal system as made up of the eight digits 0, 1, 2, 3, 4, 5, 6, 7.

When we write 356 in the octal system we mean

$$\begin{aligned} &3 \times 8^2 + 5 \times 8 + 6 \times 1 \\ &= 3 \times 64 + 5 \times 8 + 6 \times 1 \\ &= 238 \text{ in decimal notation} \end{aligned}$$

We can convert back and forth between the octal and the binary systems very simply. We need merely replace each successive block of three binary digits by the appropriate octal digit, as, for instance,

binary	0	1	0		1	1	1		0	1	1		1	1	0
octal		2				7				3				6	

People who work with binary notation in connection with computers find it easier to remember and transcribe a short sequence of octal digits than a long group of binary digits. They learn to regard patterns of three successive binary digits as an entity, so that they will think of a sequence of twelve binary digits as a succession of four patterns of three, that is, as a sequence of four octal digits.

It is interesting to note, too, that, just as a pattern of pulses and spaces can correspond to a sequence of binary digits, so a sequence of pulses of various amplitudes (0, 1, 2, 3, 4, 5, 6, 7) can correspond to a sequence of octal digits. This is illustrated in Figure IV-3. In *a*, we have the sequence of off-on, 0-1 pulses corresponding to the binary number 010111011110. The corresponding octal number is 2736, and in *b* this is represented by a sequence of four pulses of current having amplitudes 2, 7, 3, 6.

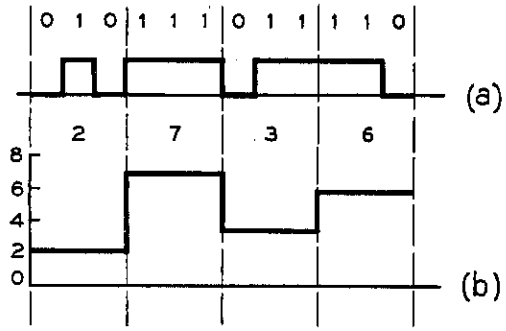

Fig. IV-3

Conversion from binary to decimal numbers is not so easy. On the average, it takes about 3.32 binary digits to represent one decimal digit. Of course we can assign four binary digits to each decimal digit, as shown in Table VIII, but this means that some patterns are wasted; there are more patterns than we use.

It is convenient to think of sequences of 0's and 1's or sequences of pulses and spaces as binary numbers. This helps us to understand how many sequences of a different length there are and how numbers written in the binary system correspond to numbers written in the octal or in the decimal system. In the transmission of information, however, the particular number assigned to a sequence of binary digits is irrelevent. For instance, if we wish merely to *transmit* representations of octal digits, we could make the assignments shown in Table IX rather than those in Table VI.

TABLE VIII

Binary Number	*Decimal Digit*
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	not used
1011	not used
1100	not used
1101	not used
1110	not used
1111	not used

TABLE IX

Sequence of Binary Digits	*Octal Digit Represented*
000	5
001	7
010	1
011	6
100	0
101	4
110	2
111	3

Here the "binary numbers" in the left column designate octal numbers of different numerical value.

In fact, there is another way of looking at such a correspondence between binary digits and other symbols, such as octal digits, a way in which we do not regard the sequence of binary digits as part of a binary number but rather as means of choosing or designating a particular symbol.

We can regard each 0 or 1 as expressing an elementary choice between two possibilities. Consider, for instance, the "tree of choice" shown in Figure IV-4. As we proceed upward from the root to the twigs, let 0 signify that we take the left branch and let 1 signify that we take the right branch. Then 0 1 1 means left, right, right and takes us to the octal digit 6, just as in Table IX.

Just as three binary digits give us enough information to determine one among eight alternatives, four binary digits can determine one among sixteen alternatives, and twenty binary digits can determine one among 1,048,576 alternatives. We can do this by assigning the required binary numbers to the alternatives in any order we wish.

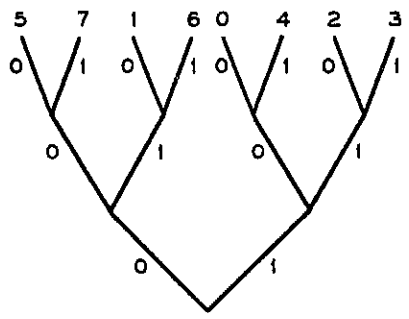

Fig. IV-4

The alternatives which we wish to specify by successions of binary digits need not of course be numbers at all. In fact, we began by considering how we might encode English text so as to transmit it electrically by sequences of pulses and spaces, which can be represented by sequences of binary digits.

A bare essential in transmitting English text letter by letter is twenty-six letters plus a space, or twenty-seven symbols in all. This of course allows us no punctuation and no Arabic numbers.

We can write out the numbers (three, not 3) if we wish and use words for punctuation, (stop, comma, colon, etc.).

Mathematics says that a choice among 27 symbols corresponds to about 4.75 binary digits. If we are not too concerned with efficiency, we can assign a different 5-digit binary number to each character, which will leave five 5-digit binary numbers unused.

My typewriter has 48 keys, including shift and shift lock. We might add two more "symbols" representing carriage return and line advance, making a total of 50. I could encode my actions in typing, capitalization, punctuation, and all (but not insertion of the paper) by a succession of choices among 50 symbols, each choice corresponding to about 5.62 binary digits. We could use 6 binary digits per character and waste some sequences of binary digits.

This waste arises because there are only thirty-two 5-digit binary numbers, which is too few, while there are sixty-four 6-digit binary numbers, which is too many. How can we avoid this waste? If we have 50 characters, we have 125,000 possible different groups of 3 ordered characters. There are 131,072 different combinations of

17 binary digits. Thus, if we divide our text into *blocks* of 3 successive characters, we can specify any possible block by a 17-digit binary number and have a few left over. If we had represented each separate character by 6 binary digits, we would have needed 18 binary digits to represent 3 successive characters. Thus, by this *block coding*, we have cut down the number of binary digits we use in encoding a given length of text by a factor 17/18.

Of course, we might encode English text in quite a different way. We can say a good deal with 16,384 English words. That's quite a large vocabulary. There are just 16,384 fourteen-digit binary numbers. We might assign 16,357 of these to different useful words and 27 to the letters of the alphabet and the space, so that we could spell out any word or sequence of words we failed to include in our word vocabulary. We won't need to put a space between words to which numbers have been assigned; it can be assumed that a space goes with each word.

If we have to spell out words very infrequently, we will use about 14 binary digits per word in this sort of encoding. In ordinary English text there are on the average about 4.5 letters per word. As we must separate words by a space, when we send the message character by character, even if we disregard capitalization and punctuation, we will require on the average 5.5 characters per word. If we encode these using 5 binary digits per character, we will use on the average 27.5 binary digits per word, while in encoding the message word by word we need only 14 binary digits per word.

How can this be so? It is because, in spelling out the message letter by letter, we have provided means for sending with equal facility all sequences of English letters, while, in sending word by word, we restrict ourselves to English words.

Clearly, the average number of binary digits per word required to represent English text depends strongly on how we encode the text.

Now, English text is just one sort of message we might want to transmit. Other messages might be strings of numbers, the human voice, a motion picture, or a photograph. If there are efficient and inefficient ways of encoding English text, we may expect that there will be efficient and inefficient ways of encoding other signals as well.

Indeed, we may be led to believe that there exists in principle some *best* way of encoding the signals from a given message source, a way which will on the average require fewer binary digits per character or per unit time than any other way.

If there is such a best way of encoding a signal, then we might use the average number of binary digits required to encode the signal as a measure of the amount of information per character or the amount of information per second of the message source which produced the signal.

This is just what is done in information theory. How it is done and further reasons for so doing will be considered in the next chapter.

Let us first, however, review very briefly what we have covered in this chapter. In communication theory, we regard coding very broadly, as representing one signal by another. Thus a radio wave can represent the sounds of speech and so form an encoding of these sounds. Encoding is, however, most simply explained and explored in the case of discrete message sources, which produce messages consisting of sequences of characters or numbers. Fortunately, we can represent a continuous signal, such as the current in a telephone line, by a number of samples of its amplitude, using, each second, twice as many samples as the highest frequency present in the signal. Further we can if we wish represent the amplitude of each of these samples approximately by a whole number.

The representation of letters or numbers by sequences of off-or-on signals, which can in turn be represented directly by sequences of the binary digits 0 and 1, is of particular interest in communication theory. For instance, by using sequences of 4 binary digits we can form 16 binary numbers, and we can use 10 of these to represent the 10 decimal digits. Or, by using sequences of 5 binary digits we can form 32 binary numbers, and we can use 27 of these to represent the letters of the English alphabet plus the space. Thus, we can transmit decimal numbers or English text by sending sequences of off-or-on signals.

We should note that while it may be convenient to regard the sequences of binary digits so used as binary numbers, the numerical value of the binary number has no particular significance; we can choose any binary number to represent a particular decimal digit.

If we use 10 of the 16 possible 5-digit binary numbers to encode the 10 decimal digits, we never use (we waste) 6 binary numbers. We could, but never do, transmit these sequences as sequences of off-or-on signals. We can avoid such waste by means of block coding, in which we encode sequences of 2, 3, or more decimal digits or other characters by means of binary digits. For instance, all sequences of 3 decimal digits can be represented by 10 binary digits, while it takes a total of 12 binary digits to represent separately each of 3 decimal digits.

Any sequence of decimal digits may occur, but only certain sequences of English letters ever occur, that is, the words of the English language. Thus, it is more efficient to encode English words as sequences of binary digits rather than to encode the letters of the words individually. This again emphasizes the gain to be made by encoding sequences of characters, rather than encoding each character separately.

All of this leads us to the idea that there may be a best way of encoding the messages from a message source, a way which calls for the least number of binary digits.