第5章

信源编码

前面介绍了信源熵和信息率失真函数的概念，了解了传送信源信息只需要具有信源极限熵或信息率失真函数大小的信息率。但是在实际通信系统中，用来传送信源信息的信息率远大于这些，那么是否能够达到或接近像信源熵或率失真函数这样的最小信息率，这就是编码定理要回答的问题之一。编码分为信源编码和信道编码，其中信源编码又分为无失真和限失真。由于这些定理都要求符号数很大才能使它的值接近所规定的值，因而这些定理被称为极限定理。一般称无失真信源编码定理为第一极限定理；称信道编码定理（包括离散和连续信道）为第二极限定理；称限失真信源编码定理为第三极限定理。完善这些定理是香农信息论的主要内容。下面分别讨论这三大定理。

由于信源符号之间存在分布不均匀和相关性，使得信源存在冗余度，信源编码的主要任务就是减少冗余，提高编码效率。具体来说，就是针对信源输出符号序列的统计特性，寻找一定的方法把信源输出符号序列变换为最短的码字序列。信源编码的基本途径有两个：使序列中的各个符号尽可能地互相独立，即解除相关性；使编码中各个符号出现的概率尽可能地相等，即概率均匀化。

信源编码的基础是信息论中的两个编码定理：**无失真编码定理**和**限失真编码定理**。前者是可逆编码的基础。可逆是指当信源符号转换成代码后，可从代码无失真地恢复原信源符号。当已知信源符号的概率特性时，可计算它的符号熵，即每个信源符号所载有的信息量。编码定理不但证明了必定存在一种编码方法，使代码的平均长度可任意接近但不能低于符号熵，而且还阐明达到该目标的途径，就是使概率与码长匹配。无失真编码或可逆编码只适用于离散信源。对于连续信源，编成代码后就无法无失真地恢复原来的连续值，因为后者的取值可有无限多个。此时只能根据率失真编码定理在失真受限制的情况下进行限失真编码。信源编码定理出现后，编码方法就趋于合理化。本章讨论离散信源编码，首先从无失真编码定理出发，讨论香农码，然后介绍限失真编码定理，最后简单介绍一些常用的信源编码方法。

5.1　编码的概念

将信源消息分成若干组，即符号序列 $\boldsymbol{x}_i$，$\boldsymbol{x}_i=(x_{i_1},x_{i_2},\cdots,x_{i_l},\cdots,x_{i_L})$，序列中的每个符号取自于符号集 A，$x_{i_L}\in\{a_1,a_2,\cdots,a_i,\cdots,a_n\}$。而每个符号序列 $\boldsymbol{x}_i$ 依照固定的码表映射成一个码字 $\boldsymbol{y}_i$，这样的码称为**分组码**，有时也叫块码。只有分组码才有对应的码表，而非分组码中则不存在码表。

如图 5-1 所示，如果信源输出的符号序列长度 $L=1$，信源符号集为 $A=(a_1,a_2,\cdots,a_n)$ 信源概率空间为

$$\begin{bmatrix}X\\P\end{bmatrix}=\begin{bmatrix}a_1 & a_2 & \cdots & a_n\\ p(a_1) & p(a_2) & \cdots & p(a_n)\end{bmatrix}$$

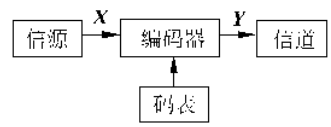


图 5-1　信源编码器示意图

需要将这样的信源符号传输。常用的一种信道就是二元信道，它的信道基本符号集为{0,1}。若将信源 X 通过这样的二元信道传输，就必须把信源符号 a_i 变换成由 0、1 符号组成的码符号序列，这个过程就是信源编码。可用不同的码符号序列，如表 5-1 所列。

表 5-1　码的不同属性

信源符号 a_i	符号出现概率 $p(a_i)$	码 1	码 2	码 3	码 4	码 5
a_1	1/2	00	0	0	1	1
a_2	1/4	01	11	10	10	01
a_3	1/8	10	00	00	100	001
a_4	1/8	11	11	01	1000	0001

一般情况下，码可分为两类：一类是固定长度的码，码中所有码字的长度都相同，如表 5-1 中的码 1 就是定长码。另一类是可变长度码，码中的码字长短不一，如表 5-1 中的其他码都是变长码。

采用分组编码方法，需要分组码具有某些属性，以保证在接收端能够迅速准确地将码译出。下面首先讨论分组码的一些直观属性。

(1) 奇异码和非奇异码

若信源符号和码字是一一对应的，则该码为非奇异码。反之为奇异码。如表 5-1 中的码 2 是奇异码，码 3 是非奇异码。

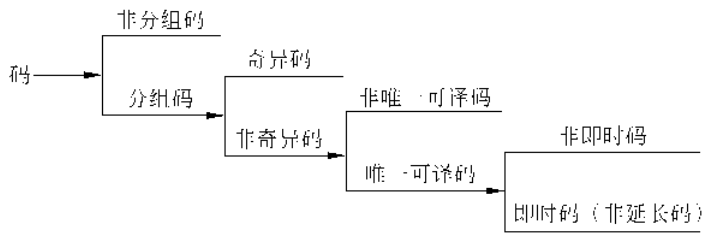
(2) 唯一可译码

任意有限长的码元序列，只能被唯一地分割成一个个的码字，便称为唯一可译码。例如{0,10,11}是一种唯一可译码。因为任意一串有限长码序列，如 100111000，只能被分割成 10,0,11,10,0,0。任何其他分割法都会产生一些非定义的码字。显然，奇异码不是唯一可译码，而非奇异码中有非唯一可译码和唯一可译码。表 5-1 中码 4 是唯一可译码，但码 3 不是唯一可译码。例如 10000100 是由码 3 的(10,0,0,01,00)产生的码流，译码时可有多种分割方法，如 10,0,00,10,0，此时就产生了歧义。

（3）非即时码和即时码

唯一可译码中又分为**非即时码**和**即时码**。如果接收端收到一个完整的码字后，不能立即译码，还需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。表 5-1 中码 4 是非即时码，而码 5 是即时码。码 5 中只要收到符号 1 就表示该码字已完整，可以立即译码。即时码又称为**非延长码**，任意一个码字都不是其他码字的前缀部分，有时称为**异前缀码**。在延长码中，有的码是唯一可译的，主要取决于码的总体结构，如表 5-1 中码 4 的延长码就是唯一可译的。

综上所述，可将码作如下分类：



通常可用码树来表示各码字的构成。对于 m 进制的码树，如图 5-2 所示。图 5-2(a)是二进码树，图 5-2(b)是三进码树。其中 A 点是树根，分成 m 个树枝，则称为 m 进码树。树枝的尽头是节点，中间节点生出树枝，终端节点安排码字。码树中自根部经过一个分枝到达 m 个节点称为一级节点。二级节点的可能个数为 m^2 个，一般 r 级节点有 m^r 个。图 5-2(a)的码树是 4 节，有 $2^4=16$ 个可能的终端节点。若将从每个节点发出的 m 个分枝分别标以 $0,1,\cdots,m-1$，则每个 r 级节点需要用 r 个 m 元数字表示。如果指定某个 r 级节点为终端节点表示一个信源符号，则该节点就不再延伸，相应的码字即为从树根到此端点的分枝标号序列，其长度为 r。这样构造的码满足即时码的条件。因为从树根到每一个终端节点所走的路径均不相同，故一定满足对前缀的限制。如果有 q 个信源符号，那么在码树上就要选择 q 个终端节点，用相应的 m 元基本符号表示这些码字。由这样的方法构造出来的码称为树码，若树码的各个分支都延伸到最后一级端点，此时将共有 m^r 个码字，这样的码树称为**满树**，如图 5-2(a)所示。否则就称为**非满树**，如图 5-2(b)所示，这时的码字就不是定长的了。总结上述关于码树和码字的对应关系，可得到如图 5-3 所示的关系图。

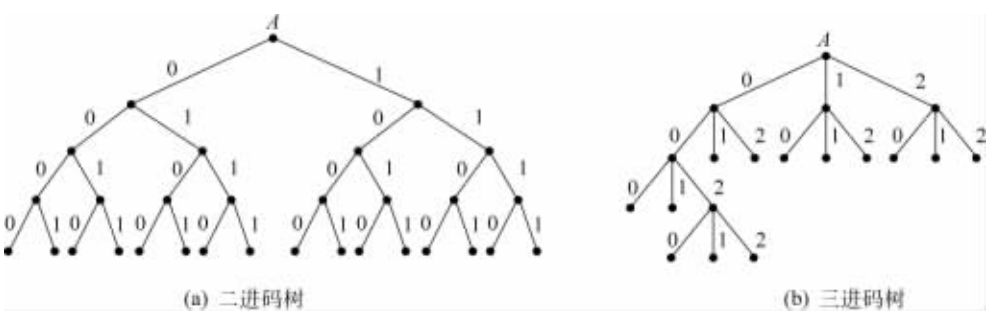


图 5-2　码树图

用树的概念可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**克劳夫特不等式**(Kraft's inequality)，即

$$\sum_{i=1}^{n} m^{-K_i} \leqslant 1 \qquad (5\text{-}1\text{-}1)$$

其中 m 是进制数，n 是信源符号数。

图 5-3　码树与码字对应关系图

上述不等式是唯一可译码存在的充要条件，必要性表现在如果是唯一可译码，则必定满足该不等式，如表 5-1 中的码 1、码 4 和码 5 等都满足不等式；充分性表现在如果满足不等式，则这种码长的唯一可译码一定存在，但并不表示所有满足不等式的码一定是唯一可译码。所以说，该不等式是唯一可译码存在的充要条件，而不是唯一可译码的充要条件。

例 5-1　用二进制对符号集$\{a_1, a_2, a_3, a_4\}$进行编码，对应的码长分别为 $K_1=1, K_2=2, K_3=2, K_4=3$，应用式(5-1-1)判断

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

因此不存在满足这种 K_i 的唯一可译码。可以用树码进行检查，由图 5-4 所示，要形成上述码字，必然在中间节点放置码字，若符号 a_1 用“0”码，符号 a_2 用“10”码，符号 a_3 用“11”码，则符号 a_4 只能是符号 a_2 或 a_3 所编码的延长码。

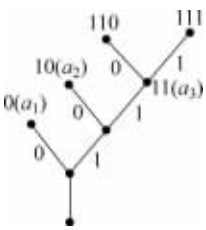


图 5-4　树码

如果将各码字长度改成 $K_1=1, K_2=2, K_3=3, K_4=3$，则此时

$$\sum_{i=1}^{4} 2^{-K_i} = 2^{-1} + 2^{-2} + 2^{-3} + 2^{-3} = 1$$

这种 K_i 的唯一可译码是存在的，如$\{0, 10, 110, 111\}$。但是必须注意，克劳夫特不等式只是用来说明唯一可译码是否存在，并不能作为唯一可译码的判据。如码字$\{0, 10, 010, 111\}$，虽然满足克劳夫特不等式，但它不是唯一可译码。

5.2　无失真信源编码定理

若信源输出符号序列的长度 $L \geqslant 1$，即

$$\boldsymbol{X} = (X_1, X_2, \cdots, X_l, \cdots, X_L), \quad X_l \in \{a_1, a_2, \cdots, a_i, \cdots, a_n\}$$

变换成由 K_L 个符号组成的码序列(有时也叫做码字，以下均用码字来叙述)。

$$\boldsymbol{Y} = (Y_1, Y_2, \cdots, Y_k, \cdots, Y_{K_L}), \quad Y_k \in \{b_1, b_2, \cdots, b_j, \cdots, b_m\}$$

变换的要求是能够无失真或无差错地从 $\boldsymbol{Y}$ 恢复 $\boldsymbol{X}$，也就是能正确地进行反变换或译码，同时希望传送 $\boldsymbol{Y}$ 时所需要的信息率最小。由于 Y_k 可取 m 种可能值，即平均每个符号输出的最大信息量为 $\log m$，K_L 长码字的最大信息量为 $K_L \log m$。用该码字表示 L 长的信源序列，则送出一个信源符号所需要的信息率平均为 $\overline{K} = \frac{K_L}{L}\log m = \frac{1}{L}\log M$，其中 $M = m^{K_L}$ 是 $\boldsymbol{Y}$ 所能编成的码字的个数。所谓信息率最小，就是找到一种编码方式使 $\frac{K_L}{L}\log m$ 最小。然而上述最小信息率为多少时，才能得到无失真的译码？若小于这个信息率是否还能无失真地译码？这就是无失真信源编码定理要研究的内容。在无失真的信源编码定理中相应的有定长编码

定理和变长编码定理，下面分别加以讨论。

5.2.1 定长编码

在定长编码中，K 是定值，且 $K=K_L$。编码的目的是寻找最小 K 值。要实现无失真的信源编码，不但要求信源符号 $\boldsymbol{X}_i, i=1,2,\cdots,q$ 与码字 $\boldsymbol{Y}_i, i=1,2,\cdots,q$ 是一一对应的，而且还要求由码字组成的码符号序列的逆变换也是唯一的。也就是说，由一个码表编出的任意一串有限长的码符号序列只能被唯一地译成所对应的信源符号序列。

定长编码定理：由 L 个符号组成的、每个符号的熵为 $H_L(\boldsymbol{X})$的无记忆平稳信源符号序列 $X_1,X_2,\cdots,X_l,\cdots,X_L$，可用 K_L 个符号 $Y_1,Y_2,\cdots,Y_k,\cdots,Y_{K_L}$（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

$$\frac{K_L}{L}\log m \geqslant H_L(\boldsymbol{X})+\varepsilon \tag{5-2-1}$$

则当 L 足够大时，必可使译码差错小于 δ；反之，当

$$\frac{K_L}{L}\log m \leqslant H_L(\boldsymbol{X})-2\varepsilon \tag{5-2-2}$$

时，译码差错一定是有限值，而当 L 足够大时，译码几乎必定出错。

这个定理的前一部分是正定理，后一部分为逆定理。定理证明略。

上述编码定理说明，当编码器容许的输出信息率，也就是当每个信源符号所必须输出的码长是

$$\overline{K}=\frac{K_L}{L}\log m \tag{5-2-3}$$

时，只要 $\overline{K}>H_L(\boldsymbol{X})$，这种编码器一定可以做到几乎无失真，也就是接收端的译码差错概率接近于零，条件是所取的符号数 L 足够大。

将上述定理的条件式(5-2-1)改写成

$$K_L\log m > LH_L(\boldsymbol{X})=H(\boldsymbol{X}) \tag{5-2-4}$$

上式大于号左边为 K_L 长码字所能携带的最大信息量，右边为 L 长信源序列携带的信息量。于是上述定理表明，只要码字所能携带的信息量大于信源序列输出的信息量，则可以使传输几乎无失真，当然条件是 L 足够大。

反之，当 $\overline{K}<H_L(\boldsymbol{X})$时，不可能构成无失真的编码，也就是不可能做一种编码器，能使接收端译码时差错概率趋于零。当 $\overline{K}=H_L(\boldsymbol{X})$时，则为临界状态，可能无失真，也可能有失真。

例如，某信源有 8 种等概率符号，$L=1$，信源序列熵达到最大值

$$H_1(X)=\log_2 8=3\text{bit/ 符号}$$

即该信源符号肯定可以用 3bit 的信息率进行无失真的编码。这就是说，如果采用二进制符号作为码字输出符号，$Y_k\in\{0,1\}$，则用 3 个 bit 就可以表示一个符号，即 $\overline{K}=3\text{bit/符号}=H_1(\boldsymbol{X})$。当信源符号输出概率不相等时，如 $p(a_i)=\{0.4,0.18,0.1,0.1,0.07,0.06,0.05,0.04\}$，则此时 $H_1(\boldsymbol{X})=2.55\text{bit/符号}$，小于 3bit。按常理，8 种符号一定要用 3bit($2^3=8$)组成的码字表示才能区别开来，而用 $\overline{K}=H_L(\boldsymbol{X})=2.55\text{bit/符号}$来表示，只有 $2^{2.55}=5.856$ 种可能码字，还有部分符号没有对应的码字，信源一旦出现这些符号，就只能用其他码字替代，

因而引起差错。差错发生的可能性就取决于这些符号出现的概率。当 L 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。

设 $\boldsymbol{x}_i=(x_{i_1},x_{i_2},\cdots,x_{i_l},\cdots,x_{i_L})$ 是信源序列的样本矢量，$x_{i_L}\in\{a_1,a_2,\cdots,a_i,\cdots,a_n\}$，则共有 n^L 种样本，把它分为两个互补的集 A_ε 和 A_ε^C，集 A_ε 中的元素（样本矢量）有与之对应的不同码字，而集 A_ε^C 中的元素没有对应的输出码字，因而会在译码时发生差错。如果允许一定的差错 δ，则编码时只需对属于 A_ε 中的 M_ε 个样本矢量赋以相应的不同码字，即输出码字的总个数 m^K 只要大于 M_ε 就可以了。在这种编码方式下，差错概率 P_e 即为集 A_ε^C 中元素发生的概率 $p(A_\varepsilon^C)$，此时要求 $p(A_\varepsilon^C)\leqslant\delta$，因而 A_ε^C 集中的样本都应是小概率事件。当 L 增大时，虽然样本数也随着增多，但小概率事件的概率将更小，有望使 $p(A_\varepsilon^C)$ 更小。根据切比雪夫不等式可推得（推导从略，见参考文献[1]）

$$P_e\leqslant\frac{\sigma^2(\boldsymbol{X})}{L\varepsilon^2}\tag{5-2-5}$$

式中 $\sigma^2(\boldsymbol{X})=E\{[I(\boldsymbol{x}_i)-H(\boldsymbol{X})]^2\}$ 为信源序列的自信息方差，ε 为一正数。当 $\sigma^2(\boldsymbol{X})$ 和 ε^2 均为定值时，只要 L 足够大，P_e 可以小于任一正数 δ，即 $\frac{\sigma^2(\boldsymbol{X})}{L\varepsilon^2}\leqslant\delta$，也就是当信源序列长度 L 满足

$$L\geqslant\frac{\sigma^2(\boldsymbol{X})}{\varepsilon^2\delta}\tag{5-2-6}$$

时，就能达到差错率要求。

说得具体一些，就是给定 ε 和 δ 后，用式(5-2-6)规定了 L 的大小，计算所有可能的信源序列样本矢量的概率 $p(\boldsymbol{x}_i)$，按概率大小排列，选用概率较大的 $\boldsymbol{x}_i$ 作为 A_ε 中的元素，直到 $p(A_\varepsilon)\geqslant1-\delta$，使 $p(A_\varepsilon^C)\leqslant\delta$。这些在 A_ε 中的元素分别用不同码字来代表，就完成了编码过程。如果取足够小的 δ，就可几乎无差错地译码，而所需的信息率就不会超过 $H_L(\boldsymbol{X})+\varepsilon$。

在连续信源的情况下，由于信源的信息量趋于无限，显然是不能用离散符号序列 $\boldsymbol{Y}$ 来完成无失真编码的，而只能进行限失真编码。

定义

$$\eta=\frac{H_L(\boldsymbol{X})}{\overline{K}}$$

为**编码效率**。即信源的平均符号熵为 $H(X)$，采用平均符号码长为 $\overline{K}$ 来编码所得的效率。编码效率总是小于 1，且最佳编码效率为

$$\eta=\frac{H_L(\boldsymbol{X})}{H_L(\boldsymbol{X})}+\varepsilon,\quad\varepsilon>0\tag{5-2-7}$$

编码定理从理论上阐明了编码效率接近 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，即

$$\frac{H_L(\boldsymbol{X})}{\frac{K_L}{L}\log m}\to1\tag{5-2-8}$$

但要在实际中实现，必须取无限长（$L\to\infty$）的信源符号进行统一编码。这样做实际上是不可能的，因 L 非常大，无法实现。下面用例子来说明。

例 5-2 设离散无记忆信源概率空间为

$$\begin{bmatrix} X \\ P \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 \\ 0.4 & 0.18 & 0.1 & 0.1 & 0.07 & 0.06 & 0.05 & 0.04 \end{bmatrix}$$

信源熵为

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

对信源符号采用定长二元编码，要求编码效率为 $\eta=90\%$，若取 $L=1$，则可算出

$$\overline{K} = 2.55 \div 90\% = 2.8\text{bit/ 符号}, \quad 2^{2.88} = 6.96 \text{ 种}$$

即每个符号用 2.8bit 进行定长二元编码，共有 6.96 种可能性，即使按 7 种可能性来算，信源符号中就有一种符号没有对应的码字，取概率最小的 a_8，差错概率为 0.04，显然太大。现采用式(5-2-7)

$$\eta = \frac{H(X)}{H(X)+\varepsilon} = 0.90$$

可以得到 $\varepsilon=0.28$

信源序列的自信息方差为

$$\sigma^2(X) = D[I(x_i)] = \sum_{i=1}^{8} p_i (\log p_i)^2 - [H(X)]^2 = 7.82\ (\text{bit})^2$$

若要求译码错误概率 $\delta \leqslant 10^{-6}$，由式(5-2-6)得

$$L \geqslant \frac{\sigma^2(X)}{\varepsilon^2 \delta} = \frac{7.82}{0.28^2 \times 10^{-6}} = 9.8 \times 10^7 \approx 10^8$$

由此可见，在对编码效率和译码错误概率的要求并不十分苛刻的情况下，就需要 $L=10^8$ 个信源符号一起进行编码，这对存储或处理技术的要求太高，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，$L=1$，则 $\overline{K}=H(X)+\varepsilon=3$，可以求得 $\varepsilon=0.45$。此时译码无差错，即 $\delta=0$。在这种情况下，式(5-2-6)就不适用了。但此时编码效率只能为 $\eta=\frac{2.55}{3}=85\%$。因此一般来说，当 L 有限时，高传输效率的定长码往往要引入一定的失真和错误，它不像变长码那样可以实现无失真编码。

5.2.2 变长编码

在变长编码中，码长 K 是变化的，可根据信源各个符号的统计特性，如概率大的符号用短码，如例 5-2 中的 a_1、a_2 可用 1 或 2bit，而对概率小的 a_7、a_8 用较长的码，这样在大量信源符号编成码后，平均每个信源符号所需的输出符号数就可以降低，从而提高编码效率。下面分别给出单个符号($L=1$)和符号序列的变长编码定理。

单个符号变长编码定理：若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 $\overline{K}$ 满足下列不等式

$$\frac{H(X)}{\log m} \leqslant \overline{K} < \frac{H(X)}{\log m} + 1 \tag{5-2-9}$$

离散平稳无记忆序列变长编码定理：对于平均符号熵为 $H_L(\boldsymbol{X})$的离散平稳无记忆信

源，必存在一种无失真编码方法，使平均信息率 $\overline{K}$ 满足不等式

$$H_L(\boldsymbol{X}) \leqslant \overline{K} < H_L(\boldsymbol{X}) + \varepsilon \tag{5-2-10}$$

其中 ε 为任意小正数。

可从式(5-2-9)推出式(5-2-10)。设用 m 进制码元作变长编码，序列长度为 L 个信源符号，则由式(5-2-9)可以得到平均码字长度 $\overline{K_L}$ 满足下列不等式

$$\frac{LH_L(\boldsymbol{X})}{\log m} \leqslant \overline{K_L} < \frac{LH_L(\boldsymbol{X})}{\log m} + 1$$

已知平均输出信息率为

$$\overline{K} = \frac{\overline{K_L}}{L}\log m$$

则 $$H_L(\boldsymbol{X}) \leqslant \overline{K} < H_L(\boldsymbol{X}) + \frac{\log m}{L}$$

当 L 足够大时，可使 $\frac{\log m}{L} < \varepsilon$，这就得到了式(5-2-10)。

用变长编码来达到相当高的编码效率，一般所要求的符号长度 L 可以比定长编码小得多。从式(5-2-10)可得编码效率的下界：

$$\eta = \frac{H_L(\boldsymbol{X})}{\overline{K}} > \frac{H_L(\boldsymbol{X})}{H_L(\boldsymbol{X}) + \frac{\log m}{L}} \tag{5-2-11}$$

例如用二进制，$m=2$，$\log_2 m = 1$，仍用前面的例 5-2，$H(X)=2.55$bit/符号，若要求 $\eta > 90\%$，则

$$\frac{2.55}{2.55 + \frac{1}{L}} = 0.9, \quad L = \frac{1}{0.28} \approx 4$$

就可以了。

编码效率总是小于 1，可以用它来衡量各种编码方法的优劣。另外，为了衡量各种编码方法与最佳码的差距，定义**码的剩余度**为

$$\gamma = 1 - \eta = 1 - \frac{H_L(\boldsymbol{X})}{\frac{\overline{K_L}}{L}\log m} = 1 - \frac{H_L(\boldsymbol{X})}{\overline{K}} \tag{5-2-12}$$

例 5-3 设离散无记忆信源的概率空间为

$$\begin{bmatrix} X \\ P \end{bmatrix} = \begin{bmatrix} a_1 & a_2 \\ \frac{3}{4} & \frac{1}{4} \end{bmatrix}$$

其信源熵为

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

若用二元定长编码(0,1)来构造一个即时码：$a_1 \to 0$，$a_2 \to 1$。这时平均码长

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

编码效率为

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

输出的信息效率为

$$R = 0.811\text{bit/ 二元码符号}$$

再对长度为 2 的信源序列进行变长编码(编码方法后面介绍),其即时码如表 5-2 所示。

表 5-2 $L=2$ 时信源序列的变长编码

序　列	序列概率	即 时 码
a_1a_1	9/16	0
a_1a_2	3/16	10
a_2a_1	3/16	110
a_2a_2	1/16	111

这个码的码字平均长度

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

每一单个符号的平均码长

$$\overline{K} = \frac{\overline{K_2}}{2} = \frac{27}{32}\text{ 二元码符号 / 信源符号}$$

其编码效率

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

输出的信息效率

$$R_2 = 0.961\text{bit/ 二元码符号}$$

可见编码复杂了一些,但信息传输效率有了提高。

用同样的方法可进一步将信源序列的长度增加,$L=3$ 或 $L=4$,对这些信源序列 $\boldsymbol{X}$ 进行编码,并求出其编码效率为

$$\eta_3 = 0.985$$

$$\eta_4 = 0.991$$

这时信息传输效率分别为

$$R_3 = 0.985\text{bit/ 二元码符号}$$

$$R_4 = 0.991\text{bit/ 二元码符号}$$

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%时,允许译码错误概率 $\delta \leqslant 10^{-5}$。则根据式(5-2-8),自信息的方差

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

所需要的信源序列长度

$$L \geqslant \frac{0.4715}{(0.811)^2}\cdot\frac{(0.96)^2}{0.04^2\times 10^{-5}} = 4.13\times 10^7$$

很明显,定长码需要的信源序列长,使得码表很大,且总存在译码差错。而变长码要求编码效率达到 96%时,只需 $L=2$。因此用变长码编码时,L 不需要很大就可达到相当高的编码效率,而且可实现无失真编码。随着信源序列长度的增加,编码的效率越来越接近于 1,编码后的信息传输率 R 也越来越接近于无噪无损二元对称信道的信道容量 $C=1$bit/二

元码符号，达到信源与信道匹配，使信道得到充分利用。

从变长编码定理可以看出，要使信源编码后的平均码长最短，就要求信源中每个符号的码长与其概率相匹配，即概率大的信息符号编以短的码字，概率小的符号编以长的码字。由于符号的自信息量 $I(x_i)$ 就是基于概率计算得到的该符号含有的信息量，因此，将式(5-2-9)中的信源熵和平均码长替换成每个信源符号的自信息量 $I(x_i)$ 和码长 K_i，则可得到一种构造最佳码长的编码方法，称为香农编码。

香农第一定理指出，选择每个码字的长度 K_i 满足下式

$$I(x_i) \leqslant K_i < I(x_i)+1, \quad \forall i$$

就可以得到这种码。其编码方法如下：

(1) 将信源消息符号按其出现的概率大小依次排列为

$$p_1 \geqslant p_2 \geqslant \cdots \geqslant p_n$$

(2) 确定满足下列不等式的整数码长 K_i 为

$$-\log_2(p_i) \leqslant K_i < -\log_2(p_i)+1$$

(3) 为了编成唯一可译码，计算第 i 个消息的累加概率

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

(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。

如图 5-5 所示，香农编码可以这样理解，累加概率 P_i 把区间[0,1)分割成许多小区间，每个小区间的长度等于各符号的概率 p_i，小区间内的任一点可用来代表该符号。

例 5-4 设信源共 7 个符号消息，其概率和累加概率如表 5-3 所列。以 $i=4$ 为例，

$$-\log_2 0.17 \leqslant K_4 < -\log_2 0.17+1$$

$$2.56 \leqslant K_4 < 3.56, \quad K_4 = 3$$

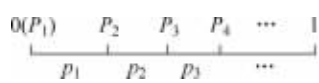


图 5-5

累加概率 $P_4=0.57$，变换成二进制为 0.1001…，由于 $K_4=3$，所以第 4 个消息的编码码字为 100。其他消息的码字可用同样方法求得，如表 5-3 所示。该信源共有 5 个 3 位的码字，各码字之间至少有一位数字不相同，故是唯一可译码。同时可以看出，这 7 个码字都不是延长码，它们都属于即时码。

表 5-3 香农码编码过程

信源消息符号 a_i	符号概率 $p(a_i)$	累加概率 P_i	$-\log p(a_i)$	码字长度 K_i	码字
a_1	0.20	0	2.32	3	000
a_2	0.19	0.2	2.39	3	001
a_3	0.18	0.39	2.47	3	011
a_4	0.17	0.57	2.56	3	100
a_5	0.15	0.74	2.74	3	101
a_6	0.10	0.89	3.34	4	1110
a_7	0.01	0.99	6.64	7	1111110

这里 $L=1, m=2$，所以信源符号的平均码长为

$$\overline{K}=\sum_{i=1}^{7}p(a_i)K_i=3.14 \text{ 码元 / 符号}$$

平均信息传输速率为

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

这种码的编码效率为 83.1%，是比较低的。

例 5-5 设信源有 3 个符号，概率分布为(0.5,0.4,0.1)，根据香农编码方法求出各个符号的码长对应为(1,2,4)，码字为(0,10,1110)。事实上，观察信源的概率分布可以构造出一个码长更短的码(0,10,11)，显然也是唯一可译码。

所以从上述两个例子可以看出，香农编码法多余度稍大，编码效率比较低，实用性不强，但它是依据编码定理而来，因此具有重要的理论意义。按照信源编码定理，若对信源序列进行编码，当序列长度 $L\to\infty$ 时，平均码长会趋于信源熵。

5.3 限失真信源编码定理

将编码器看作信道，信源编码模型如图 5-6 所示。无失真信源编码对应于无损确定信道，有失真信源编码对应于有噪信道。对于无失真信源编码，信道的输入符号个数与输出符号个数相等，呈一一对应关系，信道的损失熵 $H(X|Y)$ 和噪声熵 $H(Y|X)$ 均为零，通过信道的信息传输率 R 等于信源熵 $H(X)$，因此，从信息处理的角度来看，无失真信源编码是保熵的，只是对冗余度进行了压缩，因为冗余度是对信号携带信息能力的一种浪费。

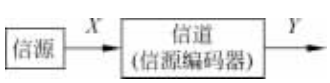


图 5-6 信源编码器示意图

有失真信源编码的中心任务是：在允许的失真范围内把编码后的信息率压缩到最小。有失真信源编码的失真范围受限，所以又称为限失真信源编码；编码后的信息率得到压缩，因此属熵压缩编码。之所以引入有失真的熵压缩编码，原因如下：

(1) 保熵编码并非总是必需的。有些情况下，信宿不需要或无能力接收信源发出的全部信息，例如人眼接收视觉信号和人耳接收听觉信号就属于这种情况，这时就没有必要进行无失真的保熵编码。

(2) 保熵编码并非总是可能的。例如对连续信号进行数字处理时，由于不可能从根本上去除量化误差，因此不可能做到保熵编码。

(3) 降低信息率有利于传输和处理，因此有必要进行熵压缩编码。例如连续信源的绝对熵为无穷大，若用离散码元来表示，需要用无穷长的码元串，传输无穷长的码元串势必造成无限延时，这种通信就无任何实际意义了。所以，对连续信源而言，熵压缩编码是绝对必需的。有失真的熵压缩编码主要针对连续信源，但其理论同样适用于离散信源。

在第 4 章讨论中，信息率失真函数给出了失真小于 D 时所必须具有的最小信息率 $R(D)$；只要信息率大于 $R(D)$，一定可以找到一种编码，使译码后的失真小于 D。

限失真信源编码定理：设离散无记忆信源 X 的信息率失真函数 $R(D)$，则当信息率

$R>R(D)$，只要信源序列长度 L 足够长，一定存在一种编码方法，其译码失真小于或等于 $D+\varepsilon$，ε 为任意小的正数。反之，若 $R<R(D)$，则无论采用什么样的编码方法，其译码失真必大于 D。

如果是二元信源，对于任意小的 $\varepsilon>0$，每一个信源符号的平均码长满足

$$R(D) \leqslant \overline{K} < R(D)+\varepsilon$$

上述定理指出，在失真限度内使信息率任意接近 $R(D)$ 的编码方法存在。然而，要使信息率小于 $R(D)$，平均失真一定会超过失真限度 D。

对于连续平稳无记忆信源，虽然无法进行无失真编码，在限失真情况下，有与上述定理一样的编码定理。

上述定理只能说明最佳编码是存在的，而具体构造编码方法却一无所知。因而就不能像无损编码那样从证明过程中引出概率匹配的编码方法。一般只能从优化的思路去求最佳编码。实际上迄今尚无合适的可实现的编码方法可接近 $R(D)$ 这个界。

5.4 常用信源编码方法简介

前面已经介绍了信源编码的两大定理，实用的编码方法需要根据信源的具体特点。在编码理论指导下，先后出现了许多性能优良的编码方法，根据信源的性质进行分类，则有信源统计特性已知或未知、无失真或限定失真、无记忆或有记忆信源的编码；按编码方法进行分类，可分为分组码或非分组码、等长码或变长码等。然而最常见的是讨论统计特性已知条件下，离散、平稳、无失真信源的编码，消除这类信源剩余度的主要方法有统计匹配编码和解除相关性编码。例如，香农码、哈夫曼码属于不等长度分组码，算术编码属于非分组码，预测编码和变换编码是以解除相关性为主的编码。对统计特性未知的信源编码称为通用编码，如 ***LZ*** 编码。对限定失真的信源编码则是以信息率失真 $R(D)$ 函数为基础，最典型的是矢量量化编码。在此简要介绍部分编码方法的基本原理。

5.4.1 哈夫曼编码

哈夫曼编码是分组编码，完全依据各字符出现的概率来构造码字。其基本原理是基于二叉树的编码思想，所有可能的输入符号在哈夫曼树上对应为一个节点，节点的位置就是该符号的哈夫曼编码。为了构造出唯一可译码，这些节点都是哈夫曼树上的终极节点，不再延伸，不会出现前缀码。具体编码方法如下：

(1) 将信源消息符号按其出现的概率大小依次排列为

$$p_1 \geqslant p_2 \geqslant \cdots \geqslant p_n$$

(2) 取两个概率最小的字母分别配以 0 和 1 两个码元，并将这两个概率相加作为一个新字母的概率，与未分配二进符号的字母一起重新排队。

(3) 对重排后的两个概率最小符号重复步骤(2)的过程。

(4) 不断继续上述过程，直到最后两个符号配以 0 和 1 为止。

(5) 从最后一级开始，向前返回得到各个信源符号所对应的码元序列，即相应的码字。

例 5-6 对例 5-4 中的信源进行哈夫曼编码，编码过程如表 5-4 所示。

表 5-4 哈夫曼码编码过程

信源符号 a_i	概率 $p(a_i)$	编码过程	码字 W_i	码长 K_i
a_1	0.20	0.20 0.26 0.35 0.39 0.61 0 1.0	10	2
a_2	0.19	0.19 0.20 0.26 0.35 0 0.39 1	11	2
a_3	0.18	0.18 0.19 0.20 0 0.26 1	000	3
a_4	0.17	0.17 0.18 0 0.19 1	001	3
a_5	0.15	0.15 0 0.17 1	010	3
a_6	0.10	0 0.11 1	0110	4
a_7	0.01	1	0111	4

该哈夫曼码的平均码长为

$$\overline{K} = \sum_{i=1}^{7} p(a_i)K_i = 2.72 \text{ 码元 / 符号}$$

编码效率为

$$\eta = \frac{H(X)}{\overline{K}} = \frac{2.61}{2.72} = 96\%$$

由此可见，与例 5-4 的香农编码相比，哈夫曼码的平均码长比较小，编码效率高，信息传输速率大。所以在压缩信源信息率的实用设备中，哈夫曼编码还是比较常用的。

以上介绍的这种编码方法输出的是二进制哈夫曼码，如果要求编出 N 进制的哈夫曼码，则应在每次最小概率合并时取 N 个符号。另外，为了得到最短平均码长，尽量减少赋长码的信源符号，有时在编码前需要对信源符号作添加，使得信源的符号数量满足 $M(N-1)+1$，M 为正整数。添加的信源符号的概率为零。这样在多次合并后就能充分利用短码，以便降低平均码长。例如要将信源 $\begin{bmatrix} X \\ P \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 & a_4 \\ p_1 & p_2 & p_3 & p_4 \end{bmatrix}$ 编成三进制的哈夫曼码，如果直接编码，形成的码长为(1,2,2,2)。如果先对信源添加 1 个符号，变成 $\begin{bmatrix} X \\ P \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 & a_4 & a_5 \\ p_1 & p_2 & p_3 & p_4 & 0 \end{bmatrix}$，这时编码形成的码长为(1,1,2,2)。

哈夫曼编码方法得到的码并非是唯一的。造成非唯一的原因如下：

- 每次对信源缩减时，赋予信源最后两个概率最小的符号，用 0 和 1 是可以任意的，所以可以得到不同的哈夫曼码，但不会影响码字的长度。
- 对信源进行缩减时，两个概率最小的符号合并后的概率与其他信源符号的概率相同时，这两者在缩减信源中进行概率排序，其位置放置次序可以是任意的，故会得到不同的哈夫曼码。此时将影响码字的长度，一般将合并的概率放在上面，这样可获得较小的码方差。

例 5-7 设有离散无记忆信源

$$\begin{bmatrix} X \\ P \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 & a_4 & a_5 \\ 0.4 & 0.2 & 0.2 & 0.1 & 0.1 \end{bmatrix}$$

可有两种哈夫曼编码方法，如表 5-5 和表 5-6 所示，码树如图 5-7(a)和(b)所示。

表 5-5　哈夫曼编码方法一

信源符号 a_i	概率 $p(a_i)$	编码过程	码字 W_i	码长 K_i
a_1	0.4	0.4 0.4 0.6 0 1.0	1	1
a_2	0.2	0.2 0.4 0 0.4 1	01	2
a_3	0.2	0.2 0 0.2 1	000	3
a_4	0.1	0 0.2 1	0010	4
a_5	0.1	1	0011	4

表 5-6　哈夫曼编码方法二

信源符号 a_i	概率 $p(a_i)$	编码过程	码字 W_i	码长 K_i
a_1	0.4	0.4 0.4 0.6 0 1.0	00	2
a_2	0.2	0.2 0.4 0 0.4 1	10	2
a_3	0.2	0.2 0 0.2 1	11	2
a_4	0.1	0 0.2 1	010	3
a_5	0.1	1	011	3

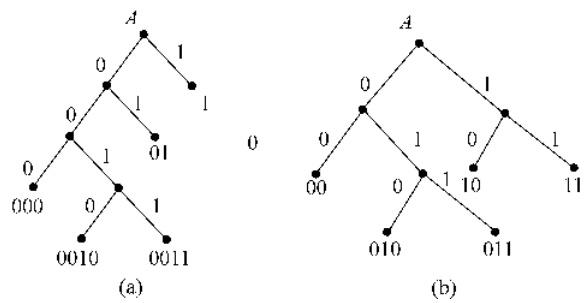


图 5-7　哈夫曼码树

由表 5-5 和表 5-6 给出的哈夫曼码的平均码长相等，即

$$\overline{K} = \sum_{i=1}^{7} p(a_i)K_i = 2.2 \text{ 码元 / 符号}$$

编码效率也相等，即

$$\eta = \frac{H(X)}{\overline{K}} = 96.5\%$$

但是两种码的质量不完全相同，可用码方差来表示，即

$$\sigma_l^2 = E[(k_i - \overline{K})^2] = \sum_{i=1}^{q} p(a_i)(k_i - \overline{K})^2$$

表 5-5 中哈夫曼码的方差为　$\sigma_{l1}^2 = 1.36$

表 5-6 中哈夫曼码的方差为　$\sigma_{l2}^2 = 0.16$

因此可见，第二种哈夫曼编码方法得到的码方差要比第一种哈夫曼编码方法得到的码方差小许多。故第二种哈夫曼码的质量要好。

从上述例子看出，进行哈夫曼编码时，为得到码方差最小的码，应使合并的信源符号位于缩减信源序列尽可能高的位置上，以减少再次合并的次数，充分利用短码。

哈夫曼码是用概率匹配方法进行信源编码。它有两个明显特点：一是哈夫曼码的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，充分利用了短码；二是缩减信源的最后二个码字总是最后一位不同，从而保证了哈夫曼码是即时码。

哈夫曼变长码的效率是相当高的，它可以单个信源符号编码或用 L 较小的信源序列编码，对编码器的设计来说也将简单得多。但是应当注意，要达到很高的效率仍然需要按长序列来计算，这样才能使平均码字长度降低。

例 5-8 信源输出两个符号，概率分布为 $P=(0.9,0.1)$，信源熵 $H(X)=H(0.9)=0.469$。采用二进制哈夫曼编码。

$L=1$，$\overline{K}_1=1$bit/符号；

$L=2$，$P'=(0.81,0.09,0.09,0.01)$，$\overline{K}_2=0.645$bit/符号；

$L=3$，$\overline{K}_3=0.533$bit/符号；

$L=4$，$\overline{K}_4=0.493$bit/符号。

随着序列长度 L 的增加，平均码长迅速降低，接近信源熵值。

但是对于信源的某一个符号而言，有时可能还会比定长码长。例如在例 5-7 中，信源符号有 5 个，采用定长码方式可用 3 个二进制符号组成码字。而用变长码时，有的码字却长达 4 个二进制符号。所以编码简单化的代价是要有大量的存储设备来缓冲码字长度的差异，这也是码方差小的码质量好的原因。设一秒送一个信源符号，输出的码字有的只有一个二进制符号，有的却有 5 个二进制符号，若希望平均每秒输出 $\overline{K}=2.61$ 个二进制符号以压缩信息率（与 3 个符号的定长码相比），就必须先把编成的码字存储起来，再按 $\overline{K}$ 的信息率输出，才能从长远来计算，输出和输入保持平衡。当存储量不够大时，就可能有时取空，有时溢出。例如信源常发出短码时，就会出现取空，就是说还没有存入就要输出。常发出长码时，就会溢出，就是存入太多，以致存满了还未取出就再要存入。所以应估计所需的存储器容量，才能使上述现象发生的概率小至可以容忍。

设 T 秒内有 N 个信源符号输出，信源输出符号速率 $S=N/T$，若符号的平均码长为 $\overline{K}$，则信道传输速率

$$R = S\overline{K} \tag{5-4-1}$$

时可以满足条件。

N 个码字的长度分别为 K_i，$i=1,2,\cdots,N$，即在此期间输入存储器 $\sum K_i$bit，输出至信道 RTbit，则在存储器内还剩 Xbit，即

$$X = \sum_{i=1}^{N} K_i - RT \tag{5-4-2}$$

已知 K_i 是随机变量，其均值和方差分别为

$$\overline{K} = E[K_i] = \sum_{j=1}^{m} p_j K_j \tag{5-4-3}$$

$$\sigma^2 = E[K_i^2] - \overline{K}^2 = \sum_{j=1}^{m} p_j K_j^2 - \overline{K}^2 \tag{5-4-4}$$

式中 m 是信源符号集的元数。当 N 足够大时，X 是许多同分布的随机变量之和。由概率论可知，它将近似于正态变量，其均值和方差分别为

$$E[X] = N\overline{K} - RT = (S\overline{K} - R)T$$

$$\sigma_x^2 = N\sigma^2$$

令
$$Y=\frac{X-E[X]}{\sigma_x} \tag{5-4-5}$$

它是标准正态变量，可得下列概率

$$P(Y > A) = P(Y < -A) = \varphi(-A) \tag{5-4-6}$$

上式中 $\varphi(-A)$是误差函数，可查表得其数值。

如果式(5-4-1)成立，则 $E[X]=0$。设起始时存储器处半满状态，而存储器容量为 $2A\sigma_x$，可由式(5-4-6)求得溢出概率和取空概率；因 $Y>A$，即 $X>A\sigma_x$，存储器将溢出；而 $Y<-A$，即 $X<-A\sigma_x$，存储器取空。这就是说，如果要求这些概率都小于 $\varphi(-A)$，存储器容量应大于 $2A\sigma_x$。例如要求溢出概率和取空概率都小于 0.001，查表得 A 应为 3.08，则存储器容量 C 应为

$$C > 6.16\sqrt{N}\sigma \tag{5-4-7}$$

当式(5-4-1)不成立时，存储器容量还要增加，在起始时存储器也不应处于半满状态。例如若 $R>S\overline{K}$，平均来说，输出大于输入，易被取空，起始状态可超过半满；反之，若 $R<S\overline{K}$，易于溢出，可不到半满。

由式(5-4-7)可见，时间 T 越长，N 越大，要求存储器的容量也越大。当容量设定后，随着时间的增长，存储器溢出和取空的概率都将增大；当 T 很大时，几乎一定会溢出或取空，造成损失；即使式(5-4-1)成立，也是如此。由此可见，对于无限长的信息，很难采用变长码而不出现差错。一般来说，变长码只适用于有限长的信息传输；即送出一段信息后，信源能停止输出，例如传真机送出一张纸上的信息后就停止。对于长信息，在实际使用时可把长信息分段发送；也可检测存储器的状态，发现将要溢出就停止信源输出，发现将要取空就插入空闲标志在信道上传送，或加快信源输出。

说变长编码可以无失真地译码，这是理想情况。如果这种变长码由信道传送时，有某一个符号错了。因为一个码字前面有某一个码元错了，就可能误认为是另一个码字而点断，结果后面一系列的码字也会译错，这常称为差错的扩散。当然也可以采用某些措施，使得错了一段以后，能恢复正常的码字分离和译码，这一般要求在传输过程中差错很少，或者加纠错用的监督码位，但是这样一来又增加了信息率。

此外，当信源有记忆时，用单个符号编制变长码不可能使编码效率接近于 1，因为信息率只能接近一维熵 H_1，而 H_∞ 一定小于 H_1。此时仍需要多个符号一起编码，才能进一步提高编码效率。但导致码表长、存储器多。

哈夫曼码在实际中已有所应用，但它仍存在一些分组码所具有的缺点。例如概率特性必须精确地测定，以此来编制码表，它若略有变化，还需更换码表。因而在实际的编码过程中，需要对原始数据扫描两遍，第一遍用来统计原始数据中各字符出现的概率，创建码表存放起来，第二遍则依据码表在扫描的同时进行编码，才能传输信息。如果将这种编码用于网络通信中，两遍扫描会引起较大的延时；如果用于数据压缩，则会降低速度。因此出现了自适应哈夫曼编码方法，其码表不是事先构造，而是随着编码的进行，不断动态地构造、调整，所以码表不仅取决于信源的特性，还与编码、解码过程相关。

另外，对于二元信源，常需多个符号合起来编码，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，尤其对于相关信源，不能令人满意，而合并的符号数增大时，码

表中的码字数很多，设备将越来越复杂。在大多数情况下，哈夫曼编码用于无失真编码，但也可以用于有失真情况。例如在符号数很多且有部分符号的概率非常小时，为了减小码表，可以将这些小概率符号合并对应同一个码字，在解码时出现的错误概率即为这些符号概率之和。

5.4.2 算术编码

以上所讨论的编码方法都是建立在符号和码字相对应的基础上的，这种编码通常称为块码或分组码。若对信源单符号进行编码，则符号间的相关性就无法考虑；若将 m 个符号合起来编码，一是会增加设备复杂度，二是 $m+1$ 个符号间以及组间符号的相关性还是无法考虑。这就使信源编码的匹配原则不能充分满足，编码效率就有所损失。

为了克服这种局限性，就需要跳出分组码的范畴，研究非分组码的编码方法。算术码即为其中之一，编码的基本思路是，将需要编码的全部数据看成某一 L 长序列，所有可能出现的 L 长序列的概率映射到[0,1]区间上，把[0,1]区间分成许多小段，每段的长度等于某一序列的概率。再在段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。这种方法与香农编码法有点类似，只是它们考虑的信源序列对象不同，算术码中的信源序列长度要长得多，或许是欲编码的整个数据文件，而香农码中的序列长度是 1。

如果信源符号集为 $A=\{a_1,a_2,\cdots,a_n\}$，L 长信源序列 $\boldsymbol{x}_i=(x_{i_1},x_{i_2},\cdots x_{i_l},\cdots x_{i_L})$，$x_{i_l}\in A$，共有 n^L 种可能序列。由于考虑的是全序列，也许是整页纸上的信息作为一个序列，因而序列长度 L 很大。实用中很难得到对应序列的概率，只能从已知的信源符号概率 $P=[p(a_1),p(a_2),\cdots,p(a_n)]=[p_1,p_2,\cdots,p_r,\cdots,p_n]$中递推得到。定义各符号的积累概率为

$$P_r=\sum_{i=1}^{r-1}p_i \tag{5-4-8}$$

显然，由上式可得 $P_1=0$，$P_2=p_1$，$P_3=p_1+p_2\cdots$，而且

$$p_r=P_{r+1}-P_r \tag{5-4-9}$$

由于 P_{r+1}和 P_r都是小于 1 的正数，可用[0,1]区间内的两个点来表示，则 p_r就是这两点间的小区间的长度，如图 5-8 所示。不同的符号有不同的小区间，它们互不重叠，所以可将这种小区间内的任一个点作为该符号的代码。以后将计算这代码所需的长度，使之能与其概率匹配。

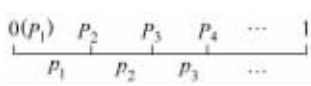


图 5-8

例如有一序列 $S=011$，这种 3 个二元符号的序列可按自然二进制数排列，000，001，010，…，则 S 的积累概率为

$$P(S)=p(000)+p(001)+p(010) \tag{5-4-10}$$

如果 S 后面接一个“0”，积累概率就成为

$$\begin{aligned}P(S,0)&=p(0000)+p(0001)+p(0010)+p(0011)+p(0100)+p(0101)\\&=p(000)+p(001)+p(010)=P(S)\end{aligned}$$

因为两个四元符号的最后一位是“0”和“1”时，根据归一律，它们的概率和应等于前 3 位的概率，即 $p(0000)+p(0001)=p(000)$等。

如果 S 后面接一个“1”，则其积累概率是

$$\begin{aligned}P(S,1) &= p(0000)+p(0001)+p(0010)+p(0011)+p(0100)+p(0101)+p(0110)\\ &= P(S)+p(0110)\\ &= P(S)+p(S)p_0\end{aligned}$$

由于单符号的积累概率为 $P_0=0, P_1=p_0$，所以上面两式可统一写作

$$P(S,r) = P(S)+p(S)P_r, r=0,1$$

这样写的式子很容易推广到多元序列($m>2$)，即可得一般的递推公式

$$P(S,a_r) = P(S)+p(S)P_r \tag{5-4-11}$$

以及序列的概率公式

$$p(S,a_r) = p(S)p_r$$

对于有相关性的序列，上面的两个递推公式也是适用的，只是上式中的单符号概率应换成条件概率。用递推公式可逐位计算序列的积累概率，而不用像式(5-4-10)那样列举所有排在前面的那些序列概率。

从以上关于积累概率 $P(S)$ 的计算中可看出，$P(S)$ 把区间[0,1)分割成许多小区间，每个小区间的长度等于各序列的概率 $p(S)$，而该小区间内的任一点可用来代表该序列，现在来讨论如何选择这个点。令

$$L = \left\lceil \log \frac{1}{p(S)} \right\rceil \tag{5-4-12}$$

其中⌈ ⌉代表大于或等于的最小整数。把积累概率 $P(S)$ 写成二进位的小数，取其前 L 位，以后如果有尾数，就进位到第 L 位，这样得到一个数 C。例如 $P(S)=0.10110001$，$p(S)=1/17$，则 $L=5$，得 $C=0.10111$。这个 C 就可作为 S 的码字。因为 C 不小于 $P(S)$，至少等于 $P(S)$。又由式(5-4-12)，可知 $p(S)\geqslant 2^{-L}$。令 $(S+1)$ 为按顺序正好在 S 后面的一个序列，则

$$P(S+1) = P(S)+p(S) \geqslant P(S)+2^{-L} > C$$

当 $P(S)$ 在第 L 位以后没有尾数时，$P(S)$ 就是 C，上式成立；如果有尾数时，该尾数就是上式的左右两侧之差，所以上式也成立。由此可见 C 必在 $P(S+1)$ 和 $P(S)$ 之间，也就是在长度为 $p(S)$ 的小区间(左闭右开的区间)内，因而是可以唯一译码。这样构成的码字，编码效率是很高的，因为已可达到概率匹配，尤其是当序列很长时。由式(5-4-12)可见，对于长序列，$p(S)$ 必然很小，L 与概率倒数的对数已几乎相等，也就是取整数所造成的差别很小，平均代码长度将接近 S 的熵值。

实际应用中，采用累积概率 $P(S)$ 表示码字 $C(S)$，符号概率 $p(S)$ 表示状态区间 $A(S)$，则有

$$\begin{cases} C(S,r) = C(S)+A(S)P_r \\ A(S,r) = A(S)p_r \end{cases} \tag{5-4-13}$$

对于二进制符号组成的序列，$r=0,1$。

实际编码过程是这样的。先置定两个存储器，起始时可令

$$A(\varphi)=1, \quad C(\varphi)=0$$

其中 φ 代表空集，即起始时码字为 0，状态区间为 1。每输入一个信源符号，存储器 C 和 A 就按照式(5-4-13)更新一次，直至信源符号输入完毕，就可将存储器 C 的内容作为该序

列的码字输出。由于 $C(S)$ 是递增的，而增量 $A(S)P_r$ 随着序列的增长而减小，因为状态区间 $A(S)$ 越来越小，与信源单符号的积累概率 P_r 的乘积就越来越小。所以 C 的前面几位一般已固定，在以后计算中不会被更新，因而可以边算边输出，只需保留后面几位用作更新。

译码也可逐位进行，与编码过程相似。

例 5-9 有 4 个符号 a、b、c、d 构成简单序列 $S=abda$，各符号及其对应概率如表 5-7 所示。

表 5-7 各符号及其对应概率

符号	符号概率 p_i	符号累积概率 P_j
a	0.100(1/2)	0.000
b	0.010(1/4)	0.100
c	0.001(1/8)	0.110
d	0.001(1/8)	0.111

算术编解码过程如下：

设起始状态为空序列 φ，则 $A(\varphi)=1$，$C(\varphi)=0$。

递推得

$$\begin{cases} C(\varphi a) = C(\varphi) + A(\varphi)P_a = 0 + 1 \times 0 = 0 \\ A(\varphi a) = A(\varphi)p_a = 1 \times 0.1 = 0.1 \end{cases}$$

$$\begin{cases} C(a,b) = C(a) + A(a)P_b = 0 + 0.1 \times 0.1 = 0.01 \\ A(a,b) = A(a)p_b = 0.1 \times 0.01 = 0.001 \end{cases}$$

$$\begin{cases} C(a,b,d) = C(a,b) + A(a,b)P_d = 0.01 + 0.001 \times 0.111 = 0.010111 \\ A(a,b,d) = A(a,b)p_d = 0.001 \times 0.001 = 0.000001 \end{cases}$$

$$\begin{cases} C(a,b,d,a) = C(a,b,d) + A(a,b,d)P_a = 0.010111 + 0.000001 \times 0 = 0.010111 \\ A(a,b,d,a) = A(a,b,d)p_a = 0.000001 \times 0.1 = 0.0000001 \end{cases}$$

计算该序列的编码码长，根据式(5-4-12)，有

$$L = \left\lceil \log \frac{1}{A(a,b,d,a)} \right\rceil = 7$$

得码长为 7，取 $C(a,b,d,a)$ 的小数点后 7 位即为编码后的码字 0101110。上述编码过程如图 5-9 所示，可用对单位区间的划分来描述。

该信源的熵为

$$H(X) = -\frac{1}{2}\log\frac{1}{2} - \frac{1}{4}\log\frac{1}{4} - 2 \times \frac{1}{8}\log\frac{1}{8} = 1.75\text{bit/ 符号}$$

编码效率 $\eta = \dfrac{1.75}{7/4} = 100\%$

译码可通过比较上述编码后的数值大小来进行，即判断码字 $C(S)$ 落在哪一个区间就可以得出一个相应的符号序列。据递推公式的相反过程译出每个符号。步骤如下：

$C(a,b,d,a)=0.0101110<0.1\in[0,0.1]$ 第一个符号为 a；

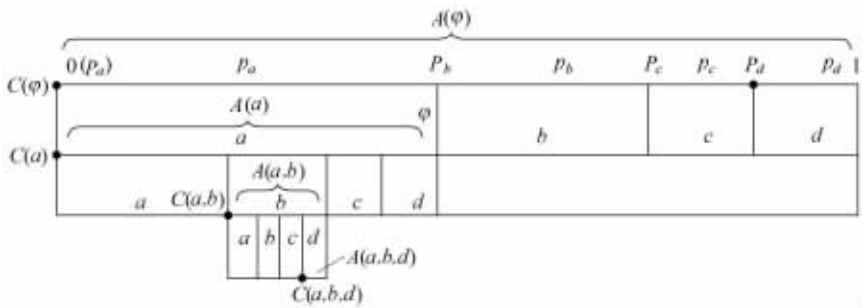


图 5-9　算术码编码过程

放大至$[0,1](\times p_{a-1})$：$C(a,b,d,a)\times 2^1=0.10111\in[0.1,0.110]$　第二个符号为 b；

去掉累积概率 P_b：　$0.10111-0.1=0.00111$；

放大至$[0,1](\times p_b^{-1})$：$0.00111\times 2^2=0.111\in[0.111,1]$　第三个符号为 d；

去掉累积概率 P_d：　$0.111-0.111=0$；

放大至$[0,1](\times p_d^{-1})$：$0\times 2^3=0\in[0,0.1]$　第四个符号为 a。

实际的编译码过程比较复杂，但原理相同。算术编码从性能上看具有许多优点，特别是所需的参数很少，不像哈夫曼编码那样需要一个很大的码表。由于二元信源的编码实现比较简单，我国最早将它应用于报纸传真的压缩设备中，获得了良好的效果。从理论上说，只要已知信源符号集及其符号概率，算术编码的平均码长可以接近符号熵。因而在实际编码时，需要预先对信源输入符号的概率进行估计，估计的精准程度将直接影响编码性能。但是事先知道精确的信源符号概率是很难的，而且是不切实际的。算术编码可以是静态的或是自适应的。在静态算术编码中，信源符号的概率是固定的。而对于一些信源概率未知或非平稳情况，常设计成自适应算术编码，在编码的过程中根据信源符号出现的频繁程度动态地修正符号概率。

5.4.3　LZ 编码

上述信源编码方法都需要精确已知信源的概率分布，一旦信源的实际分布与假设的分布有差异，编码性能就会急剧下降。但是在实际应用中，确切地获知信源统计特性有时是非常困难的，有时信源统计特性还会随时发生变化，因此就需要一种与信源统计特性无关的信源编码方法，称为通用信源编码。

1965 年苏联数学家柯尔莫戈洛夫(Kolmogolov)提出利用信源序列的结构特性来编码。而两位以色列研究者齐夫(Ziv)和伦佩尔(Lempel)独辟蹊径，完全脱离哈夫曼码和算术编码的设计思路，创造出了一系列比哈夫曼编码更有效，比算术编码更快捷的通用压缩算法。将这些算法统称为 LZ 系列算法。

Ziv 和 Lempel 于 1977 年提出了 LZ77 算法。1978 年，两人又提出了改进算法，后被命名为 LZ78 算法，该算法性能稍差，但易于实现。1984 年，韦尔奇(Welch)提出了 LZ78 算法的一个变种，即 LZW 算法。1990 年后，贝尔(Bell)等人又陆续提出了许多 LZ 系列算法的变体或改进版本。LZ 系列算法用一种巧妙的方式将字典技术应用于通用数据压缩领域，而且，可以从理论上证明 LZ 系列算法同样可以逼近信息熵的极限。

设信源符号集 $A=\{a_1,a_2,\cdots,a_K\}$ 共 K 个符号，设输入信源符号序列为 $U=(u_1,u_2,\cdots,u_L)$，编码是将此序列分成不同的段。分解是迭代进行的，在第 i 步，编码器从 s_{i-1} 短语后的第一个符号开始向后搜索在此之前未出现过的最短短语 s_i，将短语 s_i 添入字典第 i 段。由于 s_i 是此时字典中最短的新短语，所以 s_i 在去掉最后一个符号 x 后所得的前缀必定是字典中之前已经出现过的。若设此前缀是在第 $j(<i)$ 步时出现的，则对 s_i 的编码就可利用 j 和 s_i 最后一位符号 x 来表示，即为码字 (j,x)。对于段号 j，最多需要 $\lceil \log i \rceil$ bit 表示，而符号 x 只需 $\lceil \log K \rceil$ bit。若编码后的字典中短语共有 $M(U)$ 个，则 U 序列编码后输出的码流总长度为 $\sum_{i=1}^{M(U)}(\lceil \log i \rceil+\lceil \log K \rceil)$。

例 5-10 信源符号集 $A=\{a,b\}$，输入信源符号序列 $U=(abbabaabbabbaaaaba\cdots)$，编码输出二进制码流。

如表 5-8 所示，对输入序列进行分段。最先出现的是单符号 a 和 b，分别赋予段号 1 和 2，由于是单符号，没有前缀，因而码字中的 j 赋 0，则对应段号 1 和段号 2 的码字分别为 $(0,a)$ 和 $(0,b)$。接着信源序列出现符号 b，由于之前字典中已有，所以最短的新短语应为 ba，为段号 3，前缀 b 为段号 2，因此对应的码字为 $(2,a)$。按照这样的规则分解序列，直至最后。由于最终需要编成二进制码，故将信源符号 a 编码为 0，符号 b 编码为 1，再将段号用 $\lceil \log i \rceil$ 长度的二进制数表示，最后得到输出的二进制码流为 0001100110010100111000 1100…。

表 5-8 LZ 编码示例

短语	a	b	ba	baa	bb	ab	$baaa$	aba
段号	1	2	3	4	5	6	7	8
码字	$(0,a)$	$(0,b)$	$(2,a)$	$(3,a)$	$(2,b)$	$(1,b)$	$(4,a)$	$(6,a)$
二进制码	(0,0)	(0,1)	(10,0)	(11,0)	(010,1)	(001,1)	(100,0)	(110,0)

LZ 编码的编码方法非常简捷，译码也很简单，可以一边译码一边建立字典。译码时若收到的码字为 (j,x)，则在字典中找到第 j 个短语，然后加上符号 x 即可译出对应的新短语，并添入字典。因此发送时无需传输字典本身。从上例中看到，编码后输出的码流较长，编码效率不是很高，这是由于信源序列长度短，当编码的信源序列增长时，编码效率会提高。可以证明，LZ 编码的输出速率可以达到信源极限熵。

LZ 编码算法逻辑简单，硬件实现廉价，运算速度快，被 ITU 数据传输标准 V.42 所采用，并在很多计算机数据存储中得到应用，如用于计算机文件压缩的 WinZip、WinRAR 等工具。其优点在于能够有效地利用信源输出序列字符的频率、重复性和高使用率的冗余度，是一种自适应算法，只需对信源序列进行一次扫描，无须知道信源的先验统计特性，运算时间正比于序列长度。但也有缺点，一是不能有效利用位置的冗余度；二是该算法通常在序列起始段压缩效果差一些，随着长度增加效果变好。

5.4.4 游程编码

在二元序列中，只有两种符号，即“0”和“1”，这些符号可连续出现，连“0”这一段称为“0”游程，连“1”这一段称为“1”游程。它们的长度分别称为游程长度 $L(0)$ 和 $L(1)$。“0”游程和“1”游程总是交替出现的。如果规定二元序列是以“0”开始，第一个游程是“0”游程，第二个

必为“1”游程，第三个又是“0”游程……。对于随机的二元序列，各游程长度将是随机变量，其取值可为 1,2,3,…，直到无限。将任何二元序列变换成游程长度序列，这种变换是一一对应的，也就是可逆的。例如有一个二元序列 000101110010001…，可变换成下列游程序列：3113213…。

若已知二元序列是以“0”起始的，从上面的游程序列很容易恢复成原来的二元序列，包括最后一个“1”，因为长度为 3 的“0”游程之后必定是“1”。游程序列已是多元序列，各长度就可按霍夫曼编码或其他方法处理以达到压缩码率的目的。这种从二元序列转换成多元序列的方法，在实现时比前面的并元法简单。因为游程长度的计数比较容易，得到游程长度后就可从码表中找出码字输出，同时去数下一个游程长度。此外，在减弱原有序列的符号间的相关性方面，采用游程变换一般也比并元法更有效。当然，要对二元序列进行霍夫曼编码时，应先测定“0”游程长度和“1”游程长度的概率分布，或由二元序列的概率特性去计算各种游程长度的概率。

对于多元序列也存在相应的游程序列。例如 m 元序列中，可有 m 种游程。连着出现符号 a_r 的游程，其长度 $L(r)$ 就是“r”游程长度，这也是一个随机变量。用 $L(r)$ 也可构成游程序列，但是这种变换必须再加一些符号，才能成为一一对应或可逆的，与二元序列变换所得的游程序列不同，这里每个“r”游程的前面和后面出现什么符号是不确定的，除 r 外的任何符号都是可能的，因此这一游程之后是何种符号的游程就无法确定，除非插入一个标志说明后一游程的类别。所以把多元序列变换成游程序列再进行压缩编码是没有多大意义的，因为上述的附加标志可能抵消压缩编码所得的好处，对原来的多元序列直接编码，或许会更有效一些。

游程编码仍是变长码，有其固有的缺点，即需有大量的缓冲和优质的信道。此外，由于游程长度可从 1 直到无限，这在码字的选择和码表的建立方面都有困难，实际应用时尚需采取某些措施来改进。

一般情况下，游程长度越大，其概率越小；这在以前的计算中也可看到，而且将随长度的增大渐趋向零。对于小概率的码字，其长度未达到概率匹配或较长，损失不会太大，也就是对平均码字长度影响较小。这样就可对长游程不严格按霍夫曼码步骤进行；在实际应用时，常采用截断处理的方法。

游程编码只适用于二元序列，对于多元信源，一般不能直接利用游程编码，但在下面介绍的冗余位编码，也可认为是游程编码在多元信源的一种应用。

在许多信源序列中，常有不少符号不携带信息，除了它的数目或所占时长外，完全可以不传送。例如在电话通信中，讲话时常有间隙，如字句间的停顿，听对方讲话而静默；又如图像信源中，背景基本上不变，并在图像中占相当大一部分，而其值为常量相当于平均亮度，一般也可以不传送；在数据信源序列中，信息包间的间歇或某种固定模式，也属于冗余性质。这些符号可称为**冗余位**，若能删除它们，可得较大的压缩比。

设有多元信源序列

$$x_1, x_2, \cdots, x_{m1}, y, y, \cdots, y, x_{m1+1}, x_{m1+2}, \cdots x_{m2}, y, y, \cdots \tag{5-4-14}$$

其中 x 是含有信息的代码，取值于 m 元符号集 A，可称为信息位；y 是冗余位，它们可为全零，即使未曾传送在接收端也能恢复的。这样的序列可用下列两个序列来代替

$$111, \cdots, 100, \cdots, 000111, \cdots, 111000$$

和

$$x_1, x_2, \cdots, x_{m1}, x_{m1+1}, x_{m1+2}, \cdots x_{m2}, \cdots \tag{5-4-15}$$

前一个序列中，用“1”表示信息位，用“0”表示冗余位；后一个序列是取消冗余位后留下的所有信息位。显然，从式(5-4-14)变换成式(5-4-15)中的两个序列是一一对应的，也就是可逆的。如果把式(5-4-15)中的两个序列传送出去，只要没有差错，在接收端就可恢复式(5-4-14)中的多元信源序列。这样就把一个多元序列分解为一个二元序列和一个缩短了的多元序列。它们可用不同的方法来编码以利于更有效地压缩码率。

5.4.5 矢量量化编码

连续信源进行编码的主要方法是量化，即将连续的样值 x 离散化成为 y_i，$i=1,2,3,\cdots,n$。n 是量化级数，y_i 是某些实数。这样就把连续值转化为 n 个实数，可用 $0,1,2,\cdots,n-1$ 等 n 个数字来表示。离散信源也会涉及量化的问题，比如当提供的量化级数少于原来的量化级数时，也需要对该信源信号进行再次量化。在上述的这些量化中，由于 x 是一个标量，因此称为标量量化。矢量量化就是将若干个标量数据组构成一个矢量，然后在矢量空间进行整体量化，从而压缩数据。量化会引入失真，所以矢量量化是一种限失真编码，量化时必须使这些失真最小。正如前面的编码定理中看到的，将离散信源的多个符号联合编码可提高效率。连续信源也是如此，当把多个信源符号联合起来形成多维矢量，再对矢量进行标量量化时，可以充分利用各分量间的统计依赖性，同样的失真下，量化级数可进一步减少，码率可进一步压缩。在维数足够高时，矢量量化编码可以任意接近率失真理论所给出的极限。

矢量量化编码的原理是，输入 k 维随机矢量 $\boldsymbol{X}_i=(x_{i1},x_{i2},\cdots,x_{ik})$，通过一个矢量量化器 $Q(\boldsymbol{X})$，映射成对应的 k 维输出矢量 $\boldsymbol{Y}_i=(y_{i1},y_{i2},\cdots,y_{ik})$。$\boldsymbol{Y}=\{\boldsymbol{Y}_1,\boldsymbol{Y}_2,\cdots,\boldsymbol{Y}_N\}$，共有 N 种矢量组成的集合 $\boldsymbol{Y}$ 称为码书或码本。它实际上是一个长度为 N 的表，表中的每个分量 $\boldsymbol{Y}_i$ 都是一个 k 维矢量，称为码字或码矢。码书中码字的数量就称为码书的尺寸。矢量编码的过程就是在码书 $\boldsymbol{Y}$ 中搜索一个与输入矢量 $\boldsymbol{X}_i$ 最接近的码字 $\boldsymbol{Y}_i$，$\boldsymbol{Y}_i$ 就是 $\boldsymbol{X}_i$ 的矢量量化值。传输时，只需传输码字 $\boldsymbol{Y}_i$ 的下标 i。在接收端解码器中，有一个与发送端相同的码书 $\boldsymbol{Y}$，根据接收的标号 i 可简单地用查表法找到对应矢量 $\boldsymbol{Y}_i$ 作为 $\boldsymbol{X}_i$ 的近似。当码书尺寸为 N 时，传输矢量下标所需的比特数为 $\log_2 N$，平均传输矢量中一维信号所需的比特数为 $(1/k)\log_2 N$。若 $k=16$，$N=256$，则比特率为 0.5bit/维。

从编码原理中可以看出，矢量量化编码的关键技术就是码书设计和码字搜索。

1. *码书设计*

矢量量化的码书设计是把 k 维空间无遗漏地划分成 N 个互不相交的子空间 $S_1,S_2,\cdots,S_N$，并在每个子空间中找出一个最佳的矢量 $\boldsymbol{Y}_i$ 作为输出码字。码书的优化直接影响压缩效率和数据恢复质量。在接收端以量化值 $\boldsymbol{Y}_i$ 再现 $\boldsymbol{X}_i$，必然存在失真，需要满足 $d(\boldsymbol{X}_i,\boldsymbol{Y}_i)=\min(d(\boldsymbol{X}_i,\boldsymbol{Y}_j))$，$j=1,2,\cdots,N$，其中 $d(\boldsymbol{X}_i,\boldsymbol{Y}_i)$ 是输入矢量 $\boldsymbol{X}_i$ 与码字 $\boldsymbol{Y}_i$ 之间的失真测度。一般可以采用均方误差来衡量失真测度，即

$$d(\boldsymbol{X}_i,\boldsymbol{Y}_i)=\sum_{j=1}^{k}(x_{ij}-y_{ij})^2$$

整个信号的平均失真为

$$D = E[d(\boldsymbol{X},\boldsymbol{Y}_i)] = \sum_{i=1}^{N} E[d(\boldsymbol{X},\boldsymbol{Y}_i),\boldsymbol{X} \in S_i]$$

D 为每个子空间失真的统计平均，可作为衡量恢复信号质量的指标。解码失真的大小主要由码书的质量决定。码书设计的过程就是寻求把 M 个训练矢量分成 $N(N<M)$ 类的一种最佳方案(使得均方误差最小)，而把各类的质心矢量作为码书的码字。然而，在 N 和 M 比较大的情况下，搜索全部码书是根本不可能的。为了克服这个困难，各种现有的码书设计方法都采取搜索部分码书的方法得到局部最优或接近全局最优的码书。所以研究码书设计算法的目的就是寻求有效的算法尽可能找到全局最优或接近全局最优的码书以提高码书的性能，并尽可能降低计算复杂度。

LBG 算法是一种直观且有效的矢量量化码书设计算法，是由 Linde、Buzo 和 Gray 于 1980 年首先提出来的。该算法基于最佳矢量量化器设计的最佳划分和最佳码书两个必要条件，是最佳标量量化在矢量空间的推广，其物理概念清晰，算法理论严密，算法实现容易。后来人们又针对该算法的一些缺点进行改进，提出了许多性能更优的设计算法，至今仍广泛应用。

2. *码字搜索*

矢量量化的码字搜索算法就是在码书已经存在的情况下，对某个输入矢量，在码书中搜索与该输入矢量之间失真最小的码字。矢量量化中最常用的搜索方法是全搜索算法和树搜索算法。全搜索算法与码书生成算法基本相同。如果采用平方误差作为失真测度，对于 k 维矢量，每次失真计算需要 k 次乘法，$2k-1$ 次加法，因而为了对矢量进行穷尽搜索编码需要 Nk 次乘法、$N(2k-1)$ 次加法和 $N-1$ 次比较。计算复杂度由码书尺寸和矢量维数决定。对于大尺寸码书和高维矢量，计算复杂度会很大。研究码字搜索算法的主要目的就是寻找快速有效的算法以减少计算复杂度，并且尽量使得算法易于用硬件实现。

随着算法研究的进展以及超大规模集成电路技术的飞速发展，矢量量化编码器在语音编码、语音识别与合成、图像压缩等领域被广泛应用。实验证明，即使各信源符号相互独立，多维量化也可压缩信息率，这就使矢量量化成为当前连续信源编码研究的一个热点。可是当维数较大时，矢量量化尚无解析方法，只能求助于数值计算；而且联合概率密度也不易测定，还需采用训练序列等方法。一般来说，高维矢量联合很复杂，虽已有不少方法，在实用时尚有不少困难，有待进一步研究。

5.4.6 预测编码

前面介绍的编码方法都是考虑独立的信源序列。霍夫曼码对于独立多值信源符号很有效；二元序列的游程编码实际上是为了把二值序列转化成多值序列以适应霍夫曼编码；多个二元符号合并成一个符号的方法也有类似的情况。算术码对于独立二元信源序列是很有效的，对于相关信源虽然可采用条件概率来编码，以达到高效率，但这样做所引起的复杂度，往往使之难以实现。由信息论可知，对于相关性很强的信源，条件熵可远小于无条件熵，因此人们常采用尽量解除相关性的办法，使信源输出转化为独立序列，以利于进一步压缩码率。

常用的解除相关性的两种措施是**预测**和**变换**。它们既适应于离散信源，也可用于连续信源。其实两者都是序列的变换。一般来说，预测有可能完全解除序列的相关性，但必须确

知序列的概率特性；变换编码一般只解除矢量内部的相关性，但它可有许多可供选择的变换矩阵，以适应不同信源特性。这在信源概率特性未确知或非平稳时可能有利。

本节介绍预测的一般理论和方法。

预测就是从已收到的符号中提取关于未收到的符号的信息，从而预测其最可能的值作为预测值，并对它与实际值之差进行编码，达到进一步压缩码率的目的。由此可见，预测编码是利用信源的相关性来压缩码率的，对于独立信源，预测就没有可能。

预测的理论基础主要是**估计理论**。**估计**就是用实验数据组成一个统计量作为某一物理量的估值或预测值。最常见的估计是利用某一物理量在被干扰下所测定的实验值，这些值是随机变量的样值，可根据随机量的概率分布得到一个统计量作为估值。若估值的数学期望等于原来的物理量，就称这种估计为**无偏估计**；若估值与原物理量之间的均方误差最小，就称之为**最佳估计**。用来预测时，这种估计就成为均方误差最小的预测，所以也就认为这种预测是最佳的。

要实现最佳预测就需要找到计算预测值的预测函数。设有信源序列 $x_1, x_2, \cdots, x_r, x_{r+1}, \cdots$。$r$ 阶预测就是由 $x_1, x_2, \cdots, x_r$ 来预测 x_{r+1}。可令预测值为

$$x'_{r+1} = f(x_1, x_2, \cdots, x_r)$$

其中 f 是待定的预测函数。要使预测值具有最小均方误差，必须确知 $r+1$ 个变量（$x_1, x_2, \cdots, x_r, x_{r+1}$）的联合概率密度函数，这在一般情况下是困难的。因而常用线性预测的方法来达到次最佳的结果。线性预测就是预测函数为各已知信源符号的线性函数，即 x_{r+1} 的预测值

$$x'_{r+1} = f(x_1, x_2, \cdots, x_r) = \sum_{s=1}^{r} a_s x_s \tag{5-4-16}$$

并求均方误差

$$D = E(x'_{r+1} - x_{r+1})^2 \tag{5-4-17}$$

最小时的各 a_s 值。可将式(5-4-16)代入式(5-4-17)，对各 a_s 取偏导并置零，得到

$$\frac{\partial D}{\partial a_s} = -E\left\{(x_{r+1} - \sum_{s=1}^{r} a_s x_s) x_s\right\} = 0$$

只需已知信源各符号之间的相关函数即可进行运算。

最简单的预测是令

$$x'_{r+1} = x_r$$

这可称为零阶预测，常用的差值预测就属这类。高阶线性预测已在语音编码，尤其是声码器中广泛采用。如果信源是非平稳的或非概率性的，无法获得确切和恒定的相关函数，不能构成线性预测函数，可采用自适应预测的方法。一种常用的自适应预测方法是设预测函数是前几个符号值的线性组合，即令预测函数为

$$x' = \sum_{s=1}^{r} a_s x_{t-r-1-s}$$

再用已知信源序列来确定各系数 a_s，使对该序列所造成的均方误差 D 最小。此时的各系数 a_s 并不能保证对该信源发出的所有序列都适用，只有在平稳序列情况下，这种预测的均方误差可逼近线性预测时的最小值。随着序列的延长，各系数 a_s 可根据以后的 n 个符号值来

计算，因而将随序列的延长而变更，也就是可不断适应序列的变化，适用于缓变的非平稳信源序列。

利用预测值来编码的方法可分为两类：一类是用实际值与预测值之差进行编码，也称为差值编码。常用于相关性强的连续信源，也可用于离散信源。在连续信源的情况下，就是对此差值进行量化或取一组差值进行矢量量化。由于相关性很强的信源可较精确地预测待编码的值，该差值的方差将远小于原来的值，所以在同样失真要求下，量化级数可明显地减少，从而较显著地压缩码率。对于离散信源也有类似的情况。

另一类方法是根据差值的大小，决定是否需传送该信源符号。例如可规定某一可容许值 ε，当差值小于该值时可不传送。对于连续函数或相关性很强的信源序列，常有很长一串符号可以不传送而只需传送这串符号的个数，这样能大量压缩码率。这类方法一般是按信宿要求设计的，也就是失真应能满足信宿需求。

5.4.7 变换编码

变换是一个广泛的概念。在通信系统中，常希望把信号进行变换以达到某一目的。信源编码实际上就是一种变换，使之能在信道中更有效地传送。这里将讨论的变换是数学意义上的一一对应变换。**变换编码**就是经过变换后的信号的样值能更有效地编码，也就是通过变换来解除或减弱信源符号间的相关性，再将变换后的样值进行标量量化，或采用对于独立信源符号的编码方法，以达到压缩码率的目的。

首先讨论变换的一般原理，即连续函数的变换。

设有函数 $f(t)$，$0<t<T$

$$\int_0^T f^2(t)\mathrm{d}t < \infty \tag{5-4-18}$$

该函数是希尔伯特空间 $L^2(0,T)$ 的一个矢量，其维数是可数无限，它的坐标系将可用一个完备正交函数系来表征。

设有一个完备正交归一函数系 $\varphi(i,t)$，$i=0,1,2,\cdots$。正交性就是

$$\int_0^T \varphi(i,t)\varphi(j,t)\mathrm{d}t = 0, \quad i \neq j \tag{5-4-19}$$

归一性就是

$$\int_0^T \varphi^2(i,t)\mathrm{d}t = 1 \tag{5-4-20}$$

则可把 $f(t)$ 展开为

$$f(t) = \sum_{i=0}^{\infty} a_i\varphi(i,t) \tag{5-4-21}$$

其中 a_i 是待定系数，可用有限项逼近时的均方误差最小准则来求。即

$$D_n = \int_0^T \left[f(t) - \sum_{i=0}^{n-1} a_i\varphi(i,t) \right]^2 \mathrm{d}t$$

$$\frac{\partial D_n}{\partial a_i} = \int_0^T -2\left[f(t) - \sum_{i=0}^{n-1} a_i\varphi(i,t) \right]\varphi(i,t)\mathrm{d}t$$

利用函数 $\varphi(i,t)$ 的正交归一性式(5-4-19)和式(5-4-20)，可得

$$a_i = \int_0^T f(t)\varphi(i,t)\mathrm{d}t \tag{5-4-22}$$

如果

$$\lim_{n\to\infty} D_n \to 0$$

称为上述正交函数系是**完备**的，此时式(5-4-21)才成立。不然就是不完备的，因而式(5-4-22)也就不成立。

与欧几里德空间类比，可见式(5-4-21)实际上就是把函数矢量分解成各坐标分量，式(5-4-22)就相当于内积运算，把函数 $f(t)$ 投影到 $\varphi(i,t)$ 上去。

通过上述变换，就把函数 $f(t)$ 变换成一系列离散的系数 a_i，若已给定这些系数，就可用式(5-4-21)恢复函数 $f(t)$ 而不产生误差，所以这种变换是可逆的。如果只取有限个系数，恢复时就会引入误差。

我们所熟悉的傅里叶变换具有正交归一性函数系，但从解除相关性的意义上说，傅里叶变换不是一种很好的变换。要有效地解除相关性，正交函数系必须根据信源的相关函数来选择。

按均方误差最小准则来推算，有一种正交变换叫做 K-L 变换（Karhunen-Loeve transform），可使变换后的随机变量之间互不相关。一般认为 K-L 变换是压缩编码的最佳变换，评价其他变换时，常与它进行比较。K-L 变换的最大缺点是计算复杂，除了需测定相关函数和解积分方程外，变换时的运算也十分复杂，尚无快速算法可用。

以上的变换是在时间上连续的信源输出 $x(t)$ 中取一段 $(0,T)$ 进行积分运算，得到一系列系数 a_i，$i=0,1,2,\cdots$，截取有限个 n（即 $i=0,1,2,\cdots,n-1$）并对各 a_i 进行量化，达到信源编码的目的。这种方法在实际编码时较少应用，因为积分运算一般来说是比较困难的，而且除了量化各系数时将引入失真外，截取有限个系数也会引入失真。要保持失真在某一限度内，可能量化级数要有一定的增多，从而使码率有所上升。

另一种方法是先对信源输出 $x(t)$ 取样，得到一系列离散值 $x(i)$，$i=0,1,2,\cdots$，然后取 N 个样值形成一个 N 维矢量，对该矢量用矩阵进行变换，成为另一域内的 N 维矢量，以解除或减弱矢量内各分量的相关性。再对后一个分量进行标量量化或对矢量进行矢量量化来完成信源编码。此时的变换已不用积分运算而是用矩阵运算。若变换所用的矩阵选得恰当，就可达到压缩码率的要求。用矩阵来变换常称为**离散变换**。

其实取样也是一种把连续函数变换成时间上离散的一系列值的变换。此时变换所用的正交函数是单元脉冲函数 $\delta(t-i,\tau)$，$i=0,1,2,\cdots$，τ 是取样间隔。单元脉冲函数系的正交性和完备性是明显的，但这里已不是截取一段 $(0,T)$ 信源输出而是连续进行取样运算。要使变换能一一对应，也就是能无失真地恢复原来的连续函数，信源输出必须是限频的。若其最高频率是 f_m，则取样间隔 τ 必须小于 $1/2f_m$，才能使变换可逆，不然将引入失真。实际上连续信源常是限频的，尤其对信宿来说，频率大于一定值的含量，信宿已不感兴趣或已不能感受，语音和图像对人耳与人眼分别都有这种情况，所以限频的要求常是能满足的。这样既避免了积分运算，也不致引入额外失真，因此实际上常采用离散变换。

上面提到的傅里叶变换就有其相应的离散变换，只需将以前的正交函数取样即得。令取样点为 $t=kT/N$，$k=0,1,2,\cdots,N-1$，变换矩阵的元为

$$a_{ik} = w^{ik} / \sqrt{N}$$

其中 $w = e^{j2\pi/N}$，变换和反变换写成矩阵形式分别为

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{N-1} \end{bmatrix} = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & w & w^2 & \cdots & w^{(N-1)} \\ 1 & w^2 & w^4 & \cdots & w^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & w^{N-1} & w^{2(N-1)} & \cdots & w^{(N-1)^2} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N-1} \end{bmatrix} \tag{5-4-23}$$

$$\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N-1} \end{bmatrix} = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & w^* & w^{*2} & \cdots & w^{*(N-1)} \\ 1 & w^{*2} & w^{*4} & \cdots & w^{*2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & w^{*N-1} & w^{*2(N-1)} & \cdots & w^{*(N-1)^2} \end{bmatrix} \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{N-1} \end{bmatrix} \tag{5-4-24}$$

经傅里叶变换后的输出各分量间的相关系数将与原输入过程的相关函数有关。一般来说，输入过程的相关系数越接近 1，输出各分量间的相关函数越小，也就是说傅里叶变换对强相关的信源是有效的。此外各输出分量的方差将不同，有大有小，即经变换后能量有所集中，这对压缩码率也是有利的。

离散傅里叶变换虽有快速算法（FDFT 或 FFT）可减少计算量，但运算将在复数域内进行，这是不方便的，在实用中常用离散余弦变换（DCT）。尤其是对视频图像信号，其统计特性接近一阶马尔可夫链，离散余弦变换的正交矢量近似于相应的 K-L 变换的正交矢量。

余弦变换的完备正交归一函数系是

$$\varphi(0,t) = \frac{1}{\sqrt{N}}$$

$$\varphi(i,t) = \sqrt{\frac{2}{N}} \cos \frac{\pi(2i+1)}{2T}, \quad t \in (0,T)$$

对这些函数在 $(0,T)$ 内取 N 个样值，即得离散余弦变换矩阵的元

$$a_{0k} = 1/\sqrt{N}$$

$$a_{ik} = \sqrt{(2/N)} \cos[(2k+1)i\pi/N]$$

变换和反变换的矩阵形式就分别为

$$\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{N-1} \end{bmatrix} = \frac{2}{\sqrt{N}} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \cdots & \frac{1}{\sqrt{2}} \\ \cos\frac{\pi}{2N} & \cos\frac{3\pi}{2N} & \cdots & \cos\frac{2N-1}{2N}\pi \\ \vdots & \vdots & \ddots & \vdots \\ \cos\frac{N-1}{2N}\pi & \cos\frac{3(N-1)}{2N}\pi & \cdots & \cos\frac{(2N-1)(N-1)}{2N}\pi \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{bmatrix}$$

$$\begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{bmatrix} = \frac{2}{\sqrt{N}} \begin{bmatrix} \frac{1}{\sqrt{2}} & \cos\frac{\pi}{2N} & \cdots & \cos\frac{N-1}{2N}\pi \\ \frac{1}{\sqrt{2}} & \cos\frac{3\pi}{2N} & \cdots & \cos\frac{3(N-1)}{2N}\pi \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{\sqrt{2}} & \cos\frac{2N-1}{2N}\pi & \cdots & \cos\frac{(2N-1)(N-1)}{2N}\pi \end{bmatrix} \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{N-1} \end{bmatrix}$$

在离散变换中，最佳变换也是 K-L 变换。其正交矢量系和变换矩阵可根据输入矢量各分量间的相关系数来求，而不用解积分方程，只需求相关矩阵的特征值和特征矢量。容易验证经过 K-L 变换后输出矢量的相关系数为零，即它能完全解除输出矢量间的线性相关性，且各分量的方差就是各特征值，它们各不相等，下降很快。在实际编码时，后面几个分量，方差已很小，往往可以不传送，有利于压缩编码。

还有很多离散变换，如正反变换矩阵都相同的离散哈尔（Harr）变换和离散沃尔什（Walsh）变换；由有限维正交矢量系导出的广泛用于电视信号编码的斜变换和多重变换；可把信号分割成多个窄带以解除或减弱信号样值间相关性的子带编码和小波变换等。在实际应用中，需要根据信源特性来选择变换方法以达到解除相关性、压缩码率的目的。另外，还可以根据一些参数来比较各种变换方法间的性能优劣，如反映编码效率的编码增益、反映编码质量的块效应系数等。当信源的统计特性很难确知时，可用各种变换分别对信源进行变换编码，然后用实验或计算机仿真来计算这些参数。

本章小结

本章从信源编码的模型出发，介绍了信源编码的目的，引出了信息传输速率和编码效率的概念，重点论述了无失真信源编码定理，从而引出了几种最佳编码方法，并简单介绍了限失真编码定理。

编码的定义：分组码、变长码、非奇异码、唯一可译码、即时码、非延长码。

唯一可译码存在的充分和必要条件，克劳夫特不等式：$\sum_{i=1}^{n} m^{-K_i} \leqslant 1$

编码效率：$\eta = \dfrac{H_L(\boldsymbol{X})}{\overline{K}}$

无失真信源编码定理（香农第一编码定理）：

定长编码定理：$\dfrac{K_L}{L}\log m \geqslant H_L(\boldsymbol{X}) + \varepsilon$

变长编码定理：$\dfrac{LH_L(\boldsymbol{X})}{\log m} \leqslant \overline{K_L} < \dfrac{LH_L(\boldsymbol{X})}{\log m} + 1$

最佳变长码：香农（Shannon）编码

限失真信源编码定理（香农第三编码定理）：$R(D) \leqslant \overline{K} < R(D) + \varepsilon$

几种常用信源编码方法：哈夫曼（Huffman）编码、算术编码、LZ 编码、游程编码、矢量量化编码、预测编码、变换编码。

习题

5-1 将某六进制信源进行二进制编码如下，试问：

消息	概率	C_1	C_2	C_3	C_4	C_5	C_6
u_1	1/2	000	0	0	0	1	01
u_2	1/4	001	01	10	10	000	001
u_3	1/16	010	011	110	1101	001	100
u_4	1/16	011	0111	1110	1100	010	101
u_5	1/16	100	01111	11110	1001	110	110
u_6	1/16	101	011111	111110	1111	110	111

（1）这些码中哪些是唯一可译码？

（2）哪些码是非延长码（即时码）？

（3）对所有唯一可译码求出其平均码长和编码效率。

5-2 已知信源的各个消息分别为字母 A、B、C、D，现用二进制码元对消息字母作信源编码，$A\rightarrow(x_0,y_0)$，$B\rightarrow(x_0,y_1)$，$C\rightarrow(x_1,y_0)$，$D\rightarrow(x_1,y_1)$，每个二进制码元的长度为 5ms。

（1）若各个字母以等概率出现，计算在无扰离散信道上的平均信息传输速率。

（2）若各个字母的出现概率分别为 $P(A)=1/5$，$P(B)=1/4$，$P(C)=1/4$，$P(D)=3/10$，再计算在无扰离散信道上的平均信息传输速率。

（3）若字母消息改用四进制码元作为信源编码，码元幅度分别为 0、1V、2V、3V，码元长度为 10ms。重新计算（1）和（2）两种情况下的平均信息传输速率。

5-3 设信道的基本符号集合 $A=\{a_1,a_2,a_3,a_4,a_5\}$，它们的时间长度分别为 $t_1=1$，$t_2=2$，$t_3=3$，$t_4=4$，$t_5=5$（个码元时间）。用这样的信道基本符号编成消息序列，且不能出现 (a_1,a_1)，(a_2,a_2)，(a_1,a_2)，(a_2,a_1) 这 4 种符号相连的情况。

（1）若信源的消息集合为 $\{x_1,x_2,x_3,\cdots,x_7\}$，它们的出现概率分别为 $P(x_1)=1/2$，$P(x_2)=1/4$，$P(x_3)=1/8$，$P(x_4)=1/16$，$P(x_5)=1/32$，$P(x_6)=P(x_7)=1/64$。试按最佳编码原则利用上述信道来传输这些消息时的信息传输速率；

（2）求上述信源编码的编码效率。

5-4 若消息符号、对应概率分布和二进制编码如下：

消息符号：	u_0	u_1	u_2	u_3
概率：	1/2	1/4	1/8	1/8
编码：	0	10	110	111

试求：（1）消息符号熵；

（2）每个消息符号所需的平均二进码个数；

（3）若各消息符号间相互独立，求编码后对应的二进码序列中出现“0”和“1”的无条件概率 p_0 和 p_1，以及相邻码间的条件概率 $p(1|1)$、$p(0|1)$、$p(1|0)$ 和 $p(0|0)$。

5-5 某信源有 8 个符号 $\{u_1,\cdots,u_8\}$，概率分别为 1/2、1/4、1/8、1/16、1/32、1/64、

1/128、1/128，编成这样的码：000,001,010,011,100,101,110,111。

（1）求信源的符号熵 $H(u)$；

（2）求出现一个“1”或一个“0”的概率；

（3）求这种码的编码效率；

（4）求出相应的香农码；

（5）求该码的编码效率。

5-6 设无记忆二元信源，概率为 $p_0=0.005$，$p_1=0.995$。信源输出 $N=100$ 的二元序列。在长为 $N=100$ 的信源序列中只对含有 3 个或小于 3 个“0”的各信源序列构成一一对应的一组定长码。

（1）求码字所需的最小长度。

（2）考虑没有给予编码的信源序列出现的概率，该定长码引起的错误概率 P 是多少？

5-7 已知符号集合$\{x_1,x_2,x_3,\cdots\}$为无限离散消息集合，它们的出现概率分别为 $p(x_1)=1/2$，$p(x_2)=1/4$，$p(x_3)=1/8$，$\cdots$，$p(x_i)=1/2^i$，$\cdots$。

（1）用香农编码方法写出各个符号消息的码字。

（2）计算码字的平均信息传输速率。

（3）计算信源编码效率。

5-8 某信源有 6 个符号，概率分别为 3/8、1/6、1/8、1/8、1/8、1/12，试求三进码元(0,1,2)的哈夫曼码，并求出编码效率。

5-9 若某一信源有 N 个符号，并且每个符号均以等概出现，对此信源用最佳哈夫曼二元编码，问当 $N=2^i$ 和 $N=2^i+1$（i 为正整数）时，每个码字的长度等于多少？平均码长是多少？

5-10 设有离散无记忆信源 $P(X)=\{0.37,0.25,0.18,0.10,0.07,0.03\}$。

（1）求该信源符号熵 $H(X)$。

（2）用哈夫曼编码编成二元变长码，计算其编码效率。

（3）要求译码错误小于 10^{-3}，采用定长二元码要达到(2)中哈夫曼编码的效率，问需要多少个信源符号一起编？

5-11 信源符号 X 有 6 种字母，概率为 0.32,0.22,0.18,0.16,0.08,0.04。

（1）求符号熵 $H(X)$。

（2）用香农编码编成二进制变长码，计算其编码效率。

（3）用哈夫曼编码编成二进制变长码，计算其编码效率。

（4）用哈夫曼编码编成三进制变长码，计算其编码效率。

（5）若用逐个信源符号来编定长二进制码，要求能不出差错译码，求所需要的每符号的平均信息率和编码效率。

（6）当译码差错小于 10^{-3} 的定长二进制码要达到(3)中哈夫曼的效率时，估计要多少个信源符号一起编才能办到？

5-12 已知一信源包含 8 个消息符号，其出现的概率为 $P(X)=\{0.1,0.18,0.4,0.05,0.06,0.1,0.07,0.04\}$。

（1）该信源在每秒内发出 1 个符号，求该信源的熵及信息传输速率。

（2）对这 8 个符号进行哈夫曼编码，写出相应码字，并求出编码效率。

(3) 采用香农编码,写出相应码字,求出编码效率。

5-13 某信源有 9 个符号,概率分别为 1/4、1/4、1/8、1/8、1/16、1/16、1/16、1/32、1/32,用三进制符号(a,b,c)编码。

(1) 编出哈夫曼码,并求出编码效率;

(2) 若要求符号 c 后不能紧跟另一个 c,编出一种有效码,其编码效率是多少?

5-14 一信源可能发出的数字有 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信道中传输,二进制信道中传输一个码字需要 1.8 元,三进制信道中传输一个码字需要 2.7 元。

(1) 编出二进制符号的哈夫曼码,求其编码效率;

(2) 编出三进制符号的哈夫曼码,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?

5-15 有二元独立序列,已知 $p_0=0.9$,$p_1=0.1$,求这序列的符号熵。当用霍夫曼编码时,以 3 个二元符号合成一个新符号,求这种符号的平均代码长度和编码效率。设输入二元符号的速率为每秒 100 个,要求 3 分钟内溢出和取空的概率均小于 0.01,求所需的信道码率(bit/s)和存储器容量(比特数)。若信道码率已规定为 50bit/s,存储器容量将如何选择?

5-16 离散无记忆信源发出 a、b 两种符号,其概率分布为 1/4、3/4。若信源输出的序列为 *babba*,对其进行算术编码并计算编码效率。

5-17 离散无记忆信源发出 a、b 两种符号,若信源输出的序列为 *babbabbbbabbabbb*,对其进行 LZ 编码。

5-18 在电视信号中,亮度信号的黑色电平为 0,白色电平为 L。用均匀分割来量化其样值,要求峰功率信扰比大于 50dB,求每样值所需的量化比特数。