Chapter 6

Communications

We have been considering a model of an information handling system in which symbols from an input are encoded into bits, which are then sent across a "channel" to a receiver and get decoded back into symbols, as shown in Figure 6.1.

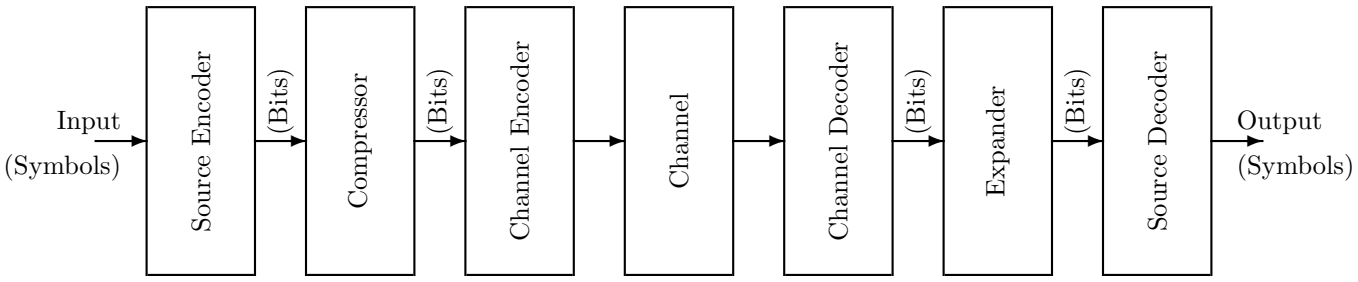

Figure 6.1: Elaborate communication system

In this chapter we focus on how fast the information that identifies the symbols can be transferred to the output. The symbols themselves, of course, are not transmitted, but only the information necessary to specify them. This is all that is necessary to enable the stream of symbols from the source to be recreated at the output.

We will model both the source and the channel in a little more detail, and then give three theorems relating to the source characteristics and the channel capacity.

6.1 Source Model

The source is assumed to produce symbols at a rate of R symbols per second. Each symbol is chosen from a finite set of possible symbols, and the index i ranges over the possible symbols. The event of the selection of symbol i will be denoted A_i.

Let us suppose that each event A_i (i.e., the selection of the symbol i) is represented by a different codeword C_i with a length L_i. For fixed-length codes such as ASCII all L_i are the same, whereas for variable-length codes, such as Huffman codes, they are generally different. Since the codewords are patterns of bits, the

Author: Paul Penfield, Jr.

number available of each length is limited. For example, there are only four distinct two-bit codewords possible, namely 00 01 10, and 11.

An important property of such codewords is that none can be the same as the first portion of another, longer, codeword – otherwise the same bit pattern might result from two or more different messages, and there would be ambiguity. A code that obeys this property is called a **prefix-condition code**, or sometimes an **instantaneous code**.

6.1.1 Kraft Inequality

Since the number of distinct codes of short length is limited, not all codes can be short. Some must be longer, but then the prefix condition limits the available short codes even further. An important limitation on the distribution of code lengths L_i was given by L. G. Kraft, an MIT student, in his 1949 Master's thesis. It is known as the Kraft inequality:

$$\sum_i \frac{1}{2^{L_i}} \leq 1 \tag{6.1}$$

Any valid set of distinct codewords obeys this inequality, and conversely for any proposed L_i that obey it, a code can be found.

As an example, this sum in the case of four distinct two-bit codewords is the sum of four terms each of which is $1/2^2 = 1/4$. In this case the equation evaluates to an equality, and there are many different ways to assign the four codewords to four different symbols. As another example, suppose there are only three symbols, and the proposed codewords are 00 01, and 11. In this case the Kraft inequality is a true inequality. However, because the sum is less than 1, the code can be made more efficient by replacing one of the codewords with a shorter one. In particular, if the symbol represented by 11 is now represented by 1 the result is still a prefix-condition code but the sum would be $(1/2^2) + (1/2^2) + (1/2) = 1$.

The Kraft inequality can be proven easily. Let L_{max} be the length of the longest codeword of a prefix-condition code. Then

$$\sum_i \frac{1}{2^{L_{max}}} = 1 \tag{6.2}$$

where the sum is over all $2^{L_{max}}$ distinct patterns of length L_{max}. At least one of these patterns is one of the codewords, but unless this happens to be a fixed-length code there are other shorter codewords. For each shorter codeword of length 1 (if there is one), replace the $L_{max} - 1$ terms corresponding to bit patterns that are prefixed by that shorter codeword with one term representing this codeword, equal to 1/2. The value of the sum is unchanged. Next, for each shorter codeword of length 2 (if any), replace the $L_{max} - 2$ terms corresponding to bit patterns that are prefixed by this codeword with a single term using its length 2. Again the sum is unchanged. Keep going with longer codewords until all shorter codewords are accounted for in the sum, and all longer bit patterns that are prefixed (and therefore cannot be valid codewords) are gone. The sum is still equal to 1. Finally, remove from the sum any remaining terms that do not correspond to codewords in use. The result is a sum over all the codewords which is less than or equal to 1. This ends the proof.

6.2 Source Entropy

As part of the source model, we assume that each symbol selection is independent of the other symbols chosen, so that the probability $p(A_i)$ does not depend on what symbols have previously been chosen (this model can, of course, be generalized in many ways). The uncertainty of the identity of the next symbol chosen, H, is the average information gained when the next symbol is made known:

$$H = \sum_i p(A_i) \log_2 \left(\frac{1}{p(A_i)} \right) \tag{6.3}$$

This quantity is also known as the entropy of the source, and is measured in bits per symbol. The information rate, in bits per second, is $H \cdot R$ where R is the rate at which the source selects the symbols, measured in symbols per second.

6.2.1 Gibbs Inequality

Here we present the Gibbs Inequality, named after the American physicist J. Willard Gibbs (1839 - 1903)[1], which will be useful to us in later proofs. This inequality states that the entropy is smaller than or equal to any other average formed using the same probabilities but a different function in the logarithm. Specifically,

$$\sum_i p(A_i) \log_2 \left(\frac{1}{p(A_i)} \right) \leq \sum_i p(A_i) \log_2 \left(\frac{1}{p'(A_i)} \right) \tag{6.4}$$

where $p(A_i)$ is any probability distribution (we will use it for source events and other distributions) and $p'(A_i)$ is any other probability distribution, or more generally any set of numbers such that

$$0 \leq p'(A_i) \leq 1 \tag{6.5}$$

and

$$\sum_i p'(A_i) \leq 1. \tag{6.6}$$

Like all probability distributions,

$$\sum_i p(A_i) = 1. \tag{6.7}$$

Equation 6.4 can be proven by noting that, because the natural logarithm is a convex function of x (i.e., one which curves downward but not upward), it is less than or equal to a straight line that is tangent to it at any point, for example the point $x = 1$, which is shown in Figure 6.2.

$$\ln x \leq (x - 1) \tag{6.8}$$

and therefore, by converting the logarithm base e to the logarithm base 2, we have

$$\log_2 x \leq (x - 1) \log_2 e \tag{6.9}$$

Then

$$\begin{aligned}
\sum_i p(A_i) \log_2 \left(\frac{1}{p(A_i)} \right) - \sum_i p(A_i) \log_2 \left(\frac{1}{p'(A_i)} \right) &= \sum_i p(A_i) \log_2 \left(\frac{p'(A_i)}{p(A_i)} \right) \\
&\leq \log_2 e \sum_i p(A_i) \left[\frac{p'(A_i)}{p(A_i)} - 1 \right] \\
&= \log_2 e \left(\sum_i p'(A_i) - \sum_i p(A_i) \right) \\
&= \log_2 e \left(\sum_i p'(A_i) - 1 \right) \\
&\leq 0
\end{aligned} \tag{6.10}$$

[1] See a biography of Gibbs at http://www-groups.dcs.st-andrews.ac.uk/%7Ehistory/Mathematicians/Gibbs.html

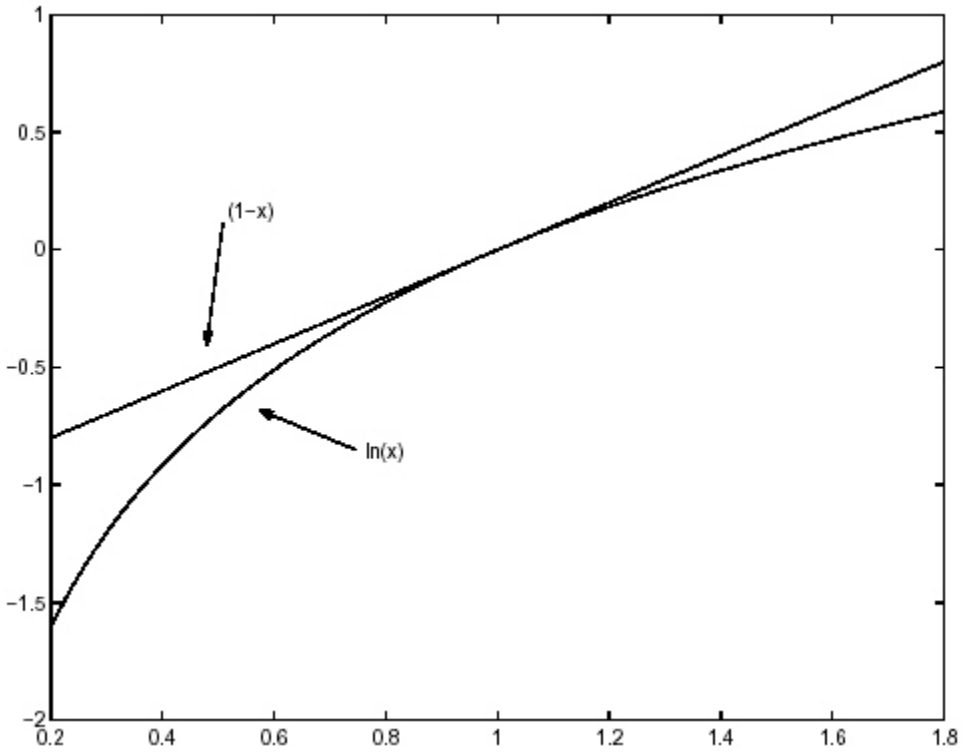

Figure 6.2: Graph of the inequality $\ln x \leq (x-1)$

6.3 Source Coding Theorem

Now getting back to the source model, note that the codewords have an average length, in bits per symbol,

$$L = \sum_i p(A_i) L_i \tag{6.11}$$

For maximum speed the lowest possible average codeword length is needed. The assignment of high-probability symbols to the short codewords can help make L small. Huffman codes are optimal codes for this purpose. However, there is a limit to how short the average codeword can be. Specifically, the Source Coding Theorem states that the average information per symbol is always less than or equal to the average length of a codeword:

$$H \leq L \tag{6.12}$$

This inequality is easy to prove using the Gibbs and Kraft inequalities. Use the Gibbs inequality with $p'(A_i) = 1/2^{L_i}$ (this is valid because the Kraft inequality assures that the terms, besides being positive, add up to no more than 1). Thus

$$\begin{aligned} H &= \sum_i p(A_i) \log_2 \left(\frac{1}{p(A_i)} \right) \\ &\leq \sum_i p(A_i) \log_2 \left(\frac{1}{p'(A_i)} \right) \\ &= \sum_i p(A_i) \log_2 2^{L_i} \end{aligned}$$

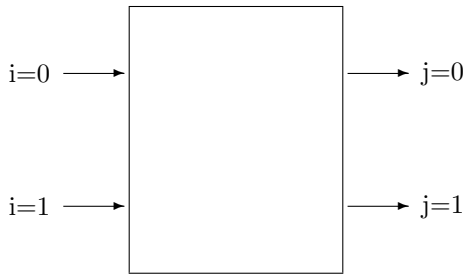

Figure 6.3: Binary Channel

$$\begin{aligned} &= \sum_i p(A_i)L_i \\ &= L \end{aligned} \tag{6.13}$$

6.4 Channel Model

A communication channel accepts input bits and produces output bits. We model the input as the selection of one of a finite number of input states (for the simplest channel, two such states), and the output as a similar event. The language of probability theory will be useful in describing channels. If the channel perfectly changes its output state in conformance with its input state, it is said to be **noiseless** because nothing affects the output except the input. Let us say that the channel has a certain maximum rate W at which its input state can be changed and the output state can follow.

The most general channel can be quite complicated. To get the basic ideas across in a simple domain, we consider for now a simple model, that of a symmetric binary channel. The word "binary" implies that it has two possible states, which we may call 0 and 1 (the minimum number that will permit any communication) and **symmetric** implies that the two states behave in the same or symmetrical ways.

We will use the index i to run over the input states, and j to index the output states. We will refer to the input events as A_i and the output events as B_j. You may picture the binary channel as something with inputs and outputs, as in the Figure 6.3, but note that the inputs are not normal signal inputs or electrical inputs to systems, but instead mutually exclusive events, only one of which is active at any one time. For simple channels such a diagram is simple because there are so few possible choices, but for more complex structures there may be so many possible inputs that the diagrams become impractical to use in detail (even though they may be retained as a useful conceptual model). For example, a logic gate with three inputs, each of which could be 0 or 1, would have eight inputs in a diagram of this sort.

For a noiseless channel, where each of n possible input states leads to exactly one output state, each new input state (W per second) can be specified with $\log_2 n$ bits. Thus for the binary channel, $n = 2$, and so the new state can be specified with one bit. The maximum rate at which information is supplied to the input is called the **channel capacity** $C = W \log_2 n$ bits per second. For the binary channel, $C = W$.

If the input is changed at a rate less than W (or, equivalently, if the information supplied at the input is less than C) then the output can follow the input, and the output events can be used to infer the identity of the input symbols at that rate. If there is an attempt to change the input more rapidly, the channel cannot follow (since W is by definition the maximum rate of change of the input) and some of the input information is lost.

6.5 Noiseless Channel Theorem

The relation between the information supplied to the input and what is available on the output is very simple. Let the input information rate, in bits per second, be denoted D (for example, D might be the

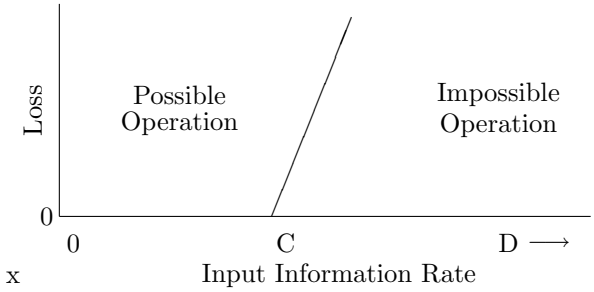

Figure 6.4: Channel Loss Diagram

entropy per symbol of a source H expressed in bits per symbol, times the rate of the source R in symbols per second). Then, if $D \leq C$ then the information available at the output can be as high as D (the information at the input), and if $D > C$ then the information available at the output cannot exceed C and so an amount at least equal to $D - C$ is lost. This result is pictured in Figure 6.4.

Note that this result places a limit on how fast information can be transmitted across a given channel. It does not indicate how to achieve results close to this limit. However, it is known how to use Huffman codes to efficiently represent streams of symbols by streams of bits. If the channel is a binary channel it is simply a matter of using that stream of bits to change the input. For other channels, with more than two possible states, operation close to the limit involves using multiple bits to control the input rapidly.

Achieving high communication speed may (like Huffman code) require representing some infrequently occurring symbols with long codewords. Therefore the rate at which individual bits arrive at the channel input may vary, and even though the average rate may be acceptable, there may be bursts of higher rate, if by coincidence several low-probability symbols happen to be adjacent. It may be necessary to provide temporary storage buffers to accommodate these bursts, and the symbols may not materialize at the output of the system at a uniform rate. Also, to encode the symbols efficiently it may be necessary to consider several of them together, in which case the first symbol would not be available at the output until several symbols had been presented at the input. Therefore high speed operation may require high latency. Different communication systems have different tolerance for latency or bursts; for example, latency of more than about 100 milliseconds is annoying in a telephone call, whereas latency of many minutes may be tolerable in e-mail. A list of the needs of practical communication systems, as shown in Section 6.9 reveals a wide variation in required speed, throughput, latency, etc.

6.6 Noisy Channel

If the channel introduces noise then the output is not a unique function of the input. We will model this case by saying that for every possible input (which are mutually exclusive states indexed by i) there may be more than one possible output event which could be chosen (although only one actually is). Which output event actually happens is a matter of chance, and we will model the channel by the probability that each of the output events occurs when one of the possible input events happens. Although it is natural to think of these **transition probabilities** c_{ji} as probabilities, they are really properties of the channel and do not depend on the probability distribution of whatever drives the channel. Like probabilities, they all have values between 0 and 1

$$0 \leq c_{ji} \leq 1 \tag{6.14}$$

They may be thought of as forming a matrix with as many rows as there are input events, and as many columns as there are output events. Because each input event must lead to exactly one output event,

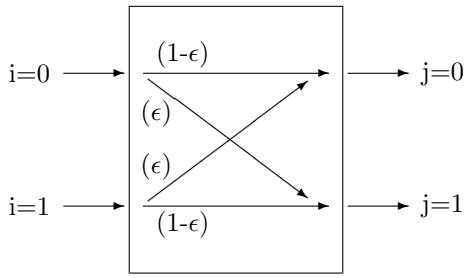

Figure 6.5: Symmetric binary channel

$$1 = \sum_j c_{ji} \tag{6.15}$$

for each i. For a noiseless channel, the various c_{ji} are all 0 except that for each value of i one of them is equal to 1.

When the channel is driven by a source with probabilities $p(A_i)$, the conditional probabilities of the output events, conditioned on the input events, is

$$p(B_j \mid A_i) = c_{ji} \tag{6.16}$$

The unconditional probability of each output $p(B_j)$ is

$$p(B_j) = \sum_i c_{ji} p(A_i) \tag{6.17}$$

The backward conditional probabilities $p(A_i \mid B_j)$ can be found using Bayes' Theorem:

$$\begin{aligned} p(A_i, B_j) &= p(B_j)p(A_i \mid B_j) \\ &= p(A_i)p(B_j \mid A_i) \\ &= p(A_i)c_{ji} \end{aligned} \tag{6.18}$$

The simplest noisy channel is the symmetric binary channel, for which we assume that there is a (hopefully small) probability e of an error, so

$$\begin{aligned} c_{01} = c_{10} &= \epsilon \\ c_{00} = c_{11} &= 1 - \epsilon \end{aligned} \tag{6.19}$$

This binary channel is called **symmetric** because the probability of an errors for all inputs are the same. If $\epsilon = 0$ then this channel is noiseless. Figure 6.3 can be extended to the noisy channel by indicating the possible transitions from input to output, as shown in Figure 6.5.

If the output B_j is in one of its (mutually exclusive) states, can the input A_i that caused it be determined? In the absence of noise, yes; there is no uncertainty about the input once the output is known. However, with noise there is some residual uncertainty. We will calculate this uncertainty in terms of the transition probabilities c_{ji} and define the information that we have learned about the input as a result of knowing the output as the **mutual information**. From that we will define the channel capacity C.

Before we know the output, what is our uncertainty U_{before} about the identity of the input event? This is the entropy of the input:

$$U_{\text{before}} = \sum_i p(A_i) \log_2 \left(\frac{1}{p(A_i)} \right) \tag{6.20}$$

After some particular output event B_j has been observed, what is the residual uncertainty $U_{\text{after}}(B_j)$ about the input event? A similar formula applies, with $p(A_i)$ replaced by the conditional backward probability $p(A_i \mid B_j)$:

$$U_{\text{after}}(B_j) = \sum_i p(A_i \mid B_j) \log_2 \left(\frac{1}{p(A_i \mid B_j)} \right) \tag{6.21}$$

The amount we have learned in the case of this particular output event is the difference between U_{before} and $U_{\text{before}}(B_j)$. The mutual information M is defined as the average, over all outputs, of the amount so learned,

$$M = U_{\text{before}} - \sum_j p(B_j) U_{\text{after}}(B_j) \tag{6.22}$$

It is not difficult to prove that $M \geq 0$ or, in other words, that our knowledge about the input is never made more uncertain by learning the output event. To prove this, the Gibbs inequality is used, for each j:

$$\begin{aligned} U_{\text{after}}(B_j) &= \sum_i p(A_i \mid B_j) \log_2 \left(\frac{1}{p(A_i \mid B_j)} \right) \\ &\leq \sum_i p(A_i \mid B_j) \log_2 \left(\frac{1}{p(A_i)} \right) \end{aligned} \tag{6.23}$$

This use of the Gibbs inequality is valid because, for each j, $p(A_i \mid B_j)$ is a probability distribution over i, and $p(A_i)$ is another probability distribution over i, different from the one doing the average. This inequality holds for every value of j and hence for the average over all j:

$$\begin{aligned} \sum_j p(B_j) U_{\text{after}}(B_j) &\leq \sum_j p(B_j) \sum_i p(A_i \mid B_j) \log_2 \left(\frac{1}{p(A_i)} \right) \\ &= \sum_{ji} p(B_j) p(A_i \mid B_j) \log_2 \left(\frac{1}{p(A_i)} \right) \\ &= \sum_{ij} p(B_j \mid A_i) p(A_i) \log_2 \left(\frac{1}{p(A_i)} \right) \\ &= \sum_i p(A_i) \log_2 \left(\frac{1}{p(A_i)} \right) \\ &= U_{\text{before}} \end{aligned} \tag{6.24}$$

We are now in a position to find M in terms of the input probability distribution and the properties of the channel. Straightforward substitution in the formula above leads to

$$M = \sum_j \left(\sum_i p(A_i) c_{ji} \right) \log_2 \left(\frac{1}{\sum_i p(A_i) c_{ji}} \right) - \sum_{ij} p(A_i) c_{ji} \log_2 \left(\frac{1}{c_{ji}} \right) \tag{6.25}$$

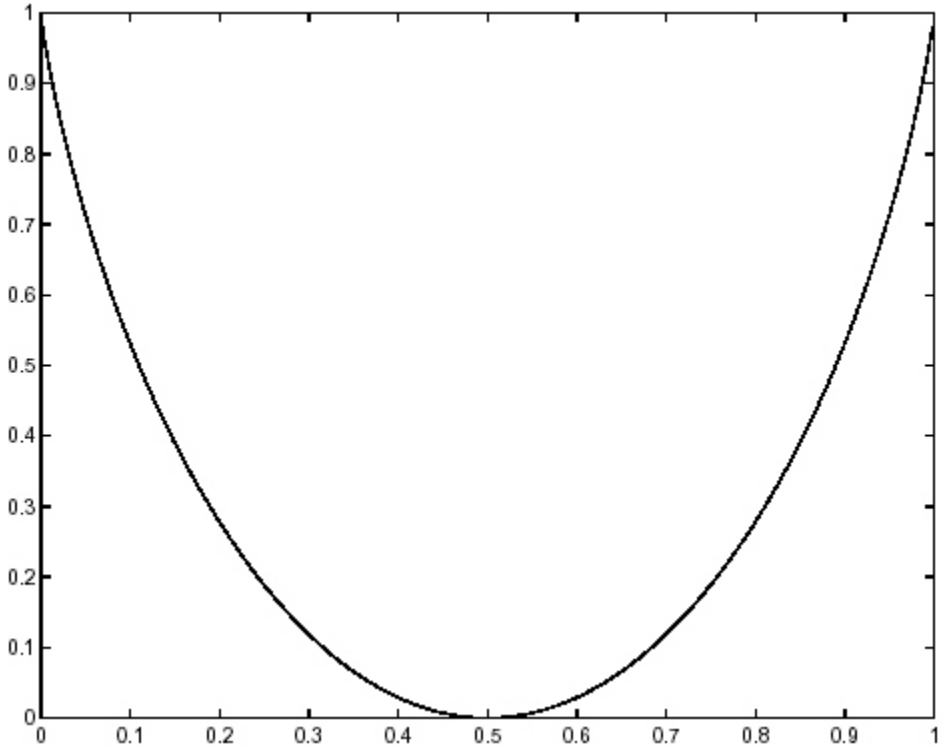

Figure 6.6: Mutual information as a function of ϵ

Rather than give a general interpretation of this, or other similar formulas, it is simpler to look at the symmetric binary channel. In this case both $p(A_i)$ and $p(B_j)$ are equal to 0.5 and so the first term in the expression for M is 1 and the second term is found in terms of ϵ:

$$M = 1 - \epsilon \log_2 \left(\frac{1}{\epsilon}\right) - (1-\epsilon) \log_2 \left(\frac{1}{(1-\epsilon)}\right) \tag{6.26}$$

which is 1 bit minus the expression for the entropy of a binary source with probabilities ϵ and $1-\epsilon$. This is a cup-shaped curve that goes from a value of 1 when $\epsilon = 0$ down to 0 at $\epsilon = 0.5$ and then back up to 1 when $\epsilon = 1$. The interpretation of this result is straightforward. When $\epsilon = 0$ (or when $\epsilon = 1$) the input can be determined exactly whenever the output is known, so there is no loss of information. The mutual information is therefore the same as the input information, 1 bit. When $\epsilon = 0.5$ each output is equally likely, no matter what the input, so learning the output event tells us nothing about the input. The mutual information therefore is 0. Figure 6.6 shows the mutual information.

6.7 Noisy Channel Capacity Theorem

The channel capacity of a noisy channel is defined in terms of the mutual information M. However, in general M depends not only on the channel (through the transfer probabilities c_{ji}) but also on the input probability distribution $p(A_i)$. It is more useful to define the channel capacity so that it depends only on the channel, so $M_{\max}$, the maximum mutual information that results from any possible input probability distribution, is used. In the case of the symmetric binary channel example, this maximum occurs when the two input probabilities are equal. Generally speaking, going away from the symmetric case offers few if any advantages in engineered systems, and in particular the fundamental limits given by the theorems in this

chapter cannot be evaded through loopholes like this. Therefore the symmetric case gives the right intuitive understanding.

The channel capacity is defined as

$$C = M_{\max} W \tag{6.27}$$

where W is the maximum rate at which the input state can be changed. Thus C is expressed in bits per second.

The channel capacity theorem, first proved by Shannon in 1948, gives a fundamental limit to the rate at which information can be transmitted through a channel. If the input information rate in bits per second D is less than C then it is possible (perhaps by dealing with long sequences of inputs together) to code the data in such a way that the error rate is as low as desired. On the other hand, if $D > C$ then this is not possible; in fact the maximum rate at which information about the input can be inferred from learning the output is C. This result is exactly the same as the result for the noiseless channel, shown in Figure 6.4.

This result is really quite remarkable. A capacity figure, which depends only on the channel, was defined and then the theorem states that a code which gives performance arbitrarily close to this capacity can be found. In conjunction with the source coding theorem, it implies that a communication channel can be designed in two stages – first, the source is encoded so that the average length of codewords is equal to its entropy, and second, this stream of bits can then be transmitted at any rate up to the channel capacity with arbitrarily low error. The channel capacity is not the same as the native rate at which the input can change, but rather is degraded from that value because of the noise.

Unfortunately, the proof of this theorem (which is not given here) does not indicate how to go about finding such a code. In other words, it is not a constructive proof, in which the assertion is proved by displaying such a code. In the half century since Shannon published this theorem, there have been numerous discoveries of better and better codes, to meet a variety of high speed data communication needs. However, there is not yet any general theory of how to design codes from scratch (such as the Huffman procedure provides for source coding).

6.8 Reversibility

It is instructive to recall which operations discussed so far involve loss of information, and which do not.

As for source coding, the use of Huffman codes or other codes for discrete sources is generally reversible, in the sense that the source decoder at the output can reconstruct the original stream of symbols without errors if presented with exactly the bit pattern generated by the source encoder. On the other hand, codes that represent symbols selected from infinite sets such as the real numbers or the integers cannot, with a finite number of bits, represent arbitrary symbols with complete accuracy. Instead, regions are defined and all points within a given region are represented in the same way. The source decoder then returns a representative member from the region, and the intent is that the difference between any two symbols in the same region are small enough.

As for compression, we have seen examples of both reversible and irreversible compression techniques. Generally speaking irreversible techniques give a greater degree of compression, and if well done the errors introduced can be small enough to not be noticed. Examples of very effective irreversible compression techniques are JPEG for images and MP3 for audio.

Now, in channel coding, two other type of irreversibility arise. First, if data is put into a channel faster than the channel capacity, some of the input information will necessarily be lost. Second, if the channel introduces noise, some information will be lost, although techniques do exist that can make this loss arbitrarily small if the channel capacity is higher than the input data rate.

In all these cases of irreversibility, information is lost, or kept unchanged in the limiting case of zero errors, or reversible compression or source coding. Never is information increased in any of the systems we have considered.

Is there a general principle here?

6.9 Detail: Communication System Requirements

The model of a communication system that we have been developing is shown in Figure 6.1. The source is assumed to emit a stream of symbols. The channel may be a physical channel between different points in space, or it may be a memory which stores information for retrieval at a later time, or it may even be a computation in which the information is processed in some way.

Naturally, different communication systems, though they all might be well described by our model, differ in their requirements. The following table is an attempt to illustrate the range of requirements that are reasonable for modern systems. It is, of course, not complete.

The systems are characterized by four measures: throughput, latency, tolerance of errors, and tolerance to nonuniform rate (bursts). Throughput is simply the number of bits per second that such a system should, to be successful, accommodate. Latency is the time delay of the message; it could be defined either as the delay of the start of the output after the source begins, or a similar quantity about the end of the message (or, for that matter, about any particular features in the message).

	Throughput (per second)	Maximum Latency	Error Tolerance	Bursts Tolerated?
Memory	??	microseconds	error-free	yes
Hard disk	MB or higher	milliseconds	error-free	yes
Conversation	??	50 ms	feedback error control	annoying
Telephone	20 kb	100 ms	noise tolerated	no
Radio broadcast	??	seconds	some noise tolerated	no
Instant message	low	seconds	error-free	yes
Compact Disc	1.4 MB	2 s	error-free	no
Internet	1 MB	5 s	error-free	yes
Print queue	1 MB	10 s	error-free	yes
Fax	14.4 kb	minutes	errors tolerated	yes
Shutter telegraph	??	1 min	error-free	yes
E-mail	N/A	1 hour	error-free	yes
Overnight delivery	large	1 day	error-free	yes
Parcel delivery	large	days	error-free	yes
Snail mail	large	days	error-free	yes

Table 6.1: Various Communication Systems