Chapter 6

Communications

We have been considering an information-handling system in which a stream of 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.

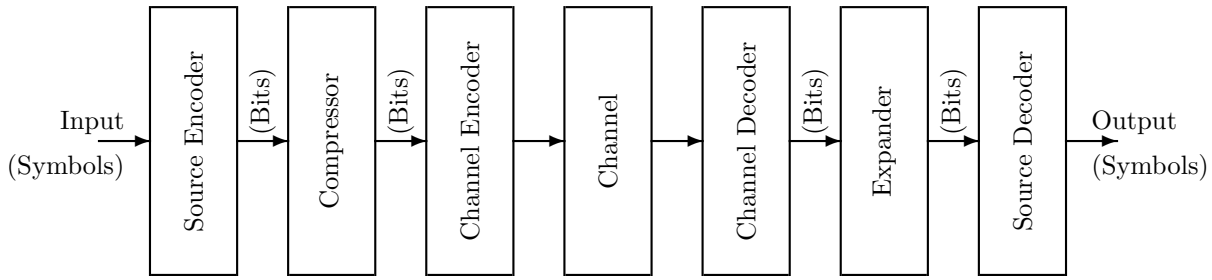

Figure 6.1: Communication system

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, only the information necessary to identify them. This is what enables the stream of symbols 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.

Let us suppose that each event A_i (the selection of the symbol i) is represented by a different codeword C_i with a length L_i (i.e., a string of L_i bits). 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 simply sequences of bits, the number available of each length is limited. For example, there are only four distinct two-bit codewords possible, namely 00, 01, 10, and 11.

Author: Paul Penfield, Jr.
This document: http://www-mtl.mit.edu/Courses/6.050/2014/notes/chapter6.pdf
Version 1.8, March 3, 2014.

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's Inequality

Since the number of distinct short codewords is limited, not all codewords can be short. Some must be longer, but then the prefix condition limits the available short codewords even further. An important limitation on the distribution of codeword lengths L_i was given by Leon G. Kraft, Jr. (1923–1989), an MIT student, in his 1949 Master's thesis. It is known as Kraft's 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 set of lengths L_i that obey it, a code can be found.

For example, suppose a code consists of the four distinct two-bit codewords 00, 01, 10, and 11. Then each $L_i = 2$ and each term in the sum in Equation 6.1 is $1/2^2 = 1/4$. In this case the equation evaluates to an equality, and there are several 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 Kraft's inequality is an 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 $(1/2^2) + (1/2^2) + (1/2^1) = 1$.

Kraft's inequality can be proven easily. Let L_{max} be the length of the longest codeword of a prefix-condition code. There are exactly $2^{L_{max}}$ different patterns of 0 and 1 of this length. Thus

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

where this sum is over these $2^{L_{max}}$ patterns, indexed by j (this equation looks unusual because the quantity being summed does not depend on j). For each codeword i of length L_{max} replace the term in Equation 6.2 by the corresponding $1/2^{L_i}$. The value of the sum is unchanged. That will happen at least once, but unless this happens to be a fixed-length code there are other shorter codewords. For each shorter codeword i of length L_i ($L_i < L_{max}$) there are exactly $2^{L_{max}-L_i}$ patterns that begin with this codeword, and none of those except the shorter codeword in question is a valid codeword (because this is a prefix-condition code). In the sum of Equation 6.2 replace the terms corresponding to those patterns by a single term equal to $1/2^{L_i}$. The sum is unchanged. Do the same with all the other short codewords. When this process is complete, there are terms in the sum corresponding to every codeword, and the sum is still equal to 1. There may be other terms corresponding to patterns that are not codewords—if so, eliminate them from the sum in Equation 6.2. The result is exactly the sum in Equation 6.1 and is less than or equal to 1. The proof is complete.

From this proof it is evident that if the sum in Kraft's inequality Equation 6.1 is less than 1, then there is at least one potential codeword that is not used, so at least one additional symbol could be added to the source without affecting the rest of the code. In this sense the code is inefficient.

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 H of the identity of the next symbol chosen is the average information gained when the next symbol is made known, Equation 5.14:

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

This quantity, also known as the **entropy** of the source, is expressed in bits per symbol (because the logarithm is base 2). The information rate, in bits per second, is HR where R is the rate at which the source selects the symbols, measured in symbols per second.

6.2.1 Gibbs' Inequality

Gibbs' inequality, named after the American physicist J. Willard Gibbs (1839–1903)[1], will be useful in later proofs. This inequality states that the entropy, Equation 6.3, is less than or equal to any other average formed using the same probabilities but a different probability distribution 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

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

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

It is also part of Gibbs' inequality that the two sides of Equation 6.4 are equal if and only if $p(A_i) = p'(A_i)$ for all i, that is, if the two probability distributions are identical.

Gibbs' inequality, Equation 6.4, can be proven easily. First, since $p(A_i)$ is a probability distribution,

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

Second, note that the natural logarithm $\ln x$ is **convex** in that when graphed, it is on or below a straight line that is tangent to it at any point (for example the point $x = 1$ as shown in Figure 6.2). Thus

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

or, when the conversion formula $\log_2 x = (\log_2 e)(\ln x)$ is used to convert Equation 6.8 to logarithm base 2,

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

Also, the two sides of Equation 6.9 are equal if and only if $x = 1$. Then

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

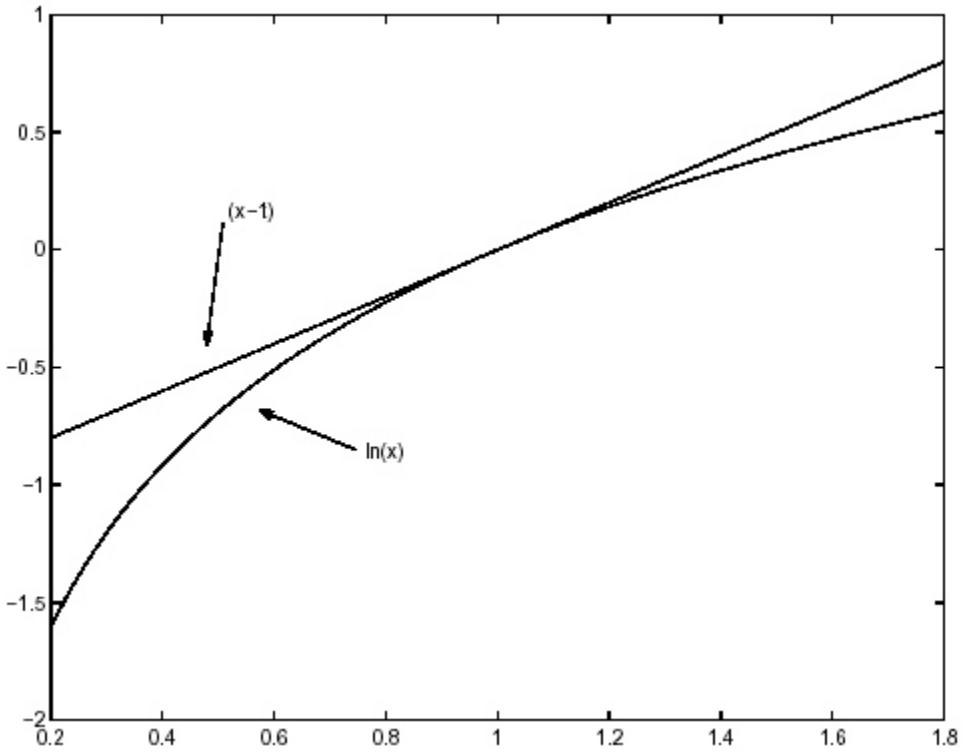

Figure 6.2: Graph demonstrating the inequality $\ln x \leq (x-1)$

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

If $p(A_i) \neq p'(A_i)$ for any i then the first inequality in Equation 6.10 is a strict inequality and therefore the two sides of Equation 6.4 are not equal. If $p(A_i) = p'(A_i)$ for all i the two sides of Equation 6.4 are obviously equal. This completes the proof of Gibbs' inequality.

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 L is wanted. Assigning high-probability symbols to the short codewords helps make L small; Huffman codes are optimal for this. However, there is a limit to how short the average codeword can be. The **Source Coding Theorem** states that the average codeword length L cannot be less than the average information per symbol:

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

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

$$\begin{aligned} H &= \sum_i p(A_i) \log_2 \left(\frac{1}{p(A_i)} \right) \\ &\leq \sum_i p(A_i) \log_2 \left(\frac{1}{p'(A_i)} \right) \\ &= \sum_i p(A_i) \log_2 2^{L_i} \\ &= \sum_i p(A_i) L_i \\ &= L \end{aligned} \tag{6.13}$$

The Source Coding Theorem can also be expressed in terms of rates of transmission in bits per second by multiplying Equation 6.12 by the symbols per second R:

$$HR \leq LR \tag{6.14}$$

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** and in that case nothing affects the output except the input. Let us say that the channel has a certain maximum rate W at which its output can follow changes at the input (just as the source has a rate R at which symbols are selected).

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 channel as something with inputs and outputs, as in 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 true at any one time. For simple channels such a diagram is simple because there are so few possible choices, but for more complicated structures there may be so many possible inputs that the diagrams become impractical (though they may be useful as a 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. The **binary** channel has two mutually exclusive input states and is the one pictured in Figure 6.3.

For a noiseless channel, where each of n possible input states leads to exactly one output state, each new input state (R 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 supplied to the input can affect the output 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 R 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 at which changes at the input affect the output) and some of the input information is lost.

6.5 Noiseless Channel Theorem

If the channel does not introduce any errors, 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 entropy per symbol of a source H expressed in bits per symbol, times the rate of the source R in symbols per second). 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 input 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 lead to 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 for email. A list of the needs of some practical communication systems, in Section 6.9, reveals a wide variation in required speed, throughput, latency, etc.

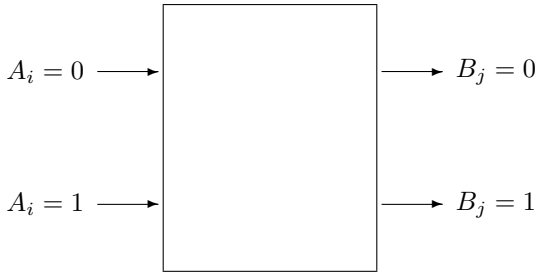

Figure 6.3: Binary Channel

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 outcome. Which actually happens is a matter of chance, and we will model the channel by the set of probabilities that each of the output events B_j occurs when one of the possible input events A_i happens. These **transition probabilities** c_{ji} are, of course, probabilities, but they are properties of the channel and do not depend on the probability distribution $p(A_i)$ of the input. Like all probabilities, they have values between 0 and 1

$$0 \leq c_{ji} \leq 1 \tag{6.15}$$

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

$$1 = \sum_j c_{ji} \tag{6.16}$$

for each i. (In other words, the sum of c_{ji} in each column i is 1.) If the channel is noiseless, for each value of i exactly one of the various c_{ji} is equal to 1 and all others are 0.

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

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

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

$$p(B_j) = \sum_i c_{ji} p(A_i) \tag{6.18}$$

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

$$\begin{aligned} p(A_i, B_j) &= p(B_j)p(A_i \mid B_j) \\ &= p(A_i)p(B_j \mid A_i) \\ &= p(A_i)c_{ji} \end{aligned} \tag{6.19}$$

The simplest noisy channel is the **symmetric binary channel**, for which there is a (hopefully small) probability ε of an error, so

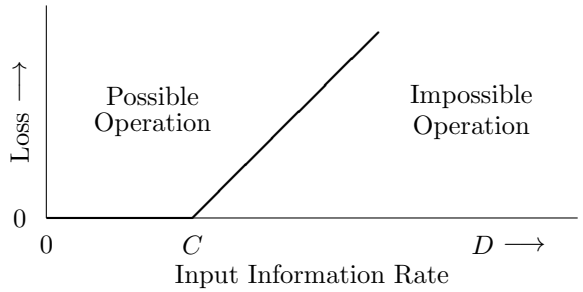

Figure 6.4: Channel Loss Diagram. For an input data rate D, either less than or greater than the channel capacity C, the minimum possible rate at which information is lost is the greater of 0 and $D - C$

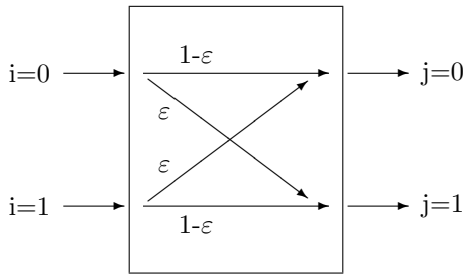

Figure 6.5: Symmetric binary channel

$$\begin{bmatrix} c_{00} & c_{01} \\ c_{10} & c_{11} \end{bmatrix} = \begin{bmatrix} 1-\varepsilon & \varepsilon \\ \varepsilon & 1-\varepsilon \end{bmatrix} \tag{6.20}$$

This binary channel is called **symmetric** because the probability of an error for both inputs is the same. If $\varepsilon = 0$ then this channel is noiseless (it is also noiseless if $\varepsilon = 1$, in which case it behaves like an inverter). Figure 6.3 can be made more useful for the noisy channel if the possible transitions from input to output are shown, as in Figure 6.5.

If the output B_j is observed to be 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** M. 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:

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

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 $p(A_i \mid B_j)$:

$$U_{\text{after}}(B_j) = \sum_i p(A_i \mid B_j) \log_2 \left(\frac{1}{p(A_i \mid B_j)} \right) \tag{6.22}$$

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

$$M = U_{\text{before}} - \sum_j p(B_j) U_{\text{after}}(B_j) \tag{6.23}$$

It is not difficult to prove that $M \geq 0$, i.e., that our knowledge about the input is not, on average, made more uncertain by learning the output event. To prove this, Gibbs' inequality is used, for each j:

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

This use of 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 therefore for the average over all j:

$$\begin{aligned} \sum_j p(B_j) U_{\text{after}}(B_j) &\leq \sum_j p(B_j) \sum_i p(A_i \mid B_j) \log_2 \left(\frac{1}{p(A_i)} \right) \\ &= \sum_{ji} p(B_j) p(A_i \mid B_j) \log_2 \left(\frac{1}{p(A_i)} \right) \\ &= \sum_{ij} p(B_j \mid A_i) p(A_i) \log_2 \left(\frac{1}{p(A_i)} \right) \\ &= \sum_i p(A_i) \log_2 \left(\frac{1}{p(A_i)} \right) \\ &= U_{\text{before}} \end{aligned} \tag{6.25}$$

We are now in a position to find M in terms of the input probability distribution and the properties of the channel. Substitution in Equation 6.23 and simplification leads to

$$M = \sum_j \left(\sum_i p(A_i) c_{ji} \right) \log_2 \left(\frac{1}{\sum_i p(A_i) c_{ji}} \right) - \sum_{ij} p(A_i) c_{ji} \log_2 \left(\frac{1}{c_{ji}} \right) \tag{6.26}$$

Note that Equation 6.26 was derived for the case where the input "causes" the output. At least, that was the way the description went. However, such a cause-and-effect relationship is not necessary. The term **mutual information** suggests (correctly) that it is just as valid to view the output as causing the input, or to ignore completely the question of what causes what. Two alternate formulas for M show that M can be interpreted in either direction:

$$\begin{aligned} M &= \sum_i p(A_i) \log_2 \left(\frac{1}{p(A_i)} \right) - \sum_j p(B_j) \sum_i p(A_i \mid B_j) \log_2 \left(\frac{1}{p(A_i \mid B_j)} \right) \\ &= \sum_j p(B_j) \log_2 \left(\frac{1}{p(B_j)} \right) - \sum_i p(A_i) \sum_j p(B_j \mid A_i) \log_2 \left(\frac{1}{p(B_j \mid A_i)} \right) \end{aligned} \tag{6.27}$$

Rather than give a general interpretation of these or similar formulas, let's simply 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 in Equation 6.26 is 1 and the second term is found in terms of ε:

$$M = 1 - \varepsilon \log_2 \left(\frac{1}{\varepsilon} \right) - (1 - \varepsilon) \log_2 \left(\frac{1}{(1 - \varepsilon)} \right) \tag{6.28}$$

which happens to be 1 bit minus the entropy of a binary source with probabilities ε and $1 - \varepsilon$. This is a cup-shaped curve that goes from a value of 1 when $\varepsilon = 0$ down to 0 at $\varepsilon = 0.5$ and then back up to 1 when $\varepsilon = 1$. See Figure 6.6. The interpretation of this result is straightforward. When $\varepsilon = 0$ (or when $\varepsilon = 1$)

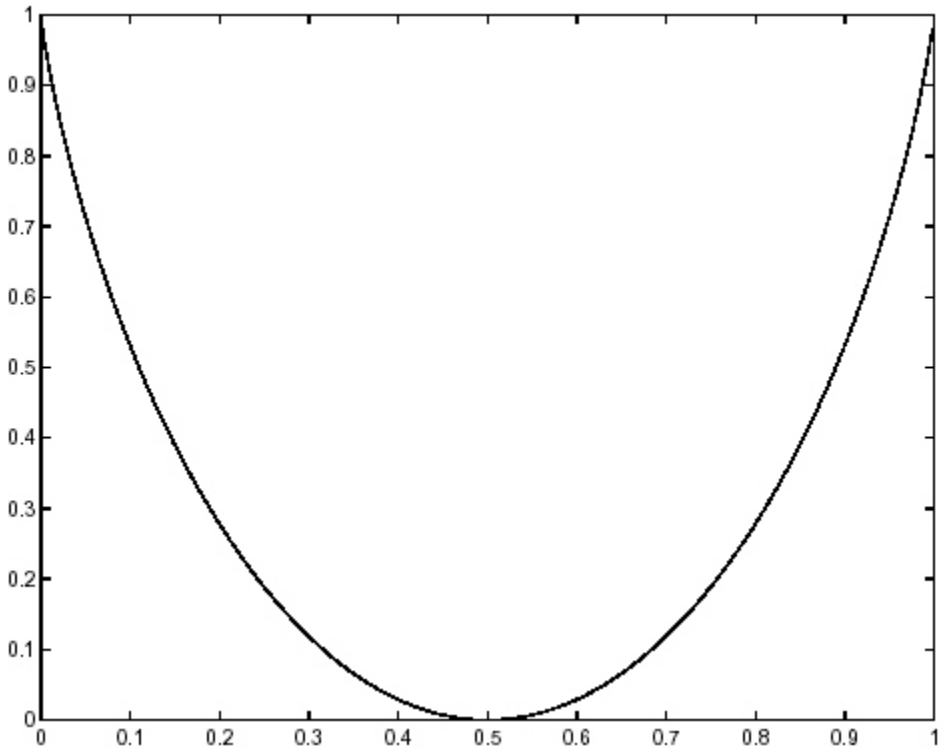

Figure 6.6: Mutual information, in bits, as a function of bit-error probability ε

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 $\varepsilon = 0.5$ each output is equally likely, no matter what the input is, so learning the output tells us nothing about the input. The mutual information is 0.

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, 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 such techniques. Therefore the symmetric case gives the right intuitive understanding.

The channel capacity is defined as

$$C = M_{\max} W \tag{6.29}$$

where W is the maximum rate at which the output state can follow changes at the input. Thus C is expressed in bits per second.

The channel capacity theorem was first proved by Claude Shannon (1916–2001) in 1948. It 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 (or close 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 the code. In the decades 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).

Shannon is rightly regarded as the greatest figure in communications in all history.[2] He established the entire field of scientific inquiry known today as information theory. He did this work while at Bell Laboratories, after he graduated from MIT with his S.M. in electrical engineering and Ph.D. in mathematics. It was he who recognized the binary digit as the fundamental element in all communications. In 1956 he returned to MIT as a faculty member. In his later life he suffered from Alzheimer's disease, and, sadly, was unable to attend a symposium held at MIT in 1998 honoring the 50th anniversary of his seminal paper (his wife Betty did attend on his behalf).

6.8 Reversibility

Which operations discussed so far involve loss of information, and which do not?

Some Boolean operations had the property that the input could not be deduced from the output. The *AND* and *OR* gates are examples. Other operations were reversible—the *XOR* gate, when the output is augmented by one of the two inputs, is an example.

Some sources may be encoded so that all possible symbols are represented by different codewords. This is always possible if the number of symbols is finite. Other sources have an infinite number of possible symbols, and these cannot be encoded exactly. Among the techniques used to encode such sources are binary codes for integers (which suffer from overflow problems) and floating-point representation of real numbers (which suffer from overflow and underflow problems and also from limited precision).

Some compression algorithms are reversible in the sense that the input can be recovered exactly from the output. One such technique is LZW, which is used for text compression and some image compression, among other things. Other algorithms achieve greater efficiency at the expense of some loss of information. Examples are JPEG compression of images and MP3 compression of audio.

Now we have seen that some communication channels are noiseless, and in that case there can be perfect transmission at rates up to the channel capacity. Other channels have noise, so perfect, reversible communication is not possible, although the error rate can be made arbitrarily small if the data rate is less than the channel capacity. For greater data rates the channel is necessarily irreversible.

In all these cases of irreversibility, information is lost, (or at best kept unchanged). Never is information increased in any of the systems we have considered.

Is there a general principle at work here?

[2] See a biography of Shannon at http://www-groups.dcs.st-andrews.ac.uk/%7Ehistory/Biographies/Shannon.html

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**, **error tolerance**, and **burst tolerance** (bursts result from a nonuniform rate of symbol selection or identification). 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). The numbers for throughput, in MB (megabytes) or kb (kilobits) per second are approximate or, in some cases, only guesses.

	Throughput (per second)	Maximum Latency	Errors Tolerated?	Bursts Tolerated?
Computer Memory	many MB	microseconds	no	yes
Hard disk	MB or higher	milliseconds	no	yes
Conversation	1 kb	50 ms	yes; feedback error control	annoying
Telephone	8 kb	100 ms	noise tolerated	no
AM Radio broadcast	32 kb	seconds	some noise tolerated	no
FM Radio broadcast	96 kb	seconds	some noise tolerated	no
Instant message (text)	low	seconds	no	yes
Audio CD	1.4 MB	2 s	no	no
HDTV	1.5 MB	2 s	no	no
Fast Ethernet	12 MB	1 s	no	no
WiFi (802.11n)	45 MB	1 s	no	no
Internet	1 MB	5 s	no	yes
Print queue	1 MB	30 s	no	yes
Fax	14.4 kb	minutes	errors tolerated	yes
Shutter telegraph station	??	5 min	no	yes
Email	N/A	1 hour	no	yes
Overnight delivery	large	1 day	no	yes
Parcel delivery	large	days	no	yes
Snail mail	large	days	no	yes

Table 6.1: Various Communication Systems