第 5 章　无失真信源编码

通信的实质是信息的传输。而高速度、高质量地传送信息是信息传输的基本问题。将信源信息通过信道传送给信宿，怎样才能做到尽可能不失真而又快速呢？这就需要解决两个问题：

（1）在不失真或允许一定失真的条件下，如何用尽可能少的符号来传送信源信息；

（2）在信道受干扰的情况下，如何增加信号的抗干扰能力，同时又使得信息传输率最大。

为了解决这两个问题，就要引入信源编码和信道编码。

一般来说，提高抗干扰能力（降低失真或错误概率）往往是以降低信息传输率为代价的；反之，要提高信息传输率常常又会使抗干扰能力减弱。二者是有矛盾的。然而在信息论的编码定理中，已从理论上证明，至少存在某种最佳的编码或信息处理方法能够解决上述矛盾，做到既可靠又有效地传输信息。这些结论对各种通信系统的设计和评价具有重大的理论指导意义。

第 4 章讨论了满足一定失真限度的限失真信源编码。那么在无失真的条件下如何让信源消息尽可能快地传递到接收端呢？这个问题引出了无失真信源编码。本章将在第 2 章信源统计特性和信源熵概念的基础上，研究离散信源的无失真编码问题，重点讨论无失真信源编码定理，并给出以香农编码、费诺编码和哈夫曼编码为代表的最佳无失真信源编码方法。

5.1　编码的基本概念

无失真信源编码是一种可逆编码，是指当信源符号转换成码字后，可从接收码字无失真地恢复原信源符号。本节主要讨论对离散信源进行无失真编码的要求和方法，涉及的基本概念有编码器的定义、码字的类型划分、即时码的构造及唯一可译码的判断等。

5.1.1　编码器的定义

编码实质上是对信源的原始符号按一定的数学规则进行的一种变换。

图 5.1.1 所示是一个信源编码器，它的输入是信源符号序列 $X^N=(X_1X_2\cdots X_N)$，序列中的每个符号 $X_i\in\{a_1,a_2,\cdots,a_n\}$；而每个符号序列依照固定的码表映射成一个码符号序列 $Y^{K_N}=(Y_1Y_2\cdots Y_{K_N})$，码符号序列中的每个符号 $Y_k\in\{b_1,b_2,\cdots,b_m\}$，一般来说，元素 b_j 是适合信道传输的，称为码符号（或者码元）。输出的码符号序列称为码字，长度 K_N 称为码字长度或简称码长。可见，编码就是从信源符号到码符号的一种映射。若要实现无失真编码，则这种映射必须是一一对应，并且是可逆的。

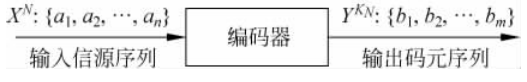


图 5.1.1　无失真信源编码器

例 5.1.1　如果信源输出的符号序列长度为 1，即信源输出符号集

$$X\in\{x_1,x_2,\cdots,x_n\}$$

信源概率空间

$$\begin{bmatrix} X \\ P \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ p(x_1) & p(x_2) & \cdots & p(x_n) \end{bmatrix}$$

假设该信道为二元信道，即信道的符号集为{0,1}。若将信源 X 通过该二元信道传输，就必须把信源符号 x_i 变换成由 0、1 符号组成的码符号序列，即要进行编码。$n=2,4,8$ 时，信源符号与码字的一种可能的一一对应关系为

$$\begin{aligned} n=2 \quad & x_1 \to 0, \quad x_2 \to 1 \\ n=4 \quad & x_1 \to 00, \quad x_2 \to 01, \quad x_3 \to 10, \quad x_4 \to 11 \\ n=8 \quad & x_1 \to 000, \quad x_2 \to 001, \quad x_3 \to 010, \quad x_4 \to 011 \\ & x_5 \to 100, \quad x_6 \to 101, \quad x_7 \to 110, \quad x_4 \to 111 \end{aligned}$$

例 5.1.2 为了传输一个由字母 A、B、C、D 组成的符号集，把每个字母编码成二元码脉冲序列，以“00”代表 A，“01”代表 B，“10”代表 C，“11”代表 D，每个二元码脉冲宽度为 5ms。

(1) 不同字母等概率出现时，计算信息的传输速率；

(2) 若每个字母出现的概率分别为 $p_A=\frac{1}{5}$，$p_B=\frac{1}{4}$，$p_C=\frac{1}{4}$，$p_D=\frac{3}{10}$，试计算信息的传输速率。

解：(1) 不同字母等概率出现时，符号集的概率空间为

$$\begin{bmatrix} X \\ P \end{bmatrix} = \begin{bmatrix} \mathrm{A} & \mathrm{B} & \mathrm{C} & \mathrm{D} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \end{bmatrix}$$

每个符号含有的平均信息量即熵为

$$H(X) = \log_2 4 = 2(\text{bit/ 符号})$$

现在用两个二元码脉冲代表一个字母，每个二元码脉冲宽度为 $\tau=5\text{ms}$，则每个字母占用 $t=2\tau=10\text{ms}$。1s 内可以传输的字母个数为

$$n = \frac{1}{t} = 100(\text{字母 /s})$$

则信息传输速率

$$R_t = nH(X) = 200(\text{b/s})$$

(2) 字母出现概率不同时，据题意其概率空间为

$$\begin{bmatrix} X \\ P \end{bmatrix} = \begin{bmatrix} \mathrm{A} & \mathrm{B} & \mathrm{C} & \mathrm{D} \\ \frac{1}{5} & \frac{1}{4} & \frac{1}{4} & \frac{3}{10} \end{bmatrix}$$

则此时每个字母含有的平均信息量为

$$H(X) = -\sum_{i=1}^{4} p(a_i)\log p(a_i) = 1.985(\text{bit/ 符号})$$

同(1)，计算得信息传输速率为

$$R_t = nH(X) = 198.5(\text{b/s})$$

可见，编码后信源的信息传输速率与信源的统计特性有关。

对于固定的信源，信源编码的方式对信源的信息传输速率有什么影响呢？

上例中，将字母 A、B、C、D 编码为由两个码符号组成的等长码。当然，也可以把每个字

母编为长度不同的码字。如

码 1：A→111；B→10；C→01；D→0

码 2：A→0；B→01；C→001；D→111

码 1 和码 2 的性能怎么评价呢？哪个最适合给定的信源，使编码后的信息传输速率最大呢？它们是否满足无失真编码的要求呢？这些问题，将在后续内容中讲到。

无失真码字的性能除了与信源的统计特性相关外，主要由码符号的类型、码长决定。图 5.1.2 是一个码分类图。

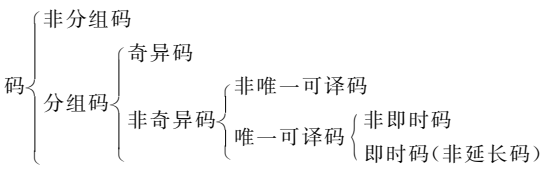


图 5.1.2　码的分类

这些码的定义如下：

分组码与非分组码——将信源消息分成若干组(符号序列)，对每一组按照固定的码表映射成一个码字，这样的码称为分组码，也叫块码。只有分组码有固定的码表，而非分组码中不存在码表。

二元码——若码符号集为 $Y=\{0,1\}$，所有码字都是一些二元序列，则称为二元码。二元码是数字通信和计算机系统中最常用的一种码。

等长码与变长码——若一组码中所有码字的长度都相同，则称为等长码或定长码。若码字的长短不一，则称为变长码。

奇异码与非奇异码——若一组码中所有码字都不相同，则称为非奇异码。非奇异码中，信源符号和码字是一一对应的；反之，若一组码中有相同的码字，则该码为奇异码。奇异码不可能是无失真的码字。

唯一可译码与非唯一可译码——若任意一串有限长的码符号序列只能唯一地被译成所对应的信源符号序列，则此码称为唯一可译码；否则，就称为非唯一可译码。

即时码与非即时码——唯一可译码可分为即时码和非即时码。如果接收端收到一个完整的码字后，不能立即译码，还要等下一个码字开始接收后才能判断是否可以译码，这样的码称为非即时码。如果收到一个完整的码字以后，就可以立即译码，则称为即时码。即时码要求任何一个码字都不是其他码字的前缀部分，也称为异前缀码，又称为非延长码。即时码一定是唯一可译码，但是非即时码并非一定不是唯一可译码，这取决于码字的总体结构，可采用前缀后缀法进行判断。

例 5.1.3　给定信源

$$\begin{bmatrix} X \\ P \end{bmatrix}=\begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ \dfrac{1}{2} & \dfrac{1}{4} & \dfrac{1}{8} & \dfrac{1}{8} \end{bmatrix}$$

对该信源进行二元编码，如表 5.1.1 所示。

表 5.1.1　二元信源编码示例

信源	码　字					
	码 1	码 2	码 3	码 4	码 5	码 6
x_1	00	0	0	1	1	0
x_2	01	10	11	10	01	01
x_3	10	00	10	100	001	001
x_4	11	11	11	1000	0001	111

码 1 中，每个码字的长度相同，为等长码。每个码字各不相同，为非奇异码。若等长码为非奇异码，则一定是唯一可译码，且为即时码。

码 2、码 3、码 4、码 5 和码 6 均为变长码。码 2 中的每个码字都不相同，为非奇异码。码字“00”有两种译码方法，既可以译成信源符号 x_3，又可以译成 x_1x_1，因此码 2 是非唯一可译码。

码 3 中有相同的码字，为奇异码，一定不是唯一可译码。

码 4 为唯一可译码，也为前缀码，短码是长码的前缀，除去前缀“1”，码字“10”的后缀“0”、码字“100”的后缀“00”及码字“1000”的后缀都不是码组中的码字，因此该非即时码为唯一可译码，这就是前缀后缀判断法。

码 5 为即时码，一定是唯一可译码。

码 6 为前缀码，短码“0”是长码“01”“001”的前缀，对应的后缀分别为“1”“01”，其中后缀“01”是码组中单独的码字，因此码 6 不是唯一可译码。码符号序列 001 既可以译成信源符号 x_3，又可以译成 x_1x_2。

5.1.2　即时码的码树构造法

即时码一定是唯一可译码。无失真信源编码要求所编的码字必须是唯一可译码，包含两层含义，一是要求信源符号与码字一一对应；二是要求码字的反变换也对应唯一的信源符号，否则就会引起译码带来的错误和失真。

即时码作为唯一可译码以其较快的译码速度得到广泛应用。那么，如何构造即时码呢？

即时码的一种简单构造方法是码树法。

对于给定码字的全体集合

$$C = \{W_1, W_2, \cdots, W_q\}$$

可以用码树来描述它。所谓树，就是既有根、枝，又有节点，如图 5.1.3 所示。图中，最上端 A 为根节点，A、B、C、D、E 皆为节点，E 为终端节点。A、B、C、D 为中间节点，中间节点不安排码字，而只在终端节点安排码字，每个终端节点所对应的码字就是从根节点出发到终端节点走过的路径上所对应的符号组成。终端节点 E，走过的路径为 ABCDE，所对应的码符号分别为 0、0、0、1，则 E 对应的码字为 0001。可以看出，按码树法构成的码一定是即时码，是非前缀的唯一可译码。

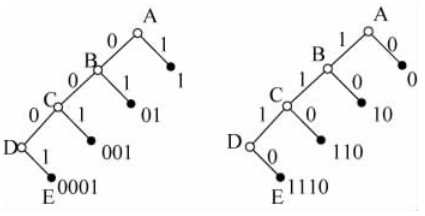


图 5.1.3　码树结构图

从码树上可以得知，当第 i 阶的节点作为终端节点，且分配码字，则码字的码长为 i。任一即时码都可以用码树来表示。当码字长度给定后，用码树法安排的即时码不是唯一的。

如图 5.1.3 中，如果把左树枝安排为 1，右树枝安排为 0，则得到不同的结果。

对一个给定的码，画出其对应的码树，如果有中间节点安排了码字，则该码一定不是即时码。这是最简单的即时码判断方法。

每个中间节点上都有 r（r 为进制数，如为二进制码树，则 $r=2$）个分支的树称为满树，否则为非满树。

即时码的码树还可以用来译码。当收到一串码符号序列后，首先从根节点出发，根据接收到的第一个码符号来选择应走的第一条路径，再根据接收到的第二个符号来选择应走的第二条路径，直到走到终端节点为止，就可以根据终端节点，立即判断出所接收的码字。然后从树根继续下一个码字的判断。这样，就可以将接收到的一串码符号序列译成对应的信源符号序列。

在表 5.1.1 给出的码组中，码 1、码 2 和码 5 对应的码树图分别如图 5.1.4(a)、(b)、(c)所示。

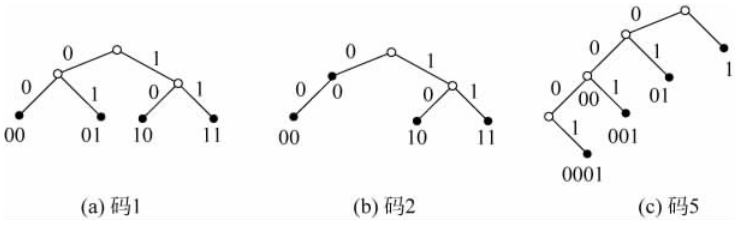


图 5.1.4　码 1、码 2 和码 5 对应的码树

可见，码 1 和码 5 的各码字位于树的终端节点，满足即时码的条件。

5.1.3　唯一可译码与克拉夫特不等式

用码树的概念可以判断、构造即时码，还可以导出唯一可译码存在的充分和必要条件，即各码字的长度 K_i 应符合克拉夫特(Kraft)不等式。

定理　对于码符号为 $Y=\{b_1,b_2,\cdots,b_m\}$ 的任意唯一可译码，其码字为 $W_1,W_2,\cdots,W_q$，所对应的码长为 $k_1,k_2,\cdots,k_q$，则必定满足克拉夫特不等式

$$\sum_{i=1}^{q} m^{-k_i} \leqslant 1 \tag{5.1.1}$$

反之，若码长满足上面的不等式，则一定存在具有这样码长的唯一可译码。式中，m 为码符号进制数。

克拉夫特不等式只是说明唯一可译码是否存在，并不能作为唯一可译码的判据，但可以作为某码组不是唯一可译码的判据。如{0,10,11,110}不满足克拉夫特不等式，则肯定不是唯一可译码；{0,10,010,111}满足克拉夫特不等式，但却不是唯一可译码。因为，如果收到码字“010”，则存在两种可能的译码方法：既可译为“010”对应的信源符号，又可译为“0”“10”对应的信源符号；而{0,10,110,111}是满足克拉夫特不等式的唯一可译码。

例 5.1.4　设二进制码树中 $X=\{x_1,x_2,x_3,x_4\}$，对应的 $k_1=1,k_2=2,k_3=2,k_4=3$，由上述定理，可得

$$\sum_{i=1}^{4} 2^{-k_i} = 2^{-1}+2^{-2}+2^{-2}+2^{-3} = \frac{9}{8} > 1$$

因此不存在满足这种码长的唯一可译码。{0,10,11,110}的码长分布如题，肯定不是唯一可译码。

根据克拉夫特不等式，结合前缀后缀法即可以判断某码组是否是唯一可译码。

(1) 等长码为唯一可译码的判断法。等长码若为非奇异码，一定是唯一可译码。

(2) 变长码为唯一可译码的判断法。首先，该变长码的各码字长度若不满足克拉夫特不等式，则一定不是唯一可译码；如果变长码的各码字长度满足克拉夫特不等式，则进一步按照前缀后缀法判断。将码 C 中所有可能的尾随后缀组成一个集合 F，当且仅当集合 F 中没有包含任一码字，则可判断此码 C 为唯一可译码。

集合 F 的构成方法：首先，观察码 C 中最短的码字是否是其他码字的前缀，若是，将其所有可能的尾随后缀排列出。而这些尾随后缀又有可能是某些码字的前缀，再将这些尾随后缀产生的新的尾随后缀列出，然后再观察这些新的尾随后缀是否是某些码字的前缀，再将产生的尾随后缀列出，依此下去，直到没有一个尾随后缀是码字的前缀为止。这样，首先获得了由最短的码字能引起的所有尾随后缀，接着，按照上述步骤将次短码字等所有码字可能产生的尾随后缀全部列出。由此得到由码 C 的所有可能的尾随后缀的集合 F。

例 5.1.5 设码 $C=\{0,10,1100,1110,1011,1101\}$，根据唯一可译码前缀后缀判断法，判断其是否是唯一可译码。

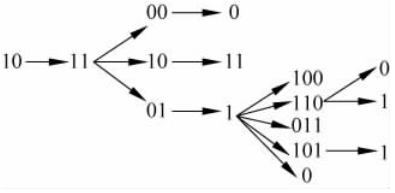
解：首先，计算克拉夫特不等式

$$\sum_{i=1}^{6} 2^{-k_i} = 2^{-1} + 2^{-2} + 2^{-4} + 2^{-4} + 2^{-4} + 2^{-4} = 1$$

因此，该码组满足克拉夫特不等式。存在该码长分布的唯一可译码。下面结合前缀后缀法判断具体码字。

(1) 先看最短的码字“0”，它不是其他码字的前缀，所以没有尾随后缀。

(2) 再观察次短码字“10”，它是码字“1011”的前缀，因此有尾随后缀



尾随后缀集合 $F=\{11,00,10,01,0,11,1,100,110,011,101\}$，其中“10”“0”为码字，故码 C 不是唯一可译码。

5.1.4 码性能评价参数

无失真码字的性能除了与信源的统计特性相关外，主要由所编码字的特性决定。无失真编码要求所编的码字是唯一可译码，码组中的各码字的码长要满足克拉夫特不等式。无失真编码器的整体性能需要用平均码长、编码信息率、压缩比、编码效率和信道剩余度评价。

1. 平均码长

如果单符号信源概率空间为

$$\begin{bmatrix} X \\ P \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ p(x_1) & p(x_2) & \cdots & p(x_n) \end{bmatrix}$$

对每个信源符号进行无失真信源编码，对应码字为 $C=\{W_1,W_2,\cdots,W_n\}$，所对应的码长为 $k_1,k_2,\cdots,k_n$。根据无失真信源编码的要求中信源符号与码字的一一映射关系，码字概率空间和码长概率空间为

$$\begin{bmatrix} C \\ P \end{bmatrix} = \begin{bmatrix} W_1 & W_2 & \cdots & W_n \\ p(x_1) & p(x_2) & \cdots & p(x_n) \end{bmatrix}$$

和

$$\begin{bmatrix} K \\ P \end{bmatrix} = \begin{bmatrix} k_1 & k_2 & \cdots & k_n \\ p(x_1) & p(x_2) & \cdots & p(x_n) \end{bmatrix} \tag{5.1.2}$$

各码字长度的数学期望，即平均码长为

$$\overline{K} = E[k_i] = \sum_{i=1}^{n} k_i p(x_i) \tag{5.1.3}$$

上述信源编码的对象是单符号信源，因此式(5.1.3)的平均码长又称为单符号码长。当然，大多数的情况下，可以对单符号信源的 N 次扩展信源进行编码，所编的码字称为 N 次扩展码。假设 N 次扩展信源的概率空间为

$$\begin{bmatrix} X^N \\ P \end{bmatrix} = \begin{bmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_q \\ p(\alpha_1) & p(\alpha_2) & \cdots & p(\alpha_q) \end{bmatrix} \tag{5.1.4}$$

对每个信源序列进行无失真信源编码，对应码字为 $C=\{W_1,W_2,\cdots,W_q\}$，所对应的码长为 $k_1,k_2,\cdots,k_q$。

各码字长度的数学期望，即平均码长为

$$\overline{K}_N = \sum_{i=1}^{q} k_i p(\alpha_i) \tag{5.1.5}$$

称 $\overline{K}_N$ 为序列码长，对应的单符号码长为

$$\overline{K} = \frac{\overline{K}_N}{N} \tag{5.1.6}$$

单符号码长的物理意义就是平均一个信源符号对应 $\overline{K}$ 个码符号。那么，单符号信源的熵也将平均分配给 $\overline{K}$ 个码符号。

2. 编码信息率

单符号信源 X 的信源熵就是信源的信息传输速率。若所编码字的单符号码长为 $\overline{K}$，则单符号信源的熵等于 $\overline{K}$ 个码符号的熵。当信源给定时，信源的熵确定，而编码后每个信源符号平均用 $\overline{K}$ 个码符号来替换。那么，每个码符号的平均信息量，即编码信息率为

$$R = \frac{H(X)}{\overline{K}} \tag{5.1.7}$$

若传输一个码符号平均需要 t 秒，则编码后信源每秒钟传输的信息量，即编码后信源的信息传输速率为

$$R_t = \frac{H(X)}{t\overline{K}} \tag{5.1.8}$$

式(5.1.7)和式(5.1.8)给出了单符号信源的编码信息率和信息传输速率；对于如图5.1.1所示的信源序列编码器，只需要把式中的单符号信源熵 $H(X)$ 替换为信源序列的符号熵 $H_N(X)$ 即可。

$$R = \frac{H_N(X)}{\overline{K}} \tag{5.1.9}$$

$$R_t = \frac{H_N(X)}{t\overline{K}} \tag{5.1.10}$$

式中，$\overline{K}=\frac{\overline{K}_N}{N}$，代入式(5.1.9)，可得

$$R = \frac{NH_N(X)}{\overline{K}_N} = \frac{H(X^N)}{\overline{K}_N} \tag{5.1.11}$$

式中，$H(X^N)$ 为无记忆信源序列的序列熵。

编码信息率既可以表示为单符号熵与单符号码长之比，也可以表示为序列熵与序列码长之比。而且，平均码长越短，编码信息率就越大，编码的信息传输速率就越高。为此，信源编码研究中感兴趣的是使平均码长为最短的码。

3. 压缩比

压缩比是衡量数据压缩程度的指标之一。目前常用的压缩比定义为

$$P_r = \frac{L_B - L_d}{L_B} \times 100\% \tag{5.1.12}$$

式中，L_B 为源代码长度；L_d 为压缩后的代码长度。

压缩比的物理意义是被压缩掉的数据占据源数据的百分比。当压缩比 P_r 接近 100% 时，压缩效果最理想。

4. 编码效率

与信源信息传输速率的定义相类似，编码后信道中传输的是码符号，因为不能完全获得码符号的概率分布，只能按照每个码符号的等概率分布进行估算。也就是说，所设计的信道要具有传输码符号等概率分布时的最大熵的传输能力或手段。

图5.1.1给出的编码器中，码符号序列 $Y^{K_N}=(Y_1Y_2\cdots Y_{K_N})$，码符号序列中的每个符号 $Y_k\in\{b_1,b_2,\cdots,b_m\}$，根据最大熵定理，当码符号等概率分布时，即 $p(b_i)=\frac{1}{m}$，每个码符号携带的平均信息量最大，等于 $\log m$bit，长度为 $\overline{K}_N$ 的码字的最大信息量为 $\overline{K}_N\log m$bit。用该码字表示长为 N 的信源序列，则送出一个信源符号所需要的信息率最大值为

$$R_{max} = \frac{\overline{K}_N}{N}\log m = \overline{K}\log m \tag{5.1.13}$$

定义编码效率

$$\eta = \frac{H_N(X)}{\overline{K}\log m} \tag{5.1.14}$$

对二元码，$m=2$，代入式(5.1.14)，可得

$$\eta = \frac{H_N(X)}{\overline{K}} \tag{5.1.15}$$

或

$$\eta = \frac{H(X^N)}{\overline{K}_N}$$

从工程观点来看，总希望通信设备经济、简单，并且单位时间内传输的信息量越大越好。

5. 信道剩余度（冗余度）

信道剩余度表示信道未被利用的程度，所以是冗余的。信道冗余度定义为

$$\gamma = 1 - \eta = 1 - \frac{H_N(X)}{\overline{K}} \tag{5.1.16}$$

可见，对于二元信源编码，编码效率与编码信息率大小相同，只是单位不同。平均码长越短，编码效率和编码信息率越高。当平均码长等于信源熵时，编码效率达到上限 100%，编码信息率达到二元对称信道的信道容量 1，消除信源冗余度实现了信源与信道的匹配。由上述分析可见，最短的平均码长与信源的统计特性有关，如果某种信源编码方法的平均码长最短，可称之为最佳无失真信源编码。一般情况下，如何使无失真信源编码的平均码长尽可能接近信源熵并可实现是被关心的问题，在解决该问题的过程中需要遵循的是无失真信源编码定理。

5.2　无失真信源编码定理

由 5.1 节可知，无失真信源编码要求所编码字必须是唯一可译码，这样才能保证无失真或无差错地从 Y 恢复 X，也就是能正确地进行译码；如果进一步考虑编码前后的信息传输率变化，无失真信源编码的编码信息率的最大值必定不能小于信源的信息率，同时希望传送 Y 时所需要的信息率最小。

无失真信源编码的编码信息率的最大值必定不能小于信源的信息率的数学描述为

$$\frac{\overline{K}_N}{N}\log m \geqslant H_N(X) \tag{5.2.1}$$

可写成

$$\overline{K}\log m \geqslant H_N(X) \tag{5.2.2}$$

即

$$\overline{K} \geqslant \frac{H_N(X)}{\log m} \tag{5.2.3}$$

上式给出了无失真信源编码平均码长的下界，对于二元编码，$\overline{K} \geqslant H_N(X)$。

5.2.1　定长编码定理

前面已经讲过，所谓信源编码，就是将信源符号序列变换成另一个序列（码字）。设信源输出符号序列长度为 N，码字的长度为 K_N，编码的目的就是要使信源的信息率最小，也就是说，要用最少的符号来代表信源。

在定长编码中，对每一个信源序列，K_N 都是定值，编码的目的是寻找最小 K_N 值。

定长编码定理　由 N 个符号组成的、每个符号熵为 $H_N(X)$ 的无记忆平稳信源符号序列 $X_1X_2\cdots X_N$，可用 K_N 个符号 $Y_1Y_2\cdots Y_{K_N}$（每个符号有 m 种可能值）进行定长编码。对任

意 $\varepsilon>0$，$\delta>0$，只要

$$\frac{\overline{K}_N}{N}\log m \geqslant H_N(X)+\varepsilon \tag{5.2.4}$$

则当 N 足够大时，必可使译码差错小于 δ，即可实现几乎无失真编码；反之，当

$$\frac{K_N}{N}\log m < H_N(X)+\varepsilon \tag{5.2.5}$$

时，译码差错一定是有限值（即不可能实现无失真编码），当 N 足够大时，译码几乎必定出错（译码错误概率近似等于 1）。这就是单符号信源无失真定长编码定理。

定长编码定理是在平稳无记忆信源的条件下论证的，但它同样适用于平稳有记忆信源，只是要求有记忆信源的极限熵存在。对于平稳有记忆信源，定理的两个不等式中的 $H_N(X)$ 应改为极限熵 $H_\infty(X)$。

对二元编码，$m=2$，式(5.2.4)成为

$$\frac{K_N}{N} \geqslant H_N(X)+\varepsilon \tag{5.2.6}$$

可见，定理给出了等长编码时平均每个信源符号所需的二元码符号的理论极限，这个极限值由信源熵 $H_N(X)$ 决定。

式(5.2.4)中，左边是输出码字每符号所能载荷的最大信息量（编码后每个信源符号所携带的最大信息量）$\frac{K_N}{N}\log m$，右边是信源序列的符号熵 $H_N(X)$。不等式两边同时乘以信源序列的长度 N，则式(5.2.4)可改写为

$$K_N\log m \geqslant NH_N(X)+\varepsilon$$

或

$$K_N\log m \geqslant H(X^N)+\varepsilon' \tag{5.2.7}$$

这个不等式左边表示长为 K_N 的码符号序列（码字）所载荷的最大信息量，而右边代表长为 N 的信源序列携带的平均信息量。这就是信源序列无失真定长编码定理。

由等长编码定理可知，只要码字传输的信息量大于信源序列携带的信息量，总可以实现几乎无失真的编码，条件是所取的符号数 N 足够大。

例如，某单符号信源有 8 种等概率符号，信源熵最大值为

$$H(X) = \log 8 = 3(\text{bit/ 符号})$$

即该信源符号肯定可以用 $K=3$ 个二元码符号进行无失真编码。但是，当信源符号概率不相等时，若 $p(x_i)=\{0.4,0.18,0.1,0.1,0.07,0.06,0.05,0.04\}$，则信源熵为

$$H(X) = -\sum_{i=1}^{8} p_i \log_2 p_i = 2.55(\text{bit/ 符号}) \tag{5.2.8}$$

小于 3bit，用 $K=2.55$ 个二元码符号表示信源时，只有 $2^{2.55}=5.856$ 种可能码字，信源的 8 个符号中还有部分符号没有对应的码字，就只能用其他码字代替，因而引起差错。差错发生的可能性取决于这些没有对应码字的信源符号出现的概率。当 N 足够大时，有些符号序列发生的概率变得很小，使得差错概率达到足够小。根据切比雪夫不等式可推导得到等长编码的译码错误概率

$$P_\varepsilon \leqslant \frac{\sigma^2(X)}{N\varepsilon^2} \tag{5.2.9}$$

其中，

$$\varepsilon = \frac{1-\eta}{\eta} H_N(X) \tag{5.2.10}$$

且

$$\sigma^2(X) = E\{[I(x_i) - H(X)]^2\} \tag{5.2.11}$$

为信源符号的自方差，称作自信息方差。

当 $\sigma^2(X)$ 和 ε 均为定值时，只要 N 足够大，P_ε 可以小于任一整数 δ，即

$$\frac{\sigma^2(X)}{N\varepsilon^2} \leqslant \delta \tag{5.2.12}$$

此时要求信源序列长度必满足

$$N \geqslant \frac{\sigma^2(X)}{\varepsilon^2 \delta} \tag{5.2.13}$$

只要 δ 足够小，就可以几乎无差错地译码，当然代价是 N 变得更大。

无失真信源编码定理从理论上阐明了编码效率接近于 1 的理想编码器的存在性，它使输出符号的信息率与信源熵之比接近于1，但要在实际中实现，则要求信源符号序列的 N 非常大，进行统一编码才行，这往往是不现实的。

例 5.2.1　设离散无记忆信源概率空间为

$$\begin{bmatrix} X \\ P \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 \\ 0.4 & 0.18 & 0.1 & 0.1 & 0.07 & 0.06 & 0.05 & 0.04 \end{bmatrix}$$

信源熵为

$$H(X) = -\sum_{i=1}^{8} p_i \log_2 p_i = 2.55(\text{bit/符号})$$

自信息方差为

$$\begin{aligned} \sigma^2(X) &= E\{[I(x_i) - H(X)]^2\} = \sum_{i=1}^{8} p_i [-\log_2 p_i - H(X)]^2 \\ &= \sum_{i=1}^{8} p_i \{(\log_2 p_i)^2 + 2H(X)\log_2 p_i + [H(X)]^2\} \\ &= \sum_{i=1}^{8} p_i (\log_2 p_i)^2 + 2H(X)\sum_{i=1}^{8} p_i \log_2 p_i + [H(X)]^2 \sum_{i=1}^{8} p_i \\ &= \sum_{i=1}^{8} p_i (\log_2 p_i)^2 - [H(X)]^2 = 7.82 \end{aligned}$$

对信源符号采用定长二元编码，要求编码效率 $\eta = 90\%$，无记忆信源有 $H_N(X) = H(X)$，因此

$$\eta = \frac{H(X)}{H(X) + \varepsilon} \times 100\% = 90\%$$

可以得到 $\varepsilon = 0.28$。

如果要求译码错误概率 $\delta \leqslant 10^{-6}$，则

$$N \geqslant \frac{\sigma^2(X)}{\varepsilon^2 \delta} = 9.8 \times 10^7 \approx 10^8$$

由此可见，在对编码效率和译码错误概率的要求不是十分苛刻的情况下，就需要 $N=$

10^8 个信源符号一起进行编码，这对存储和处理技术的要求过高，目前还无法实现。

如果用 3bit 来对上述信源的 8 个符号进行定长二元编码，$N=1$，此时可实现译码无差错，但编码效率只有$(2.55/3)\times100\%=85\%$。因此，一般来说，当信源序列的长度 N 有限时，高传输效率的定长编码往往要引入一定的失真和译码错误。解决的办法是采用变长编码。

5.2.2 变长编码定理（香农第一定理）

在变长编码中，码长是变化的。对同一信源，其即时码或唯一可译码可以有许多种。究竟哪一种好呢？从高速传输信息的观点来考虑，当然希望选择由短的码符号组成的码字，就是用平均码长来作为选择准则。

1. 单个符号变长编码定理

若一离散无记忆信源的符号熵为 $H(X)$，每个信源符号用 m 进制码元进行变长编码，一定存在一种无失真编码方法，其码字平均长度 $\overline{K}$ 满足下面的不等式：

$$\frac{H(X)}{\log m}\leqslant\overline{K}<\frac{H(X)}{\log m}+1 \tag{5.2.14}$$

2. 离散平稳无记忆序列变长编码定理（香农第一定理）

对于平均符号熵为 $H_N(X)$的离散平稳无记忆信源，必存在一种无失真信源编码方法，使平均码长 $\overline{K}$ 满足下面的不等式：

$$\frac{H_N(X)}{\log m}\leqslant\overline{K}<\frac{H_N(X)}{\log m}+1 \tag{5.2.15}$$

上面的两个定理实际上是一样的，可以由第一个推导出第二个。设用 m 进制码元做变长编码，信源序列长度为 N 个符号，则该序列所对应的码字的平均长度 $\overline{K}_N$ 满足下面的不等式：

$$\frac{H(X^N)}{\log m}\leqslant\overline{K}_N<\frac{H(X^N)}{\log m}+\varepsilon \tag{5.2.16}$$

式中，ε 为任意小正数。式中利用了符号码长和序列码长的关系 $\overline{K}_N=N\overline{K}$。

式(5.2.15)也可以表示为

$$H_N(X)\leqslant\overline{R}<H_N(X)+\frac{\log m}{N} \tag{5.2.17}$$

式中，$\overline{R}=\overline{K}\log m$，为编码后每个码符号所能携带的最大信息量，是码符号的最大平均编码信息率。

可见，码符号的最大编码信息率需大于信源的熵（编码前每个信源符号携带的信息量）。式(5.2.17)中，当 N 足够大时，可使$\frac{\log m}{N}<\varepsilon$，故有

$$H_N(X)\leqslant\overline{R}<H_N(X)+\varepsilon \tag{5.2.18}$$

对于二元码，式(5.2.18)重写为

$$H_N(X)\leqslant\overline{K}<H_N(X)+\varepsilon \tag{5.2.19}$$

香农第一编码定理给出了码字平均长度的下界和上界，但并不是说大于这个上界就不能构成唯一可译码，而是因为编码时总是希望 $\overline{K}$ 尽可能短。定理说明当平均码长小于上界时，唯一可译码也存在。也就是说，定理给出的是最佳码的最短平均码长，并指出这个最短的平均码长与信源熵是有关的。

变长编码的编码效率为

$$\eta = \frac{H_N(X)}{\overline{K}} > \frac{H_N(X)}{H_N(X) + \frac{\log m}{N}} \tag{5.2.20}$$

或者

$$\eta = \frac{H(X^N)}{\overline{K}_N}$$

例 5.2.2　设离散无记忆信源的概率空间为

$$\begin{bmatrix} X \\ P \end{bmatrix} = \begin{bmatrix} x_1 & x_2 \\ \frac{3}{4} & \frac{1}{4} \end{bmatrix}$$

其信源熵为

$$H(X) = \frac{1}{4}\log_2 4 + \frac{3}{4}\log_2 \frac{4}{3} = 0.811(\text{bit/符号})$$

若用二元定长编码(0,1)来构造一个即时码：$x_1 \to 0$，$x_2 \to 1$，这时平均码长为

$$\overline{K} = 1(\text{二元码符号/信源符号})$$

编码效率为

$$\eta = \frac{H(X)}{\overline{K}} = 0.811$$

输出的编码信息率为

$$R = 0.811(\text{bit/符号})$$

再对长度为 2 的信源序列进行变长编码，其即时码如表 5.2.1 所示。

表 5.2.1　长度为 2 的信源序列对应的即时码

序　列	序列概率	即时码	序　列	序列概率	即时码
x_1x_1	9/16	0	x_2x_1	3/16	110
x_1x_2	3/16	10	x_2x_2	1/16	111

这个码的平均长度为

$$\overline{K_2} = \frac{9}{16}\times 1 + \frac{3}{16}\times 2 + \frac{3}{16}\times 3 + \frac{1}{16}\times 3 = \frac{27}{16}(\text{二元码符号/信源序列})$$

单个信源符号所编码字的平均码长为

$$\overline{K} = \frac{\overline{K}_2}{2} = \frac{27}{32}(\text{二元码符号/信源符号})$$

其编码效率和编码信息率分别为

$$\eta_2 = \frac{32 \times 0.811}{27} = 0.961$$

$$R_2 = 0.961(\text{bit/码符号})$$

这说明，虽然编码复杂了，但信息传输率和效率有了提高。同样，可以求得信源序列长度增加到 3 和 4 时，进行变长编码所得的编码效率和信息传输速率分别为

$$\eta_3 = 0.985, \quad R_3 = 0.985(\text{bit/码符号})$$
$$\eta_4 = 0.991, \quad R_4 = 0.991(\text{bit/码符号})$$

如果对这一信源采用定长二元码编码，要求编码效率达到96%，允许译码错误概率 $\delta \leqslant 10^{-5}$，则可以算出自信息方差为

$$\sigma^2(X) = \sum_{i=1}^{2} p_i (\log p_i)^2 - [H(X)]^2 = 0.4715$$

ε 为

$$\varepsilon = \frac{H(X)}{\eta} - H(X) = \frac{0.811}{0.96} - 0.811 = \frac{0.811 \times 0.04}{0.96}$$

需要的信源序列长度为

$$L \geqslant \frac{\sigma^2(X)}{\varepsilon^2 \delta} = 4.13 \times 10^7$$

可以看出，使用定长编码时，为了使编码效率较高(96%)，需要对非常长的信源序列进行编码，且总存在译码差错。而使用变长编码，使编码效率达到96%，只要 $L=2$ 就行了，且可以实现无失真编码。当然，变长编码的译码相对来说要复杂一些。

5.2.3 最佳变长编码及 MATLAB 实现

香农第一定理给出了信源熵与编码后的平均码长之间的关系，同时也指出可以通过编码使平均码长达到极限值，因此，香农第一定理是一个极限定理。但定理中并没有告知如何来构造这种码。

根据最佳码的编码思想：选择每个码字长度 k_i 满足

$$k_i = \left\lfloor \log \frac{1}{p(x_i)} \right\rfloor \tag{5.2.21}$$

式中，$\lfloor \cdot \rfloor$ 表示向上取整。

由单符号变长编码定理可知，这样选择的码长一定满足克拉夫特不等式，所以一定存在唯一可译码。然后，按照这个码长 k_i，用树图法就可以编出相应的一组码(即时码)。

式(5.2.21)证明：

根据单符号变长编码定理，$\overline{K} \geqslant \frac{H(X)}{\log m}$，即

$$H(X) - \overline{K}\log m \leqslant 0$$

展开式为

$$\begin{aligned} H(X) - \overline{K}\log m &= -\sum_{i=1}^{n} p(x_i)\log p(x_i) - \log m \sum_{i=1}^{n} p(x_i)k_i \\ &= -\sum_{i=1}^{n} p(x_i)\log p(x_i) + \sum_{i=1}^{n} p(x_i)\log m^{-k_i} \\ &\xlongequal{\text{詹姆斯不等式}} \sum_{i=1}^{n} p(x_i)\log \frac{m^{-k_i}}{p(x_i)} \leqslant \log \sum_{i=1}^{n} p(x_i)\frac{m^{-k_i}}{p(x_i)} = \log \sum_{i=1}^{n} m^{-k_i} \end{aligned}$$

因为总可以找到一种唯一可译码，其码长满足克拉夫特不等式，所以

$$H(X) - \overline{K}\log m \leqslant \log \sum_{i=1}^{n} m^{-k_i} \leqslant 0$$

得

$$\overline{K} \geqslant \frac{H(X)}{\log m}$$

上式成立的充要条件是

$$\frac{m^{-k_i}}{p(x_i)} = 1, \quad \text{对所有 } i$$

即

$$p(x_i) = m^{-k_i}$$

两边取对数，得

$$k_i = -\frac{\log p(x_i)}{\log m} = = -\log_m p(x_i)$$

证明完毕。

下面将介绍三种无失真信源编码方法：香农编码、费诺编码以及哈夫曼编码。这三种码的平均码长都比较短。

1. 香农编码方法

因为平均码长是各个码字长度的概率平均，可以想象，应该使出现概率大的信源符号编码后码长尽量短一些，出现概率小的信源符号编码后码长长一些，保证平均码长较短。三种编码方法的出发点都是如此，这也是变长编码的思想。

香农编码严格意义上来说不是最佳编码。

香农编码是采用信源符号的累计概率分布函数来分配码字的。

设信源符号集 $X=\{x_1, x_2, \cdots, x_n\}$，并设所有的 $P(x)>0$，则香农编码方法如下：

(1) 将信源消息符号按其出现的概率大小依次排列：$p(x_1) \geqslant p(x_2) \geqslant \cdots \geqslant p(x_n)$；

(2) 确定满足下列不等式的整数码长 k_i：

$$-\log p(x_i) \leqslant k_i < -\log p(x_i) + 1$$

(3) 为了编成唯一可译码，计算第 i 个消息的累加概率：

$$P_i = \sum_{k=1}^{i-1} p(x_k)$$

(4) 将累加概率 P_i 变换成二进制数；

(5) 取 P_i 二进制数的小数点后 k_i 位即为该消息符号的二进制码字。

可以证明，这样得到的编码一定是唯一可译码，且码长比较短，接近于最佳编码。也可以不对信源消息符号按概率大小排列，这时香农编码方法如下：

(1) 求出修正累计概率分布函数为

$$P_i = \sum_{k=1}^{i-1} p(x_k) + \frac{1}{2} p(x_i) x_k, \quad x_i \in X$$

(2) 确定满足下式的码长：

$$k_i = \left\lceil \log \frac{1}{p(x_i)} \right\rceil + 1$$

(3) 将修正累加概率 P_i 变换成二进制数。

(4) 取 P_i 二进制小数点后 k_i 位即为该消息符号的二进制编码。

例 5.2.3　设信源共由 7 个符号组成，其概率和香农编码过程如表 5.2.2 所示。

表 5.2.2　香农编码过程

信源符号 x_i	符号概率 $p(x_i)$	累加概率 P_i	$-\log p(x_i)$	码字长度 k_i	码　　字
x_1	0.20	0	2.34	3	000
x_2	0.19	0.2	2.41	3	001
x_3	0.18	0.39	2.48	3	011
x_4	0.17	0.57	2.56	3	100
x_5	0.15	0.74	2.74	3	101
x_6	0.10	0.89	3.34	4	1110
x_7	0.01	0.99	6.66	7	1 111 110

以 $i=4$ 为例，第 4 个信源符号所编码的码长为

$$-\log 0.17 \leqslant k_4 < -\log_2 0.17 + 1$$

即

$$2.56 \leqslant k_4 < 3.56$$

取

$$k_4 = 3$$

第 4 个信源符号累加概率 $P_i=0.57$，变成二进制数，为 0.1001…，取 3 位，得第 4 个信源符号所编码字为：100。

小数变二进制数的方法：用 P_i 乘以 2，如果整数部分有进位，则小数点后第一位为 1，否则为 0，将其小数部分再做同样的处理，得到小数点后的第二位，依此类推，直到得到满足要求的位数，或者没有小数部分为止。

例如，现在 $P_i=0.57$，乘以 2 为 1.14，整数部分有进位，所以小数点后第一位为 1，将小数部分即 0.14 再乘以 2，得 0.28，没有整数进位，所以小数点后第二位为 0，依此类推，可得到其对应的二进制数为 0.1001…。

由表 5.2.2 可以看出，编码所得的码字没有相同的，所以是非奇异码，也没有一个码字是其他码字的前缀，所以是即时码、唯一可译码。

香农编码的 MATLAB 程序：主程序 shanoncode.m，子程序 deczbin.m。

```
%%% 主程序 shanoncode.m
clc
clear all
n = input('输入信源符号个数 n = ');
p = zeros(1,n);
while(1)
for i = 1:n
        fprintf('请输入第 % 个符号的概率:',i);
p(1,i) = input('p = ');
end
if sum(p)~ = 1
        disp('输入概率不符合概率分布')
continue
else
        y = fliplr(sort(p));
        d = zeros(n,4);
D(:,1) = y;
```

```
for i = 2:n
            D(1,2) = 0;                          % 令第一行第二行的元素为 0
            D(i,2) = D(i - 1,1) + D(i - 1,2);    % 第二行其余的元素用此式求得,即为累加概率
end
for i = 1:n
            D(i,3) = - log2(D(i,1));             % 求第三列的元素
            D(i,4) = ceil(D(i,3));               % 求第四列的元素,对 D(i,3)向无穷方向取最小正整数
end
        D
        A = D(:,2)';                             % 取出 D 中第二列元素
        B = D(:,4)';
for j = 1:n
                C = deczbin(A(j),B(j))           % 生成码字
end
end
break
end

function [C] = deczbin(A,B) % 对累加概率求二进制的函数
C = zeros(1,B); % 生成零矩阵用于存储生成的二进制数,对二进制的每一位进行操作 temp = A;
 % temp 赋值
for i = 1:B              % 累加概率转化为二进制,循环求二进制的每一位,B 控制生成二进制的位数
temp = temp * 2;
if temp > 1
temp = temp - 1;
C(1,i) = 1;
else
C(1,i) = 0;
end
end
```

程序运行结果：

```
输入信源符号个数 n = 7
请输入第 1 个符号的概率:p = 0.2
请输入第 2 个符号的概率:p = 0.19
请输入第 3 个符号的概率:p = 0.18
请输入第 4 个符号的概率:p = 0.17
请输入第 5 个符号的概率:p = 0.15
请输入第 6 个符号的概率:p = 0.1
请输入第 7 个符号的概率:p = 0.01
D =
        0.2000         0   2.3219   3.0000
        0.1900    0.2000   2.3959   3.0000
        0.1800    0.3900   2.4739   3.0000
        0.1700    0.5700   2.5564   3.0000
        0.1500    0.7400   2.7370   3.0000
        0.1000    0.8900   3.3219   4.0000
        0.0100    0.9900   6.6439   7.0000
```

```
C =   0   0   0
C =   0   0   1
C =   0   1   1
C =   1   0   0
C =   1   0   1
C =   1   1   1   0
C =   1   1   1   1   1   1   0
```

例 5.2.3 中香农编码的性能可以用平均码长和平均信息传输率、编码效率评价。

平均码长为

$$\overline{K} = \sum_{i=1}^{7} p(x_i)k_i = 3.14(\text{二元码符号 / 信源符号})$$

平均信息传输率为

$$R = \frac{H(X)}{\overline{K}} = \frac{2.61}{3.14} = 0.831(\text{bit/ 码符号})$$

编码效率为

$$\eta = \frac{H(X)}{\overline{K}} = 83.1\%$$

压缩之前 7 个符号，平均每个符号需要 3bit 表示，经香农编码压缩之后的平均码字长度为 3.14，因此压缩比为

$$P_r = \frac{3-3.14}{3} \times 100\% = -4.67\%$$

香农编码的效率不高，本例中压缩比为负值，并没有对信源进行压缩，实用意义不大，但对其他编码方法有很好的理论指导意义。

2. 费诺编码方法

费诺编码也不是最佳编码方法，但有时可以得到最佳编码。

费诺编码方法如下：首先，将信源符号以概率递减的次序排列起来，将排列好的信源符号分成两组，使每一组的概率之和相接近，并各赋予一个二元码符号“0”或者“1”；然后，将每一组的信源符号再分成两组，使每一小组的符号概率之和也接近相等，并又分别赋予一个二元码符号。依此下去，直到每一个小组只剩下一个信源符号为止。这样，信源符号所对应的码符号序列则为编得的码字。

例 5.2.4 给定信源的费诺编码过程如表 5.2.3 所示。

表 5.2.3 费诺编码过程

消息符号	符号概率	第一次分组	第二次分组	第三次分组	第四次分组	码字	码长
x_1	0.20	0	0			00	2
x_2	0.19		1	0		010	3
x_3	0.18			1		011	3
x_4	0.17	1	0			10	2
x_5	0.15		1	0		110	3
x_6	0.10			1	0	1110	4
x_7	0.01				1	1111	4

费诺编码的 MATLAB 程序：

```
% fanocode.m
clc;
clear all;
fprintf('.....费诺编码程序.....\n');
fprintf('请输入信源符号的个数:');
N = input('N = '); % 输入信源符号的个数
s = 0;l = 0;H = 0;
for i = 1:N
    fprintf('请输入第 %d 个符号的概率:',i);
    p(i) = input('p = ');                    % 输入信源符号概率分布向量,0 < p(i)< 1
if p(i)<= 0||p(i)>= 1
   error('请注意 P 的范围是 0 < P < 1')
end
s = s + p(i);
H = H + ( - p(i) * log2(p(i)));              % 计算信源信息熵
end
if(s~ = 1)
error('信源符号概率和不等 1')
end
tic;
for i = 1:N - 1                              % 按概率分布大小对信源排序
for j = i + 1:N
if p(i)< p(j)
m = p(j);p(j) = p(i);p(i) = m;
end
end
end
x = f1(1,N,p,1);
for i = 1:N                                  % 计算平均码长
L(i) = length(find(x(i,:)));
    l = l + p(i) * L(i);
end
n = H/l;                                     % 计算编码效率
fprintf('编码后所得码字:\n');
disp(x)                                      % 显示按概率降序排列的码字
fprintf('平均码长:K = \n');
disp(l)                                      % 显示平均码长
fprintf('信息熵:H(X) = \n');disp(H)          % 显示信息熵
fprintf('编码效率:η = \n');disp(n)           % 显示编码效率
```

程序运行结果：

```
..... 费诺编码程序 .....
请输入信源符号的个数:N = 7
```

```
请输入第 1 个符号的概率:p = 0.2
请输入第 2 个符号的概率:p = 0.19
请输入第 3 个符号的概率:p = 0.18
请输入第 4 个符号的概率:p = 0.17
请输入第 5 个符号的概率:p = 0.15
请输入第 6 个符号的概率:p = 0.1
请输入第 7 个符号的概率:p = 0.01
编码后所得码字:
00 010 011 10 110 1110 1111
平均码长:K =  2.7400
信息熵:H(X) =  2.6087
编码效率:η =  0.9521
```

由表 5.2.3 可以求得，该费诺编码的平均码长为

$$\overline{K} = \sum_{i=1}^{7} p(x_i)k_i = 2.74(\text{二元码符号 / 信源符号})$$

信息传输率为

$$R = \frac{H(X)}{\overline{K}} = \frac{2.61}{2.74} = 0.953(\text{bit/ 码符号})$$

编码效率为

$$\eta = 95.3\%$$

压缩之前 7 个符号，平均每个符号需要 3bit 表示，经费诺编码压缩之后的平均码字长度为 2.74，因此压缩比为

$$P_r = \frac{3-2.74}{3} \times 100\% = 8.67\%$$

3. 哈夫曼编码方法

1952 年哈夫曼提出了一种构造最佳编码的方法。它是一种最佳的逐个符号的编码方法。其编码步骤如下：

(1) 将 n 个信源符号按概率分布的大小，以递减次序排列起来，设

$$p(x_1) \geqslant p(x_2) \geqslant \cdots \geqslant p(x_n)$$

(2) 用“0”和“1”码符号分别代表概率最小的两个信源符号，并将这两个概率最小的符号合并成一个符号，合并的符号概率为两个符号概率之和，从而得到只包含 $n-1$ 个符号的新信源，称为缩减信源。

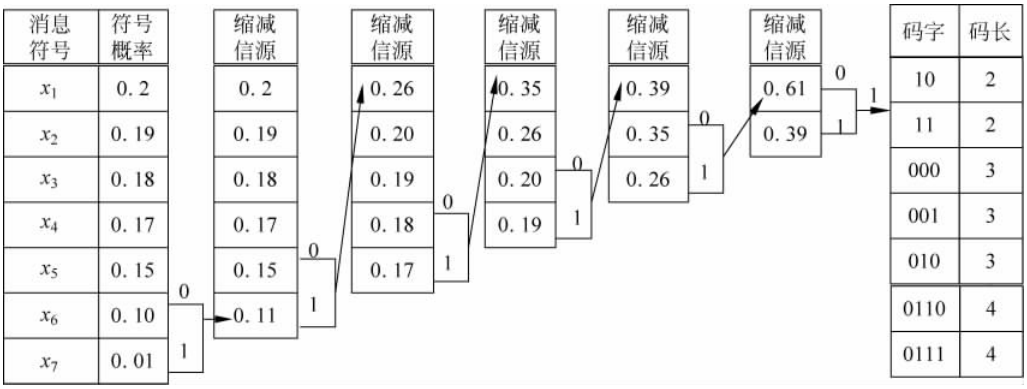
(3) 把缩减信源的符号仍旧按概率大小以递减次序排列，再将其概率最小的两个信源符号分别用“0”和“1”表示，并将其合并成一个符号，概率为两符号概率之和，这样又形成了 $n-2$ 个符号的缩减信源。

(4) 依此继续下去，直至信源只剩下两个符号为止。将这最后两个信源符号分别用“0”和“1”表示。

(5) 从最后一级缩减信源开始，向前返回，就得出各信源符号所对应的码符号序列，即对应的码字。

例 5.2.5　给定信源的哈夫曼编码过程如表 5.2.4 所示。

表 5.2.4　哈夫曼编码过程



哈夫曼编码也可以用树图法实现，如图 5.2.1 所示。

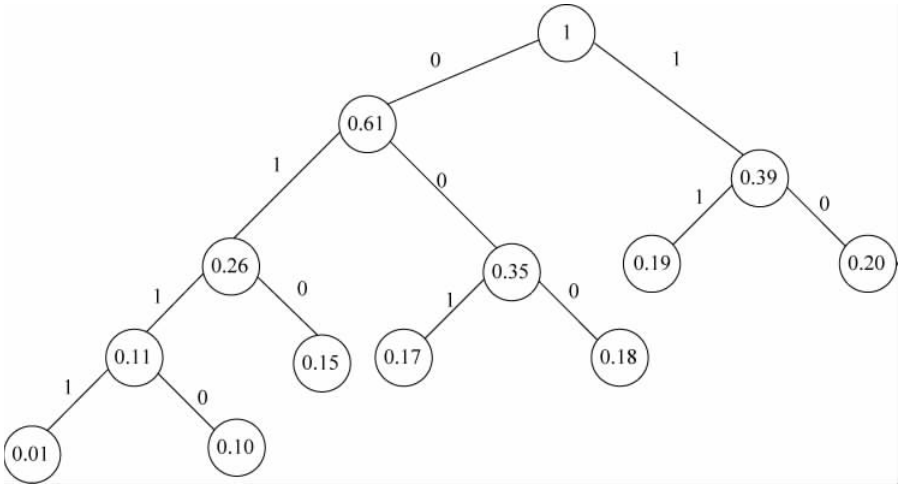


图 5.2.1　树图法哈夫曼编码过程

哈夫曼编码的 MATLAB 程序：

```
% huffmancode.m
clc;
clear all;
n = input('Enter the number of source symbols:');
A = zeros(1,n);
for i = 1:n
A(1,i) = input( 'input source symbol probability:');
end
if sum(A)~ = 1
disp('Input probability is not in conformity with the probability distribution')
break
else A = fliplr(sort(A));                    % 按降序排列
T = A;
[m,n] = size(A)
B = zeros(n,n - 1);                          % 空的编码表(矩阵)
```

```
for i = 1:n
    B(i,1) = T(i);                              % 生成编码表的第一列
end
r = B(i,1) + B(i - 1,1);                        % 最后两个元素相加
T(n - 1) = r;
T(n) = 0;
T = fliplr(sort(T));
t = n - 1;
for j = 2:n - 1                                 % 生成编码表的其他各列
for i = 1:t
B(i,j) = T(i);
end
        K = find(T == r);
        B(n,j) = K(end);        % 从第二列开始,每列的最后一个元素记录特征元素在该列的位置
        r = (B(t - 1,j) + B(t,j));              % 最后两个元素相加
T(t - 1) = r;
T(t) = 0;
        T = fliplr(sort(T));
        t = t - 1;
end
B;                                              % 输出编码表
END1 = sym('[0,1]');                            % 给最后一列的元素编码
END = END1;
t = 3;
d = 1;
for j = n - 2: - 1:1                            % 从倒数第二列开始依次对各列元素编码
for i = 1:t - 2
if i > 1 & B(i,j) == B(i - 1,j)
           d = d + 1;
else
            d = 1;
end
B(B(n,j + 1),j + 1) = - 1;
temp = B(:,j + 1);
       x = find(temp == B(i,j));
END(i) = END1(x(d));
end
    y = B(n,j + 1);
END(t - 1) = [char(END1(y)),'0'];
END(t) = [char(END1(y)),'1'];
    t = t + 1;
    END1 = END;
end
    A                                           % 排序后的原概率序列
   END                                          % 编码结果
for i = 1:n
    [a,b] = size(char(END(i)));
L(i) = b;
end
avlen = sum(L. * A)                             % 平均码长
H1 = log2(A);
H = - A * (H1')                                 % 熵
P = H/avlen                                     % 编码效率
end
```

程序运行结果为

```
Enter the number of source symbols:7
input source symbol probability:0.2
input source symbol probability:0.19
input source symbol probability:0.18
input source symbol probability:0.17
input source symbol probability:0.15
input source symbol probability:0.1
input source symbol probability:0.01
m = 1
n = 7
A = 0.2000 0.1900 0.1800 0.1700 0.1500 0.1000 0.0100
END = [10, 11, 000, 001, 010, 0110, 0111]
avlen = 2.7200
H = 2.6087
P = 0.9591
```

哈夫曼编码还可以调用 huffmanenco 和 huffmandeco 函数实现编码和译码：

```
ENCO = HUFFMANENCO(SIG, DICT)———HUFFMANENCO Encode an input signal using Huffman coding
       algorithm.
DECO = HUFFMANDECO(COMP, DICT)———HUFFMANDECO Huffman decoder.
```

上例的 MATLAB 程序如下：

```
Clear all;
clc;
symbols = [1:7];
p = [0.2 0.19 0.18 0.17 0.15 0.1 0.01];
entropy = - p * log2(p');
[dict,avg_len] = huffmandict(symbols,p)
eta = entropy/avg_len;
source_len = 1000;
seq = randsrc(1,source_len,[1 2 3 4 5 6 7;0.2 0.19 0.18 0.17 0.15 0.1 0.01]);
comp = huffmanenco(seq,dict);
[s,codeseq_len] = size(comp)
actual_avg_len = codeseq_len/source_len
actual_eta = entropy/actual_avg_len
dcomp = huffmandeco(comp,dict);
```

输出：

```
dict =  % 编码映射表,第 1 列为信源序号,第 1 列对应码字长度
  [1]  [1x2 double][1,0]
  [2]  [1x2 double] [1,1]
  [3]  [1x3 double][0,0,0]
  [4]  [1x3 double][0,0,1]
  [5]  [1x3 double][0,1,0]
  [6]  [1x4 double] [0,1,1,0]
  [7]  [1x4 double][0,1,1,1]
avg_len = 2.7200                          % 平均码长
codeseq_len = 2750                        % 码序列长度
```

```
actual_avg_len = 2.7500                    % 实际平均码长
actual_eta = 0.9486                        % 实际平均码长
```

平均码长为

$$\overline{K} = \sum_{i=1}^{7} p(x_i)k_i = 2.72(\text{二元码符号 / 信源符号})$$

信息传输率为

$$R = \frac{H(X)}{\overline{K}} = \frac{2.61}{2.72} = 0.9596(\text{bit/ 码符号})$$

压缩之前 7 个符号，平均每个符号需要 3bit 表示，经哈夫曼编码压缩之后的平均码字长度为 2.72，因此压缩比为

$$P_r = \frac{3-2.72}{3} \times 100\% = 9.33\%$$

与香农编码、费诺编码相比，哈夫曼编码的平均码长较短，编码效率和压缩比较高。

从表 5.2.4 可以看出，哈夫曼编码方法得到的码一定是即时码。因为这种编码方法不会使任一码字的前缀为码字。这一点在用码树形式表示时看得更清楚。图 5.2.1 是用码树形式进行哈夫曼编码的过程，由于代表信源符号的节点都是终端节点，因此其编码不可能是其他终端节点对应的码字的前缀。

另外，由于哈夫曼编码总是把概率大的符号安排在离根节点近的终端节点，所以其码长比较小，因此得到的编码整体平均码长就比较小。

哈夫曼编码得到的码不是唯一的，因为每次对缩减信源中两个概率最小的符号编码时，“0”和“1”的安排是任意的。另外，当两个符号的概率相同时，排列的次序也是随意的，所以可能导致不同的编码结果，但最后的平均码长一定是一样的。在这种情况下，怎么样来判断一个码的好坏呢？可以引进码字长度 k_i 偏离平均长度 $\overline{K}$ 的方差，即码方差 σ^2：

$$\sigma^2 = E[(k_i - \overline{K})^2] = \sum_{i=1}^{n} p(x_i)\,(k_i - \overline{K})^2$$

哈夫曼编码时，一般将合并的概率放在上面，这样可获得较小的码方差。

例 5.2.6 对信源{0.4 0.2 0.2 0.1 0.1}进行哈夫曼编码的过程如表 5.2.5 和表 5.2.6 所示。两个表中合并符号概率与原信源符号概率相同时，表 5.2.5 将合并信源符号概率排在上面，表 5.2.6 将合并信源符号概率放在下面。

表 5.2.5 哈夫曼编码过程 1

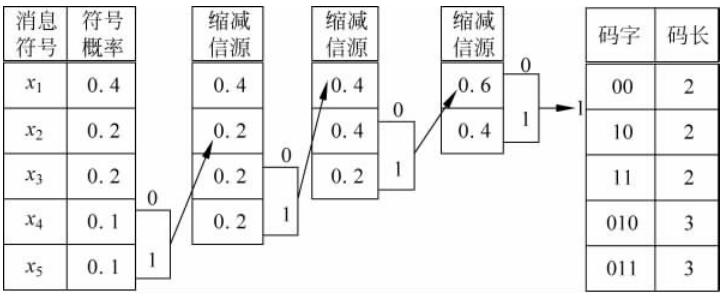


表 5.2.6　哈夫曼编码过程 2

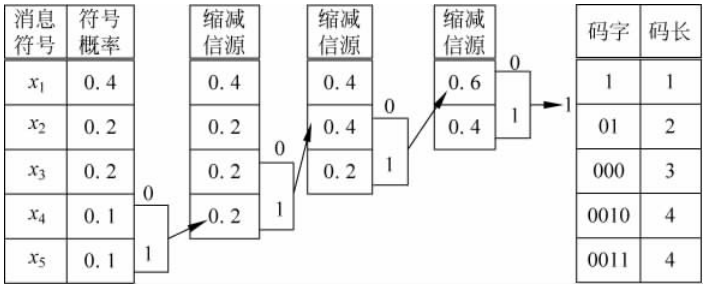


表 5.2.5 所编哈夫曼编码字的平均码长和码方差分别为

$$\overline{K} = 0.4\times 2 + 0.2\times 2 + 0.2\times 2 + 0.1\times 3 + 0.1\times 3 = 2.2(\text{二元码符号 / 信源符号})$$

$$\sigma_k^2 = 0.4\times 0.2^2 + 0.2\times 0.2^2 + 0.2\times 0.2^2 + 0.1\times 0.8^2 + 0.1\times 0.8^2 = 0.16$$

表 5.2.6 所编哈夫曼编码字的平均码长和码方差分别为

$$\overline{K} = 0.4\times 1 + 0.2\times 2 + 0.2\times 3 + 0.1\times 4 + 0.1\times 4 = 2.2(\text{二元码符号 / 信源符号})$$

$$\sigma_k^2 = 0.4\times 1.2^2 + 0.2\times 0.2^2 + 0.2\times 0.8^2 + 0.1\times 1.8^2 + 0.1\times 1.8^2 = 1.36$$

可见，将合并概率放在同概率信源符号的上面能提供较小的码方差。因此哈夫曼编码中应将合并概率放在同概率信源符号的上面，以保证较好的编码性能。

从以上的编码实例中可以看出，哈夫曼编码具有以下三个特点：

(1) 哈夫曼编码方法保证了概率大的符号对应于短码，概率小的符号对应于长码，且短码得到充分利用。

(2) 每次缩减信源的最后两个码字总是最后一位码元不同，前面各位码元相同。

(3) 每次缩减信源的最长两个码字有相同的码长。

这三个特点保证了所得的哈夫曼编码一定是最佳编码。

5.3　实用的无失真信源编码方法

无失真信源编码主要用于离散信源或数字信号，如文字、数字、数据等。它们要求无失真地数据压缩，且无失真地可逆恢复。目的是增加单位时间内传送的信息量，即提高信息传输的效率。香农第一定理给出了无失真信源压缩的理论极限，由定理可知，信源的熵是信源进行无失真编码的理论极限值，存在最佳的无失真信源编码方法使编码后信源的信息传输率任意接近信源的熵。香农第一定理并没有告知如何找到最佳的无失真信源编码方法，但是，从降低信源冗余度的途径考虑，只要寻找到去除信源符号相关性或者改变信源符号概率分布不均匀性的方法和手段，就能找到最佳无失真信源编码的具体方法和实用码型。

5.2.3 节已经从香农第一定理内容中信源熵与平均码长的关系推导出码长与信源各符号自信息量的关系，讨论了香农编码、费诺编码和哈夫曼编码的原理和方法。它们都是当信源的统计特性一定时，能达到或接近无失真信源压缩极限的典型编码方法。上述编码主要适用于多元信源和无记忆信源。当信源给定时，哈夫曼编码是最佳编码，它在实际中已有所应用，但它仍存在一些分组码所具备的缺点。例如，概率特性必须得到精确的测定，若略有

变化，还需要更新码表；对于二元信源，必须对其 N 次扩展信源进行编码，才能取得好的效果，但当合并的符号数不大时编码效率提高不多，特别是二元相关信源；采用哈夫曼编码等分组编码方法会使编译码设备变得复杂，而且没有充分利用扩展信源符号间的相关性，导致这些编码方法的编码效率提高不多。

游程编码、算术编码和字典码是针对相关信源的有效无失真信源编码方法，属于非分组码，尤其适用于二元相关信源。

5.3.1 游程编码及其 MATLAB 实现

对于二元相关信源，输出的信源符号序列中往往会出现多个“0”或“1”符号，游程编码尤其适合对这类信源的编码。游程编码已在图文传真、图像传输等实际通信工程技术中得到应用，实际工程技术中也常与其他变长编码方法进行联合编码，如哈夫曼编码、MH 编码等，能进一步压缩信源，提高传输效率。

游程编码是无失真信源编码，它能够把二元序列编成多元码。

在二元序列中，只有“0”和“1”两种码元，把连续出现的“0”称为“0”游程，连续出现的“1”称为“1”游程。连续出现“0”或者“1”码元的个数称为游程长度(Run-Length，RL)，“0”游程长度记为 $L(0)$，“1”游程长度记为 $L(1)$。这样，一个二元序列可以转换成游程序列，游程序列中游程长度一般都用自然数标记。

例如，二元序列 000111110000000111110001000000 可以变换成多元序列 3 574 316。

若规定游程必须从“0”游程开始，第一个游程是“0”游程，则第二个游程必为“1”游程，第三个又是“0”游程……。上述变换是可逆、无失真的。如果连“0”或连“1”非常多，则可以达到信源压缩的目的。

一般传输信道为二元离散信道，游程序列中的各游程长度必须变换成二元码序列。等长游程编码就是将游程长度编成二进制的自然数。上例中，$\max[L(0),L(1)]=7$，则用三位二进制码来编码，上例的游程序列对应的码序列为

011　101　111　100　011　001　110

可见，信源序列由原来的 29 个二元符号，编成 21 个二元码符号，信源序列得到压缩，游程长度越长及长游程较多时压缩效果越好。

为提高压缩比，变长游程编码游程映射变换后常和哈夫曼编码或 MH 编码结合。联合编码过程如下：

首先对游程映射的各多元序列，测定 $L(0)$ 和 $L(1)$ 的概率分布，以游程长度为元素，构造一个新的多元信源，一般 $L(0)$ 和 $L(1)$ 应建立各自的信源；然后对 $L(0)$ 构成的多元信源进行哈夫曼编码，得到不同游程长度映射的码字，从而将游程序列变换成码字序列；同样，对 $L(1)$ 构成的多元信源进行哈夫曼编码，这样就可以得到 $L(0)$ 信源与 $L(1)$ 信源的码字和码表，而且两码表中的码字一般是不同的。在上述编码过程中，考虑到编码的复杂度以及长游程概率随游程长度减小的特点，对较大的游程长度，采用截断处理的方法，将大于一定长度的长游程统一用等长码编码。

游程编码一般不直接应用于多灰度值的图像，因其压缩比很低，但比较适合于黑白图文、图片等二值图像的编码。游程编码与其他一些编码方法的混合使用，能达到较好的压缩效果。如在彩色静止图像压缩的国际标准化算法 JPEG 中，采用了游程编码和离散余弦变

换(Discrete Cosine Transform,DCT)及哈夫曼编码的联合编码方法。

例 5.3.1　二值图像游程编码算法的 MATLAB 实现。

```
clc
clear all
image1 = imread('C:\Program Files\MATLAB71\work\1\girl.jpg'); % 读入图像
figure(1);
imshow(image1);                                    % 显示原图像
% 以下程序是将原图像转换为二值图像
image2 = image1(:);                                % 将原始图像写成一维的数据并设为 image2
image2length = length(image2);                     % 计算 image2 的长度
for i = 1:1:image2length                           % for 循环,目的在于转换为二值图像
Ifimage2(i)>= 127
   image2(i) = 255;
else
   image2(i) = 0;
   end
end
image3 = reshape(image2,146,122);                  % 重建二维数组图像,并设为 image3
figure(2);
imshow(image3);                                    % 以下程序为对原图像进行游程编码,压缩
X = image3(:);                                     % 令 X 为新建的二值图像的一维数据组
x = 1:1:length(X);                                 % 显示游程编码之前的图像数据
figure(3);
plot(x,X(x));
j = 1;
image4(1) = 1;
for z = 1:1:(length(X) - 1)                        % 游程编码程序段
ifX(z) == X(z + 1)
  image4(j) = image4(j) + 1;
else
  data(j) = X(z);                                  % data(j)代表相应的像素数据
  j = j + 1;
  image4(j) = 1;
  end
end
data(j) = X(length(X));                            % 最后一个像素数据赋给 data
image4length = length(image4);
y = 1:1:image4length;
figure(4);
plot(y,image4(y));
PR = (image2length - image4length)/ (image2length; % 压缩比
l = 1;
for m = 1:image4length
  for n = 1:1:image4(m);
  rec_image(1) = data(m);
   l = l + 1;
   end
  end
   u = 1:1:length(rec_image);
  figure(5)
  plot(u,rec_image(u));
```

二值图像游程编码算法的 MATLAB 实现所涉及的原图像、原图像转换的二值图像、二

值图像灰度数据以及游程编码结果如图 5.3.1～图 5.3.4 所示。

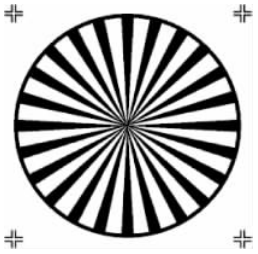


图 5.3.1　原图像

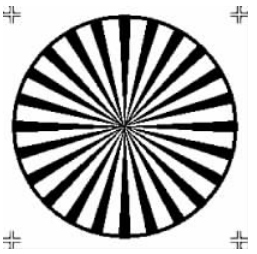


图 5.3.2　原图像转换为二值图像

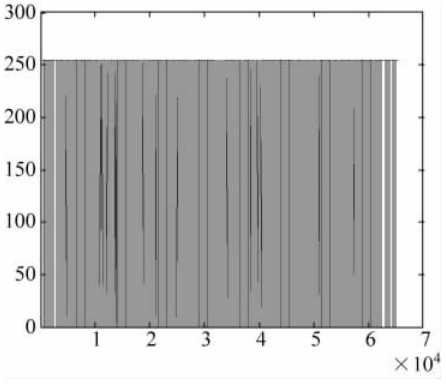


图 5.3.3　二值图像灰度数据

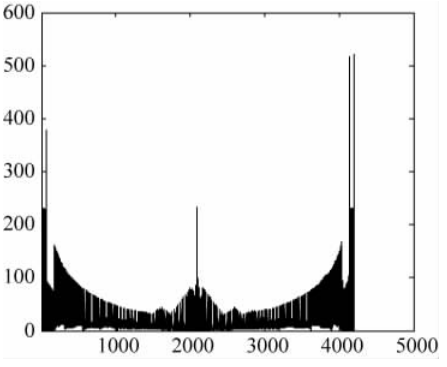


图 5.3.4　游程编码结果

仿真结果显示，压缩比 PR＝93.58％。

5.3.2　算术编码及其 MATLAB 实现

算术编码也是一种无失真信源编码方法。

前面讨论的无失真信源编码方法，都是针对单个信源符号的编码，当信源符号之间有相关性时，这些编码方法由于没有考虑到符号之间的相关性，因此编码效率就不可能很高。解决的办法是对信源序列进行编码，编码效率随序列长度增大而提高。但序列中符号之间的相关性以及序列之间的相关性无法考虑，无法真正满足信源编码的匹配原则。比如，某个符号的概率为 0.8，该符号只需要－log0.8＝0.322 位二元码符号编码，但香农编码等最佳变长编码会为其分配一位 0 或一位 1 的码字。是否存在与该符号概率相匹配的编码呢？

为了解决这个问题，需要跳出分组码的局限，研究非分组码。算术编码就是一种非分组编码方法。其基本思路：从全序列出发，将不同的信源序列的概率映射到[0,1]区间上，使每个序列对应区间上的一点，即把区间[0,1]分成许多互不重叠的小区间，不同的信源序列对应不同的小区间，每个小区间的长度等于某一序列的概率。在每个小区间内取一个二进制小数用作码字，其长度可与该序列的概率匹配，达到高效率编码的目的。可以证明，只要这些小区间互不重叠，就可以编得即时码。

可见，算术编码的主要编码方法就是计算信源符号序列所对应的小区间。下面将讨论如何找出信源符号序列所对应的区间。

设信源符号集 $A=\{a_1,a_2,\cdots,a_q\}$，其相应的概率分布为 $p(a_i)$，$p(a_i)>0(i=1,2,\cdots,q)$。定义信源符号的累积分布函数为

$$F(a_k)=\sum_{i=1}^{k-1}p(a_i)$$

则

$$F(a_1)=0,\quad F(a_2)=p(a_1),\quad F(a_3)=p(a_1)+p(a_2),\cdots$$

对二元序列，有

$$F(0)=0,F(1)=p(0)$$

现在，来计算二元信源序列 s 的累积分布函数。只讨论二元无记忆信源，结果可推广到一般情况。

(1) 初始时，在[0,1)区间内由 $F(1)$ 划分成二个子区间[0,$F(1)$)和[$F(1)$,1)，$F(1)=p(0)$。子区间[0,$F(1)$]的宽度为 $A(0)=p(0)$，子区间[$F(1)$,1)的宽度为 $A(1)=p(1)$。子区间[0,$F(1)$]对应于信源符号“0”，子区间[$F(1)$,1)对应于信源符号“1”。若输入符号序列的第一个符号为 $s=$“0”，即落入相应的区间为[0,$F(1)$)，得 $F(s=\text{“0”})=F(0)=0$。即某序列累积概率分布函数为该序列所对应区间的下界值。

(2) 当输入的第二个符号为“1”时，$s=$“01”，$s=$“01”所对应的区间是在[0,$F(1)$)中进行分割。符号序列“00”对应的区间宽度为 $A(00)=A(0)p(0)=p(0)p(0)$；符号序列“01”对应的区间宽度为 $A(01)=A(0)p(1)=p(0)p(1)=p(01)$，也等于 $A(01)=A(0)-A(00)$。“00”对应的区间为[0,$F(s=\text{“01”})$)；“01”对应的区间为[$F(s=\text{“01”})$,$F(1)$)。其中，$F(s=\text{“01”})$是符号序列“01”区间的下界值。可见，$F(s=\text{“01”})=p(0)p(0)$正是符号序列 $s=$“01”的累积分布函数。

(3) 当输入符号序列中第三个符号为“1”时，因前面已输入序列为 $s=$“01”，所以可记做输入序列为 $s1=$“011”(若第三个符号输入为“0”，可记作 $s0=$“010”)。现在，输入序列 $s1=$“011”所对应的区间是对区间[$F(s)$,$F(1)$)进行分割。序列 $s0=$“010”对应的区间宽度为 $A(s0=\text{“010”})=A(s=\text{“01”})p(0)=A(s)p(0)$，其对应的区间为[$F(s)$,$F(s)+A(s)p(0)$)，而序列 $s1=$“011”对应的区间宽度为

$$A(s1=\text{“011”})=A(s)p(1)=A(s=\text{“01”})-A(s0=\text{“010”})$$

即 $A(s1=\text{“011”})=A(s)-A(s0)$，其对应的区间为[$F(s)+A(s)p(0)$,$F(1)$)。可得，符号序列 $s1=$“011”的累积概率分布函数为 $F(s1)=F(s)+A(s)p(0)$。

(4) 若第三个符号输入为“0”，由上述分析可得，符号序列 $s0=$“010”的区间下界值仍为 $F(s)$，所以符号 $s0=$“010”的累积概率分布函数为 $F(s0)=\boldsymbol{F}(s)$。

现已输入三个符号串，将这个符号序列标为 s，接着输入第四个符号为“0”或“1”，又可计算出 $s0=$“0110”或 $s1=$“0111”对应的子区间及其累积概率分布函数，如图 5.3.5 所示。

根据前面的分析，可归纳出：

当已知前面输入符号序列 s，若接着输入一个符号“r”，则二元信源符号序列的累积概率分布函数的递推公式为

$$F(sr)=F(s)+p(s)F(r)\quad(r=0,1)\tag{5.3.1}$$

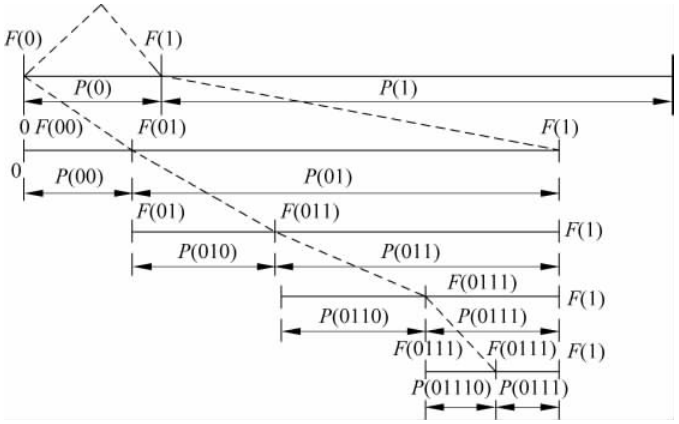


图 5.3.5　信源符号序列的累积概率分布函数 $F(s)$及对应的区间 A

同样，可得信源符号序列所对应区间宽度的递推公式为

$$A(sr) = p(sr) = p(s)p(r) \tag{5.3.2}$$

由此可得，信源符号序列对应的区间宽度等于该符号序列的概率。

例 5.3.2　设二元无记忆信源 $X\in\{0,1\}$，其 $p(0)=1/4$，$p(1)=3/4$。对二元序列 $s=11111100$ 做算术编码。

解：根据式(5.3.1)和式(5.3.2)，二元序列 $s=11111100$ 的累积概率分布函数为

$$\begin{aligned}
F(11111100) &= F(1111110) + p(1111110)F(0)\\
&= F(111111) + p(111111)F(0) + p(1111110)F(0)\\
&= F(11111) + p(11111)F(1)\\
&= F(1111) + p(1111)F(1)\\
&= F(111) + p(111)F(1)\\
&= F(11) + p(11)F(1)\\
&= F(1) + p(1)F(1) + p(11)F(1) + p(111)F(1) + p(1111)F(1) +\\
&\quad p(11111)F(1) + p(111111)F(0) + p(1111110)F(0)\\
&= p(0) + p(10) + p(110) + p(1110) + p(11110) + p(111110)
\end{aligned}$$

其对应的区间宽度为

$$A(11111100) = p(11111100)$$

由于累积概率分布函数和子区间宽度都是递推公式，因此在实际应用中，只需要两个存储器，把 $p(s)$和 $F(s)$存储下来，然后随着符号的输入，不断地更新两个存储器中的数值。因为在编码过程中，每输入一个符号就要进行乘法和加法运算，所以称这种编码方法为算术编码。

很容易将其推广到多元信源序列。可以得到一般信源序列的累积概率分布函数和区间宽度的递推公式为

$$\begin{cases} F(sa_k) = F(s) + p(s)F(a_k) \\ A(sa_k) = p(sa_k) = p(s)p(a_k) \end{cases} \tag{5.3.3}$$

通过关于信源符号序列的累积概率分布函数计算，$F(s)$可以把区间$[0,1)$分割成许多小

区间，每个小区间的长度等于各信源序列的概率 $F(s)$，不同的信源符号序列对应于不同的区间$[F(s),F(s)+p(s))$。可取小区间内的一点来代表这个序列。下面讨论如何选择这个点。

将符号序列的累积概率分布函数写成二进制小数，取小数点后 l 位，若后面有尾数，则进位到第 l 位，这样得到的一个数 C，并使 l 满足

$$l=\left\lceil \log\frac{1}{p(s)} \right\rceil \tag{5.3.4}$$

设 $C=0.z_1z_2\cdots z_l$，z_i 取 0 或者 1，得符号 s 的码字为 $z_1z_2\cdots z_l$。这样选取的数值 C，根据二进制小数截去位数的影响，得

$$C-F(s)<\frac{1}{2^l}$$

当 $F(s)$在 l 位以后没有尾数时，$C=F(s)$。另外，由 $l=\left\lceil \log\frac{1}{p(s)} \right\rceil$可知，$p(s)\geqslant\frac{1}{2^l}$，则信源符号序列 s 对应区间的上界为

$$F(s)+p(s)=F(s)+\frac{1}{2^l}>C$$

可见，数值 C 在区间$[F(s),F(s)+p(s))$内。不同的信源序列对应的不同区间(左封右开的区间)是不重叠的，所以编得的码是即时码。符号序列 s 的平均码长满足

$$-\sum_s p(s)\log p(s)\leqslant\bar{L}=\sum_s p(s)l(s)<-\sum_s p(s)\log p(s)+1$$

平均每个信源符号的码长为

$$\frac{H(s)}{n}\leqslant\frac{\bar{L}}{n}<\frac{H(s)}{n}+\frac{1}{n}$$

对无记忆信源，有

$$H(s)=nH_L(s)$$

因此有

$$H_L(s)\leqslant\frac{\bar{L}}{n}<H_L(s)+\frac{1}{n}$$

可以看出，算术编码的编码效率是比较高的。当信源符号序列很长时，n 很大，平均码长接近于信源的符号熵。

例 5.3.2 中，符号序列 $s=11111100$ 对应的累积概率分布函数为

$$\begin{aligned}F(11111100)&=p(0)+p(10)+p(110)+p(1110)+p(11110)+p(111110)\\&=3367/4096=0.82\,202=0.110\,100\,100\,111\end{aligned}$$

且

$$p(11111100)=3^6/4^8$$

$$l=\left\lceil \log\frac{4^8}{3^6} \right\rceil=7$$

得符号序列 s 的码字 $C=1101010$。

因此，平均码字长度为

$$\bar{K}=\frac{7}{8}(\text{二元码符号 / 字符})$$

编码效率为

$$\eta = \frac{H(X)}{\overline{K}} = \frac{0.811}{7/8} = 92.7\%$$

算术编码可以通过硬件电路实现，上述乘法运算可以通过右移来实现，因此在算术编码算法中只有加法和移位运算。

例 5.3.3 设二元无记忆信源 $X\in\{0,1\}$，其 $p(0)=1/4$，$p(1)=3/4$。对二元序列 $s=1011$ 做算术编码。

解：(1) 二进制信源只有两个符号“0”和“1”，且 $p(0)=1/4$，$p(1)=3/4$；

(2) 设 C 为子区间的左端起始位置，A 为子区间的宽度，符号“0”的子区间为[0,1/4)，符号“1”的子区间为[1/4,1)；

(3) 初始子区间为[0,1)，$C=0$，$A=1$，子区间按以下各步依次缩小：

步序	符号	C	A
1	1	$0+1\times1/4=1/4$	$1\times3/4=3/4$
2	0	1/4	$3/4\times1/4=3/16$
3	1	$1/4+3/16\times1/4=19/64$	$3/16\times3/4=9/64$
4	1	$19/64+9/64\times1/4=85/256$	$9/64\times3/4=27/256$

该过程如图 5.3.6 所示。

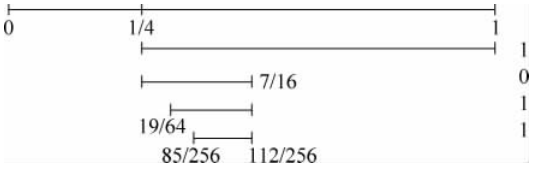


图 5.3.6 算术编码过程

最后的子区间左端(起始位置)为

$$C = 85/256 = 0.01\,010\,101$$

最后的子区间右端(终止位置)为

$$C+A = 112/256 = 0.01\,110\,000$$

编码结果为子区间头、尾之间取值，其值为 0.011，可编码为 011，原来 4 个符号 1011 现被压缩为三个符号 011。

例 5.3.4 一个离散二元无记忆信源，符号集为{0,1}，其中 $p(0)=0.1$，$p(1)=0.9$，信源序列长度 100，是 1111111011 的重复。

MATLAB 中的 arithenco 和 arithdeco 函数可以方便地用来实现算术编码和译码。

MATLAB 程序如下：

```
clear all;
clc;
seq = repmat([2 2 2 2 2 2 2 1 2 2],1,10)
counts = [10 90];
len = 100;
code = arithenco(seq,counts)
s2 = length(code)
```

```
dseq = arithdeco(code,counts,len)
comp_ratio = (len - s2)/len
```

输出：

```
code = % 编码码字
   Columns 1 through 21
1    0    0    0    1    1    0    0    0    0    1    1    1    0    1    1    1
0    0    1    0
   Columns 22 through 42
1    1    1    1    0    1    0    1    1    1    1    1    0    1    1    1    0
1    0    0    1
   Columns 43 through 55
0    1    0    1    0    0    0    1    1    1    1    0    1
s2 = 55                                   % 编码序列长度
comp_ratio = 45 %                         % 压缩比
```

若对上例中满足信源分布、长度为 1000 的序列进行算术编码，MATLAB 程序如下：

```
clear all;
clc;
counts = [10 90];
len = 1000;
seq = randsrc(1,len,[1 2;0.1 0.9]);
code = arithenco(seq,counts);
s2 = length(code)
dseq = arithdeco(code,counts,len);
comp_ratio = (len - s2)/len
```

输出：

```
s2 = 519
comp_ratio = 48.1 %
```

若信源序列长度改为 1000，则编码序列长度为 481，压缩比为 48.1%。可见，信源序列越长，压缩效果越好。

5.3.3　MH 编码及其 MATLAB 实现

MH(Modified Huffman)编码是将游程编码和哈夫曼编码相结合，是修正的哈夫曼编码，它是一行一行地对文件传真数据进行编码。此种方法利用了同行像素的同色性，为了保证收/发图文颜色同步，规定每行总是从白游程开始。在大多数文件中黑游程总比白游程短，因此两者的编码位数不同。MH 码表如表 5.3.1 和表 5.3.2 所示。

表 5.3.1　MH 码表：组合基干码

游 程 长 度	白游程码字	黑游程码字	游 程 长 度	白游程码字	黑游程码字
64	11011	000001111	640	011010100	0000001110011
128	10010	000011001000	704	011010101	0000001110100
192	010111	000011001001	768	011010110	0000001110101
256	0110111	000001011011	832	011010111	0000001110110
320	00110110	000000110011	896	011011000	0000001110111

续表

游 程 长 度	白游程码字	黑游程码字	游 程 长 度	白游程码字	黑游程码字
384	00110111	000000110100	1280	011011001	0000001010010
448	01100100	000000110101	1344	011011010	0000001010011
512	01100101	0000001101100	1408	011011011	0000001010100
576	01101000	0000001101101	1472	010011000	0000001010101
640	01100111	0000001001010	1536	010011001	0000001011010
704	011001100	0000001001011	1600	010011010	0000001011011
768	011001101	0000001001100	1664	011000	0000001100100
832	011010010	0000001001101	1728	010011011	0000001100101
896	011010011	0000001110010	EOL	000000000001	000000000001

表 5.3.2　MH 码表：结尾码

游 程 长 度	白游程码字	黑游程码字	游 程 长 度	白游程码字	黑游程码字
0	00110101	0001100000110111	32	00011011	000001101010
1	000111	010	33	00010010	000001101011
2	0111	11	34	00010011	000011010010
3	1000	10	35	00010100	000011010011
4	1011	011	36	00010101	000011010100
5	1100	0011	37	00010110	000011010101
6	1110	0010	38	00010111	000011010110
7	1111	00011	39	000101000	000011010111
8	10011	000101	40	00101001	000001101100
9	10100	000100	41	00101010	000001101101
10	00111	0000100	42	00101011	000011011010
11	01000	0000101	43	00101100	000011011011
12	001000	0000111	44	00101101	000001010100
13	000011	00000100	45	00000100	000001010101
14	110100	00000111	46	00000101	000001010110
15	110101	000011000	47	00001010	000001010111
16	101010	0000010111	48	00001011	000001100100
17	101011	0000011000	49	01010010	000001100101
18	0100111	0000001000	50	01010011	000001010010
19	0001100	00001100111	51	01010100	000001010011
20	0001000	00001101000	52	01010101	000000100100
21	0010111	00001101100	53	00100100	000000110111
22	0000011	00000110111	54	00100101	000000111000
23	0000100	00000101000	55	01011000	000000100111
24	0101000	00000010111	56	01011001	000000101000
25	0101011	00000011000	57	01011010	000001011000
26	0010011	000011001010	58	01011011	000001011001
27	0100100	000011001011	59	01001010	000000101011
28	0011000	000011001100	60	01001011	000000101100
29	00000010	000011001101	61	00110010	000001011010
30	00000011	000001101000	62	00110011	000001100110
31	00011010	000001101001	63	00110100	000001100111

MH 编码规则如下：

(1) 游程长度在 0～63 时，码字直接用该游程长度对应的结尾码表示。

(2) 游程长度在 64～1728 时，每个游程的码字分成两部分：前面是组合基干码，后面为结尾码。

(3) 规定每行总是从白游程开始，若第一游程为黑游程，则在行首加上长度为 0 的白游程码字，每行结束时用一个结束码 EOL 作标记。

(4) 传真文件传输时，每页文件的第一个数据前加一个结尾码，每页结尾连续使用 6 个结尾码表示结尾。

(5) 为了实现同步传输，规定 T 为每个编码行的最小传输时间，一般规定 T 最小为 20ms，最大为 5s。若行的传输时间小于 T，则在结尾码之前添上足够的 0 码元作为填充码。

按照上述编码规则，传真信息总的传送数据格式如图 5.3.7 所示。

图 5.3.7　传真信息总的传送数据格式

按照 MH 码表进行编码，一次只压缩一扫描行，各行独立不相关，因此是一种一维编码方案。译码时每一行的 MH 码都应恢复出 1728 个像素，否则有错。

例 5.3.5　设有一页传真文件，其中某一扫描线上的像素点如图 5.3.8 所示。

75 个白	5 个黑	9 个白	18 个黑	1621 个白

图 5.3.8　传真文件中某一扫描线上的像素点

求：(1) 该扫描行的 MH 编码；

(2) 编码后的比特总数；

(3) 本编码行的数据压缩比。

解：(1) 根据编码的 3 个规则，表 5.3.1 所示的 MH 码表。

- 75 个白：RL＝75，用规则(2)。组合基干码为 64(白)对应的 11011；补充结尾码为 75－64＝11(白)所对应的 01000。所以 75 个白对应码字为：1101101000。
- 5 个黑：RL＝5，用规则(1)。结尾码为 5(黑)对应的 0011。即为答案。
- 9 个白：结尾码为 9(白)对应的 10100。
- 18 个黑：结尾码为 18(黑)对应的 0000001000。
- 1621 个白：组合基干码为 1600(白)对应的 010011010；补充结尾码为 1621－1600＝21(白)所对应的 0010111。所以 1621 个白对应码字为：0100110100010111。
- EOL：结束码，为保证收/发同色，规定每行用一个结束码终止，查表可得为 000000000001。

最后得到的编码结果为

数据：75 白　　5 黑　　9 白　　18 黑　　1621 白 EOL

码字：1101101000；0011；10100；0000001000；0100110100010111；000000000001

(2) 将码字数一下，编码后的比特总数为 57bit。

(3) 压缩前数据总比特：75＋5＋9＋18＋1621＝1728bit，所以数据压缩比：1728∶57＝30.316∶1。

5.3.4　无损压缩的 JPEG 标准

JPEG(Joint Photographic Experts Group)是联合图像专家组的简称，它由两个标准机构——欧洲电信标准组织(CCIT)和国际标准化组织(ISO)联合组成。JPEG 是一个适用于彩色、单色多灰度或连续色调静态数字图像的压缩标准，已广泛应用于电视图像序列的帧内图像压缩编码以及照相机、打印机等方面的图像处理领域，是目前应用最广泛的静态图像压缩方法。

JPEG 中允许四种编解码模式：

(1) 基于 DCT 的顺序模式(sequential DCT-based)；

(2) 基于 DCT 的渐进模式(progressive DCT-based)；

(3) 无失真模式(lossless)；

(4) 层次模式(hierarchical)。

其中，模式(1)和(2)是基于 DCT 的有损压缩；模式(3)是基于线性预测的无损压缩；模式(4)可以是 DCT 与线性预测的分层混合。

在本章的前面部分，讨论了用于无损压缩的编码算法。应用这些技术可用少于源数据的比特数来存储或传输其所有的信息内容。传送源数据所有信息所必需的最小比特数取决于信源的熵。

以图像存储和传输的压缩技术为例，考虑 JPEG 图像压缩标准中的无损压缩，涉及 29 种不同的图像压缩编码系统的描述，以满足不同的用户对压缩质量和压缩相对计算时间的不同要求。其中，哈夫曼编码和算术编码是两种应用熵编码的压缩方法。算术编码与哈夫曼编码一样，通过利用数据的概率特征，使信息在传输或存储时使用比源数据更少的比特数。当数据只用到较小的字母集时算术编码的优势是它更接近于对数据流压缩的熵界。当符号出现的概率可表示为 1/2 的整数次幂时哈夫曼编码最优，压缩比最高。算术编码的构造与这些特定的概率值没有相关性，而且其编译码的计算量要大得多。用户可以选择使用哈夫曼编码或算术编码。

基准模式 JPEG 编码器如图 5.3.9 所示，它是无损编码和有损编码相混合的图像压缩格式。基于哈夫曼编码的无损编码的压缩比较低，所以 JPEG 主要应用有损压缩中的 DCT，将由 DCT 得到的参数进行 DPCM、量化、Z 形扫描，然后通过游程编码、哈夫曼编码等，提高压缩比。

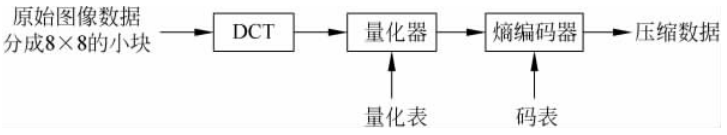


图 5.3.9　基准模式 JPEG 编码器

随着多媒体应用领域的激增，传统 JPEG 压缩技术已经无法满足人们对多媒体图像传输的要求，更高压缩比以及更多新功能的新一代静态图像压缩技术 JPEG2000 应运而生。JPEG2000 正式名称是“ISO 15444”，由 JPEG 组织负责制定，JPEG2000 标准中采用小波变换为主的多解析编码方式，能够同时支持无损和有损压缩。在实现高压缩比的目标并具备“感兴趣区域”特性方面，JPEG2000 与 JPEG 相比优势明显，且向下兼容，其应用领域将越来越广泛。

习　　题

1. 数据压缩的一个基本问题是“我们要压缩什么?”，你对此如何理解?

2. 某信源 $X\in\{a_1,a_2,a_3,a_4\}$，各符号概率为 $p(a_1)=0.6$，$p(a_2)=0.2$，$p(a_3)=0.1$，$p(a_4)=0.1$。其对应的三组码 A、B、C 如下：

码组 A：00　01　10　11

码组 B：1　01　110　101

码组 C：0　10　110　111

(1) 试判断这些码中哪些是唯一可译码，为什么?

(2) 对所有唯一可译码，求其平均码长。

3. 有线电报通信采用的莫尔斯编码把英语的 26 个字母和 5 个标点符号通过简单的编码表达出来。编码单元的长度与字母和标点符号出现的频度有关。统计发现英文字母的出现频率如表 5.1 所示。

表 5.1　英文字母的出现频率

字母	A	B	C	D	E	F	G	H	I	J	K	L	M
出现频率/%	8.167	1.492	2.782	4.253	12.7.2	2.228	2.015	6.094	6.966	0.153	0.772	4.025	2.406
字母	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
出现频率/%	6.749	7.507	1.929	0.095	5.987	6.327	9.056	2.758	0.978	2.360	0.15	1.974	0.074

(1) 若信源符号统计独立，求每个电报符号二元无失真编码的最佳码长；

(2) 考虑符号间的相关性，且每个电报符号携带的平均信息量是 $H_\infty=1.4$bit，则应该如何进行二元无失真编码？与(1)相比，编码效率有什么变化?

4. 已知信源 $X\in\{x_1,x_2,x_3,x_4,x_5,x_6\}$，若 $p(x_1)=0.37$，$p(x_2)=0.25$，$p(x_3)=0.18$，$p(x_4)=0.10$，$p(x_5)=0.07$，$p(x_6)=0.03$。

(1) 分别用香农编码、费诺编码和哈夫曼编码写出各符号的二元码字，并计算编码效率和信息传输率；

(2) 从(1)中能得出什么结论?

(3) 若该信源每秒钟发出 2 个符号，求信源的信息传输速率。

5. 给定信源 $\begin{bmatrix}X\\P\end{bmatrix}=\begin{bmatrix}s_1 & s_2\\0.8 & 0.2\end{bmatrix}$，

(1) 对该信源进行哈夫曼编码，求所编出的码字和编码效率；

(2) 对该信源的二次扩展信源 X^2 进行哈夫曼编码，求所编出的码字和编码效率；

（3）如对该信源的三次扩展信源 X^3 进行哈夫曼编码，预测其编码效率与(1)、(2)相比会发生什么变化，为什么？

6. 已知信源 $X\in\{x_1,x_2,x_3,x_4\}$，若 $p(x_1)=1/2$，$p(x_2)=1/4$，$p(x_3)=1/8$，$p(x_4)=1/8$，用香农编码、费诺编码和哈夫曼编码写出各符号的二元码字，并计算编码效率和信息传输率。

7. 有二元独立信源，已知 $p(0)=0.9$，$p(1)=0.1$，求信源的熵。当用哈夫曼编码时，以三个二元符号合成一个新符号，求这个新符号的平均码长和编码效率。设输入二元符号的速率是 100 个/s，若信道码率已规定为 50b/s，存储器容量（比特数）将如何选择？

8. 有二元平稳马尔可夫链，已知 $p(0/0)=0.8$，$p(1/1)=0.7$，求它的符号熵。用三个符号合成一个新符号来编哈夫曼编码，求这个新符号的平均码长和编码效率。

9. 设二元无记忆信源 $X\in\{0,1\}$，其中 $p(0)=\frac{1}{4}$，$p(1)=\frac{3}{4}$。对二元序列 11111100 做算术编码。

10. 一信源可发出的数字有 1,2,3,4,5,6,7，对应的概率分别为 $p(1)=p(2)=1/3$，$p(3)=p(4)=1/9$，$p(5)=p(6)=p(7)=1/27$，在二进制或三进制无噪声信道中传输，若二进制信道中传输一个码符号需要 1.8 元，三进制信道中传输一个码符号需要 2.7 元。

（1）编出二进制符号的霍夫曼编码，求其编码效率。

（2）编出三进制符号的费诺编码，求其编码效率。

（3）根据(1)和(2)的结果，确定在哪种信道中传输可得到较小的花费。

11. 设有一页传真文件，其中某一扫描行上的像素点如下所示：

| ←73 白→ | ←7 黑→ | ←白→ | ←18 黑→ | ←1619 白→ |

（1）写出该扫描行的 MH 码；

（2）求编码后该行的总比特数；

（3）求本行编码压缩比。