1
Source Coding

Words, words, words. (*Hamlet*)

This chapter considers how the information emanating from a source can be encoded, so that it can later be decoded unambiguously and without delay. These two requirements lead to the concepts of uniquely decodable and instantaneous codes. We shall find necessary and sufficient conditions for a code to have these properties, we shall see how to construct such codes, and we shall prove Kraft's and McMillan's inequalities, which essentially say that such codes exist if and only if they have sufficiently long code-words.

1.1 Definitions and Examples

Information theory is concerned with the transmission of information from a sender, through a channel, to a receiver. The sender and receiver could be people or machines. In most cases they are different, but when information is being stored for later retrieval, the receiver could be the sender at some future time. We will assume that the information comes from a *source* $\mathcal{S}$, which emits a sequence $\mathbf{s} = X_1X_2X_3\ldots$ of symbols X_n; for instance, X_n might be the n-th symbol in some message, or the outcome of the n-th repetition of some experiment. In practice, this sequence will always be finite (nothing lasts for ever), but for theoretical purposes it is sometimes useful also to consider infinite

sequences. We will assume that each symbol X_n is a member of some fixed finite set $S = \{s_1, \ldots, s_q\}$, called the *source alphabet* of $\mathcal{S}$. For simplicity we will also assume that the probability $\Pr(X_n = s_i)$, that the n-th symbol X_n is s_i, depends only on i but not on n, so we write

$$\Pr(X_n = s_i) = p_i$$

for $i = 1, \ldots, q$. Thus different symbols may have different probabilities, but these remain constant in time (so $\mathcal{S}$ is *stationary*), and do not depend on the preceding symbols X_m where $m < n$ (so $\mathcal{S}$ is *memoryless*). In more advanced forms of this theory, such factors are taken into consideration, but we will ignore them here. As with any probability distribution, the probabilities p_i must satisfy

$$p_i \geq 0 \quad \text{and} \quad \sum_{i=1}^{q} p_i = 1 \tag{1.1}$$

(so each $p_i \leq 1$). In statistical terms, one can regard $\mathcal{S}$ as a sequence of independent, identically distributed random variables X_n, with probability distribution (p_i).

Example 1.1

$\mathcal{S}$ is an unbiased die[1], $S = \{1, \ldots, 6\}$ with $q = 6$, $s_i = i$ for $i = 1, \ldots, 6$, X_n is the outcome of the n-th throw, and $p_i = 1/6$ for $i = 1, \ldots, 6$. A biased die is similar, but with different probabilities p_i.

Example 1.2

$\mathcal{S}$ is the weather at a particular place, with X_n representing the weather on day n. For simplicity, we could let S consist of $q = 3$ types of weather (good, moderate and bad, for instance), so p_i $(i = 1, 2, 3)$ is the probability of each type, say $p_1 = 1/4$, $p_2 = 1/2$, $p_3 = 1/4$. (Here we ignore seasonal variations, which may cause the probability distribution (p_i) to vary in time.)

Example 1.3

$\mathcal{S}$ is a book, S consists of all the symbols used (letters, punctuation marks, numerals, etc.), X_n is the n-th symbol in the book, and p_i is the frequency of the i-th symbol in the source alphabet. (Here we ignore the effect of preceding symbols on probabilities: for instance, in English text the symbol "q" is almost always followed by "u".)

To encode a source, we use a finite *code alphabet* $T = \{t_1, \ldots, t_r\}$ consisting of r *code-symbols* t_j. In general, this is distinct from the source alphabet $S =$

[1] The singular of dice, as in "the die is cast".

$\{s_1, \ldots, s_q\}$, since it depends on the technology of the channel rather than the source. We call r the *radix* (meaning "root"), and we refer to the code as an r-ary code. In many examples, $r = 2$ and the code is called *binary*. Most binary codes, such as ASCII (used in computing), have $T = \mathbf{Z}_2 = \{0, 1\}$, the set of integers mod (2). Codes with $r = 3$ are called *ternary*. We encode $\mathcal{S}$ by assigning a code-word w_i (a finite sequence of code-symbols) to each symbol $s_i \in S$; to encode $\mathbf{s} = X_1 X_2 X_3 \ldots$ we represent each $X_n = s_i$ by its code-word w_i, giving a sequence $\mathbf{t}$ of symbols from T. For conciseness, we do not separate the code-words in $\mathbf{t}$ with punctuation marks or blanks; if these are used, they must be regarded as elements of T appearing at the beginning or end of each w_i. Thus Morse, which appears to be binary, is actually a ternary code: the three symbols are •, — and a blank.

Example 1.4

If $\mathcal{S}$ is an unbiased die, as in Example 1.1, take $T = \mathbf{Z}_2$ and let w_i be the binary representation of the source-symbol $s_i = i$ $(i = 1, \ldots, 6)$. Thus $w_1 = 1$, $w_2 = 10, \ldots, w_6 = 110$, so a sequence of throws such as $\mathbf{s} = 53214$ is encoded as $\mathbf{t} = 10111101100$.

For clearer exposition, we will occasionally break our rule not to use punctuation, and insert full stops or brackets to show how $\mathbf{t}$ is decomposed into code-words: in Example 1.4 we could write $\mathbf{t} = 101.11.10.1.100$, for instance. This is purely for the reader's benefit, and the punctuation symbols should not be regarded as part of $\mathbf{t}$.

We need to define codes more precisely. A *word* w in T is a finite sequence of symbols of T, its *length* $|w|$ is the number of symbols. The set of all words in T is denoted by T^*; this includes the *empty word*, of length 0, which we will denote by ε. The set of all non-empty words in T is denoted by T^+. Thus

$$T^* = \bigcup_{n \geq 0} T^n \qquad \text{and} \qquad T^+ = \bigcup_{n > 0} T^n,$$

where $T^n = T \times \cdots \times T$ (with n factors) is the set of words of length n. A *source code* (or simply a *code*) $\mathcal{C}$ is a function $S \to T^+$, that is, an assignment of code-words $w_i = \mathcal{C}(s_i) \in T^+$ to the symbols $s_i \in S$. Many properties of codes depend only on the code-words w_i, and not on the particular correspondence between them and the symbols s_i, so we will often regard $\mathcal{C}$ as simply a finite set of words $w_1, \ldots, w_q$ in T^+. If S^* is defined by analogy with T^*, then one can extend $\mathcal{C}$ to a function $S^* \to T^*$ in the obvious way, encoding each $\mathbf{s}$ in S^* by using $\mathcal{C}$ to encode its successive symbols:

$$\mathbf{s} = s_{i_1} s_{i_2} \ldots s_{i_n} \mapsto \mathbf{t} = w_{i_1} w_{i_2} \ldots w_{i_n} \in T^*.$$

The image of this function is the set

$$C^* = \{w_{i_1} w_{i_2} \dots w_{i_n} \in T^* \mid \text{each } w_{i_j} \in C,\ n \geq 0\}.$$

We denote the length $|w_i|$ of w_i by l_i, so each $l_i \geq 1$. The *average word-length* of C is

$$L(C) = \sum_{i=1}^{q} p_i l_i \,. \tag{1.2}$$

Example 1.5

The code C in Example 1.4 has $l_1 = 1$, $l_2 = l_3 = 2$ and $l_4 = l_5 = l_6 = 3$, so

$$L(C) = \frac{1}{6}(1 + 2 + 2 + 3 + 3 + 3) = \frac{7}{3}.$$

The aim is to construct codes C for which

(a) there is easy and unambiguous decoding $\mathbf{t} \mapsto \mathbf{s}$,

(b) the average word-length $L(C)$ is small.

The rest of this chapter considers criterion (a), and the next chapter considers (b).

1.2 Uniquely Decodable Codes

A code C is *uniquely decodable* (= *uniquely decipherable*, or *u.d.* for short) if each $\mathbf{t} \in T^*$ corresponds under C to at most one $\mathbf{s} \in S^*$; in other words, the function $C : S^* \to T^*$ is one-to-one, so each $\mathbf{t}$ in its image C^* can be decoded uniquely. We will always assume that the code-words w_i in C are distinct, for if $w_i = w_j$ with $i \neq j$ then $\mathbf{t} = w_i$ could represent either s_i or s_j, so C is not uniquely decodable. Under this assumption, the definition of unique decodability of C is that whenever

$$u_1 \dots u_m = v_1 \dots v_n$$

with $u_1, \dots, u_m, v_1, \dots, v_n \in C$, we have $m = n$ and $u_i = v_i$ for each i. In algebraic terms we are saying that each code-sequence $\mathbf{t} \in C^*$ can be factorised in a unique way as a product of code-words.

Example 1.6

In Example 1.4, the binary coding of a die is not uniquely decodable: for instance, $\mathbf{t} = 11$ could be decomposed into code-words as 1.1 or 11, representing $\mathbf{s} = 11$ or $\mathbf{s} = 3$ (two throws of 1, or one throw of 3). We could remedy this by using a different code, with 3-digit binary representations of the source-symbols:

$$1 \mapsto 001,\ 2 \mapsto 010,\ \ldots\ ,\ 6 \mapsto 110.$$

Then $\mathbf{s} = 11 \mapsto \mathbf{t} = 001001$ whereas $\mathbf{s} = 3 \mapsto \mathbf{t} = 011$. More generally, we have:

Theorem 1.7

If the code-words w_i in $\mathcal{C}$ all have the same length, then $\mathcal{C}$ is uniquely decodable.

Proof

Let l be the common word-length. If some $\mathbf{t} \in \mathcal{C}^*$ factorises as $u_1 \ldots u_m = v_1 \ldots v_n$ with each $u_i, v_j \in \mathcal{C}$, then $lm = |\mathbf{t}| = ln$, so $m = n$. Now u_1 and v_1 both consist of the first l symbols of $\mathbf{t}$, so $u_1 = v_1$, and similarly $u_i = v_i$ for all i. □

If all the code-words in $\mathcal{C}$ have the same length l, we call $\mathcal{C}$ a *block code* of length l. We will study such codes in detail in Chapters 5–7. The converse of Theorem 1.7 is false:

Example 1.8

The binary code $\mathcal{C}$ given by

$$s_1 \mapsto w_1 = 0,\ s_2 \mapsto w_2 = 01,\ s_3 \mapsto w_3 = 011$$

has variable lengths, but is still uniquely decodable. Within $\mathbf{t}$, each symbol 0 indicates the start of a code-word w_i, and i is 1 plus the number of subsequent 1s. For instance,

$$\mathbf{t} = 001011010011 = 0.01.011.01.0.011 \quad \Rightarrow \quad \mathbf{s} = s_1 s_2 s_3 s_2 s_1 s_3.$$

In effect, we are using the symbol $0 \in T$ here as a punctuation mark.

We are going to state a necessary and sufficient condition for a code $\mathcal{C}$ to be uniquely decodable. We use induction to define a sequence $\mathcal{C}_0, \mathcal{C}_1, \ldots$ of sets of non-empty words, so $\mathcal{C}_n \subseteq T^+$ for all n. Specifically, we define $\mathcal{C}_0 = \mathcal{C}$, and

$$\mathcal{C}_n = \{\, w \in T^+ \mid uw = v \text{ where } u \in \mathcal{C},\ v \in \mathcal{C}_{n-1} \text{ or } u \in \mathcal{C}_{n-1},\ v \in \mathcal{C} \,\} \qquad (1.3)$$

for each $n \geq 1$; we then define

$$C_\infty = \bigcup_{n=1}^{\infty} C_n. \tag{1.4}$$

This definition may look a little daunting at first, but it should become clearer if we take it step by step: we start with $C_0 = C$, we then construct each C_n ($n \geq 1$) in terms of its predecessor C_{n-1}, and finally we take $C_\infty = C_1 \cup C_2 \cup \cdots$. Note that for $n = 1$ the definition of C_n can be simplified: since $C_{n-1} = C_0 = C$ the two conditions separated by the word "or" in (1.3) are identical, so

$$C_1 = \{\, w \in T^+ \mid uw = v \text{ where } u, v \in C \,\}.$$

Note also that if $C_{n-1} = \emptyset$ then $C_n = \emptyset$, so iterating this gives $C_{n+1} = C_{n+2} = \cdots = \emptyset$.

Example 1.9

Let $C = \{0, 01, 011\}$ as in Example 1.9. Then (1.3) gives $C_1 = \{1, 11\}$: we have $1 \in C_1$ since $01.1 = 011$ with $01, 011 \in C = C_0$, and $11 \in C_1$ since $0.11 = 011$ with $0, 011 \in C = C_0$. At the next stage, with $n = 2$, inspection shows that there is no $w \in T^+$ satisfying $uw = v$ where $u \in C$, $v \in C_1$ or vice versa. Thus $C_2 = \emptyset$, so $C_n = \emptyset$ for all $n \geq 2$ and hence $C_\infty = C_1 = \{1, 11\}$ by (1.4).

From the definition of C_∞, it is conceivable that the construction of this set might take infinitely many steps, requiring a new set C_n to be constructed for each $n \geq 1$. Exercise 1.1 shows that one can always construct C_∞ in finitely many steps, as in Example 1.9.

Exercise 1.1

Prove that if C has code-words of lengths $l_1, \ldots, l_q$, and $w \in C_n$ for some n, then $|w| \leq l = \max(l_1, \ldots, l_q)$. Deduce that each C_n is finite, and the sequence of sets $C_0, C_1, \ldots$ is eventually periodic. How does this help in the construction of C_∞ ?

Exercise 1.2

Construct the sets C_n and C_∞ for the ternary code $C = \{02, 12, 120, 20, 21\}$. Do the same for $C = \{02, 12, 120, 21\}$.

We can now give a necessary and sufficient condition for unique decodability. The *Sardinas–Patterson Theorem* [SP53] is as follows.

Theorem 1.10

A code $\mathcal{C}$ is uniquely decodable if and only if the sets $\mathcal{C}$ and $\mathcal{C}_\infty$ are disjoint.

Before considering a proof, let us apply this result in some simple cases.

Example 1.11

If $\mathcal{C} = \{0, 01, 011\}$ as in Examples 1.8 and 1.9, then $\mathcal{C}_\infty = \{1, 11\}$ which is disjoint from $\mathcal{C}$. It follows from Theorem 1.10 that $\mathcal{C}$ is uniquely decodable, as we have already seen.

Example 1.12

Let $\mathcal{C}$ be the ternary code $\{01, 1, 2, 210\}$. Using (1.3) we find that $\mathcal{C}_1 = \{10\}$, $\mathcal{C}_2 = \{0\}$ and $\mathcal{C}_3 = \{1\}$, so $1 \in \mathcal{C} \cap \mathcal{C}_\infty$ and thus $\mathcal{C}$ is not uniquely decodable (there is no need to calculate $\mathcal{C}_n$ for $n > 3$). To find an example of non-unique decodability, note that $10 \in \mathcal{C}_1$ since $2 \in \mathcal{C}$ and $2.10 = 210 \in \mathcal{C}$, then $0 \in \mathcal{C}_2$ since $1 \in \mathcal{C}$ and $1.0 = 10 \in \mathcal{C}_1$, and then $1 \in \mathcal{C}_3$ since $0 \in \mathcal{C}_2$ and $0.1 = 01 \in \mathcal{C}$. Putting these equations together we get

$$210.1 = 2.10.1 = 2.1.0.1 = 2.1.01,$$

so the code-sequence $\mathbf{t} = 2101$ can be decoded as 210.1 or as 2.1.01.

Exercise 1.3

Determine whether or not the codes $\mathcal{C} = \{02, 12, 120, 20, 21\}$ and $\mathcal{C} = \{02, 12, 120, 21\}$ considered in Exercise 1.2 are uniquely decodable. If $\mathcal{C}$ is not uniquely decodable, find a code-sequence which can be decoded in at least two ways.

Since the proof of the Sardinas–Patterson Theorem is rather long, we will give it in Appendix A; here instead, we will give two typical arguments to illustrate the ideas involved.

($\Rightarrow$) Suppose that $\mathcal{C} \cap \mathcal{C}_\infty \neq \emptyset$, say $w \in \mathcal{C} \cap \mathcal{C}_2$; thus $uw = v$ with $u \in \mathcal{C}$ and $v \in \mathcal{C}_1$ or vice versa, and for simplicity let us assume that the first case holds (see Exercise 1.4 for the second case). Then $u'v = v'$ where $u', v' \in \mathcal{C}$, so the sequence $\mathbf{t} = u'uw \in T^*$ could represent a sequence $\mathbf{s}$ of three source-symbols (since $u', u, w \in \mathcal{C}$) or one source-symbol (since $u'uw = u'v = v' \in \mathcal{C}$). Thus decoding is not unique.

($\Leftarrow$) Suppose that we have an instance of non-unique decoding of the form $\mathbf{t} = u_1u_2 = v_1v_2$, where $u_1, u_2, v_1, v_2 \in \mathcal{C}$. We cannot have $|u_1| = |v_1|$, for this

would give $u_1 = v_1$ and so $u_2 = v_2$. Renumbering if necessary, we may therefore assume that $|u_1| > |v_1|$, so $u_1 = v_1 w$ where $|w| > 0$. Then $w \in \mathcal{C}_1$, so $u_2 \in \mathcal{C}_2$ since $wu_2 = v_2$. Thus $u_2 \in \mathcal{C} \cap \mathcal{C}_\infty$, so $\mathcal{C}$ and $\mathcal{C}_\infty$ are not disjoint.

Exercise 1.4

Suppose that $w \in \mathcal{C} \cap \mathcal{C}_2$, where $uw = v$ with $u \in \mathcal{C}_1$ and $v \in \mathcal{C}$. Give an example of a code-sequence which can be decoded in more than one way.

Exercise 1.5

A code $\mathcal{C}$ exhibits non-unique decodability in the form $012120.120 = 01.212.01.20$; find an element of $\mathcal{C} \cap \mathcal{C}_\infty$.

Exercise 1.6

Suppose that $w \in \mathcal{C} \cap \mathcal{C}_3$. By considering the various reasons why one could have $w \in \mathcal{C} \cap \mathcal{C}_3$, give examples of code-sequences which cannot be decoded uniquely.

The general arguments in the proof of Theorem 1.10 are similar to those outlined above, but they are rather more complicated since they have to deal with infinitely many different cases. Fortunately, there is a simpler necessary and sufficient condition for another important type of code, which we will consider in the next section.

We have defined unique decodability to mean that all finite code-sequences t can be decoded uniquely, but one could also consider the stronger requirement that this should be true for *all* code-sequences, finite or infinite. A theorem due to Even [Ev63], Levenshtein [Le64] and Riley [Ri67] shows that this happens if and only if $\mathcal{C} \cap \mathcal{C}_\infty = \emptyset$ and $\mathcal{C}_n = \emptyset$ for some $n \geq 1$. (These are also necessary and sufficient conditions for $\mathcal{C}$ to be *uniquely decodable with bounded delay*, meaning that there is a constant d such that if two code-sequences agree in their first d symbols, then they have the same first code-word; thus decoding can begin after a delay of at most d symbols. We will consider a stronger condition in the next section.)

Example 1.13

In Example 1.9 above, both conditions are satisfied, so all code-sequences are decoded uniquely. For an example where all finite code-sequences are decoded uniquely, but some infinite ones are not, see Exercise 1.7.

Exercise 1.7

For each of the ternary codes $\mathcal{C}$ in Exercise 1.2, determine whether or not all infinite code-sequences can be decoded uniquely. If not, give an example of such non-unique decoding.

For the remainder of this book, we will restrict our attention to finite code-sequences.

1.3 Instantaneous Codes

Before defining instantaneous codes, let us consider a few examples.

Example 1.14

Consider the binary code $\mathcal{C}$ given by

$$s_1 \mapsto 0,\ s_2 \mapsto 01,\ s_3 \mapsto 11.$$

Using the notation of §1.2, we have $\mathcal{C}_1 = \mathcal{C}_2 = \cdots = \{1\}$, so $\mathcal{C}_\infty = \{1\}$; thus $\mathcal{C} \cap \mathcal{C}_\infty = \emptyset$, so $\mathcal{C}$ is uniquely decodable by Theorem 1.10. Now suppose that we receive a finite message beginning $\mathbf{t} = 0111\ldots$. Although we know that this can be decoded uniquely, we cannot start to decode it until we come to the end of the block of consecutive 1s: if the number of 1s in this block is even, the decomposition of $\mathbf{t}$ must be $0.11.11.11\ldots$, and the decoded message must begin $\mathbf{s} = s_1 s_3 s_3 s_3 \ldots$; however, if the number of 1s is odd, the decomposition must be $01.11.11.11\ldots$, so $\mathbf{s} = s_2 s_3 s_3 s_3 \ldots$. In a practical situation, this delay in decoding could cause difficulties. We say that $\mathcal{C}$ is not instantaneous.

Example 1.15

The Prime Minister's telex prints RUSSIANS DECLARE WAR ... ; a quick decision is made, a button is pressed, and within minutes there are some very loud explosions. Soon, everyone is dead. Meanwhile, the telex continues printing ... RINGTON VODKA TO BE EXCELLENT.

Exercise 1.8

Show that the binary code $\mathcal{C} = \{0, 01, 011, 111\}$ is uniquely decodable; how should the receiver react on receiving a sequence starting $0111\ldots1\ldots$?

Example 1.16

Consider the binary code $\mathcal{D}$ given by

$$s_1 \mapsto 0,\ s_2 \mapsto 10,\ s_3 \mapsto 11,$$

the reverse of the code $\mathcal{C}$ in Example 1.14. We can see this is uniquely decodable, either by Theorem 1.10, or because $\mathcal{C}$ is. It is also instantaneous, in the sense

that we can decode a received message $\mathbf{t}$ as we go along: a 0 indicates w_1, which we decode as s_1, and a 1 indicates the start of $w_2 = 10$ or $w_3 = 11$, decoded as s_2 or s_3 as soon as we know the next symbol. Thus any code-word in $\mathbf{t}$ can be decoded as soon as it arrives, without delay.

Exercise 1.9

Is this also true for the code $\mathcal{D} = \{0, 10, 110, 111\}$, the reverse of the code $\mathcal{C}$ in Exercise 1.8?

Now for the formal definition: a code $\mathcal{C}$ is *instantaneous* if, for each sequence of code-words $w_{i_1}, w_{i_2}, \ldots, w_{i_n}$, every code-sequence beginning $\mathbf{t} = w_{i_1} w_{i_2} \ldots w_{i_n} \ldots$ is decoded uniquely as $\mathbf{s} = s_{i_1} s_{i_2} \ldots s_{i_n} \ldots$, no matter what the subsequent symbols in $\mathbf{t}$ are. Thus the code $\mathcal{C}$ in Example 1.14 is not instantaneous: a sequence $\mathbf{t} = w_1 w_3 \ldots = 011 \ldots$ might be decoded as $\mathbf{s} = s_1 s_3 \ldots$ or as $s_2 s_3 \ldots$, depending on the subsequent symbols. The code $\mathcal{D}$ in Example 1.16 is instantaneous: once $w_{i_1} w_{i_2} \ldots w_{i_n}$ is received, we know that it represents $s_{i_1} s_{i_2} \ldots s_{i_n}$, regardless of what comes next. By definition, every instantaneous code is uniquely decodable; Example 1.14 shows that the converse is false.

A code $\mathcal{C}$ is a *prefix code* if no code-word w_i is a prefix (initial segment) of any code-word w_j $(i \neq j)$; equivalently, $w_j \neq w_i w$ for any $w \in T^*$, that is, $\mathcal{C}_1 = \emptyset$ in the notation of §1.2. Thus $\mathcal{C}$ is not a prefix code in Example 1.14 (since 0 is a prefix of 01), but the reversed code $\mathcal{D}$ in Example 1.16 is a prefix code.

Theorem 1.17

A code $\mathcal{C}$ is instantaneous if and only if it is a prefix code.

Proof

($\Rightarrow$) If $\mathcal{C}$ is not a prefix code, say w_i is a prefix of w_j, then a code-sequence beginning $\mathbf{t} = w_i \ldots$ might be decoded as $\mathbf{s} = s_i \ldots$ or as $\mathbf{s} = s_j \ldots$, so $\mathcal{C}$ is not instantaneous.

($\Leftarrow$) If $\mathcal{C}$ is a prefix code, and $\mathbf{t}$ starts with $w_i \ldots$, then $\mathbf{s}$ must start with s_i, since no code-word w_j $(j \neq i)$ is a prefix of w_i or has w_i as a prefix. We can continue like this, decoding successive code-words in $\mathbf{t}$ as we receive them, so $\mathcal{C}$ is instantaneous. □

Examples 1.14 and 1.16 are illustrations of this result.

1.4 Constructing Instantaneous Codes

In order to understand the construction of instantaneous codes, it is useful to regard the set T^* of words in T as a graph, that is, a set of points (called *vertices*), some pairs of which are joined by edges. In this case, the vertices are the words $w \in T^*$, and each w is joined by an edge to the r words $wt_1, \ldots, wt_r$ formed by adding a single symbol $t_i \in T$ to the end of w. One can visualise this graph as growing upwards, with the empty word ε at the bottom, and the words of length l at level l above ε; in Graph Theory, such a graph is called an *r-ary rooted tree*. (A *tree* is a connected graph with no circuits; here the root is ε.) Figure 1.1 shows the binary tree T^*, up to level $l = 3$, with $T = \mathbf{Z}_2$.

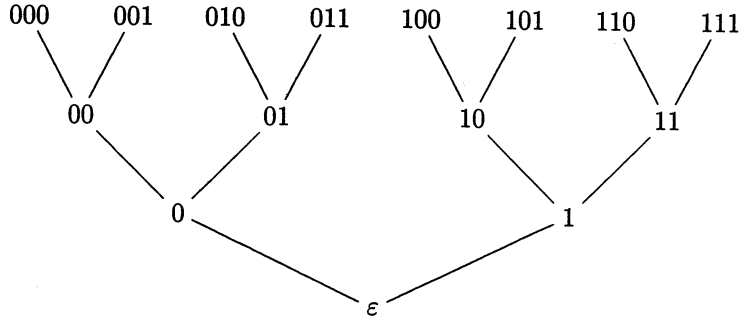

Figure 1.1

Exercise 1.10

Draw the ternary tree T^*, up to level $l = 3$, with $T = \mathbf{Z}_3$.

A code $\mathcal{C}$ can be regarded as a finite set of vertices of the tree T^*. A word w_i is a prefix of w_j if and only if the vertex w_i is dominated by the vertex w_j, that is, there is an upward path in T^* from w_i to w_j, so it follows from Theorem 1.17 that $\mathcal{C}$ is instantaneous if and only if no vertex $w_i \in \mathcal{C}$ is dominated by a vertex $w_j \in \mathcal{C}$ $(i \neq j)$. We can use this criterion to construct instantaneous codes, choosing vertices in T^* one at a time so that no vertex dominates (or is dominated by) any predecessor.

Example 1.18

Let us find an instantaneous binary code $\mathcal{C}$ for a source $\mathcal{S}$ with five symbols $s_1, \ldots, s_5$. First let us try $s_1 \mapsto w_1 = 0$, so 0 is a vertex in $\mathcal{C}$. If $\mathcal{C}$ is to be a prefix code, then no other vertex in $\mathcal{C}$ can dominate 0, so they must all dominate 1 (i.e. the other code-words must begin with 1). If we try $s_2 \mapsto w_2 = 1$ then

no further code-words can be added, since they would dominate w_1 or w_2. Instead, let us try $s_2 \mapsto w_2 = 10$. Then $s_3 \mapsto w_3 = 11$ is impossible, since it allows no further code-words, so let us try $s_3 \mapsto w_3 = 110$. Continuing, we find the possibility $s_4 \mapsto w_4 = 1110$, $s_5 \mapsto w_5 = 1111$. This gives an instantaneous binary code $C = \{0, 10, 110, 1110, 1111\}$ with word-lengths $l_i = 1, 2, 3, 4, 4$, shown in Figure 1.2. (This is not the only possibility: for instance, the binary code $\{00, 01, 10, 110, 111\}$ is also instantaneous.)

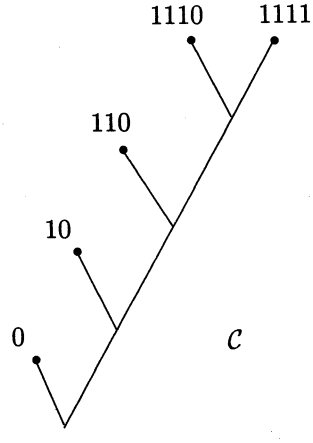

Figure 1.2

Example 1.19

Is there an instantaneous binary code for this source S with word-lengths $1, 2, 3, 3, 4$? Again, we use the binary tree T^*. Any choice of a code-word w_1 of length $l_1 = 1$, that is, a vertex of height 1, eliminates half of the vertices in T^* as possible code-words $w_2, \ldots, w_5$, namely all those dominating w_1, so a proportion $1 - \frac{1}{2} = \frac{1}{2}$ remains. A choice of w_2 at height 2 eliminates a further $1/2^2 = 1/4$ of T^*, leaving a proportion $1 - \frac{1}{2} - \frac{1}{4} = \frac{1}{4}$. Any choices of w_3 and w_4 at level 3 eliminate $1/2^3 + 1/2^3 = 1/4$ of T^*, so no vertices are left for w_5. The difficulty is that the sum of the proportions of T^* above each w_i exceeds 1:

$$\frac{1}{2} + \frac{1}{2^2} + \frac{1}{2^3} + \frac{1}{2^3} + \frac{1}{2^4} > 1.$$

Thus there is no instantaneous binary code for S with word-lengths $1, 2, 3, 3, 4$. There is, however, an instantaneous ternary code with these word-lengths: if $r = 3$ then choices of $w_1, \ldots, w_5$ eliminate proportions $1/3$, $1/3^2$, $1/3^3$, $1/3^3$, $1/3^4$ of the ternary tree T^*, where $|T| = 3$. Since

$$\frac{1}{3} + \frac{1}{3^2} + \frac{1}{3^3} + \frac{1}{3^3} + \frac{1}{3^4} \leq 1,$$

such choices are possible.

Exercise 1.11

Find an instantaneous ternary code with word-lengths $1, 2, 3, 3, 4$. Is there one with word-lengths $1, 1, 2, 2, 2, 2$?

This concept of the "proportion" of an infinite tree T^* is useful, but imprecise. By making it more precise, we can use arguments like those above to give necessary and sufficient conditions for the existence of instantaneous r-ary codes with given word-lengths.

1.5 Kraft's Inequality

Motivated by the examples in §1.4, we have the following result, known as *Kraft's inequality* [Kr49]:

Theorem 1.20

There is an instantaneous r-ary code C with word-lengths $l_1, \ldots . l_q$ if and only if

$$\sum_{i=1}^{q} \frac{1}{r^{l_i}} \leq 1 . \tag{1.5}$$

Proof

($\Leftarrow$) Renumbering the word-lengths if necessary, we may assume that $l_1 \leq \cdots \leq l_q$. Let $l = \max(l_1, \ldots, l_q)$, and consider the part $T^{\leq l} = T^0 \cup T^1 \cup \cdots \cup T^l$ of T^* up to height l. This is a finite tree: at each height $h = 0, 1, \ldots, l$ it has r^h vertices, the elements of T^h, or words of length h; its "leaves" are the r^l vertices at maximum height l.

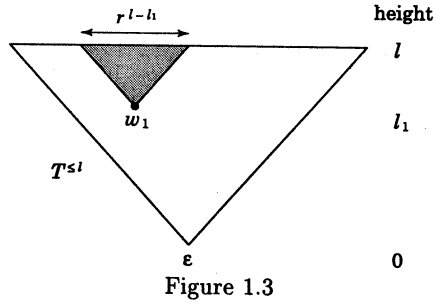

Figure 1.3

Let us assign any vertex w_1 at height l_1 to s_1, giving a code-word of length l_1, and then prune (delete) w_1 and all of $T^{\leq l}$ above w_1, since we cannot use

such vertices again; in particular, this removes r^{l-l_1} leaves, the words of length l beginning with w_1 (see Figure 1.3). If $q > 1$ then

$$r^{l-l_1} < r^l \sum_{i=1}^{q} \frac{1}{r^{l_i}} \leq r^l,$$

by (1.5), so at least one leaf of $T^{\leq l}$ remains unpruned. The first l_2 symbols of this leaf give us a code-word w_2 of length l_2, which is not above w_1 in the tree. We now prune w_2 and all of $T^{\leq l}$ above it, thus removing a further r^{l-l_2} leaves (no leaf can be above both w_1 and w_2). We continue like this, choosing code-words and then pruning, so that no code-word is above or below any other. After k code-words $w_1, \ldots, w_k$ have been chosen, where $k < q$, we have pruned $r^{l-l_1} + \cdots + r^{l-l_k}$ of the r^l leaves; since

$$\sum_{i=1}^{k} r^{l-l_i} < r^l \sum_{i=1}^{q} \frac{1}{r^{l_i}} \leq r^l,$$

at least one leaf remains, so it is possible to choose w_{k+1} of length l_{k+1} as a prefix of such a leaf. We can therefore continue choosing code-words until w_q has been chosen. Throughout this process, the prefix condition is satisfied, so the resulting code $\mathcal{C} = \{w_1, \ldots, w_q\}$ is instantaneous by Theorem 1.17.

($\Rightarrow$) If $\mathcal{C}$ is instantaneous then it is a prefix code (by Theorem 1.17), so every leaf of $T^{\leq l}$ is above at most one code-word of $\mathcal{C}$. Each code-word w_i of $\mathcal{C}$ is below r^{l-l_i} leaves (where $l_i = |w_i|$), so summing over all w_i in $\mathcal{C}$ we find that there are $\sum_i r^{l-l_i}$ leaves above code-words. Since there are r^l leaves in $T^{\leq l}$, we must have

$$\sum_{i=1}^{q} r^{l-l_i} \leq r^l,$$

so dividing by r^l we get (1.5). □

For illustrative examples of Kraft's inequality, see §1.4.

1.6 McMillan's Inequality

We have seen that the class of uniquely decodable codes is strictly larger than the class of instantaneous codes, so one might expect a weaker necessary and sufficient condition than (1.5) for the existence of a uniquely decodable r-ary code with specified word-lengths. Surprisingly, this is not the case. *McMillan's inequality* [McM56] is as follows:

Theorem 1.21

There is a uniquely decodable r-ary code $\mathcal{C}$ with word-lengths $l_1, \ldots, l_q$ if and only if

$$\sum_{i=1}^{q} \frac{1}{r^{l_i}} \leq 1 . \tag{1.6}$$

Proof

($\Leftarrow$) By Theorem 1.20 there is an instantaneous r-ary code with word-lengths l_i; being instantaneous, this code is uniquely decodable.

($\Rightarrow$) Let $\mathcal{C}$ be a uniquely decodable r-ary code with word-lengths $l_1, \ldots, l_q$; define

$$K = \sum_{i=1}^{q} \frac{1}{r^{l_i}} , \tag{1.7}$$

and let

$$l = \max(l_1, \ldots, l_q),$$
$$m = \min(l_1, \ldots, l_q).$$

Now consider the expansion of

$$K^n = \left(\sum_{i=1}^{q} \frac{1}{r^{l_i}} \right)^n ,$$

where $n \geq 1$. This is a sum of terms

$$\frac{1}{r^{l_{i_1}}} \times \frac{1}{r^{l_{i_2}}} \times \cdots \times \frac{1}{r^{l_{i_n}}} = \frac{1}{r^j}$$

where

$$j = l_{i_1} + \cdots + l_{i_n} . \tag{1.8}$$

Here $mn \leq j \leq ln$ since $m \leq l_i \leq l$ for all i, so collecting terms together we can write

$$K^n = \sum_{j=mn}^{ln} \frac{N_{j,n}}{r^j} .$$

The coefficient $N_{j,n}$ of $1/r^j$ is the number of ways of writing j in the form (1.8) as a sum of n word-lengths (possibly with repetitions); equivalently, $N_{j,n}$ is the number of sequences $w_{i_1}, \ldots, w_{i_n}$ of n code-words of $\mathcal{C}$, of total length j. Each such sequence determines a code-sequence $\mathbf{t} = w_{i_1} \ldots w_{i_n}$ of length j, and since $\mathcal{C}$ is uniquely decodable, each $\mathbf{t}$ arises from at most one such sequence of code-words. Thus $N_{j,n}$ is at most equal to the number of code-sequences $\mathbf{t}$ of length j, that is, $N_{j,n} \leq r^j$. Since K^n is a sum of $ln+1-mn$ terms $N_{j,n}/r^j \leq 1$, we have

$$K^n \leq (l - m)n + 1 \tag{1.9}$$

for all $n \geq 1$. Now K, l and m are independent of n, so if $K > 1$ then the left-hand side in this inequality grows exponentially, while the right-hand side grows only linearly. This contradicts (1.9) for sufficiently large n, so we must have $K \leq 1$. □

The above proof is due to Karush [Ka61]; the original proof used complex functions (see [McM56] or [Re61, pp. 147–8]). Theorems 1.20 and 1.21 immediately imply:

Corollary 1.22

There is an instantaneous r-ary code with word-lengths $l_1, \ldots, l_q$ if and only if there is a uniquely decodable r-ary code with these word-lengths.

1.7 Comments on Kraft's and McMillan's Inequalities

Comment 1.23

Theorems 1.20 and 1.21 do *not* say that an r-ary code with word-lengths $l_1, \ldots, l_q$ is instantaneous or uniquely decodable if and only if $\sum r^{-l_i} \leq 1$. For instance, the binary code $\mathcal{C} = \{0, 01, 011\}$ has $l_i = 1, 2, 3$, so $\sum r^{-l_i} = \frac{7}{8} \leq 1$; however, $\mathcal{C}$ is not a prefix code, so it is not instantaneous. Similarly, one can find a binary code with these word-lengths, which is not uniquely decodable: $\{0, 01, 001\}$ is an obvious example. However:

Comment 1.24

Theorems 1.20 and 1.21 assert that if $\sum r^{-l_i} \leq 1$, then there exist codes with these parameters which *are* instantaneous and uniquely decodable: for instance, the binary code $\{0, 10, 110\}$ is a prefix code, and hence satisfies both conditions.

Comment 1.25

If an r-ary code $\mathcal{C}$ is uniquely decodable, then it need not be instantaneous, but by Corollary 1.22 there must be an instantaneous r-ary code with the same word-lengths. For instance the binary code $\mathcal{C} = \{0, 01, 11\}$ in Example 1.14 is uniquely decodable but not instantaneous; with the same word-lengths we have the instantaneous code $\mathcal{D} = \{0, 10, 11\}$ in Example 1.16.

Comment 1.26

The summand r^{-l_i} in $K = \sum r^{-l_i}$ corresponds to the rather imprecise notion of the "proportion" of the tree T^* above a vertex w_i of height l_i, as used in §1.4. This interpretation helps to explain why we need $K \leq 1$.

1.8 Supplementary Exercises

Exercise 1.12

Is there an instantaneous code over $\mathbf{Z}_3$ with word-lengths l_i =1, 2, 2, 2, 2, 2, 3, 3, 3, 3? Construct one with $l_i = 1, 2, 2, 2, 2, 2, 3, 3, 3$; how many such codes are there?

Exercise 1.13

The binary code $\mathcal{C} = \{0, 10, 11\}$ is used; how many code-sequences of length j can be formed from $\mathcal{C}$? (Hint: find and solve a recurrence relation for this number N_j.)

Exercise 1.14

Suppose that $|T| = r$, $1 \leq l_1 \leq \cdots \leq l_q$, and $\sum r^{-l_i} \leq 1$. In how many ways can one choose words $w_1, \ldots, w_q \in T^*$ so that $|w_i| = l_i$ and $\{w_1, \ldots, w_q\}$ is an instantaneous code?

Exercise 1.15

A code is *exhaustive* if every sufficiently long sequence of code-symbols begins with a code-word (equivalently, every infinite sequence of code-symbols can be decomposed into code-words). Find an equivalent condition in terms of the leaves of the tree $T^{\leq l}$ in the proof of Theorem 1.20. Which of the codes in the examples in this chapter are exhaustive?

Exercise 1.16

Show that if an r-ary code $\mathcal{C}$ with word-lengths $l_1, \ldots, l_q$ is exhaustive, then $\sum r^{-l_i} \geq 1$, with equality if and only if $\mathcal{C}$ is instantaneous.

Exercise 1.17

Let $\mathcal{C}$ be an r-ary code with word-lengths $l_1, \ldots, l_q$. Show that any two of the following conditions imply the third:

(a) $\mathcal{C}$ is instantaneous;

(b) $\mathcal{C}$ is exhaustive;

(c) $\sum r^{-l_i} = 1$.

Show that no one of these conditions implies any other.