Shannon s Theory 44
2.1 Perfect Secrecy 44
Example 2.1 41
FIGURE 2.1 50
2.2 Entropy 51
2.2.1 Huffman Encodings and Entropy 53
2.3 Properties of Entropy 56
2.4 Spurious Keys and Unicity Distance ... 59
2.5 Product Cryptosystems 64
FIGURE 2.2 65
2.6 Notes 67
Exercises 67

2

Shannon's Theory

In 1949, Claude Shannon published a paper entitled "Communication Theory of Secrecy Systems" in the *Bell Systems Technical Journal*. This paper had a great influence on the scientific study of cryptography. In this chapter, we discuss several of Shannon's ideas.

2.1 Perfect Secrecy

There are two basic approaches to discussing the security of a cryptosystem.

computational security

This measure concerns the computational effort required to break a cryptosystem. We might define a cryptosystem to be *computationally secure* if the best algorithm for breaking it requires at least N operations, where N is some specified, very large number. The problem is that no known practical cryptosystem can be proved to be secure under this definition. In practice, people will call a cryptosystem "computationally secure" if the best known method of breaking the system requires an unreasonably large amount of computer time (but this is of course very different from a proof of security). Another approach is to provide evidence of computational security by reducing the security of the cryptosystem to some well-studied problem that is thought to be difficult. For example, it may be able to prove a statement of the type "a given cryptosystem is secure if a given integer n cannot be factored." Cryptosystems of this type are sometimes termed "provably secure," but it must be understood that this approach only provides a proof of security relative to some other problem, not an absolute proof of security. [1]

[1] This is a similar situation to proving that a problem is NP-complete: it proves that the given problem is at least as difficult as any other NP-complete problem, but it does not provide an absolute proof of the computational difficulty of the problem.

unconditional security

This measure concerns the security of cryptosystems when there is no bound placed on the amount of computation that Oscar is allowed to do. A cryptosystem is defined to be *unconditionally secure* if it cannot be broken, even with infinite computational resources.

When we discuss the security of a cryptosystem, we should also specify the type of attack that is being considered. In Chapter 1, we saw that neither the **Shift Cipher**, the **Substitution Cipher** nor the **Vigenère Cipher** is computationally secure against a ciphertext-only attack (given a sufficient amount of ciphertext).

What we will do in this section is to develop the theory of cryptosystems that are unconditionally secure against a ciphertext-only attack. It turns out that all three of the above ciphers are unconditionally secure if only one element of plaintext is encrypted with a given key!

The unconditional security of a cryptosystem obviously cannot be studied from the point of view of computational complexity, since we allow computation time to be infinite. The appropriate framework in which to study unconditional security is probability theory. We need only elementary facts concerning probability; the main definitions are reviewed now.

DEFINITION 2.1 *Suppose* $\mathbf{X}$ *and* $\mathbf{Y}$ *are random variables. We denote the probability that* $\mathbf{X}$ *takes on the value* x *by* $p(x)$*, and the probability that* $\mathbf{Y}$ *takes on the value* y *by* $p(y)$*. The* ***joint probability*** $p(x, y)$ *is the probability that* $\mathbf{X}$ *takes on the value* x *and* $\mathbf{Y}$ *takes on the value* y*. The* ***conditional probability*** $p(x|y)$ *denotes the probability that* $\mathbf{X}$ *takes on the value* x *given that* $\mathbf{Y}$ *takes on the value* y*. The random variables* $\mathbf{X}$ *and* $\mathbf{Y}$ *are said to be* ***independent*** *if* $p(x, y) = p(x)p(y)$ *for all possible values* x *of* $\mathbf{X}$ *and* y *of* $\mathbf{Y}$*.*

Joint probability can be related to conditional probability by the formula

$$p(x, y) = p(x|y)p(y).$$

Interchanging x and y, we have that

$$p(x, y) = p(y|x)p(x).$$

From these two expressions, we immediately obtain the following result, which is known as Bayes' Theorem.

THEOREM 2.1 ***(Bayes' Theorem)***
If $p(y) > 0$*, then*

$$p(x|y) = \frac{p(x)p(y|x)}{p(y)}.$$

COROLLARY 2.2
X *and* **Y** *are independent variables if and only if* $p(x|y) = p(x)$ *for all* x, y.

Throughout this section, we assume that a particular key is used for only one encryption. Let us suppose that there is a probability distribution on the plaintext space, $\mathcal{P}$. We denote the *a priori* probability that plaintext x occurs by $p_{\mathcal{P}}(x)$. We also assume that the key K is chosen (by Alice and Bob) using some fixed probability distribution (often a key is chosen at random, so all keys will be equiprobable, but this need not be the case). Denote the probability that key K is chosen by $p_{\mathcal{K}}(K)$. Recall that the key is chosen before Alice knows what the plaintext will be. Hence, we make the reasonable assumption that the key K and the plaintext x are independent events.

The two probability distributions on $\mathcal{P}$ and $\mathcal{K}$ induce a probability distribution on $\mathcal{C}$. Indeed, it is not hard to compute the probability $p_{\mathcal{C}}(y)$ that y is the ciphertext that is transmitted. For a key $K \in \mathcal{K}$, define

$$C(K) = \{e_K(x) : x \in \mathcal{P}\}.$$

That is, $C(K)$ represents the set of possible ciphertexts if K is the key. Then, for every $y \in \mathcal{C}$, we have that

$$p_{\mathcal{C}}(y) = \sum_{\{K:y\in C(K)\}} p_{\mathcal{K}}(K)p_{\mathcal{P}}(d_K(y)).$$

We also observe that, for any $y \in \mathcal{C}$ and $x \in \mathcal{P}$, we can compute the conditional probability $p_{\mathcal{C}}(y|x)$ (i.e., the probability that y is the ciphertext, given that x is the plaintext) to be

$$p_{\mathcal{C}}(y|x) = \sum_{\{K:x=d_K(y)\}} p_{\mathcal{K}}(K).$$

It is now possible to compute the conditional probability $p_{\mathcal{P}}(x|y)$ (i.e., the probability that x is the plaintext, given that y is the ciphertext) using Bayes' Theorem. The following formula is obtained:

$$p_{\mathcal{P}}(x|y) = \frac{p_{\mathcal{P}}(x) \sum_{\{K:x=d_K(y)\}} p_{\mathcal{K}}(K)}{\sum_{\{K:y\in C(K)\}} p_{\mathcal{K}}(K)p_{\mathcal{P}}(d_K(y))}.$$

Observe that all these calculations can be performed by anyone who knows the probability distributions.

We present a toy example to illustrate the computation of these probability distributions.

Example 2.1
Let $\mathcal{P} = \{a, b\}$ with $p_{\mathcal{P}}(a) = 1/4, p_{\mathcal{P}}(b) = 3/4$. Let $\mathcal{K} = \{K_1, K_2, K_3\}$ with $p_{\mathcal{K}}(K_1) = 1/2, p_{\mathcal{K}}(K_2) = p_{\mathcal{K}}(K_3) = 1/4$. Let $\mathcal{C} = \{1, 2, 3, 4\}$, and suppose the encryption functions are defined to be $e_{K_1}(a) = 1, e_{K_1}(b) = 2$; $e_{K_2}(a) = 2, e_{K_2}(b) = 3$; and $e_{K_3}(a) = 3, e_{K_3}(b) = 4$. This cryptosystem can be represented by the following *encryption matrix*:

	a	b
K_1	1	2
K_2	2	3
K_3	3	4

We now compute the probability distribution $p_{\mathcal{C}}$. We obtain

$$\begin{aligned} p_{\mathcal{C}}(1) &= \frac{1}{8} \\ p_{\mathcal{C}}(2) &= \frac{3}{8} + \frac{1}{16} = \frac{7}{16} \\ p_{\mathcal{C}}(3) &= \frac{3}{16} + \frac{1}{16} = \frac{1}{4} \\ p_{\mathcal{C}}(4) &= \frac{3}{16}. \end{aligned}$$

Now we can compute the conditional probability distributions on the plaintext, given that a certain ciphertext has been observed. We have:

$$\begin{aligned} p_{\mathcal{P}}(a|1) &= 1 & p_{\mathcal{P}}(b|1) &= 0 \\ p_{\mathcal{P}}(a|2) &= \frac{1}{7} & p_{\mathcal{P}}(b|2) &= \frac{6}{7} \\ p_{\mathcal{P}}(a|3) &= \frac{1}{4} & p_{\mathcal{P}}(b|3) &= \frac{3}{4} \\ p_{\mathcal{P}}(a|4) &= 0 & p_{\mathcal{P}}(b|4) &= 1. \end{aligned}$$

□

We are now ready to define the concept of perfect secrecy. Informally, perfect secrecy means that Oscar can obtain no information about plaintext by observing the ciphertext. This idea is made precise by formulating it in terms of the probability distributions we have defined, as follows.

DEFINITION 2.2 *A cryptosystem has* ***perfect secrecy*** *if* $p_{\mathcal{P}}(x|y) = p_{\mathcal{P}}(x)$ *for all* $x \in \mathcal{P}$, $y \in \mathcal{C}$. *That is, the a posteriori probability that the plaintext is* x, *given that the ciphertext* y *is observed, is identical to the a priori probability that the plaintext is* x.

In Example 2.1, the perfect secrecy property is satisfied for the ciphertext 3, but not for the other three ciphertexts.

We next prove that the **Shift Cipher** provides perfect secrecy. This seems quite obvious intuitively. For, if we are given any ciphertext element $y \in \mathbb{Z}_{26}$, then any plaintext element $x \in \mathbb{Z}_{26}$ is a possible decryption of y, depending on the value of the key. The following theorem gives the formal statement and proof using probability distributions.

THEOREM 2.3
Suppose the 26 *keys in the* **Shift Cipher** *are used with equal probability* $1/26$. *Then for any plaintext probability distribution, the* **Shift Cipher** *has perfect secrecy.*

PROOF Recall that $\mathcal{P} = \mathcal{C} = \mathcal{K} = \mathbb{Z}_{26}$, and for $0 \leq K \leq 25$, the encryption rule e_K is $e_K(x) = x + K \bmod 26$ ($x \in \mathbb{Z}_{26}$). First, we compute the distribution $p_{\mathcal{C}}$. Let $y \in \mathbb{Z}_{26}$; then

$$\begin{aligned} p_{\mathcal{C}}(y) &= \sum_{K \in \mathbb{Z}_{26}} p_{\mathcal{K}}(K) p_{\mathcal{P}}(d_K(y)) \\ &= \sum_{K \in \mathbb{Z}_{26}} \frac{1}{26} p_{\mathcal{P}}(y - K) \\ &= \frac{1}{26} \sum_{K \in \mathbb{Z}_{26}} p_{\mathcal{P}}(y - K). \end{aligned}$$

Now, for fixed y, the values $y - K \bmod 26$ comprise a permutation of $\mathbb{Z}_{26}$, and $p_{\mathcal{P}}$ is a probability distribution. Hence we have that

$$\begin{aligned} \sum_{K \in \mathbb{Z}_{26}} p_{\mathcal{P}}(y - K) &= \sum_{y \in \mathbb{Z}_{26}} p_{\mathcal{P}}(y) \\ &= 1. \end{aligned}$$

Consequently,

$$p_{\mathcal{C}}(y) = \frac{1}{26}$$

for any $y \in \mathbb{Z}_{26}$.

Next, we have that

$$p_{\mathcal{C}}(y|x) = p_{\mathcal{K}}(y - x \bmod 26)$$

$$= \frac{1}{26}$$

for every x, y, since for every x, y the unique key K such that $e_K(x) = y$ is $K = y - x \bmod 26$. Now, using Bayes' Theorem, it is trivial to compute

$$\begin{aligned} p_{\mathcal{P}}(x|y) &= \frac{p_{\mathcal{P}}(x)p_{\mathcal{C}}(y|x)}{p_{\mathcal{C}}(y)} \\ &= \frac{p_{\mathcal{P}}(x)\frac{1}{26}}{\frac{1}{26}} \\ &= p_{\mathcal{P}}(x), \end{aligned}$$

so we have perfect secrecy. ∎

So, the **Shift Cipher** is "unbreakable" provided that a new random key is used to encrypt every plaintext character.

Let us next investigate perfect secrecy in general. First, we observe that, using Bayes' Theorem, the condition that $p_{\mathcal{P}}(x|y) = p_{\mathcal{P}}(x)$ for all $x \in \mathcal{P}$, $y \in \mathcal{C}$ is equivalent to $p_{\mathcal{C}}(y|x) = p_{\mathcal{C}}(y)$ for all $x \in \mathcal{P}$, $y \in \mathcal{C}$. Now, let us make the reasonable assumption that $p_{\mathcal{C}}(y) > 0$ for all $y \in \mathcal{C}$ (if $p_{\mathcal{C}}(y) = 0$, then ciphertext y is never used and can be omitted from $\mathcal{C}$). Fix any $x \in \mathcal{P}$. For each $y \in \mathcal{C}$, we have $p_{\mathcal{C}}(y|x) = p_{\mathcal{C}}(y) > 0$. Hence, for each $y \in \mathcal{C}$, there must be at least one key K such that $e_K(x) = y$. It follows that $|\mathcal{K}| \geq |\mathcal{C}|$. In any cryptosystem, we must have $|\mathcal{C}| \geq |\mathcal{P}|$ since each encoding rule is an injection. In the boundary case $|\mathcal{K}| = |\mathcal{C}| = |\mathcal{P}|$, we can give a nice characterization of when perfect secrecy can be obtained. This characterization is originally due to Shannon.

THEOREM 2.4

Suppose $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ is a cryptosystem where $|\mathcal{K}| = |\mathcal{C}| = |\mathcal{P}|$. Then the cryptosystem provides perfect secrecy if and only if every key is used with equal probability $1/|\mathcal{K}|$, and for every $x \in \mathcal{P}$ and every $y \in \mathcal{C}$, there is a unique key K such that $e_K(x) = y$.

PROOF Suppose the given cryptosystem provides perfect secrecy. As observed above, for each $x \in \mathcal{P}$ and $y \in \mathcal{C}$ there must be at least one key K such that $e_K(x) = y$. So we have the inequalities:

$$\begin{aligned} |\mathcal{C}| &= |\{e_K(x) : K \in \mathcal{K}\}| \\ &\leq |\mathcal{K}|. \end{aligned}$$

But we are assuming that $|\mathcal{C}| = |\mathcal{K}|$. Hence, it must be the case that

$$|\{e_K(x) : K \in \mathcal{K}\}| = |\mathcal{K}|.$$

That is, there do not exist two distinct keys K_1 and K_2 such that $e_{K_1}(x) = e_{K_2}(x) = y$. Hence, we have shown that for any $x \in \mathcal{P}$ and $y \in \mathcal{C}$, there is exactly

FIGURE 2.1
One-time Pad

Let $n \geq 1$ be an integer, and take $\mathcal{P} = \mathcal{C} = \mathcal{K} = (\mathbb{Z}_2)^n$. For $K \in (\mathbb{Z}_2)^n$, define $e_K(x)$ to be the vector sum modulo 2 of K and x (or, equivalently, the exclusive-or of the two associated bitstrings). So, if $x = (x_1, \ldots, x_n)$ and $K = (K_1, \ldots, K_n)$, then

$$e_K(x) = (x_1 + K_1, \ldots, x_n + K_n) \bmod 2.$$

Decryption is identical to encryption. If $y = (y_1, \ldots, y_n)$, then

$$d_K(y) = (y_1 + K_1, \ldots, y_n + K_n) \bmod 2.$$

one key K such that $e_K(x) = y$.

Denote $n = |\mathcal{K}|$. Let $\mathcal{P} = \{x_i : 1 \leq i \leq n\}$ and fix a $y \in \mathcal{C}$. We can name the keys $K_1, K_2, \ldots, K_n$, in such a way that $e_{K_i}(x_i) = y$, $1 \leq i \leq n$. Using Bayes' theorem, we have

$$\begin{aligned} p_{\mathcal{P}}(x_i|y) &= \frac{p_{\mathcal{C}}(y|x_i)p_{\mathcal{P}}(x_i)}{p_{\mathcal{C}}(y)} \\ &= \frac{p_{\mathcal{K}}(K_i)p_{\mathcal{P}}(x_i)}{p_{\mathcal{C}}(y)}. \end{aligned}$$

Consider the perfect secrecy condition $p_{\mathcal{P}}(x_i|y) = p_{\mathcal{P}}(x_i)$. From this, it follows that $p_{\mathcal{K}}(K_i) = p_{\mathcal{C}}(y)$, for $1 \leq i \leq n$. This says that the keys are used with equal probability (namely, $p_{\mathcal{C}}(y)$). But since the number of keys is $|\mathcal{K}|$, we must have that $p_{\mathcal{K}}(K) = 1/|\mathcal{K}|$ for every $K \in \mathcal{K}$.

Conversely, suppose the two hypothesized conditions are satisfied. Then the cryptosystem is easily seen to provide perfect secrecy for any plaintext probability distribution, in a similar manner as the proof of Theorem 2.3. We leave the details for the reader. ∎

One well-known realization of perfect secrecy is the **Vernam One-time Pad**, which was first described by Gilbert Vernam in 1917 for use in automatic encryption and decryption of telegraph messages. It is interesting that the **One-time Pad** was thought for many years to be an "unbreakable" cryptosystem, but there was no proof of this until Shannon developed the concept of perfect secrecy over 30 years later.

The description of the **One-time Pad** is given in Figure 2.1.

Using Theorem 2.4, it is easily seen that the **One-time Pad** provides perfect secrecy. The system is also attractive because of the ease of encryption and

decryption.

Vernam patented his idea in the hope that it would have widepread commercial use. Unfortunately, there are major disadvantages to unconditionally secure cryptosystems such as the **One-time Pad**. The fact that $|\mathcal{K}| \geq |\mathcal{P}|$ means that the amount of key that must be communicated securely is at least as big as the amount of plaintext. For example, in the case of the **One-time Pad**, we require n bits of key to encrypt n bits of plaintext. This would not be a major problem if the same key could be used to encrypt different messages; however, the security of unconditionally secure cryptosystems depends on the fact that each key is used for only one encryption. (This is the reason for the term "one-time" in the **One-time Pad**.)

For example, the **One-time Pad** is vulnerable to a known-plaintext attack, since K can be computed as the exclusive-or of the bitstrings x and $e_K(x)$. Hence, a new key needs to be generated and communicated over a secure channel for every message that is going to be sent. This creates severe key management problems, which has limited the use of the **One-time Pad** in commercial applications. However, the **One-time Pad** has seen application in military and diplomatic contexts, where unconditional security may be of great importance.

The historical development of cryptography has been to try to design cryptosystems where one key can be used to encrypt a relatively long string of plaintext (i.e., one key can be used to encrypt many messages) and still maintain (at least) computational security. One such system is the **Data Encryption Standard**, which we will study in Chapter 3.

2.2 Entropy

In the previous section, we discussed the concept of perfect secrecy. We restricted our attention to the special situation where a key is used for only one encryption. We now want to look at what happens as more and more plaintexts are encrypted using the *same* key, and how likely a cryptanalyst will be able to carry out a successful ciphertext-only attack, given sufficient time.

The basic tool in studying this question is the idea of entropy, a concept from information theory introduced by Shannon in 1948. Entropy can be thought of as a mathematical measure of information or uncertainty, and is computed as a function of a probability distribution.

Suppose we have a random variable $\mathbf{X}$ which takes on a finite set of values according to a probability distribution $p(\mathbf{X})$. What is the information gained by an event which takes place according to distribution $p(\mathbf{X})$? Equivalently, if the event has not (yet) taken place, what is the uncertainty about the outcome? This quantity is called the entropy of $\mathbf{X}$ and is denoted by $H(\mathbf{X})$.

These ideas may seem rather abstract, so let's look at a more concrete example.

Suppose our random variable $\mathbf{X}$ represents the toss of a coin. The probability distribution is $p(heads) = p(tails) = 1/2$. It would seem reasonable to say that the information, or entropy, of a coin toss is one bit, since we could encode *heads* by 1 and *tails* by 0, for example. In a similar fashion, the entropy of n independent coin tosses is n, since the n coin tosses can be encoded by a bit string of length n.

As a slightly more complicated example, suppose we have a random variable $\mathbf{X}$ that takes on three possible values x_1, x_2, x_3 with probabilities $1/2, 1/4, 1/4$ respectively. The most efficient "encoding" of the three possible outcomes is to encode x_1 as 0, to encode x_2 as 10 and to encode x_3 as 11. Then the average number of bits in an encoding of $\mathbf{X}$ is

$$\frac{1}{2} \times 1 + \frac{1}{4} \times 2 + \frac{1}{4} \times 2 = \frac{3}{2}.$$

The above examples suggest that an event which occurs with probability 2^{-n} can be encoded as a bit string of length n. More generally, we could imagine that an event occurring with probability p might be encoded by a bit string of length approximately $-\log_2 p$. Given an arbitrary probability distribution $p_1, p_2, \ldots, p_n$ for a random variable $\mathbf{X}$, we take the weighted average of the quantities $-\log_2 p_i$ to be our measure of information. This motivates the following formal definition.

DEFINITION 2.3 *Suppose $\mathbf{X}$ is a random variable which takes on a finite set of values according to a probability distribution $p(\mathbf{X})$. Then, the* ***entropy*** *of this probability distribution is defined to be the quantity*

$$H(\mathbf{X}) = -\sum_{i=1}^{n} p_i \log_2 p_i.$$

If the possible values of $\mathbf{X}$ are x_i, $1 \leq i \leq n$, then we have

$$H(\mathbf{X}) = -\sum_{i=1}^{n} p(\mathbf{X} = x_i) \log_2 p(\mathbf{X} = x_i).$$

REMARK Observe that $\log_2 p_i$ is undefined if $p_i = 0$. Hence, entropy is sometimes defined to be the relevant sum over all the non-zero probabilities. Since $\lim_{x \to 0} x \log_2 x = 0$, there is no real difficulty with allowing $p_i = 0$ for some i. However, we will implicitly assume that, when computing the entropy of a probability distribution p_i, the sum is taken over the indices i such that $p_i \neq 0$. Also, we note that the choice of two as the base of the logarithms is arbitrary: another base would only change the value of the entropy by a constant factor. ∎

Note that if $p_i = 1/n$ for $1 \leq i \leq n$, then $H(\mathbf{X}) = \log_2 n$. Also, it is easy to see that $H(\mathbf{X}) \geq 0$, and $H(\mathbf{X}) = 0$ if and only if $p_i = 1$ for some i and $p_j = 0$ for all $j \neq i$.

Let us look at the entropy of the various components of a cryptosystem. We can think of the key as being a random variable $\mathbf{K}$ that takes on values according to the probability distribution $p_{\mathcal{K}}$, and hence we can compute the entropy $H(\mathbf{K})$. Similarly, we can compute entropies $H(\mathbf{P})$ and $H(\mathbf{C})$ associated with plaintext and ciphertext probability distributions, respectively.

To illustrate, we compute the entropies of the cryptosystem of Example 2.1.

Example 2.1 ***(Cont.)***
We compute as follows:

$$\begin{aligned} H(\mathbf{P}) &= -\frac{1}{4}\log_2\frac{1}{4} - \frac{3}{4}\log_2\frac{3}{4} \\ &= -\frac{1}{4}(-2) - \frac{3}{4}(\log_2 3 - 2) \\ &= 2 - \frac{3}{4}(\log_2 3) \\ &\approx 0.81. \end{aligned}$$

Similar calculations yield $H(\mathbf{K}) = 1.5$ and $H(\mathbf{C}) \approx 1.85$. □

2.2.1 Huffman Encodings and Entropy

In this section, we discuss briefly the connection between entropy and Huffman encodings. As the results in this section are not relevant to the cryptographic applications of entropy, it may be skipped without loss of continuity. However, this discussion may serve to further motivate the concept of entropy.

We introduced entropy in the context of encodings of random events which occur according to a specified probability distribution. We first make these ideas more precise. As before, $\mathbf{X}$ is a random variable which takes on a finite set of values, and $p(\mathbf{X})$ is the associated probability distribution.

An *encoding* of $\mathbf{X}$ is any mapping

$$f : \mathbf{X} \to \{0,1\}^*,$$

where $\{0,1\}^*$ denotes the set of all finite strings of 0's and 1's. Given a finite list (or string) of events $x_1 \ldots x_n$, we can extend the encoding f in an obvious way by defining

$$f(x_1 \ldots x_n) = f(x_1) \,||\, \ldots \,||\, f(x_n)$$

where $||$ denotes concatenation. In this way, we can think of f as a mapping

$$f : \mathbf{X}^* \to \{0,1\}^*.$$

Now, suppose a string $x_1 \ldots x_n$ is produced by a *memoryless source*, such that each x_i occurs according to the probability distribution on $\mathbf{X}$. This means that the

probability of any string $x_1 \ldots x_n$ is computed to be $p(x_1) \times \ldots \times p(x_n)$. (Notice that this string need *not* consist of distinct values, since the source is memoryless. As a simple example, consider a sequence of n tosses of a fair coin.)

Now, given that we are going to encode strings using the mapping f, it is important that we are able to decode in an unambiguous fashion. Thus it should be the case that the encoding f is injective.

Example 2.2
Suppose $\mathbf{X} = \{a, b, c, d\}$, and consider the following three encodings:

$$\begin{array}{cccccccccccc} f(a) & = & 1 & f(b) & = & 10 & f(c) & = & 100 & f(d) & = & 1000 \\ g(a) & = & 0 & g(b) & = & 10 & g(c) & = & 110 & g(d) & = & 111 \\ h(a) & = & 0 & h(b) & = & 01 & h(c) & = & 10 & h(d) & = & 11 \end{array}$$

It can be seen that f and g are injective encodings, but h is not. Any encoding using f can be decoded by starting at the end and working backwards: every time a 1 is encountered, it signals the end of the current element.

An encoding using g can be decoded by starting at the beginning and proceeding sequentially. At any point where we have a substring that is an encoding of a, b, c, or d, we decode it and chop off the substring. For example, given the string 10101110, we decode 10 to b, then 10 to b, then 111 to d, and finally 0 to a. So the decoded string is $bbda$.

To see that h is not injective, it suffices to give an example:

$$h(ac) = h(ba) = 010.$$

□

From the point of view of ease of decoding, we would prefer the encoding g to f. This is because decoding can be done sequentially from beginning to end if g is used, so no memory is required. The property that allows the simple sequential decoding of g is called the prefix-free property. (An encoding g is *prefix-free* if there do *not* exist two elements $x, y \in \mathbf{X}$, and a string $z \in \{0,1\}^*$ such that $g(x) = g(y) \,||\, z$.)

The discussion to this point has not involved entropy. Not surprisingly, entropy is related to the efficiency of an encoding. We will measure the efficiency of an encoding f as we did before: it is the weighted average length (denoted by $\ell(f)$) of an encoding of an element of $\mathbf{X}$. So we have the following definition:

$$\ell(f) = \sum_{x \in \mathbf{X}} p(x)|f(x)|,$$

where $|y|$ denotes the length of a string y.

Now, our fundamental problem is to find an injective encoding, f, that minimizes $\ell(f)$. There is a well-known algorithm, known as Huffman's algorithm, that accomplishes this goal. Moreover, the encoding f produced by Huffman's algorithm is prefix-free, and

$$H(\mathbf{X}) \leq \ell(f) < H(\mathbf{X}) + 1.$$

Thus, the value of the entropy provides a close estimate to the average length of the optimal injective encoding.

We will not prove the results stated above, but we will give a short, informal description of Huffman's algorithm. Huffman's algorithm begins with the probability distribution on the set $\mathbf{X}$, and the code of each element is initially empty. In each iteration, the two elements having lowest probability are combined into one element having as its probability the sum of the two smaller probabilities. The smaller of the two elements is assigned the value "0" and the larger of the two elements is assigned the value "1." When only one element remains, the coding for each $x \in \mathbf{X}$ can be constructed by following the sequence of elements "backwards" from the final element to the initial element x.

This is easily illustrated with an example.

Example 2.3

Suppose $\mathbf{X} = \{a, b, c, d, e\}$ has the following probability distribution: $p(a) = .05$, $p(b) = .10$, $p(c) = .12$, $p(d) = .13$ and $p(e) = .60$. Huffman's algorithm would proceed as indicated in the following table:

a	b	c	d	e
.05	.10	.12	.13	.60
0	1			
.15		.12	.13	.60
		0	1	
.15		.25		.60
0		1		
.40				.60
0				1
1.0				

This leads to the following encodings:

x	$f(x)$
a	000
b	001
c	010
d	011
e	1

Thus, the average length encoding is

$$\begin{aligned}\ell(f) &= .05 \times 3 + .10 \times 3 + .12 \times 3 + .13 \times 3 + .60 \times 1 \\ &= 1.8.\end{aligned}$$

Compare this to the entropy:

$$\begin{aligned}H(\mathbf{X}) &= .2161 + .3322 + .3671 + .3842 + .4422 \\ &= 1.7402.\end{aligned}$$

□

2.3 Properties of Entropy

In this section, we prove some fundamental results concerning entropy. First, we state a fundamental result, known as Jensen's Inequality, that will be very useful to us. Jensen's Inequality involves concave functions, which we now define.

DEFINITION 2.4 *A real-valued function f is* ***concave*** *on an interval I if*

$$f\left(\frac{x+y}{2}\right) \geq \frac{f(x)+f(y)}{2}$$

for all $x, y \in I$. f is ***strictly concave*** *if on an interval I if*

$$f\left(\frac{x+y}{2}\right) > \frac{f(x)+f(y)}{2}$$

for all $x, y \in I$, $x \neq y$.

Here is Jensen's Inequality, which we state without proof.

THEOREM 2.5 ***(Jensen's Inequality)***
Suppose f is a continuous strictly concave function on the interval I,

$$\sum_{i=1}^{n} a_i = 1,$$

and $a_i > 0$, $1 \leq i \leq n$. Then

$$\sum_{i=1}^{n} a_i f(x_i) \leq f\left(\sum_{i=1}^{n} a_i x_i\right),$$

where $x_i \in I$, $1 \leq i \leq n$. Further, equality occurs if and only if $x_1 = \ldots = x_n$.

We now proceed to derive several results on entropy. In the next theorem, we make use of the fact that the function $\log_2 x$ is strictly concave on the interval $(0, \infty)$. (In fact, this follows easily from elementary calculus since the second deriviative of the logarithm function is negative on the interval $(0, \infty)$.)

THEOREM 2.6

Suppose $\mathbf{X}$ *is a random variable having probability distribution* $p_1, p_2, \ldots, p_n$, *where* $p_i > 0$, $1 \leq i \leq n$. *Then* $H(\mathbf{X}) \leq \log_2 n$, *with equality if and only if* $p_i = 1/n$, $1 \leq i \leq n$.

PROOF Applying Jensen's Inequality, we have the following:

$$\begin{aligned} H(\mathbf{X}) &= -\sum_{i=1}^{n} p_i \log_2 p_i \\ &= \sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i} \\ &\leq \log_2 \sum_{i=1}^{n} \left(p_i \times \frac{1}{p_i} \right) \\ &= \log_2 n. \end{aligned}$$

Further, equality occurs if and only if $p_i = 1/n$, $1 \leq i \leq n$. ∎

THEOREM 2.7

$H(\mathbf{X}, \mathbf{Y}) \leq H(\mathbf{X}) + H(\mathbf{Y})$, *with equality if and only if* $\mathbf{X}$ *and* $\mathbf{Y}$ *are independent events.*

PROOF Suppose $\mathbf{X}$ takes on values x_i, $1 \leq i \leq m$, and $\mathbf{Y}$ takes on values y_j, $1 \leq j \leq n$. Denote $p_i = p(\mathbf{X} = x_i)$, $1 \leq i \leq m$, and $q_j = p(\mathbf{Y} = y_j)$, $1 \leq j \leq n$. Denote $r_{ij} = p(\mathbf{X} = x_i, \mathbf{Y} = y_j)$, $1 \leq i \leq m$, $1 \leq j \leq n$ (this is the joint probability distribution).

Observe that

$$p_i = \sum_{j=1}^{n} r_{ij}$$

$(1 \leq i \leq m)$ and

$$q_j = \sum_{i=1}^{m} r_{ij}$$

$(1 \leq j \leq n)$. We compute as follows:

$$H(\mathbf{X}) + H(\mathbf{Y}) = -\left(\sum_{i=1}^{m} p_i \log_2 p_i + \sum_{j=1}^{n} q_j \log_2 q_j \right)$$

$$= -\left(\sum_{i=1}^{m} \sum_{j=1}^{n} r_{ij} \log_2 p_i + \sum_{j=1}^{n} \sum_{i=1}^{m} r_{ij} \log_2 q_j \right)$$
$$= -\sum_{i=1}^{m} \sum_{j=1}^{n} r_{ij} \log_2 p_i q_j.$$

On the other hand,

$$H(\mathbf{X}, \mathbf{Y}) = -\sum_{i=1}^{m} \sum_{j=1}^{n} r_{ij} \log_2 r_{ij}.$$

Combining, we obtain the following:

$$\begin{aligned} H(\mathbf{X}, \mathbf{Y}) - H(\mathbf{X}) - H(\mathbf{Y}) &= \sum_{i=1}^{m} \sum_{j=1}^{n} r_{ij} \log_2 \frac{1}{r_{ij}} + \sum_{i=1}^{m} \sum_{j=1}^{n} r_{ij} \log_2 p_i q_j \\ &= \sum_{i=1}^{m} \sum_{j=1}^{n} r_{ij} \log_2 \frac{p_i q_j}{r_{ij}} \\ &\leq \log_2 \sum_{i=1}^{m} \sum_{j=1}^{n} p_i q_j \\ &= \log_2 1 \\ &= 0. \end{aligned}$$

(Here, we apply Jensen's Inequality, using the fact that the r_{ij}'s form a probability distribution.)

We can also say when equality occurs: it must be the case that there is a constant c such that $p_i q_j / r_{ij} = c$ for all i, j. Using the fact that

$$\sum_{j=1}^{n} \sum_{i=1}^{m} r_{ij} = \sum_{j=1}^{n} \sum_{i=1}^{m} p_i q_j = 1,$$

it follows that $c = 1$. Hence, equality occurs if and only if $r_{ij} = p_i q_j$, i.e., if and only if

$$p(\mathbf{X} = x_i, \mathbf{Y} = y_j) = p(\mathbf{X} = x_i) p(\mathbf{Y} = y_j),$$

$1 \leq i \leq m$, $1 \leq j \leq n$. But this says that $\mathbf{X}$ and $\mathbf{Y}$ are independent. ∎

We next define the idea of conditional entropy.

DEFINITION 2.5 *Suppose* $\mathbf{X}$ *and* $\mathbf{Y}$ *are two random variables. Then for any fixed value* y *of* $\mathbf{Y}$*, we get a (conditional) probability distribution* $p(\mathbf{X}|y)$*. Clearly,*

$$H(\mathbf{X}|y) = -\sum_x p(x|y) \log_2 p(x|y).$$

We define the ***conditional entropy*** $H(\mathbf{X}|\mathbf{Y})$ *to be the weighted average (with respect to the probabilities* $p(y)$*) of the entropies* $H(\mathbf{X}|y)$ *over all possible values* y*. It is computed to be*

$$H(\mathbf{X}|\mathbf{Y}) = -\sum_y \sum_x p(y)p(x|y) \log_2 p(x|y).$$

The conditional entropy measures the average amount of information about $\mathbf{X}$ that is revealed by $\mathbf{Y}$.

The next two results are straightforward; we leave the proofs as exercises.

THEOREM 2.8
$H(\mathbf{X}, \mathbf{Y}) = H(\mathbf{Y}) + H(\mathbf{X}|\mathbf{Y})$.

COROLLARY 2.9
$H(\mathbf{X}|\mathbf{Y}) \leq H(\mathbf{X})$*, with equality if and only if* $\mathbf{X}$ *and* $\mathbf{Y}$ *are independent.*

2.4 Spurious Keys and Unicity Distance

In this section, we apply the entropy results we have proved to cryptosystems. First, we show a fundamental relationship exists among the entropies of the components of a cryptosystem. The conditional entropy $H(\mathbf{K}|\mathbf{C})$ is called the *key equivocation*, and is a measure of how much information about the key is revealed by the ciphertext.

THEOREM 2.10
Let $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ *be a cryptosystem. Then*

$$H(\mathbf{K}|\mathbf{C}) = H(\mathbf{K}) + H(\mathbf{P}) - H(\mathbf{C}).$$

PROOF First, observe that $H(\mathbf{K}, \mathbf{P}, \mathbf{C}) = H(\mathbf{C}|\mathbf{K}, \mathbf{P}) + H(\mathbf{K}, \mathbf{P})$. Now, the key and plaintext determine the ciphertext uniquely, since $y = e_K(x)$. This implies that $H(\mathbf{C}|\mathbf{K}, \mathbf{P}) = 0$. Hence, $H(\mathbf{K}, \mathbf{P}, \mathbf{C}) = H(\mathbf{K}, \mathbf{P})$. But $\mathbf{K}$ and $\mathbf{P}$ are independent, so $H(\mathbf{K}, \mathbf{P}) = H(\mathbf{K}) + H(\mathbf{P})$. Hence,

$$H(\mathbf{K}, \mathbf{P}, \mathbf{C}) = H(\mathbf{K}, \mathbf{P}) = H(\mathbf{K}) + H(\mathbf{P}).$$

In a similar fashion, since the key and ciphertext determine the plaintext uniquely (i.e., $x = d_K(y)$), we have that $H(\mathbf{P}|\mathbf{K},\mathbf{C}) = 0$ and hence $H(\mathbf{K},\mathbf{P},\mathbf{C}) = H(\mathbf{K},\mathbf{C})$.

Now, we compute as follows:

$$\begin{aligned} H(\mathbf{K}|\mathbf{C}) &= H(\mathbf{K},\mathbf{C}) - H(\mathbf{C}) \\ &= H(\mathbf{K},\mathbf{P},\mathbf{C}) - H(\mathbf{C}) \\ &= H(\mathbf{K}) + H(\mathbf{P}) - H(\mathbf{C}), \end{aligned}$$

giving the desired formula. ∎

Let us return to Example 2.1 to illustrate this result.

Example 2.1 ***(Cont.)***
We have already computed $H(\mathbf{P}) \approx 0.81$, $H(\mathbf{K}) = 1.5$ and $H(\mathbf{C}) \approx 1.85$. Theorem 2.10 tells us that $H(\mathbf{K}|\mathbf{C}) \approx 1.5 + 0.81 - 1.85 \approx 0.46$. This can be verified directly by applying the definition of conditional entropy, as follows. First, we need to compute the probabilities $p(K_i|j)$, $1 \le i \le 3$, $1 \le j \le 4$. This can be done using Bayes' Theorem, and the following values result:

$$\begin{array}{lll} p(K_1|1) = 1 & p(K_2|1) = 0 & p(K_3|1) = 0 \\ p(K_1|2) = \frac{6}{7} & p(K_2|2) = \frac{1}{7} & p(K_3|2) = 0 \\ p(K_1|3) = 0 & p(K_2|3) = \frac{3}{4} & p(K_3|3) = \frac{1}{4} \\ p(K_1|4) = 0 & p(K_2|4) = 0 & p(K_3|4) = 1. \end{array}$$

Now we compute

$$H(\mathbf{K}|\mathbf{C}) = \frac{1}{8} \times 0 + \frac{7}{16} \times 0.59 + \frac{1}{4} \times 0.81 + \frac{3}{16} \times 0 = 0.46,$$

agreeing with the value predicted by Theorem 2.10. □

Suppose $(\mathcal{P},\mathcal{C},\mathcal{K},\mathcal{E},\mathcal{D})$ is the cryptosystem being used, and a string of plaintext $x_1x_2\ldots x_n$ is encrypted with one key, producing a string of ciphertext $y_1y_2\ldots y_n$. Recall that the basic goal of the cryptanalyst is to determine the key. We are looking at ciphertext-only attacks, and we assume that Oscar has infinite computational resources. We also assume that Oscar knows that the plaintext is a "natural" language, such as English. In general, Oscar will be able to rule out certain keys, but many "possible" keys may remain, only one of which is the correct key. The remaining possible, but incorrect, keys are called *spurious keys*.

For example, suppose Oscar obtains the ciphertext string $WNAJW$, which has been obtained by encryption using a shift cipher. It is easy to see that there are only two "meaningful" plaintext strings, namely *river* and *arena*, corresponding respectively to the possible encryption keys F $(= 5)$ and W $(= 22)$. Of these two keys, one will be the correct key and the other will be spurious. (Actually, it is moderately difficult to find a ciphertext of length 5 for the **Shift Cipher** that has two meaningful decryptions; the reader might search for other examples.)

Our goal is to prove a bound on the expected number of spurious keys. First, we have to define what we mean by the entropy (per letter) of a natural language L, which we denote H_L. H_L should be a measure of the average information per letter in a "meaningful" string of plaintext. (Note that a random string of alphabetic characters would have entropy (per letter) equal to $\log_2 26 \approx 4.76$.) As a "first-order" approximation to H_L, we could take $H(\mathbf{P})$. In the case where L is the English language, we get $H(\mathbf{P}) \approx 4.19$ by using the probability distribution given in Table 1.1.

Of course, successive letters in a language are not independent, and correlations among successive letters reduce the entropy. For example, in English, the letter "Q" is always followed by the letter "U." For a "second-order" approximation, we would compute the entropy of the probability distribution of all digrams and then divide by 2. In general, define $\mathbf{P}^n$ to be the random variable that has as its probability distribution that of all n-grams of plaintext. We make use of the following definitions.

DEFINITION 2.6 *Suppose L is a natural language. The **entropy** of L is defined to be the quantity*

$$H_L = \lim_{n \to \infty} \frac{H(\mathbf{P}^n)}{n}$$

*and the **redundancy** of L is defined to be*

$$R_L = 1 - \frac{H_L}{\log_2 |\mathcal{P}|}.$$

REMARK H_L measures the entropy per letter of the language L. A random language would have entropy $\log_2 |\mathcal{P}|$. So the quantity R_L measures the fraction of "excess characters," which we think of as redundancy. ▮

In the case of the English language, a tabulation of a large number of digrams and their frequencies would produce an estimate for $H(\mathbf{P}^2)$. $H(\mathbf{P}^2) \approx 3.90$ is one estimate obtained in this way. One could continue, tabulating trigrams, etc. and thus obtain an estimate for H_L. In fact, various experiments have yielded the empirical result that $1.0 \leq H_L \leq 1.5$. That is, the average information content in English is something like one to one and a half bits per letter!

Using 1.25 as our estimate of H_L gives a redundancy of about 0.75. This means that the English language is 75% redundant! (This is not to say that one can arbitrarily remove three out of every four letters from English text and hope to still be able to read it. What is *does* mean is that it is possible to find a Huffman encoding of n-grams, for a large enough value of n, which will compress English text to about one quarter of its original length.)

Given probability distributions on $\mathcal{K}$ and $\mathcal{P}^n$, we can define the induced probability distribution on $\mathcal{C}^n$, the set of n-grams of ciphertext (we already did this in the case $n = 1$). We have defined $\mathbf{P}^n$ to be a random variable representing an n-gram of plaintext. Similarly, define $\mathbf{C}^n$ to be a random variable representing an n-gram of ciphertext.

Given $\mathbf{y} \in \mathbf{C}^n$, define

$$K(\mathbf{y}) = \{K \in \mathcal{K} : \exists \mathbf{x} \in \mathcal{P}^n, p_{\mathcal{P}^n}(\mathbf{x}) > 0, e_K(\mathbf{x}) = \mathbf{y}\}.$$

That is, $K(\mathbf{y})$ is the set of keys K for which $\mathbf{y}$ is the encryption of a meaningful string of plaintext of length n, i.e., the set of "possible" keys, given that $\mathbf{y}$ is the ciphertext. If $\mathbf{y}$ is the observed sequence of ciphertext, then the number of spurious keys is $|K(\mathbf{y})| - 1$, since only one of the "possible" keys is the correct key. The average number of spurious keys (over all possible ciphertext strings of length n) is denoted by $\overline{s}_n$. Its value is computed to be

$$\begin{aligned}
\overline{s}_n &= \sum_{\mathbf{y} \in \mathcal{C}^n} p(\mathbf{y})(|K(\mathbf{y})| - 1) \\
&= \sum_{\mathbf{y} \in \mathcal{C}^n} p(\mathbf{y})|K(\mathbf{y})| - \sum_{\mathbf{y} \in \mathcal{C}^n} p(\mathbf{y}) \\
&= \sum_{\mathbf{y} \in \mathcal{C}^n} p(\mathbf{y})|K(\mathbf{y})| - 1.
\end{aligned}$$

From Theorem 2.10, we have that

$$H(\mathbf{K}|\mathbf{C}^n) = H(\mathbf{K}) + H(\mathbf{P}^n) - H(\mathbf{C}^n).$$

Also, we can use the estimate

$$H(\mathbf{P}^n) \approx nH_L = n(1 - R_L)\log_2 |\mathcal{P}|,$$

provided n is reasonably large. Certainly,

$$H(\mathbf{C}^n) \leq n \log_2 |\mathcal{C}|.$$

Then, if $|\mathcal{C}| = |\mathcal{P}|$, it follows that

$$H(\mathbf{K}|\mathbf{C}^n) \geq H(\mathbf{K}) - nR_L \log_2 |\mathcal{P}|. \tag{2.1}$$

Next, we relate the quantity $H(\mathbf{K}|\mathbf{C}^n)$ to the number of spurious keys, $\overline{s}_n$. We compute as follows:

$$\begin{aligned} H(\mathbf{K}|\mathbf{C}^n) &= \sum_{\mathbf{y}\in\mathcal{C}^n} p(\mathbf{y})H(\mathbf{K}|\mathbf{y}) \\ &\leq \sum_{\mathbf{y}\in\mathcal{C}^n} p(\mathbf{y}) \log_2 |K(\mathbf{y})| \\ &\leq \log_2 \sum_{\mathbf{y}\in\mathcal{C}^n} p(\mathbf{y})|K(\mathbf{y})| \\ &= \log_2(\overline{s}_n + 1), \end{aligned}$$

where we apply Jensen's Inequality (Theorem 2.5) with $f(x) = \log_2 x$. Thus we obtain the inequality

$$H(\mathbf{K}|\mathbf{C}^n) \leq \log_2(\overline{s}_n + 1). \tag{2.2}$$

Combining the two inequalities (2.1) and (2.2), we get that

$$\log_2(\overline{s}_n + 1) \geq H(\mathbf{K}) - nR_L \log_2 |\mathcal{P}|.$$

In the case where keys are chosen equiprobably (which maximizes $H(\mathbf{K})$), we have the following result.

THEOREM 2.11

Suppose $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ is a cryptosystem where $|\mathcal{C}| = |\mathcal{P}|$ and keys are chosen equiprobably. Let R_L denote the redundancy of the underlying language. Then given a string of ciphertext of length n, where n is sufficiently large, the expected number of spurious keys, $\overline{s}_n$, satisfies

$$\overline{s}_n \geq \frac{|\mathcal{K}|}{|\mathcal{P}|^{nR_L}} - 1.$$

The quantity $|\mathcal{K}|/|\mathcal{P}|^{nR_L} - 1$ approaches 0 exponentially quickly as n increases. Also, note that the estimate may not be accurate for small values of n, especially since $H(\mathbf{P}^n)/n$ may not be a good estimate for H_L if n is small.

We have one more concept to define.

DEFINITION 2.7 *The* ***unicity distance*** *of a cryptosystem is defined to be the value of n, denoted by n_0, at which the expected number of spurious keys becomes zero; i.e., the average amount of ciphertext required for an opponent to be able to uniquely compute the key, given enough computing time.*

If we set $\overline{s}_n = 0$ in Theorem 2.11 and solve for n, we get an estimate for the unicity distance, namely

$$n_0 \approx \frac{\log_2 |\mathcal{K}|}{R_L \log_2 |\mathcal{P}|}.$$

As an example, consider the **Substitution Cipher**. In this cryptosystem, $|\mathcal{P}| = 26$ and $|\mathcal{K}| = 26!$. If we take $R_L = 0.75$, then we get an estimate for the unicity distance of

$$n_0 \approx 88.4/(0.75 \times 4.7) \approx 25.$$

This suggests that, given a ciphertext string of length at least 25, (usually) a unique decryption is possible.

2.5 Product Cryptosystems

Another innovation introduced by Shannon in his 1949 paper was the idea of combining cryptosystems by forming their "product." This idea has been of fundamental importance in the design of present-day cryptosystems such as the **Data Encryption Standard**, which we study in the next chapter.

For simplicity, we will confine our attention in this section to cryptosystems in which $\mathcal{C} = \mathcal{P}$: cryptosystems of this type are called *endomorphic*. Suppose $\mathbf{S}_1 = (\mathcal{P}, \mathcal{P}, \mathcal{K}_1, \mathcal{E}_1, \mathcal{D}_1)$ and $\mathbf{S}_2 = (\mathcal{P}, \mathcal{P}, \mathcal{K}_2, \mathcal{E}_2, \mathcal{D}_2)$ are two endomorphic cryptosystems which have the same plaintext (and ciphertext) spaces. Then the *product* . of $\mathbf{S}_1$ and $\mathbf{S}_2$, denoted by $\mathbf{S}_1 \times \mathbf{S}_2$, is defined to be the cryptosystem

$$(\mathcal{P}, \mathcal{P}, \mathcal{K}_1 \times \mathcal{K}_2, \mathcal{E}, \mathcal{D}).$$

A key of the product cryptosystem has the form $K = (K_1, K_2)$, where $K_1 \in \mathcal{K}_1$ and $K_2 \in \mathcal{K}_2$. The encryption and decryption rules of the product cryptosystem are defined as follows: For each $K = (K_1, K_2)$, we have an encryption rule E_K defined by the formula

$$e_{(K_1,K_2)}(x) = e_{K_2}(e_{K_1}(x)),$$

and a decryption rule defined by the formula

$$d_{(K_1,K_2)}(y) = d_{K_1}(d_{K_2}(y)).$$

That is, we first encrypt x with e_{K_1}, and then "re-encrypt" the resulting ciphertext with e_{K_2}. Decrypting is similar, but it must be done in the reverse order:

$$\begin{aligned} d_{(K_1,K_2)}(e_{(K_1,K_2)}(x)) &= d_{(K_1,K_2)}(e_{K_2}(e_{K_1}(x))) \\ &= d_{K_1}(d_{K_2}(e_{K_2}(e_{K_1}(x)))) \end{aligned}$$

FIGURE 2.2
Multiplicative Cipher

Let $\mathcal{P} = \mathcal{C} = \mathbb{Z}_{26}$ and let

$$\mathcal{K} = \{a \in \mathbb{Z}_{26} : \gcd(a, 26) = 1\}.$$

For $a \in \mathcal{K}$, define

$$e_a(x) = ax \bmod 26$$

and

$$d_a(y) = a^{-1}y \bmod 26$$

$(x, y \in \mathbb{Z}_{26})$.

$$= d_{K_1}(e_{K_1}(x))$$
$$= x.$$

Recall also that cryptosystems have probability distributions associated with their keyspaces. Thus we need to define the probability distribution for the keyspace $\mathcal{K}$ of the product cryptosystem. We do this in a very natural way:

$$p_{\mathcal{K}}(K_1, K_2) = p_{\mathcal{K}_1}(K_1) \times p_{\mathcal{K}_2}(K_2).$$

In other words, choose K_1 using the distribution $p_{\mathcal{K}_1}$, and then independently choose K_2 using the distribution $p_{\mathcal{K}_2}$.

Here is a simple example to illustrate the definition of a product cryptosystem. Suppose we define the **Multiplicative Cipher** as in Figure 2.2.

Suppose $\mathbf{M}$ is the **Multiplicative Cipher** (with keys chosen equiprobably) and $\mathbf{S}$ is the **Shift Cipher** (with keys chosen equiprobably). Then it is very easy to see that $\mathbf{M} \times \mathbf{S}$ is nothing more than the **Affine Cipher** (again, with keys chosen equiprobably). It is slightly more difficult to show that $\mathbf{S} \times \mathbf{M}$ is also the **Affine Cipher** with equiprobable keys.

Let's prove these assertions. A key in the **Shift Cipher** is an element $K \in \mathbb{Z}_{26}$, and the corresponding encryption rule is $e_K(x) = x + K \bmod 26$. A key in the **Multiplicative Cipher** is an element $a \in \mathbb{Z}_{26}$ such that $\gcd(a, 26) = 1$; the corresponding encryption rule is $e_a(x) = ax \bmod 26$. Hence, a key in the product cipher $\mathbf{M} \times \mathbf{S}$ has the form (a, K), where

$$e_{(a,K)}(x) = ax + K \bmod 26.$$

But this is precisely the definition of a key in the **Affine Cipher**. Further, the probability of a key in the **Affine Cipher** is $1/312 = 1/12 \times 1/26$, which is the

product of the probabilities of the keys a and K, respectively. Thus $\mathbf{M} \times \mathbf{S}$ is the **Affine Cipher**.

Now let's consider $\mathbf{S} \times \mathbf{M}$. A key in this cipher has the form (K, a), where

$$e_{(K,a)}(x) = a(x + K) = ax + aK \bmod 26.$$

Thus the key (K, a) of the product cipher $\mathbf{S} \times \mathbf{M}$ is identical to the key (a, aK) of the **Affine Cipher**. It remains to show that each key of the **Affine Cipher** arises with the same probability $1/312$ in the product cipher $\mathbf{S} \times \mathbf{M}$. Observe that $aK = K_1$ if and only if $K = a^{-1}K_1$ (recall that $\gcd(a, 26) = 1$, so a has a multiplicative inverse). In other words, the key (a, K_1) of the **Affine Cipher** is equivalent to the key $(a^{-1}K_1, a)$ of the product cipher $\mathbf{S} \times \mathbf{M}$. We thus have a bijection between the two key spaces. Since each key is equiprobable, we conclude that $\mathbf{S} \times \mathbf{M}$ is indeed the **Affine Cipher**.

We have shown that $\mathbf{M} \times \mathbf{S} = \mathbf{S} \times \mathbf{M}$. Thus we would say that the two cryptosystems *commute*. But not all pairs of cryptosystems commute; it is easy to find counterexamples. On the other hand, the product operation is always *associative*: $(\mathbf{S}_1 \times \mathbf{S}_2) \times \mathbf{S}_3 = \mathbf{S}_1 \times (\mathbf{S}_2 \times \mathbf{S}_3)$.

If we take the product of an (endomorphic) cryptosystem $\mathbf{S}$ with itself, we obtain the cryptosystem $\mathbf{S} \times \mathbf{S}$, which we denote by $\mathbf{S}^2$. If we take the n-fold product, the resulting cryptosystem is denoted by $\mathbf{S}^n$. We call $\mathbf{S}^n$ an *iterated* cryptosystem.

A cryptosystem $\mathbf{S}$ is defined to be *idempotent* if $\mathbf{S}^2 = \mathbf{S}$. Many of the cryptosystems we studied in Chapter 1 are idempotent. For example, the **Shift, Substitution, Affine, Hill, Vigenère** and **Permutation Ciphers** are all idempotent. Of course, if a cryptosystem $\mathbf{S}$ is idempotent, then there is no point in using the product system $\mathbf{S}^2$, as it requires an extra key but provides no more security.

If a cryptosystem is not idempotent, then there is a potential increase in security by iterating several times. This idea is used in the **Data Encryption Standard**, which consists of 16 iterations. But, of course, this approach requires a non-idempotent cryptosystem to start with. One way in which simple non-idempotent cryptosystems can sometimes be constructed is to take the product of two different (simple) cryptosystems.

REMARK It is not hard to show that if $\mathbf{S}_1$ and $\mathbf{S}_2$ are both idempotent and they commute, then $\mathbf{S}_1 \times \mathbf{S}_2$ will also be idempotent. This follows from the following algebraic manipulations:

$$\begin{aligned}(\mathbf{S}_1 \times \mathbf{S}_2) \times (\mathbf{S}_1 \times \mathbf{S}_2) &= \mathbf{S}_1 \times (\mathbf{S}_2 \times \mathbf{S}_1) \times \mathbf{S}_2 \\ &= \mathbf{S}_1 \times (\mathbf{S}_1 \times \mathbf{S}_2) \times \mathbf{S}_2 \\ &= (\mathbf{S}_1 \times \mathbf{S}_1) \times (\mathbf{S}_2 \times \mathbf{S}_2) \\ &= \mathbf{S}_1 \times \mathbf{S}_2.\end{aligned}$$

(Note the use of the associative property in this proof.)

So, if $\mathbf{S}_1$ and $\mathbf{S}_2$ are both idempotent, and we want $\mathbf{S}_1 \times \mathbf{S}_2$ to be non-idempotent, then it is necessary that $\mathbf{S}_1$ and $\mathbf{S}_2$ not commute. ∎

Fortunately, many simple cryptosystems are suitable building blocks in this type of approach. Taking the product of substitution-type ciphers with permutation-type ciphers is a commonly used technique. We will see a realization of this in the next chapter.

2.6 Notes

The idea of perfect secrecy and the use of entropy techniques in cryptography was pioneered by Shannon [SH49]. Product cryptosystems are also discussed in this paper. The concept of entropy was defined by Shannon in [SH48]. Good introductions to entropy, Huffman coding and related topics can be found in the books by Welsh [WE88] and Goldie and Pinch [GP91].

The results of Section 2.4 are due to Beauchemin and Brassard [BB88], who generalized earlier results of Shannon.

Exercises

2.1 Let n be a positive integer. A *Latin square* of order n is an $n \times n$ array L of the integers $1, \ldots, n$ such that every one of the n integers occurs exactly once in each row and each column of L. An example of a Latin square of order 3 is as follows:

1	2	3
3	1	2
2	3	1

Given any Latin square L of order n, we can define a related cryptosystem. Take $\mathcal{P} = \mathcal{C} = \mathcal{K} = \{1, \ldots, n\}$. For $1 \leq i \leq n$, the encryption rule e_i is defined to be $e_i(j) = L(i, j)$. (Hence each row of L gives rise to one encryption rule.)

Give a complete proof that this Latin square cryptosystem achieves perfect secrecy.

2.2 Prove that the **Affine Cipher** achieves perfect secrecy.

2.3 Suppose a cryptosystem achieves perfect secrecy for a particular plaintext probability distribution p_0. Prove that perfect secrecy is maintained for *any* plaintext probability distribution.

2.4 Prove that if a cryptosystem has perfect secrecy and $|\mathcal{K}| = |\mathcal{C}| = |\mathcal{P}|$, then every ciphertext is equally probable.

2.5 Suppose $\mathbf{X}$ is a set of cardinality n, where $2^k \leq n < 2^{k+1}$, and $p(x) = 1/n$ for all $x \in \mathbf{X}$.

(a) Find a prefix-free encoding of $\mathbf{X}$, say f, such that $\ell(f) = k + 2 - 2^{k+1}/n$.

HINT Encode $2^{k+1} - n$ elements of $\mathbf{X}$ as strings of length k, and encode the remaining elements as strings of length $k+1$.

(b) Illustrate your construction for $n = 6$. Compute $\ell(f)$ and $H(\mathbf{X})$ in this case.

2.6 Suppose $\mathbf{X} = \{a, b, c, d, e\}$ has the following probability distribution: $p(a) = .32$, $p(b) = .23$, $p(c) = .20$, $p(d) = .15$ and $p(e) = .10$. Use Huffman's algorithm to find the optimal prefix-free encoding of $\mathbf{X}$. Compare the length of this encoding to $H(\mathbf{X})$.

2.7 Prove that $H(\mathbf{X}, \mathbf{Y}) = H(\mathbf{Y}) + H(\mathbf{X}|\mathbf{Y})$. Then show as a corollary that $H(\mathbf{X}|\mathbf{Y}) \leq H(\mathbf{X})$, with equality if and only if $\mathbf{X}$ and $\mathbf{Y}$ are independent.

2.8 Prove that a cryptosystem has perfect secrecy if and only if $H(\mathbf{P}|\mathbf{C}) = H(\mathbf{P})$.

2.9 Prove that, in any cryptosystem, $H(\mathbf{K}|\mathbf{C}) \geq H(\mathbf{P}|\mathbf{C})$. (Intuitively, this result says that, given a ciphertext, the opponent's uncertainty about the key is at least as great as his uncertainty about the plaintext.)

2.10 Consider a cryptosystem in which $\mathcal{P} = \{a, b, c\}$, $\mathcal{K} = \{K_1, K_2, K_3\}$ and $\mathcal{C} = \{1, 2, 3, 4\}$. Suppose the encryption matrix is as follows:

	a	b	c
K_1	1	2	3
K_2	2	3	4
K_3	3	4	1

Given that keys are chosen equiprobably, and the plaintext probability distribution is $p_{\mathcal{P}}(a) = 1/2, p_{\mathcal{P}}(b) = 1/3, p_{\mathcal{P}}(c) = 1/6$, compute $H(\mathbf{P}), H(\mathbf{C}), H(\mathbf{K}), H(\mathbf{K}|\mathbf{C})$ and $H(\mathbf{P}|\mathbf{C})$.

2.11 Compute $H(\mathbf{K}|\mathbf{C})$ and $H(\mathbf{K}|\mathbf{P}, \mathbf{C})$ for the **Affine Cipher**.

2.12 Consider a **Vigenère Cipher** with keyword length m. Show that the unicity distance is $1/R_L$, where R_L is the redundancy of the underlying language. (This result is interpreted as follows. If n_0 denotes the number of alphabetic characters being encrypted , then the "length" of the plaintext is n_0/m, since each plaintext element consists of m alphabetic characters. So, a unicity distance of $1/R_L$ corresponds to a plaintext consisting of m/R_L alphabetic characters.)

2.13 Show that the unicity distance of the **Hill Cipher** (with an $m \times m$ encryption matrix) is less than m/R_L (Note that the number of alphabetic characters in a plaintext of this length is m^2/R_L.)

2.14 A **Substitution Cipher** over a plaintext space of size n has $|\mathcal{K}| = n!$ Stirling's formula gives the following estimate for $n!$:

$$n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n.$$

(a) Using Stirling's formula, derive an estimate of the unicity distance of the **Substitution Cipher**.

(b) Let $m \geq 1$ be an integer. The **m-gram Substitution Cipher** is the **Substitution Cipher** where the plaintext (and ciphertext) spaces consist of all 26^m m-grams. Estimate the unicity distance of the **m-gram Substitution Cipher** if $R_L = 0.75$.

2.15 Prove that the **Shift Cipher** is idempotent.

2.16 Suppose $\mathbf{S}_1$ is the **Shift Cipher** (with equiprobable keys, as usual) and $\mathbf{S}_2$ is the **Shift Cipher** where keys are chosen with respect to some probability distribution $p_{\mathcal{K}}$ (which need not be equiprobable). Prove that $\mathbf{S}_1 \times \mathbf{S}_2 = \mathbf{S}_1$.

2.17 Suppose $\mathbf{S}_1$ and $\mathbf{S}_2$ are **Vigenère Ciphers** with keyword lengths m_1, m_2 respectively, where $m_1 > m_2$.

(a) If $m_2 \mid m_1$, then show that $\mathbf{S}_2 \times \mathbf{S}_1 = \mathbf{S}_1$.

(b) One might try to generalize the previous result by conjecturing that $\mathbf{S}_2 \times \mathbf{S}_1 = \mathbf{S}_3$, where $\mathbf{S}_3$ is the **Vigenère Cipher** with keyword length $\text{lcm}(m_1, m_2)$. Prove that this conjecture is false.

HINT If $m_1 \not\equiv 0 \bmod m_2$, then the number of keys in the product cryptosystem $\mathbf{S}_2 \times \mathbf{S}_1$ is less than the number of keys in $\mathbf{S}_3$.