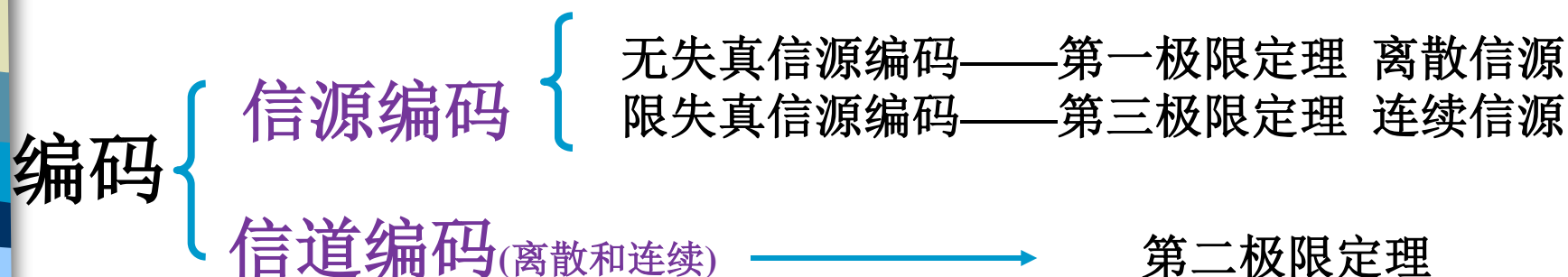


第5章 信源编码

前面介绍了信源熵和信息率失真函数的概念，弄清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，那么能否达到或接近像信源极限熵或率失真函数这样的最小信息率呢，这就是编码定理要回答的问题之一。

第5章 信源编码



信源编码

在不失真或允许一定失真条件下，如何用尽可能少的符号来传送信源信息，以便提高信息传输率。

信道编码

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。

第5章 信源编码

信源编码的作用可归纳为：

- (1) **符号变换**：使信源的输出符号与信道的输入符号相匹配；
- (2) **信息匹配**：使信息传输率达到信道容量；
- (3) **冗余度压缩**：使编码效率等于或接近100%。

- (2) **信息匹配**: 使信息传输率达到信道容量;
 (3) **冗余度压缩**: 使编码效率等于或接近100%。

例 3-12 离散无记忆信源, 输出符号概率分布如下表。 $H(X) = 1.75$ bit/信源符号

通过一个无噪无损二元离散信道进行传输。

信道容量 $C = 1$ bit/信道符号

	x_1	x_2	x_3	x_4
$p(x_i)$	$1/2$	$1/4$	$1/8$	$1/8$
编码1	00	01	10	11
编码2	000	001	010	011

编码1: 信道传输率 $R_1 = \frac{H(X)}{2} = \frac{1.75}{2} = 0.875$ bit/信道符号

$$\text{绝对冗余度} = C - R_1 = 1 - 0.875 = 0.125$$

$$\text{相对冗余度} = \frac{C - R_1}{C} = 12.5\%$$

编码2: 信道传输率 $R_2 = \frac{H(X)}{3} = 0.583$ bit/信道符号

$$\text{绝对冗余度} = C - R_2 = 1 - 0.583 = 0.417$$

$$\text{相对冗余度} = \frac{C - R_2}{C} = 41.7\%$$

第5章 信源编码

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

限失真编码定理

△无失真编码可精确复制信源输出的消息，只适用于离散信源

△对于连续信源，只能在失真受限制的情况下进行限失真编码

第5章 信源编码

- 1 5.1 编码的概念
- 2 5.2 无失真信源编码定理
- 3 5.3 限失真信源编码定理
- 4 5.4 常用信源编码方法简介

5.1 编码的概念

分组码(Block Codes)，也叫块码

- 将信源消息分成若干组，即符号序列 \mathbf{x}_i ,

$$\mathbf{x}_i = (x_{i1}x_{i2}\dots x_{il}\dots x_{iL}),$$

$$x_{il} \in A = \{a_1, a_2, \dots, a_i, \dots, a_n\}$$

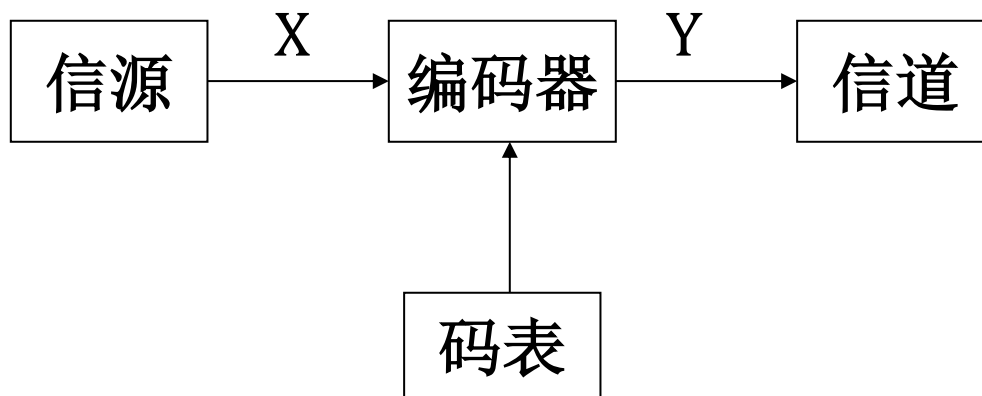
- 每个符号序列 \mathbf{x}_i 依照固定码表映射成一个码字 \mathbf{y}_i ,

$$\mathbf{y}_i = (y_{i1}y_{i2}\dots y_{ik}\dots y_{iK}),$$

$$y_{ik} \in B = \{b_1, b_2, \dots, b_i, \dots, b_m\}$$

- 只有分组码才有对应的码表，而非分组码中则不存在码表。

分组码的码表



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

信源编码

如图5-1所示，如果信源输出符号序列长度 $L=1$ ，信源符号集 $A = \{a_1, a_2, \dots, a_i, \dots, a_n\}$ 。信源概率空间为

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

- 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，它的基本符号集为 $\{0, 1\}$ 。
- 若将信源 X 通过二元信道传输，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。
- 如果 $L=2$ ，码表多大？

定长码与变长码

码可分为两类：

一、固定长度的码，码中所有码字的长度都相同，如下表中的码1就是**定长码(Fixed Length Codes)**。

二、可变长度码，码中的码字长短不一，如下表中码2就是**变长码(Variable Length Codes)**。

信源 符号 a_i	信源符号出 现概率 $p(a_i)$	码表	
		码1	码2
a_1	$p(a_1)$	00	0
a_2	$p(a_2)$	01	01
a_3	$p(a_3)$	10	001
a_4	$p(a_4)$	11	111

码的属性 (1) 奇异码与非奇异码

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。

(1) 奇异码(Singular Codes)和非奇异码 (Nonsingular Codes)

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。

下表中的码2是奇异码，其他都是非奇异码。

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

码的属性 (2) 唯一可译码

(2) 唯一可译码 (Uniquely Decodable Codes)

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：{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，**对于码表{0,10,11}而言**。

唯一可译码

- 奇异码不是唯一可译码

如码2 {0,11,00,11}

- 非奇异码

— 唯一可译码 — 码4

— 非唯一可译码 — 码3

码2	码3	码4
0	0	1
11	10	10
00	00	100
11	01	1000

码3 (0, 10, 00, 01)

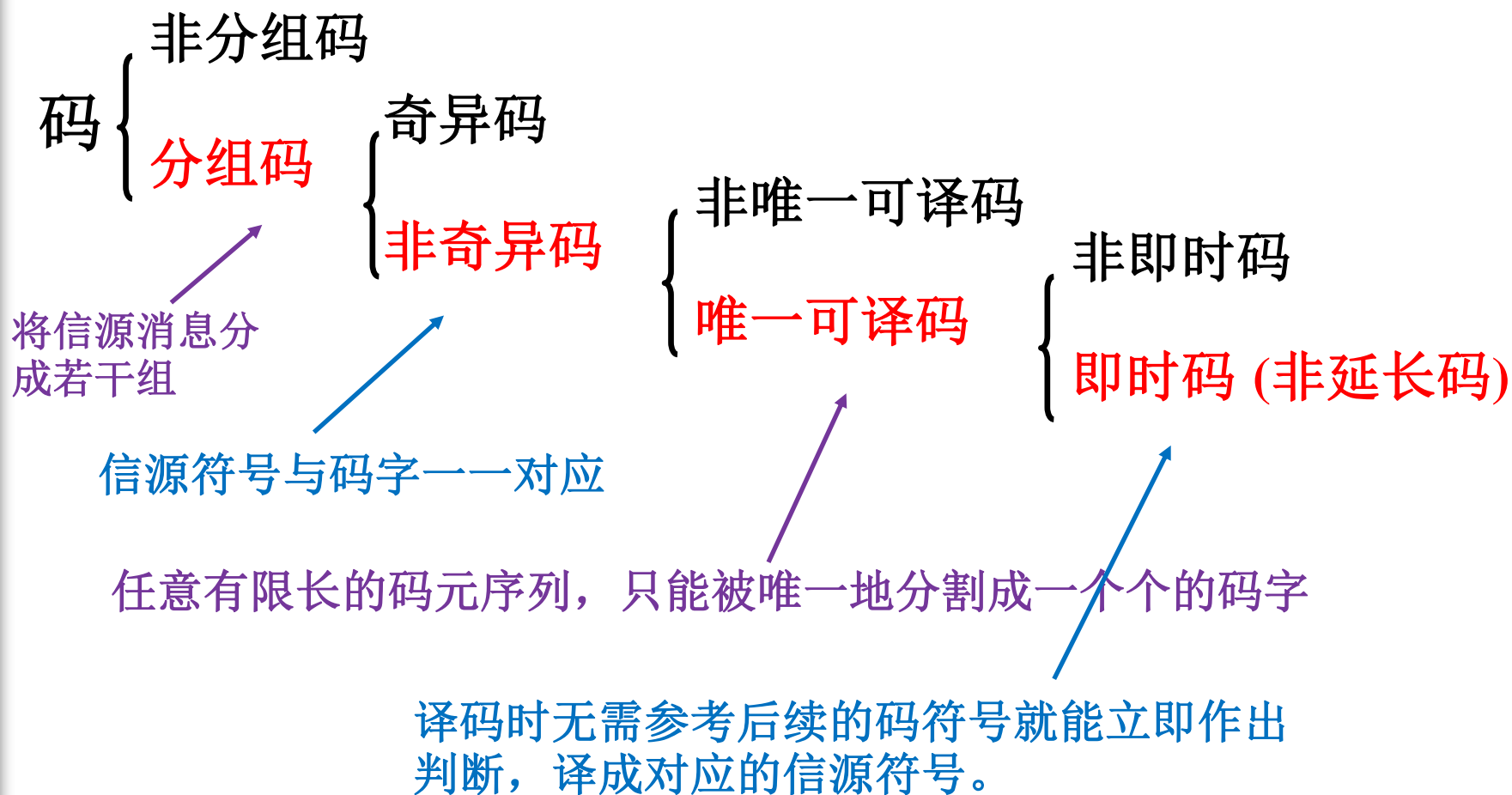
例如10000100是由码3的(10,0,0,01,00)产生的码流
译码时可有多种分割方法，如10,0,00,10,0，此时就
产生了歧义。

即时码和非即时码

码4	码5
1	1
10	01
100	001
1000	0001

- **唯一可译码**：分为**非即时码**和**即时码**。
- **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，还需等下一个码字开始接收后才能判断是否可以译码，如码4。
- **即时码**：(非延长码)(异前缀码)
 - 只要收到符号就表示该码字已完整，可以立即译码。如码5。
 - 即时码又称为**非延长码**(Undelayed Codes)，任意一个码字都不是其它码字的**前缀**部分，有时叫做**异前缀码**(Prefix Codes)。
 - **判断**：任意一个码字都不是其它码字的前缀部分
- 在延长码中，有的码是唯一可译的，取决于码的总体结构，如码4所示延长码就是唯一可译的。

码的分类



即时码及其树图构造法

- 即时码（非延长码或异前缀码）是唯一可译码的一类子码。
- 即时码可用树图来构造，构造的要点是：
 - 最上端为树根A，从根出发向下伸出树枝，树枝总数等于 m （进制数），树枝的尽头为节点。
 - 从每个节点再伸出 m 个树枝，当某个节点被安排为码字后，就不再伸枝，这节点为终端节点。能再伸枝的节点成为中间节点。一直继续下去，直至都不能伸枝为止。
 - 每个节点所伸出的树枝标上码符号，从根出发到终端点所走路径对应的码符号序列则为终端节点的码字。

码树

- 码树

— 表示各码字的构成

r 级节点有 m^r 个

一级节点

二级节点

树根—码字的起点

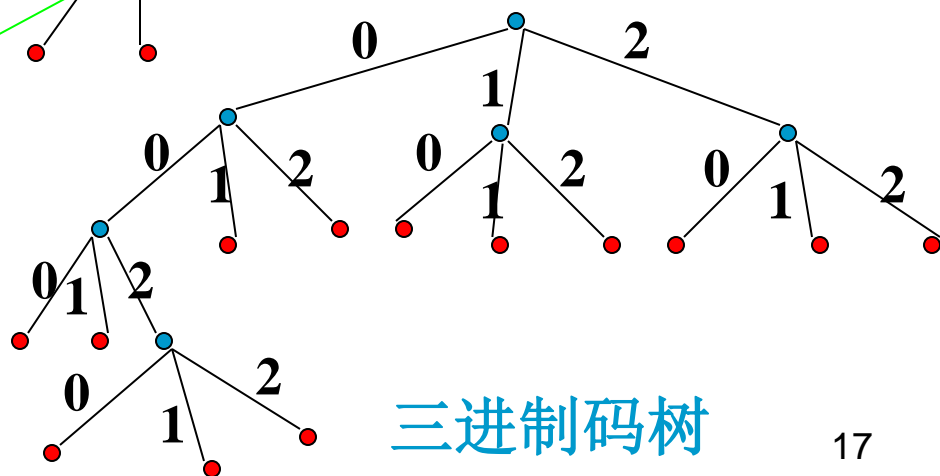
分成 m 个树枝—码的进制数

中间节点—码字的一部分

终端节点—码字1101

二进制码树

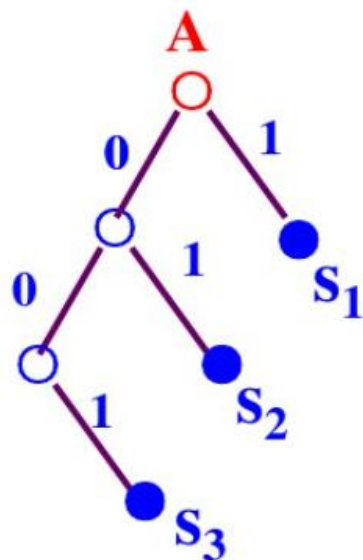
节数—码长



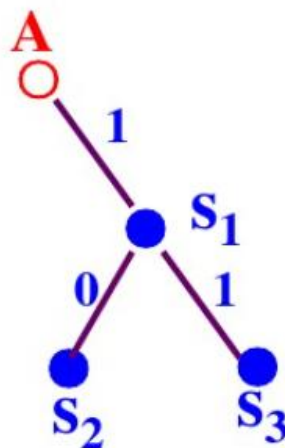
三进制码树

1) 二元（进制）码树

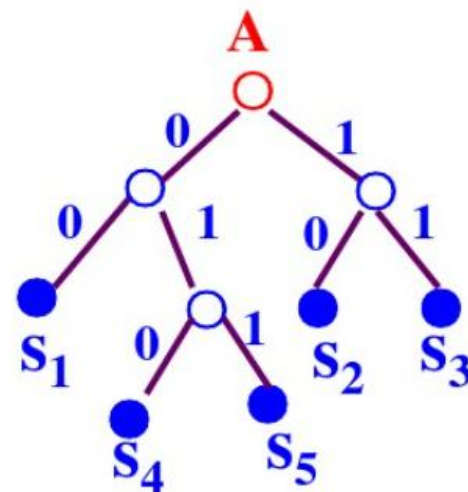
例1 $s_1 \rightarrow 1$
 $s_2 \rightarrow 01$
 $s_3 \rightarrow 001$



例2 $s_1 \rightarrow 1$
 $s_2 \rightarrow 10$
 $s_3 \rightarrow 11$



例3 $s_1 \rightarrow 00$
 $s_2 \rightarrow 10$
 $s_3 \rightarrow 11$
 $s_4 \rightarrow 010$
 $s_5 \rightarrow 011$



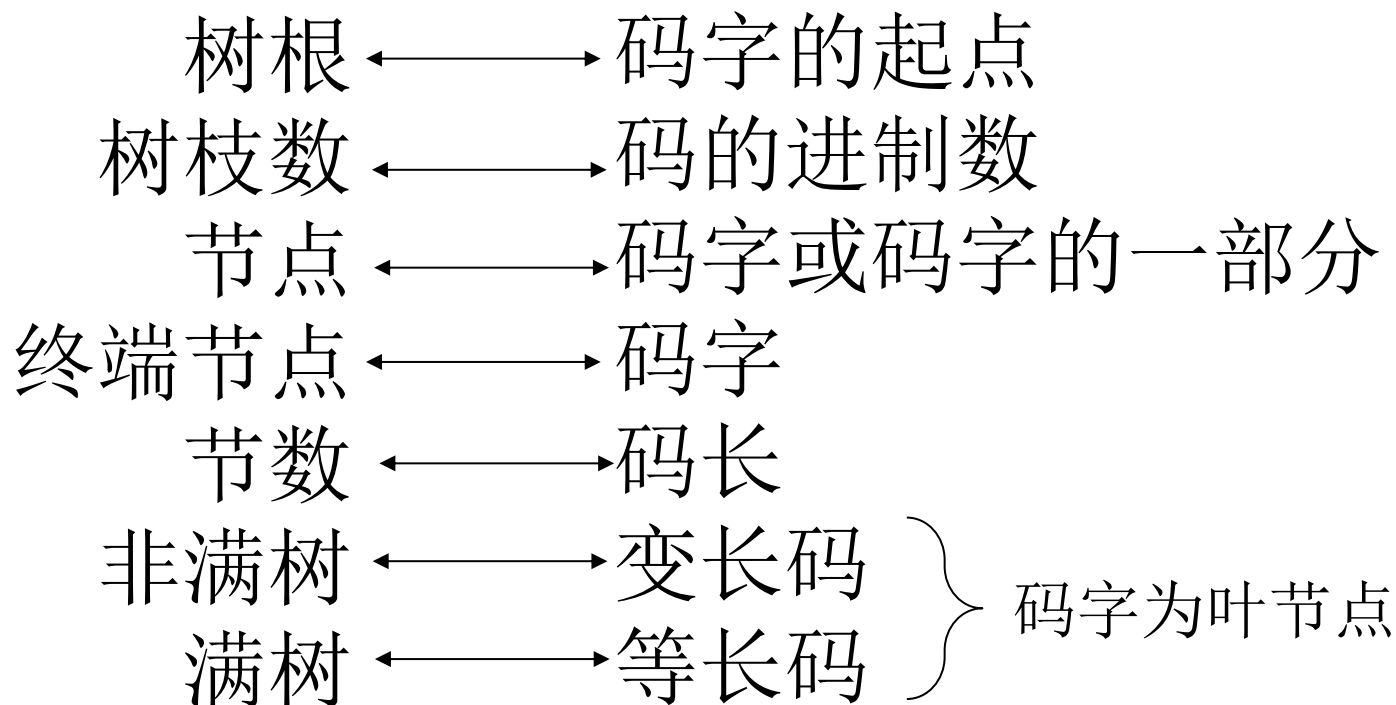
2) 用码树图构造码

在树的生长过程中，节点生出树枝，各树枝旁标出相应的码符，为了清晰起见相同码符的树枝方向相同，终端节点●表示信源符号，从树根到终端节点所经过的树枝旁的码符按经过的顺序组成的序列构成码字。

3) 用码树图判断即时码

如果表示信源符号的终端节点不再延伸，或到达任一信源符号终端节点的路径不经过其它的终端节点，这样构造的码满足即时码条件。

码树与码字对应关系图



克劳夫特不等式

满树：

- 每个节点上都有 m 个分枝的树，且都延伸到最后一级端点——等长码

非满树：

- 变长码

唯一可译码存在的充分和必要条件是各码字的长度 K_i 应符合克劳夫特不等式(Kraft Inequality):

$$\sum_{i=1}^n m^{-K_i} \leq 1$$

式中，
 m 是编码进制数
 n 是信源符号数

克劳夫特不等式

Kraft不等式是**唯一可译码存在**的充要条件

必要性表现在如果码是唯一可译码，则必定满足该不等式，如码4和码5；

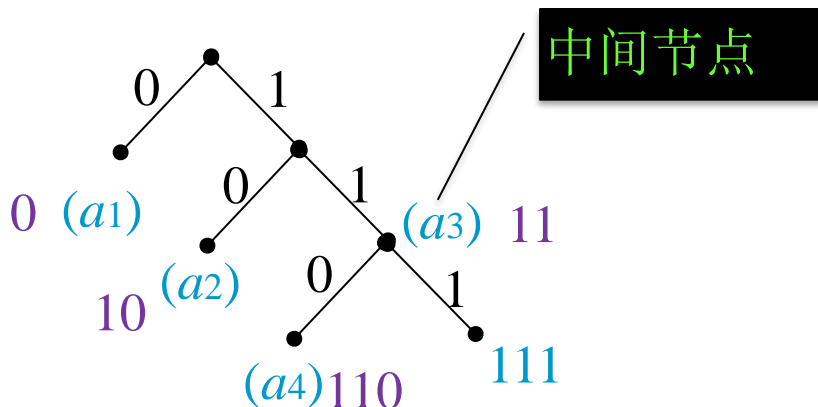
充分性表现在如果满足该不等式，则**这种码长**的唯一可译码一定存在，但并不表示所有满足不等式的一定是**唯一可译码**。

所以说，该不等式是唯一可译码**存在**的充要条件，而**不是**判别一个编码是否唯一可译码的充要条件。

克劳夫特不等式

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Kraft不等式，得：

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



不存在满足这种 K_i 的唯一可译码

这样的码字长度就存在唯一可译码

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

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

克劳夫特不等式

必须注意:

Kraft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。

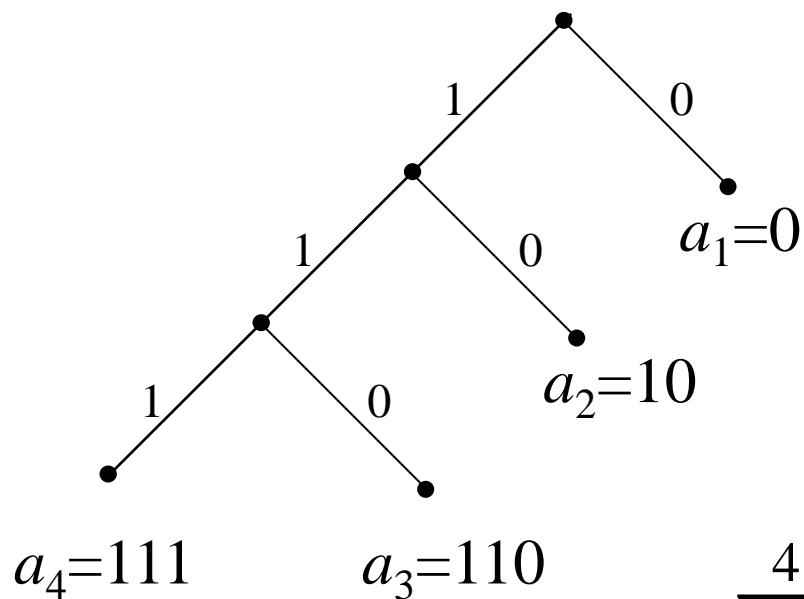
$$K_1=1, K_2=2, K_3=3, K_4=3$$

$\{0, 10, 110, 111\}$ 唯一可译码;

$\{0, 10, 010, 111\}$

不是唯一可译码;

均满足克劳夫特不等式



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

唯一可译码的判断方法1

观察是否是非奇异码。若是奇异码则一定不是唯一可译码。

计算是否满足**Kraft**不等式。若不满足一定不是唯一可译码。

将码画成一棵树图，观察是否满足即时码的树图的构造，若满足则是唯一可译码。

不满足即时可译条件的唯一可译码怎样判断？

唯一可译码的判断方法—尾随后缀法

A.A.Sardinas和G.W.Patterson 于1957年提出下述算法用于判断码C的唯一可译性。

A_1	A_2	A_3	A_4
B_1	B_2	B_3	

A_i, B_i 都是码字。当且仅当某个有限长的码符号序列能译成不同码字序列时，此码不是唯一可译码

此时， A_1 一定是 B_1 的前缀，而 B_1 的尾随后缀一定是另一码字 A_2 的前缀；而 A_2 的尾随后缀又是其他码字的前缀...

最后，码符号序列的尾部（ B_3 的尾随后缀）一定是一个码字（ A_4 ）

构造尾随后缀集合F的方法

- 1、考查 C 中所有的码字，若 W_i 是 W_j 的前缀，则将相应的后缀作为一个尾随后缀码放入集合 F_0 中；
- 2、考查 C 和 F_i 两个集合， $W_i \in C$ 是 $W_j \in F_i$ 的前缀或 $W_i \in F_i$ 是 $W_j \in C$ 的前缀，则将相应的后缀作为尾随后缀码放入集合 F_{i+1} 中；
- 3、 $F = \bigcup_i F_i$ 即为码 C 的尾随后缀集合；
- 4、若 F 中出现了 C 中的元素，则算法终止，返回假（ C 不是唯一可译码）；否则若 F 中没有出现新的元素，则返回真。

例

判断码 C : $\{0, 1001, 1011, 1101, 1111, 011\}$ 是否是唯一可译码。

构造尾随后缀集合 F :

码字“0”是“011”的前缀，其尾随后缀“11”是码字“1101”和“1111”的前缀，得尾随后缀为01和11，其中0是码字，所以码 C 不是唯一可译码。

尾随后缀集合 $F = \{11, 01, 1, 001, 011, 111\}$ ，011是一个码字，因此不唯一可译

$0 \longrightarrow 11 \longrightarrow 01 \longrightarrow 1 \longrightarrow 001, 011, 111$

练习

编码 C_4

0

10

1101

1100

1001

1111

(1) 此编码是否是唯一可译码？

(2) 假设符号等概率分布，求平均码长。

第5章 信源编码

- 1 5.1 编码的概念
- 2 5.2 无失真信源编码定理
- 3 5.3 限失真信源编码定理
- 4 5.4 常用信源编码方法简介

无失真信源编码

- 信源编码器输入的消息序列:

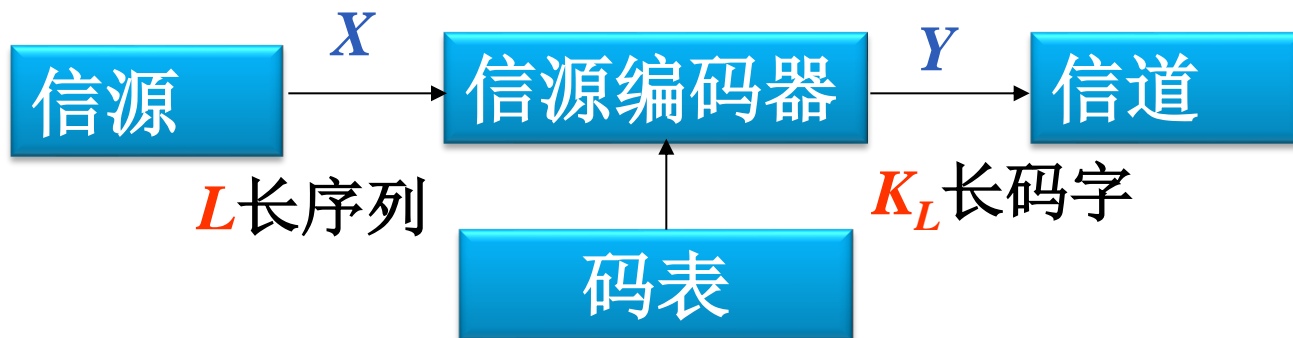
$$X=(X_1 X_2 \dots X_l \dots X_L), \quad X_l \in \{a_1, \dots, a_n\},$$

输入的消息总共有 n^L 种可能的组合

- 输出的码字为:

$$Y=(Y_1 Y_2 \dots Y_k \dots Y_{K_L}), \quad Y_k \in \{b_1, \dots, b_m\}$$

输出的码字总共有 m^{K_L} 种可能的组合。



无失真信源编码

- Y_k 有 m 种可能取值，所以平均每个符号输出的最大信息量为 $\log m$ (等概分布)，
- 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，
- 则传送一个信源符号需要的平均信息率为

$$\bar{K} = \frac{K_L}{L} \log m = \frac{1}{L} \log M \quad \text{bit/信源符号}$$

$M = m^{K_L}$ 是 Y 所能编成的码字的个数

无失真信源编码

- 实现**无失真**的信源编码，要求：
 - 信源符号 $X_1 X_2 \dots X_l \dots X_L$
 - 码字 $Y_1 Y_2 \dots Y_k \dots Y_{K_L}$ } 是一一对应的
- 能够无失真或**无差错地从Y恢复X**，也就是能正确地进行反变换或译码；
- 传送Y时所需要的**信息率最小**
信息率最小就是找到一种编码方式使

$$\bar{K} = \frac{K_L}{L} \log m = \frac{1}{L} \log M \quad \text{最小}$$

无失真信源编码

所谓**信息率最小**，就是找到一种编码方式使 \bar{K} 最小。

无失真信源编码定理研究的内容：

- 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？

定长编码定理

变长编码定理

定长编码

- 码长 K 是定值，且是唯一可译码
- 由 L 个符号组成的、每个符号的熵为 $H_L(X)$ 的无记忆平稳信源符号序列 $X_1X_2\dots X_l\dots X_L$ （每个符号 n 种可能值）
- 输入的消息总共有 n^L 种可能的组合；
- 可用 $K_L=K$ 个符号 $Y_1, Y_2, \dots, Y_k, \dots$ ，（每个符号有 m 种可能值）进行定长编码；
- 输出的码字总共有 m^K 种可能的组合。
- 若要无失真，必须满足： $n^L \leq m^K$

定长编码

- 若对信源进行定长编码，必须满足：

$$n^L \leq m^K \quad \text{或} \quad \frac{K}{L} \geq \frac{\log n}{\log m}$$

- 只有当 K 长的码符号序列数 m^K 大于或等于信源的符号数 n^L 时，才可能存在定长非奇异码。
- 例如英文电报有27个符号， $n=27$ ， $L=1$ ， $m=2$ (二元编码)

$$K \geq L \frac{\log_2 n}{\log_2 m} = \log_2 27 \approx 5$$

每个英文电报符号至少
要用5位二元符号编码

定长编码

- 实际英文电报符号信源，在考虑了符号出现的概率以及符号之间的依赖性后，平均每个英文电报符号所提供的信息量约等于1.4比特，大大小于5比特。
- 编码后5个二元符号只携带约1.4比特信息量。
- 定长编码的信息传输效率极低。

5.2.1 定长编码定理 \overline{K}

无记忆平稳信源平均符号熵为 $H_L(X)$ ，对任意 $\delta > 0, \varepsilon > 0$ ，只要

$$\frac{K_L}{L} \log m \geq H_L(\mathbf{X}) + \varepsilon$$

则当 **L** 足够大时，必可使译码差错小于 δ ；反之，当

$$\frac{K_L}{L} \log m \leq H_L(\mathbf{X}) - 2\varepsilon$$

时，译码差错一定是有限值，而 **L** 足够大时，译码几乎必定出错。

定长编码定理含义

(1)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**二进制码长**是

$$\bar{K} = \frac{K_L}{L} \log m = \frac{1}{L} \log M$$

时，只要 $\bar{K} > H_L(X)$ ，这种编码器一定可以做到**几乎无失真**，也就是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**。

定长编码定理含义

(2) 将定理的条件改写成：

$$K_L \log m > LH_L(X) = H(X)$$

其中：左边： K_L 长码字所能携带的最大信息，

右边： L 长信源序列携带的信息量。

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

例：信源输出8种符号，

$L=1$ ，等概率时， $H_1(X)=\log_2 8=3$ 比特/符号，
可用3比特的信息率进行**无失真**的编码。

$p(a_i)=\{0.4, 0.18, 0.1, 0.1, 0.07, 0.06, 0.05, 0.04\}$ ，则此时 $H_1(X)=2.55$ 比特/符号， $\bar{K} > H_1(X) = 2.55, 2^{2.55}=5.856$

例题

- 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 $\bar{K} = H_L(X) = 2.55$ 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，因而引起错误。
- 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。

信源长度 L

对定长编码，若要实现几乎无失真编码，则信源长度必须满足：

$$L \geq \frac{\sigma^2(\mathbf{X})}{\varepsilon^2 \delta}$$

$\sigma^2(X) = E\{[I(x_i) - H(X)]^2\}$ – 信源序列的自信息方差

δ : 差错率要求 (如 10^{-6})

$$\varepsilon = \bar{K} - H_L(X)$$

编码效率

编码效率定义为

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

即信源的平均符号熵为 $H_L(X)$ ，采用平均二进制符号码长为 \bar{K} 来编码，所得的效率。

无失真编码效率总是小于1，且最佳编码效率为

$$\eta = \frac{H_L(X)}{H_L(X) + \varepsilon}, \quad \varepsilon > 0$$

编码定理从理论上阐明了编码效率接近1的理想定长编码器的存在性，它使输出符号的信息率与信源熵之比接近于1，即只要 L 足够大

$$\frac{H_L(X)}{\frac{K_L}{L} \log m} \rightarrow 1$$

但要在实际中实现， L 必须取无限长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。

例5-2 设离散无记忆信源概率空间为

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

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

对信源符号采用**定长**二元编码，要求编码效率为**90%**，若取**L=1**，则可算出 $\eta = \frac{H_L(X)}{\bar{K}}$

$$\bar{K} = 2.55 \div 90\% = 2.83 \text{ 比特/符号}$$

$2^{2.83} = 7.11$ ，共有7.11种码字，按照7来算，信源符号中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**，太大。

采用式(5-2-7)

$$\eta = \frac{H(X)}{H(X) + \varepsilon} = 0.90, \quad \Rightarrow \varepsilon = 0.28$$

信源序列的自信息方差为

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

若要求译码错误概率 $\delta \leq 10^{-6}$

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

- 在差错率和编码效率要求并不十分苛刻的条件下，就需要 $L=10^8$ 个信源符号进行联合编码，这显然是很难实现的。 L 太长。

若用3bit来对上述信源的8个符号进行定长二元编码，

$L=1$ ，则 $\bar{K} = H(X) + \varepsilon = 3$, $\varepsilon = 0.45$

此时译码无差错，即 $\delta=0$ 。

编码效率为 $\eta = \frac{2.55}{3} = 85\%$

效率太低

当 L 有限时，要做到高编码效率、低错误率，对于定长编码来说是不可能做到的。

5.2.2 变长编码定理（香农第一定理）

在变长编码中，码长 K_i 是变化的

m进制平均码长： $\overline{K'} = \sum_i p(a_i) K_i$

根据信源各个符号的统计特性，如**概率大的符号用短码，概率小的用较长的码**，使得编码后平均码长降低，从而提高编码效率。（统计匹配）

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（ m 进制）码字平均长度 $\overline{K'}$ 满足下列不等式

$$\frac{H(X)}{\log m} \leq \overline{K'} < \frac{H(X)}{\log m} + 1$$

离散平稳无记忆序列变长编码定理

$$\frac{H(X)}{\log m} \leq \overline{K'} < \frac{H(X)}{\log m} + 1 \rightarrow \frac{LH_L(X)}{\log m} \leq \overline{K_L} < \frac{LH_L(X)}{\log m} + 1$$

对于平均符号熵为 $H_L(X)$ 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

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

$\overline{K} = \frac{\overline{K_L}}{L} \log m$, 当 L 足够大时, 可使 $\frac{\log m}{L} < \varepsilon$, 得到

$$H_L(X) \leq \overline{K} < H_L(X) + \varepsilon$$

其中 ε 为任意小正数。

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。

变长编码效率

变长编码效率的**下界**:

$$\eta = \frac{H_L(X)}{\overline{K}} > \frac{H_L(X)}{H_L(X) + \frac{\log m}{L}}$$

无失真编码效率**总是小于1**，可以用它来衡量各种编码方法的优劣。

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

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

对某一信源和某一码符号集，若有一个唯一可译码，其平均长度小于所有其他唯一可译码的平均长度，则称该码为**最佳码**或**紧致码**。

- 若对例5-2用变长码实现，要求 $\eta > 90\%$ ，用二进制进行编码， $m=2$ ， $\log_2 m=1$ 。

- 由
$$\eta = \frac{H_L(X)}{\overline{K}} > \frac{H_L(X)}{H_L(X) + \frac{\log m}{L}}$$

$$0.9 = \frac{2.55}{2.55 + \frac{1}{L}}$$

得 $L=4$

例5.3 设离散无记忆信源的概率空间为

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

$$H(X) = \frac{1}{4} \log 4 + \frac{3}{4} \log \frac{4}{3} = 0.81 \text{ lbit / 符号}$$

续例5.3

若 $L=1$,用二元定长编码(0, 1)来构造一个即时码: $a_1 \rightarrow 0$, $a_2 \rightarrow 1$ 。

平均码长 $\bar{K} = 1$ 二元码符号/信源符号

编码效率为 $\eta = \frac{H(X)}{\bar{K}} = 0.811$

输出的信息传输率为

$u=0.811$ 比特/二元码符号

续例5.3

对L=2的信源序列进行**变长编码**（编码方法后面介绍），其即时码如下表

$$\begin{aligned}\overline{K}_2 &= \frac{9}{16} \times 1 + \frac{3}{16} \times 2 + \frac{3}{16} \times 3 + \frac{1}{16} \times 3 \\ &= \frac{27}{16} \quad \text{二元码符号/信源序列}\end{aligned}$$

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

$$\text{编码效率 } \eta_2 = \frac{32 \times 0.811}{27} = 0.961$$

$$\text{信息传输率 } u_2 = 0.811 \div \overline{K} = 0.961 \text{ 比特/二元码符号}$$

x_i	$p(x_i)$	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

- 用同样的方法，进一步地将信源序列的长度增加到 $L=3$ 或 $L=4$ ，对这些信源序列 X 进行编码，并求出其编码效率分别为

$$\eta_3 = 0.985 \quad \eta_4 = 0.991$$

- 相应的**信息传输率**分别为

$$u_3 = 0.985 \text{ 比特/二元码符号}$$

$$u_4 = 0.991 \text{ 比特/二元码符号}$$

- 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 $\delta \leq 10^{-5}$
- 先计算自信息的方差：

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

- 所需要的信源序列长度

$$\eta = \frac{H_L(X)}{H_L(X) + \varepsilon}, \quad \varepsilon > 0 \quad \varepsilon = H_L(X)(1 - \eta)/\eta$$

$$L \geq \frac{\sigma^2(x)}{\varepsilon^2 \delta} = \frac{0.4715}{(0.811)^2} \cdot \frac{(0.96)^2}{0.04^2 \times 10^{-5}} = 4.13 \times 10^7$$

分析例5-3

从以上结果可以看出，定长码需要的信源序列长，这使得码表很大，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 不需要很大就可以达到相当高的编码效率，而且可以实现无失真编码。

随着信源序列长度的增加，编码效率越来越接近于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 $C=1\text{bit/二元码符号}$ ，达到信源与信道匹配，使信道得到充分利用。

香农编码

- 香农第一定理指出了平均码长与信源之间的关系，同时也指出了可以通过编码使平均码长达到极限值，这是一个很重要的极限定理。
- 香农第一定理指出，选择每个码字的长度 K_i 满足下式：

$$K_i = \left\lceil \log \frac{1}{p(x_i)} \right\rceil \quad \leftarrow \text{取整}$$

或：

$$I(x_i) \leq K_i < I(x_i) + 1$$

这种编码方法称为香农编码。

编码步骤

(1) 将信源消息符号按其出现的概率大小依次排列

$$p_1 \geq p_2 \geq \cdots \geq p_n$$

(2) 依照下列不等式确定整数的码长 K_i

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

(3) 为了编成**唯一可译码**，计算第 i 个消息的累加概率

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

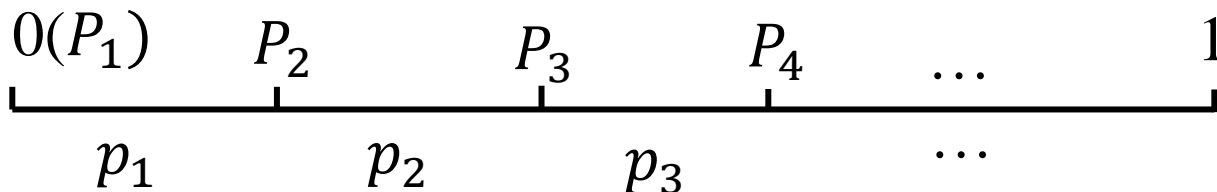
(4) 将累加概率 P_i 变换成二进制数

(5) 取 P_i 二进数的小数点后 K_i 位即为该消息符号的二进制码字

香农编码图示

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
小区间的任一点可用来代表该符号

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



例5-4

设信源共7个符号，其概率和累加概率如下表所示

信源符号 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	3.32	4	1110
a_7	0.01	0.99	6.64	7	1111110

$$-\log 0.17 \leq K_4 < -\log 0.17 + 1$$

$$2.56 \leq K_4 < 3.56, K_4 = 3$$

$$P_4 = 0.57_D = 0.1001 \dots_B$$

补充

指数	分数	十进制	二进制
2^{-1}	$1/2^1$.5	.1
2^{-2}	$1/2^2$.25	.01
2^{-3}	$1/2^3$.125	.001
2^{-4}	$1/2^4$.0625	.0001
2^{-5}	$1/2^5$.03125	.0000 1
2^{-6}	$1/2^6$.015625	.0000 01
...

例如：

$$0.2 = .125 + .0625 + \dots \rightarrow .001 + .0001 + \dots = .0011\dots$$

...

$$0.57 = .5 + .0625 + \dots \rightarrow .1 + .0001 + \dots = .1001\dots$$

...

$$0.99 = .5 + .25 + .125 + .0625 + .03125 + .015625 + \dots$$

$$\begin{array}{ccccccc} \swarrow & \swarrow & \downarrow & \downarrow & \downarrow & \searrow & \\ .75 & \rightarrow & .875 & \rightarrow & .9375 & \rightarrow & .96875 \rightarrow .984375 \end{array}$$

$$0.1 + 0.01 + 0.001 + 0.0001 + 0.00001 + 0.000001 + \dots = 0.1111110\dots$$

小数转化为二进制：乘2取整。

例：将 $(0.57)_D$ 转换为二进制数。

$$(0.57)_D = (0.10010\dots)_B$$

$$0.57 \times 2 = 1.14$$

$$0.14 \times 2 = 0.28$$

$$0.28 \times 2 = 0.56$$

$$0.56 \times 2 = 1.12$$

$$0.12 \times 2 = 0.24$$

...

例5.4 续

信源熵

$$H(X) = -\sum_{i=1}^7 p(a_i) \log p(a_i) = 2.61 \quad \text{比特/符号}$$

平均码长

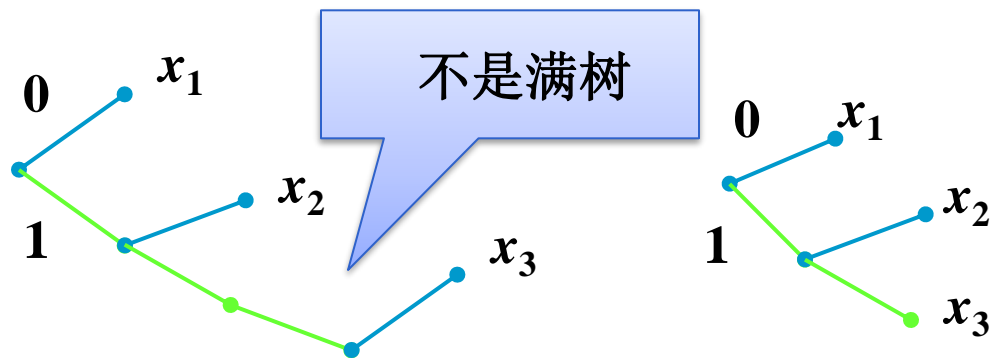
$$\bar{K} = \sum_{i=1}^7 p(a_i) K_i = 3.14 \quad \text{码元/符号}$$

信息率

$$u = \frac{H(X)}{\bar{K}} = \frac{2.61}{3.14} = 0.831 \text{ 比特/码元}$$

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。
- 所以香农编码法**多余度稍大，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。



第5章 信源编码

- 1 5.1 编码的概念
- 2 5.2 无失真信源编码定理
- 3 5.3 限失真信源编码定理
- 4 5.4 常用信源编码方法简介

5.3 限失真信源编码定理

无失真信源编码是保熵的：通过信道的信息传输率 R 等于信源熵 $H(X)$ 。

有失真信源编码属熵压缩编码，即编码后的信息率得到压缩。

之所以采用有失真的熵压缩编码，是基于以下原因：

- 保熵编码并非总是必需的；
- 保熵编码并非总是可能的；
- 降低信息率有利于传输和处理。

限失真信源编码定理

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 的信息率失真函数为 $R(D)$ ，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 。

限失真信源编码定理

如果是二元信源编码，对于任意小的 ε ，每一个信源符号的平均码长满足如下公式：

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

在失真限度内使信息率任意接近 $R(D)$ 的编码方法是存在的。然而，如果使信息率小于 $R(D)$ ，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，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。

限失真信源编码定理

限失真信源编码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。因而就不能象无失真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。

第5章 信源编码

- 1 5.1 编码的概念
- 2 5.2 无失真信源编码定理
- 3 5.3 限失真信源编码定理
- 4 5.4 常用信源编码方法简介

5.4 常用信源编码方法简介

实用化技术不同于单纯理论探讨；实用技术不是只追求理论上的性能（如：压缩比），还要考虑工程上的可实现性。

为了与复杂的实际信源统计匹配，实际的信源编码方法往往都不是单一的方法，而是多种方法的组合应用。

本节简单介绍几种常用的信源编码方法。

- 哈夫曼编码
- 算术编码
- LZ编码

5.4.1 哈夫曼编码

- 哈夫曼(Huffman)编码是分组码
- 依据各符号出现的**概率**来构造码字
- 基于二叉树的编码思想，所有可能的符号在哈夫曼数上对应为一个节点，节点的位置就是该符号的码字
- 这些节点都是终极节点，不再延伸，不会出现前缀码，因而唯一可译
- 哈夫曼编码是先给每一符号一片**树叶**，逐步合并成节点直到树根。
- 哈夫曼编码是一种效率比较高的**变长无失真信源编码**方法。

哈夫曼（Huffman）码

编码步骤:

(1) 将信源消息符号按其出现的概率大小依次排列,

$$p_1 \geq p_2 \geq \cdots \geq p_n$$

(2) 取两个概率最小的符号分别配以0和1两个码元, 并将这两个概率相加作为一个新符号的概率, 与未分配的二进符号的符号重新排队。

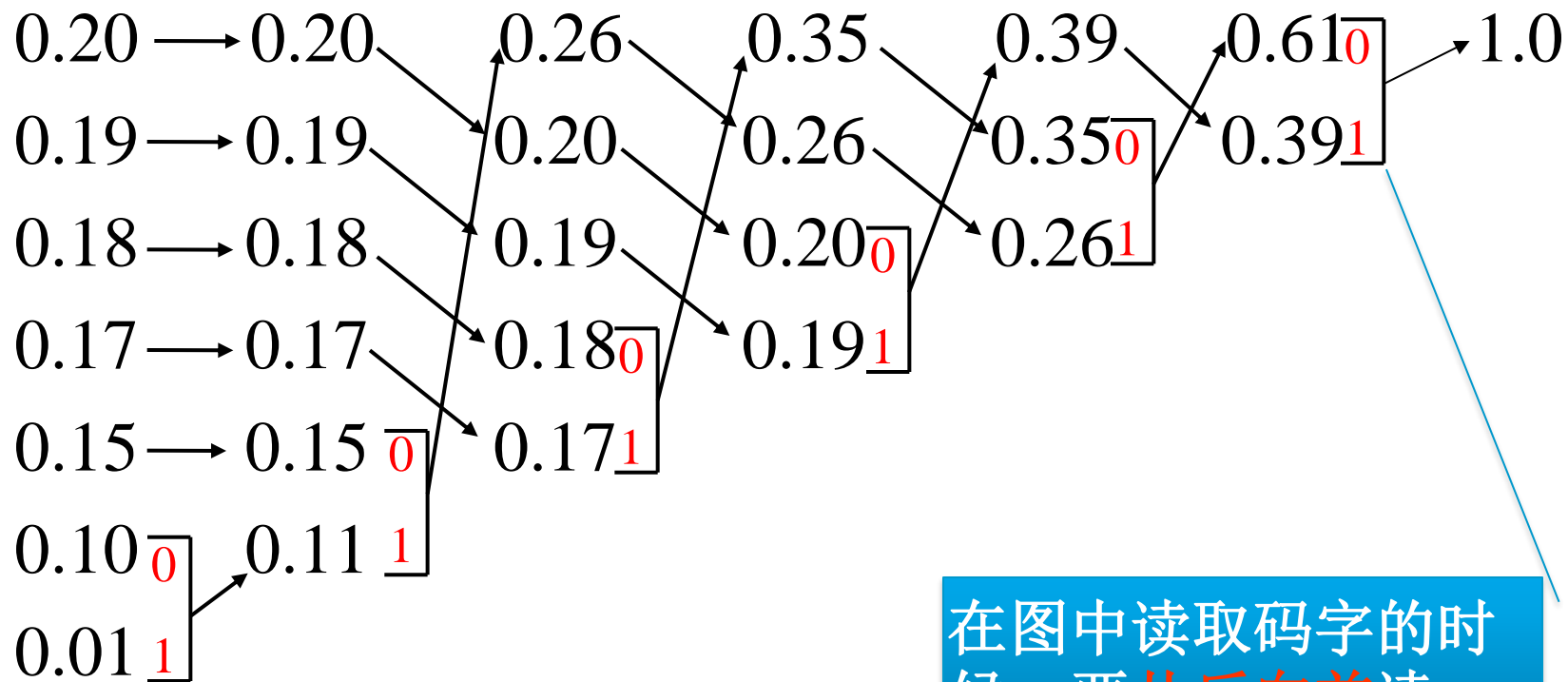
(3) 对重排后的两个概率最小符号重复步骤(2)的过程。

(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。

例题

例5-6 对例5.4中的信源进行哈夫曼编码



在图中读取码字的时候，要**从后向前**读。

编码结果

信源符号 a_i	概率 $p(a_i)$	码字 W_i	码长 K_i
a_1	0.20	10	2
a_2	0.19	11	2
a_3	0.18	000	3
a_4	0.17	001	3
a_5	0.15	010	3
a_6	0.10	0110	4
a_7	0.01	0111	4

编码结果

平均码长

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

编码效率

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

- 例5-4 香农编码

$$\overline{K}_1 = 3.14 \text{码元/符号} \quad \eta_1 = 83\%$$

哈夫曼编码特点

哈夫曼编码方法得到的码并非唯一的。

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，若两个概率最小的符号合并后的概率与其它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，其位置放置次序是可以任意的，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**。

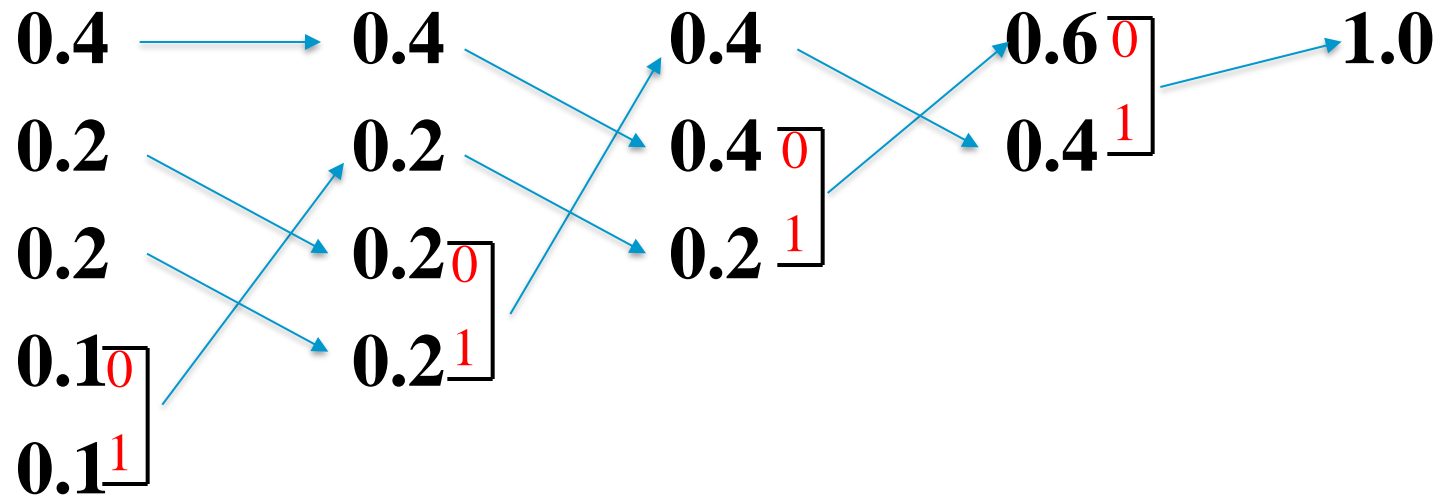
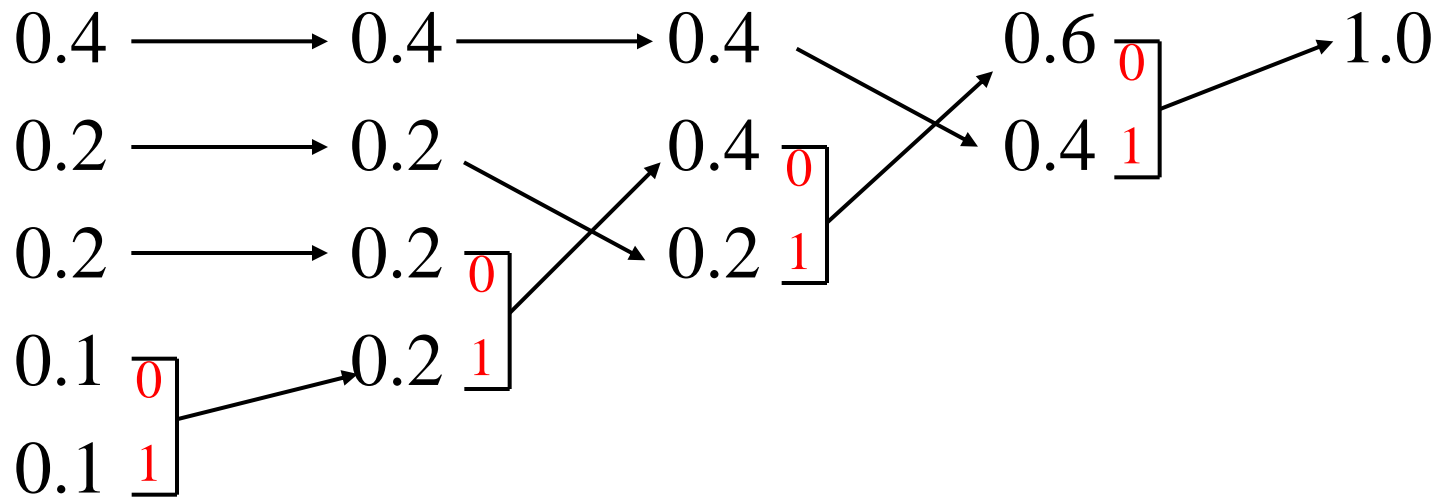
不同的编法得到的码字长度 K_i 也不尽相同。

例题

例5-7 设有离散无记忆信源，对其进行哈夫曼编码。

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

两种哈夫曼编码比较



两种哈夫曼编码比较

信源符 a_i	概率 $p(a_i)$	码字 W_{i1}	码长 K_{i1}	码字 W_{i2}	码长 K_{i2}
a_1	0.4	1	1	00	2
a_2	0.2	01	2	10	2
a_3	0.2	000	3	11	2
a_4	0.1	0010	4	010	3
a_5	0.1	0011	4	011	3

两种哈夫曼编码比较

- 编法一的平均码长为

$$\overline{K}_1 = \sum_{i=1}^5 p(x_i) K_i = 0.4 \times 1 + 0.2 \times 2 + 0.2 \times 3 + 0.1 \times 4 \times 2 = 2.2 \text{码元/符号}$$

- 编法二的平均码长为

$$\overline{K}_2 = \sum_{i=1}^5 p(x_i) K_i = 0.4 \times 2 + 0.2 \times 2 \times 2 + 0.1 \times 3 \times 2 = 2.2 \text{码元/符号}$$

- 单符号信源编二进制哈夫曼码，编码效率主要决定于信源熵和平均码长之比。
- 对相同的信源编码，其熵是一样的，采用不同的编法，得到的平均码长可能不同。
- 平均码长越短，编码效率就越高。
- 两种编法的平均码长相同，所以编码效率相同。

两种哈夫曼编码比较

但两种码的质量是不相同的，用**码方差**描述

$$\sigma_l^2 = E\left[\left(K_i - \bar{K}\right)^2\right] = \sum_{i=1}^q p(a_i) (K_i - \bar{K})^2$$

第一种方法的码方差为

$$\sigma_{l1}^2 = 1.36$$

第二种方法的码方差为

$$\sigma_{l2}^2 = 0.16$$

可见，第二种方法的码方差比第一种方法的码方差小许多，也就是说第二种方法的**Huffman**码的质量要好。

两种哈夫曼编码比较

- 第一种方法编出的5个码字有4种不同的码长；
- 第二种方法编出的码字只有2种不同的码长；
- 第二种编码方法更简单、更容易实现，所以更好。
- 结论：
- 在哈夫曼编码过程中，对缩减信源符号按概率由大到小的顺序重新排列时，**应使合并后的新符号尽可能排在靠前的位置**，这样可使合并后的新符号重复编码次数减少，使短码得到充分利用。

将下列信源编成3进制哈夫曼码

$$\begin{bmatrix} X \\ P \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 & a_4 \\ p_1 & p_2 & p_3 & p_4 \end{bmatrix}$$

如果直接编码，得到的码长为（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）

N进制哈夫曼编码，为了得到最短平均码长，有时需要对信源符号作添加，使信源符号数量满足

$$N + k(N - 1) \quad k \text{ 为整数}$$

例题：三进制哈夫曼编码

对如下单符号离散无记忆信源进行三进制哈夫曼编码

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

这里： $N=3$, $3+2k=9$ ($k=3$)

添加一个概率为0的符号

三进制哈夫曼编码

信源符号 x_i	符号概率 $p(x_i)$	编码过程	码字
x_1	0.40	<p>0.40 → 0.40 → 0.40 → 0.40 0</p> <p>0.18 → 0.18 → 0.22 → 0.38 1</p> <p>0.10 → 0.10 → 0.18 → 0.22 2</p> <p>0.10 → 0.10 → 0.10 0</p> <p>0.10 → 0.09 → 0.10 1</p> <p>0.07 → 0.07 2</p> <p>0.06 → 0.06 0</p> <p>0.05 → 0 1</p> <p>0.04 → 1</p>	0
x_2	0.18		10
x_3	0.10		11
x_4	0.10		12
x_5	0.07		21
x_6	0.06		22
x_7	0.05		200
x_8	0.04		201

码字的平均码长为

$$\overline{K}_L = \sum_{i=1}^8 p(x_i) K_i = 0.4 \times 1 + (0.18 + 0.1 + 0.1 + 0.07 + 0.06) \times 2 + (0.05 + 0.04) \times 3 = 1.69 \text{ 码元/符号}$$

- 每个符号的平均**二进制**码长（信息率）为

$$\overline{K} = \frac{\overline{K}_L}{L} \log_2 3 = \frac{1.69}{1} \log_2 3 = 2.68 \text{ bit/符号}$$

- 编码效率为

$$\eta = \frac{H(X)}{\overline{K}} = \frac{2.55}{2.68} = 95.2\%$$

三进制的信息率
与平均码长不等

- 哈夫曼的编码效率相当高，对编码器的要求也简单得多。

用哈夫曼方法对信源序列编码

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

$L=1, \bar{K} = 1$ bit/符号

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

$L=3, \bar{K}=0.533$ bit/符号

$L=4, \bar{K}=0.494$ bit/符号

随着序列长度 L 的增加，平均（二进制）码长迅速降低，接近信源熵值。

变长码与存储器容量

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 \bar{K} ，则信道传输速率需要

$$R_t = S\bar{K}$$

N个码字的长度分别为 $K_i, i = 1, \dots, N$ ，即在此期间输入存储器 $\sum K_i \text{bit}$ ，输出至信道 $R_t T \text{bit}$ ，则在存储器里还剩

$$X = \sum_{i=1}^N K_i - R_t T$$

已知 K_i 是随机变量，其均值和方差为

$$\bar{K} = E[K_i] = \sum_{j=1}^m p_j K_j \quad m: \text{信源符号集的元数}$$

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

式中 m 为信源符号集的元数。当 N 足够大时， X 是许多独立同分布的随机变量之和，它近似于正态分布，其均值和方差分别为

$$E[X] = N\bar{K} - R_t T$$

$$\sigma_X^2 = N\sigma^2$$

$$p_X(x) = \frac{1}{\sqrt{2\pi}\sigma_X} e^{-\frac{(x-E[X])^2}{2\sigma_X^2}}$$

若信道速率满足 $R_t = S\bar{K}$ ， $E[X]=0$ 。假设存储器容量为 $2A\sigma_x$ ，起始时存储器为半满，则溢出概率为

$$P(X > A\sigma_X) = \varphi(-A)$$

取空概率为

$$P(X < -A\sigma_X) = \varphi(-A)$$

若要求溢出和取空概率 <0.001 ，查表得 A 为3.08，则存储器容量为

$$C > 2A\sigma_X = 2A\sqrt{N}\sigma = 6.16\sqrt{N}\sigma$$

变长编码与存储器容量

$$C > 2A\sigma_X = 2A\sqrt{N}\sigma = 6.16\sqrt{N}\sigma$$

码方差 σ 越大，要求存储器的容量也越大

时间越长， N 越大，要求存储器的容量也越大

存储器容量设定后，随着时间的增长，存储器溢出和取空的概率都将增大

一般来说，变长码只适用于有限长的信息传输（如传真）

实际使用时，可把长信息分段发送，也可以实时检测存储器的状态，调整输出

费诺（Fano）编码方法

费诺（Fano）编码方法（属于概率匹配编码），编码步骤：

(1) 将信源消息符号按其出现的**概率大小依次排列**：

$$p_1 \geq p_2 \geq \cdots \geq p_n$$

(2) 将依次排列的信源符号**按概率值分为两大组**，使两个组的概率之和近于相同，并对各组赋予一个二进制码元“0”和“1”；

(3) 将每一大组的信源符号**进一步再分成两组**，使划分后的两个组的概率之和近于相同，并又赋予两个组一个二进制符号“0”和“1”。

(4) 如此**重复**，直至每个组只剩下一个信源符号为止。

(5) 信源符号所**对应的码字**即为费诺码。

例题

对例5-4中的信源进行费诺编码。

消息符号 a_i	各个消息概率 $p(a_i)$	第一次分组	第二次分组	第三次分组	第四次分组	二元码字	码长 K_i
a_1	0.20	0	0			00	2
a_2	0.19		1	0		010	3
a_3	0.18			1		011	3
a_4	0.17	1	0			10	2
a_5	0.15		1	0		110	3
a_6	0.10			1	0	1110	4
a_7	0.01				1	1111	4

例题

平均码长

$$\overline{K} = \sum_{i=1}^7 p(a_i) K_i = 2.74 \quad \text{码元/符号}$$

信息效率

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

- 费诺码比较适合于每次分组概率都很接近的信源，特别是对每次分组概率都相等的信源进行编码时，可达到理想的编码效率。

例题

有一单符号离散无记忆信源

$$\begin{bmatrix} X \\ P(X) \end{bmatrix} = \begin{Bmatrix} x_1, & x_2, & x_3, & x_4, & x_5, & x_6 & x_7 & x_8 \\ 1/4 & 1/4 & 1/8 & 1/8 & 1/16 & 1/16 & 1/16 & 1/16 \end{Bmatrix}$$

- 对该信源编二进制费诺码。
- 编码过程如表：

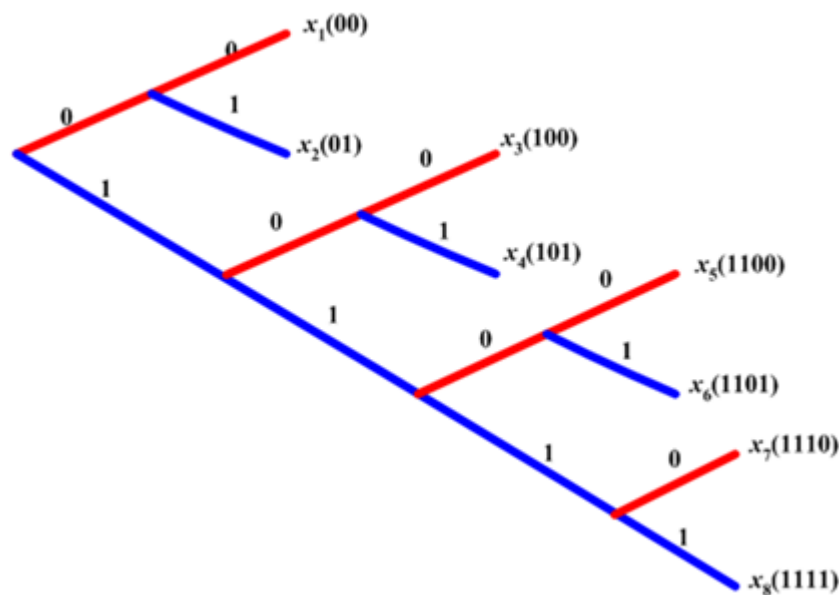
信源符号	概率	编码				码字	码长
x_1	0.25	0	0			00	2
x_2	0.25		1			01	2
x_3	0.125	1	0	0		100	3
x_4	0.125			1		101	3
x_5	0.0625		0	0	0	1100	4
x_6	0.0625				1	1101	4
x_7	0.0625		1	0	0	1110	4
x_8	0.0625				1	1111	4

例题

- 信源熵为 $H(X)=2.75$ (比特/符号)
- 平均码长为

$$\bar{K} = (0.25 + 0.25) \times 2 + 0.12 \times 2 \times 3 + 0.0625 \times 4 \times 4 = 2.75(\text{码元/符号})$$

- 编码效率为
 $\eta=1$
- 之所以如此，是因为每次所分两组的概率恰好相等。



费诺码的特点

费诺码的编码方法实际上是一种构造码树的方法，所以费诺码是即时码。

费诺码考虑了信源的统计特性，使概率大的信源符号能对应码长短的码字，从而有效地提高了编码效率。

但是费诺编码方法不一定使短码得到充分利用。尤其当信源符号较多时，若有一些符号概率分布很接近时，分两大组的组合方法会很多。可能某种分大组的结果，会使后面小组的“概率和”相差较远，使平均码长增加。

分组码的局限性

概率特性必须得到精确的测定，它若略有变化，还需更换码表。

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。

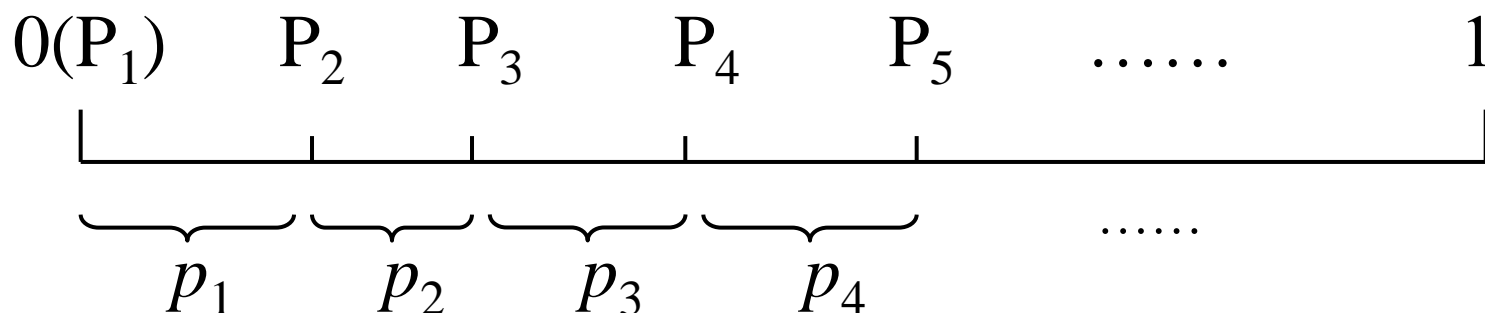
而合并的符号数增大时，码表中的码字数很多，设备将越来越复杂。

因此在实用中常需作一些改进，有必要研究非分组码。

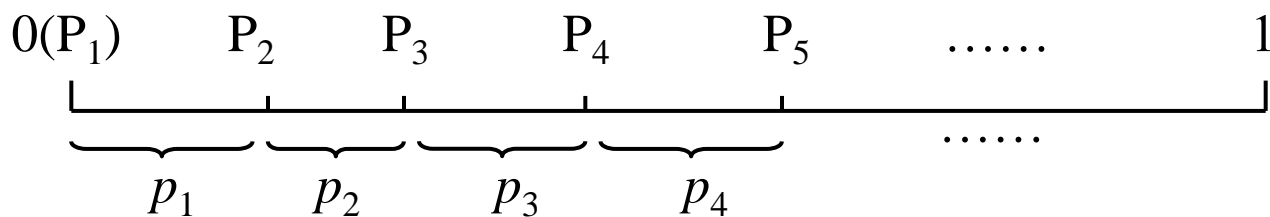
5.4.2 算术编码

算术码是一种非分组码，其基本思路是

累积概率 $P(S)$ 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 $p(S)$ ，小区间内的任一点可用来代表这序列



与香农码的区别：香农码考虑单个符号，算术编码考虑的是整个数据文件



$$L = \left\lceil \log \frac{1}{p(S)} \right\rceil$$

$\lceil x \rceil$ 代表大于或等于 x 的最小整数。

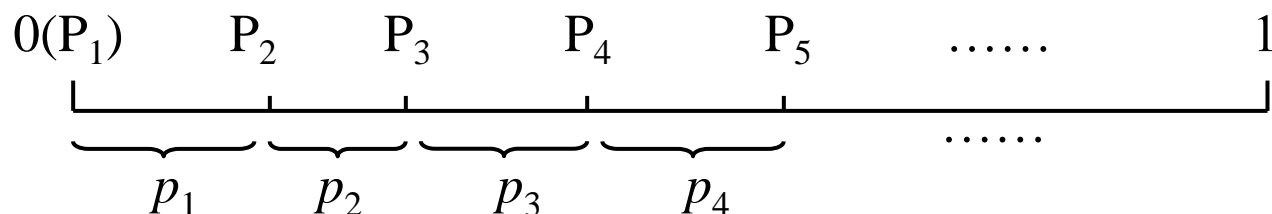
把积累概率 $P(S)$ 写成二进制的小数，取其前 L 位；如果有尾数，就进位到第 L 位，这样得到一个数 C

例如 $P(S)=0.10110001$ ， $p(S)=1/17$ ，则 $L=5$ ，

得 $C=10111$

这个 C 就可作为 S 的码字

如何求序列的概率 $p(S)$ 和累积概率 $P(S)$



信源符号集 $A = \{a_1, a_2, \dots, a_n\}$, L 长信源序列共有 n^L 种可能序列, 因为序列长度 L 很大, 很难得到对应序列的概率和累积概率, 只能从已知的信源符号概率中递推得到。

信源符号概率分布为

$$P = [p(a_1), p(a_2), \dots, p(a_n)] = [p_1, p_2, \dots, p_n]$$

定义各符号的累积概率为

$$P_r = \sum_{i=1}^{r-1} p_i, \quad \text{且有} \quad p_r = P_{r+1} - P_r$$

累积概率递推公式

有二元符号序列 $S=011$, 把3个二元符号的序列按自然二进制数排列, 000, 001, 010, 011,..., 则S的累积概率为

$$P(S) = p(000) + p(001) + p(010)$$

如果S后面接一个“0”, 累积概率就成为

$$P(S, 0) = P(0110) = p(0000) + p(0001) + p(0010) + p(0011) + p(0100) + p(0101)$$

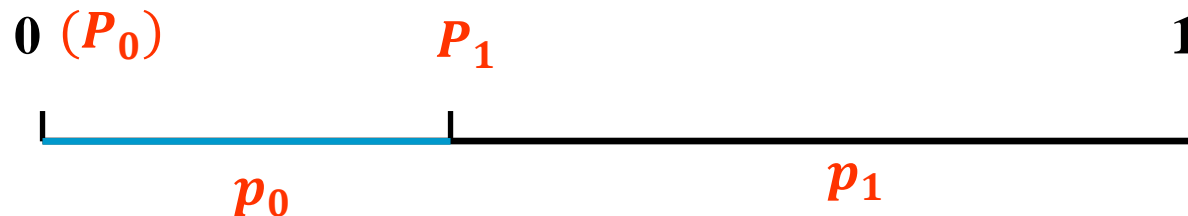
由归一律, $p(0000) + p(0001) = p(000)$ 等, 得

$$P(S, 0) = P(0110) = p(000) + p(001) + p(010) = P(S)$$

同理,

$$\begin{aligned} P(S, 1) &= P(0111) = p(0000) + p(0001) + p(0010) + p(0011) + p(0100) + p(0101) + p(0110) \\ &= P(S) + p(0110) = P(S) + p(S)p_0 \end{aligned}$$

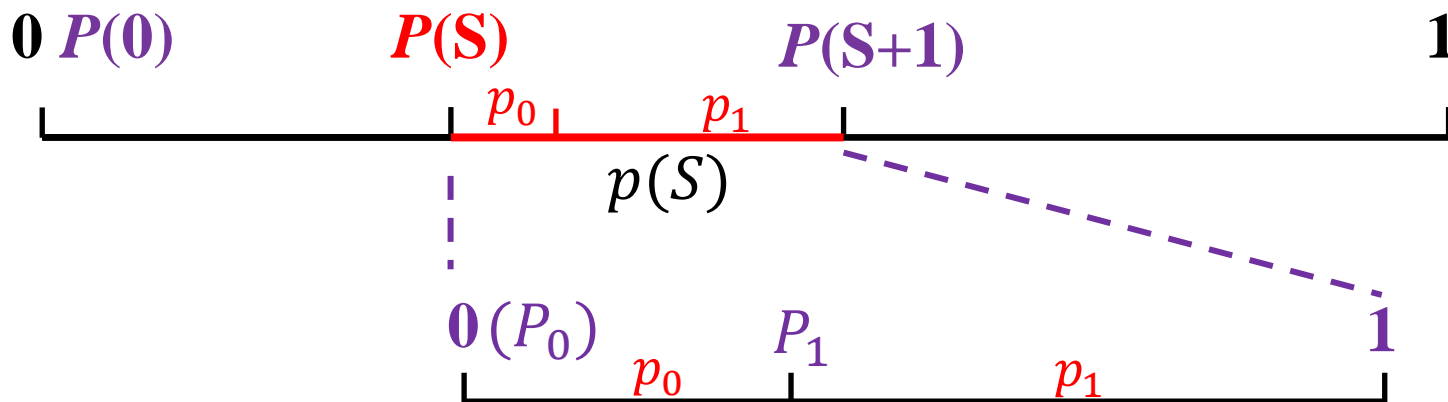
当 $A=\{0,1\}$, 即二元信源时: $P_0=0$; $P_1=p_0$



$$P(S, 0) = P(S) = P(S) + p(S)P_0$$

$$P(S, 1) = P(S) + p(S)p_0 = P(S) + p(S)P_1$$

统一写作 $P(S, r) = P(S) + p(S)P_r, \quad r = 0, 1$



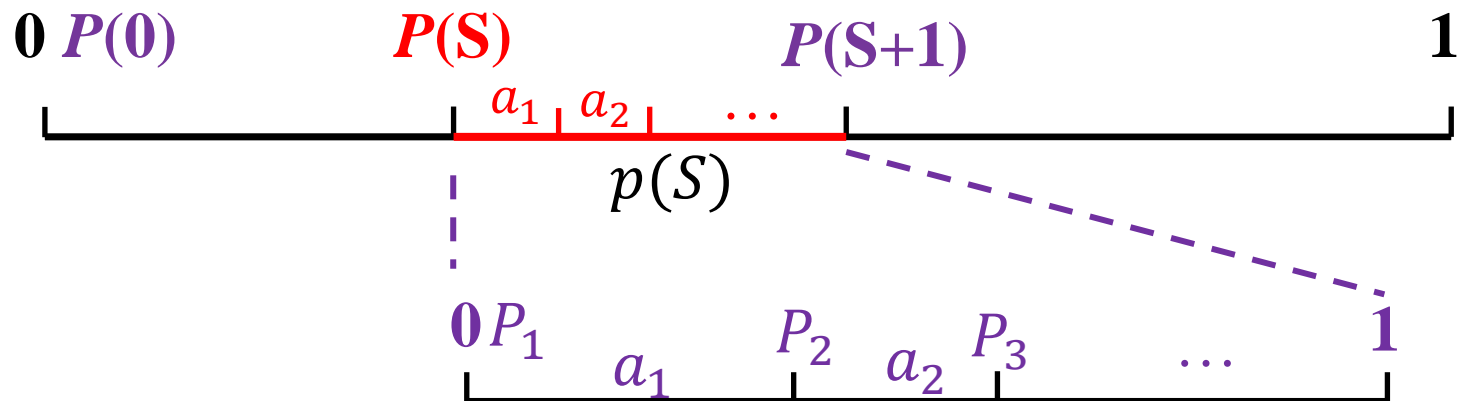
P_r 把状态区间 $p(S)$ 按概率比例划分

一般递推公式

推广到多元序列，即可得到一般的递推公式

$$P(S, a_r) = P(S) + p(S)P_r$$

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



P_r 把状态区间 $p(S)$ 按概率比例划分

随着序列的长度不断增加， $p(S)$ 越来越小， $P(S)$ 所在区间的长度就越短，也就可以更加精确地确定 $P(S)$ 的位置

编码过程

实际应用中，用码字 $C(S)$ 表示累积概率 $P(S)$ ，用状态区间 $A(S)$ 表示序列的概率 $p(S)$ ，则递推公式为

$$\begin{cases} C(S, r) = C(S) + A(S)P_r \\ A(S, r) = A(S)p_r \end{cases}$$

1. 设置两个存储器，起始时令 $A(\varphi) = 1, C(\varphi) = 0$, φ 表示空集，即起始时码字为0，状态区间为1
2. 每输入一个信源符号，存储器C和A就按照上式更新一次，直至信源符号输入完毕

3. 确定码长

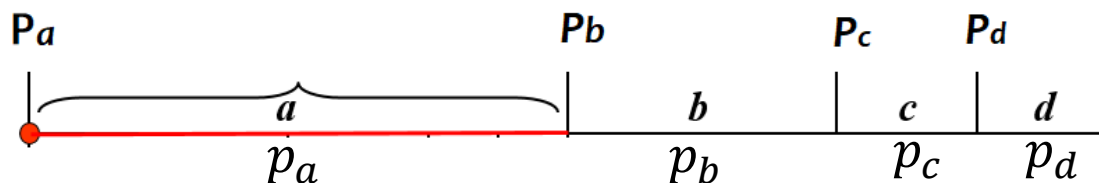
$$L = \left\lceil \log \frac{1}{A(S)} \right\rceil$$

4. 取存储器C小数点后L位的内容作为该序列的码字输出。（如果有尾数，则进位到第L位）

在编码过程中, 每输入一个符号要进行**乘法**和**加法**运算, 所以称为算术编码。

例5-9有四个符号 a, b, c, d 构成简单序列 $S = abda$ ，各符号及其对应概率如下表，算术编解码过程如下：

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



设起始状态为空序列 φ ,

则 $A(\varphi) = 1$, $C(\varphi) = 0$ 。

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

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

$$\begin{cases} C(abd) = C(ab) + A(ab)P_d \\ \quad = 0.01 + 0.001 \times 0.111 = 0.010111 \\ A(abd) = A(ab)p_d \\ \quad = 0.001 \times 0.001 = 0.000001 \end{cases}$$

$$\begin{cases} C(abda) = C(abd) + A(abd)P_a \\ \quad = 0.010111 + 0.0000001 \times 0 = 0.010111 \\ A(abda) = A(abd)p_a \\ \quad = 0.0000001 \times 0.1 = 0.00000001 \end{cases}$$

$$L = \left\lceil \log \frac{1}{A(abcd)} \right\rceil = 7$$

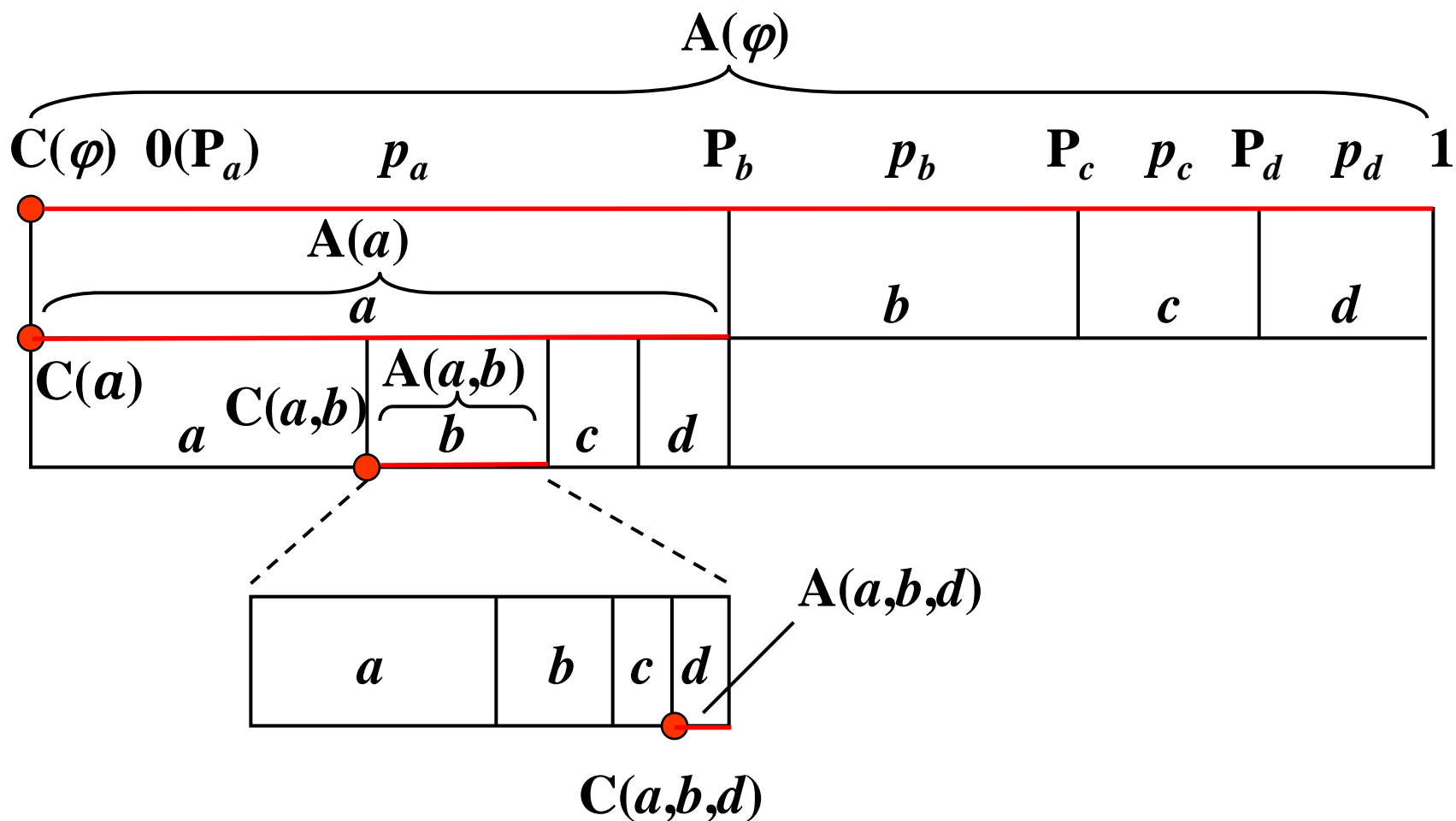
取 $C(abda)$ 的小数点后7位即为编码后的码字**0101110**。

该信源的熵为

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

$$\text{编码效率 } \eta = \frac{1.75}{\frac{7}{4}} = 100\%$$

算术编码过程



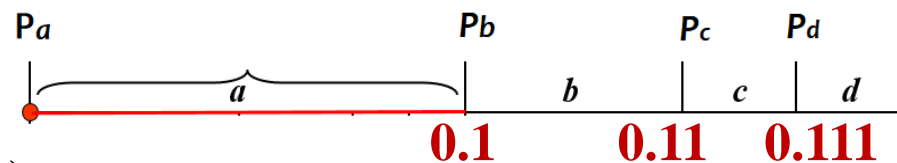
译码

译码可通过比较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

根据递推公式的相反过程译出每个符号

译码过程



- $C(S)=0.010111 < 0.1 \in [0, 0.1)$

第一个符号为 a

去掉累积概率 P_a : $0.010111 - 0 = 0.010111$

- 放大至 $[0, 1](\times p_a^{-1})$: $0.010111 \times 2^1 = 0.10111 \in [0.1, 0.110)$

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 $[0, 1](\times p_d^{-1})$: $0 \times 2^4 = 0 \in [0, 0.1)$

第四个符号为 a

算术编码小结

算术编码的编码效率很高，当信源符号序列很长时， L 很大，平均码长接近信源熵。

从性能上来看，算术编码具有许多优点，它所需的参数较少、编码效率高、编译码简单，不象哈夫曼码那样需要一个很大的码表。算术编码在图像数据压缩标准（如**JPEG**）中得到广泛的应用。

例题

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。

5.4.3 LZ编码

对于统计特性确知的平稳信源，已有huffman编码和算术编码高效编码方法，其平均码长可逼近信源的平均符号熵，而且实现困难不算太大，所以已进入实用。

要确知信源的统计特性相当困难。**通用编码**旨在信源统计特性不知时，对信源进行编码，而且编码效率很高。

两位以色列研究者J. Ziv和A. Lempel独辟蹊径，完全脱离Huffman及算术编码的设计思路，创造出了一系列比Huffman编码更有效，比算术编码更快捷的通用压缩算法——LZ算法。

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。

LZ编码

1965年苏联数学家Kolmogolov提出利用信源序列的结构特性来编码。

LZ77算法

1977年, J. Ziv and A. Lempel, “A Universal Algorithm for Sequential Data Compression”, IEEE Transaction on Information Theory, May 1977)。这个算法一般称为LZ77。

1978年, 两人又提出了改进算法, 称为LZ78算法

1982年, LZSS算法作为LZ77的改良被提出, 大量的压缩软件使用了这个算法, 如zip, arj, lharc等

1984年, Welch 提出了LZ78的改良LZW算法, LZW也得到广泛的应用, 如GIF图像中的压缩算法, Unix下的compress程序。但由于LZW存在专利问题, 此后的应用并没有LZSS广泛。

LZ77, LZ78, 这两个经典文献提出的算法成为了当代几乎所有流行压缩软件的算法核心

LZ78 编码步骤

设信源符号集 $A = (a_1, a_2, \dots, a_K)$ 共 K 个符号，输入信源符号序列 $U = (u_1, u_2, \dots, u_L)$ ，编码是将此序列分成不同的段。

分段规则

尽可能取最少个相连的信源符号，并保证各段都不相同

- 1、在第 i 步，从 s_{i-1} 短语后的第一个符号开始向后搜索此前未出现过的最短短语 s_i ，将该短语添入字典第 i 段
- 2、假设 s_i 去掉最后一个符号 x 后所得的前缀是在第 j 步出现的短语
- 3、对 s_i 的编码为 (j, x) 。对段号 j ，用 $\lceil \log i \rceil$ 比特来表示，符号 x 用 $\lceil \log K \rceil$ 比特来表示

例题：编码

设 $U=\{a_1, a_2, a_3, a_4\}$ ，信源序列为 $\underline{a_1}\underline{a_2}\underline{a_1}\underline{a_3}\underline{a_2}\underline{a_4}\underline{a_2}\underline{a_4}\underline{a_3}\underline{a_1}\underline{a_1}\underline{a_4}\cdots$

按照分段规则，可以分为 $a_1, a_2, a_1a_3, a_2a_4, a_2a_4a_3, a_1a_1, a_4$

符号编码表

a_1	a_2	a_3	a_4
00	01	10	11

段号 <i>i</i>	短语	<i>j</i>	<i>x</i>	码字	编码
1	a_1	0	a_1	$(0, a_1)$	0,00
2	a_2	0	a_2	$(0, a_2)$	0,01
3	a_1a_3	1	a_3	$(1, a_3)$	01,10
4	a_2a_4	2	a_4	$(2, a_4)$	10,11
5	$a_2a_4a_3$	4	a_3	$(4, a_3)$	100,10
6	a_1a_1	1	a_1	$(1, a_1)$	001,00
7	a_4	0	a_4	$(0, a_4)$	000,11

对 s_i 的编码为 (j, x) 。对段号 j ，用 $\lceil \log i \rceil$ 比特来表示，符号 x 用 $\lceil \log K \rceil$ 比特来表示。

例题-译码

码序列 00000101101011100100010000011

符号编码

a_1	a_2	a_3	a_4
00	01	10	11

段号 <i>i</i>	短语	j	x	编码
1	a_1	0	a_1	000
2	a_2	0	a_2	001
3	a_1a_3	1	a_3	0110
4	a_2a_4	2	a_4	1011
5	$a_2a_4a_3$	4	a_3	10010
6	a_1a_1	1	a_1	00100
7	a_4	0	a_4	00011

例5-10 信源符号集 $A=\{a, b\}$ ，输入信源符号序列 $U=(abbabaabbabbbaaaaba\dots)$

解：对输入序列 U 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 $(0, b)$ 。如表5-8。

接着 U 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 $(2, a)$ 。

对 s_i 的编码为 (j, x) 。对段号 j ，用 $\lceil \log i \rceil$ 比特来表示，符号 x 用 $\lceil \log K \rceil$ 比特来表示

表5-8 LZ编码示例

$U=(abbabaabbabbaaaaba\dots)$

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)


此例中编码后输出的码流较长，编码效率不是很高，这是因为信源长度短。当信源长度增长时，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译码

- LZ编码的方法非常简捷，译码也很简单。可以一边译码一边建立字典；收到码字 (j, x) ，则在字典中找到第 j 个短语，加上符号 x 即可译出对应的新短语，并添入字典，因此发送时无需发送字典本身

LZ编码小结

- LZ编码算法逻辑简单，硬件实现廉价，运算速度快，在数据传输和计算机数据存储中得到广泛的应用。
- 目前该算法以及它们的各种变体几乎垄断了整个通用数据压缩领域，我们熟悉的WinZIP、WinRAR、gzip等压缩工具以及ZIP、GIF、PNG 等文件格式都是 LZ 系列算法的受益者。



离散无记忆信源发出 ab 两种符号，若信源输出的序列为 $babbabbbbabbabbbb$ ，对其进行LZ编码。

5.4.4 游程编码（RLE, run-length encoding）

在二元序列中，连0段称为0游程

连1段称为1游程

二元码序列： 000101110010001 ...

可变换成下列游程序列： 31132131 ...

若已知二元序列以0起始，从游程序列很容易恢复成原来的二元序列；

游程序列是多元序列，各长度可按哈夫曼编码或其它方法处理以达到压缩码率的目的。

传真编码

文件图纸要根据清晰度要求来决定空间扫描分辨率。国际标准规定，一张A4幅面文件应该有1188或2376条扫描线，每条扫描线有1728个像素。因此，A4文件纸约有2.05M像素/公文纸。从节省传送时间和存储空间来说，必须进行数据压缩。

CCITT（国际电报电话咨询委员会）将MH (Modified Huffman) 码作为文件传真一维压缩编码的国际标准。

MH码结合了游程编码和哈夫曼编码

■ MH编码方法：

1. 黑白游程分别对应不同的编码表；
2. 游程长度在 $0 \sim 63$ 时，码字直接用相应的终端码（结尾码）表示；

例：一行中连续19个白，接着连续30个黑，即白游程长度为19，黑游程长度为30. 查表得码字为

0001100 ; 000001101000

MH码表（一） 结尾码

RL长度	白游程码字	黑游程码字	RL长度	白游程码字	黑游程码字	RL长度	白游程码字	黑游程码字
0	00110101	0000110111	21	001011	00001101100	42	00101011	000011011010
1	000111	010	22	0000011	00000110111	43	00101100	000011011011
2	0111	11	23	0000100	00000101000	44	00101101	000001010100
3	1000	10	24	0101000	00000010111	45	00000100	000001010101
4	1011	011	25	0101011	00000011000	46	00000101	000001010110
5	1100	0011	26	0010011	000011001010	47	00001010	000001010111
6	1110	0010	27	0100100	000011001011	48	00001011	000001100100
7	1111	00011	28	0011000	000011001100	49	01010010	000001100101
8	10011	000101	29	00000010	000011001101	50	01010011	000001010010
9	10100	000100	30	00000011	000001101000	51	01010100	000001010011
10	00111	0000100	31	00011010	000001101001	52	01010101	000000100100
11	01000	0000101	32	00011011	000001101010	53	00100100	000000110111
12	001000	0000111	33	00010010	000001101011	54	00100101	000000111000
13	000011	00000100	34	00010011	000011010010	55	01011000	000000100111
14	110100	00000111	35	00010100	000011010011	56	01011001	000000101000
15	110101	000011000	36	00010101	000011010100	57	01011010	000001011000
16	101010	0000010111	37	00010110	000011010101	58	01011011	000001011001
17	101011	0000011000	38	00010111	000011010110	59	01001010	000000101011
18	0100111	0000001000	39	00101000	000011010111	60	01001011	000000101100
19	0001100	00001100111	40	00101001	000001101100	61	00110010	000001011010
20	0001000	00001101000	41	00101010	000001101101	62	00110011	000001100110
						63	00110100	000001100111

MH码表（二）组合基干码

RL长度	白游程码字	黑游程码字	RL长度	白游程码字	黑游程码字
64	11011	0000001111	960	011010100	000000111001
128	10010	000011001000	1024	011010101	0000001110100
192	010111	000011001001	1088	011010110	0000001110101
256	0110111	000001011011	1152	011010111	0000001110110
320	00110110	000000110011	1216	011011000	0000001110111
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

■ MH编码方法:

3. 游程长度在64~1728, 用一个组合码加上一个
结尾码为相应码字;

例: 白游程长度为65 ($=64+1$), 查表得码字,

11011 | 000111

黑游程长度为856 ($=832+24$), 查表得码字,

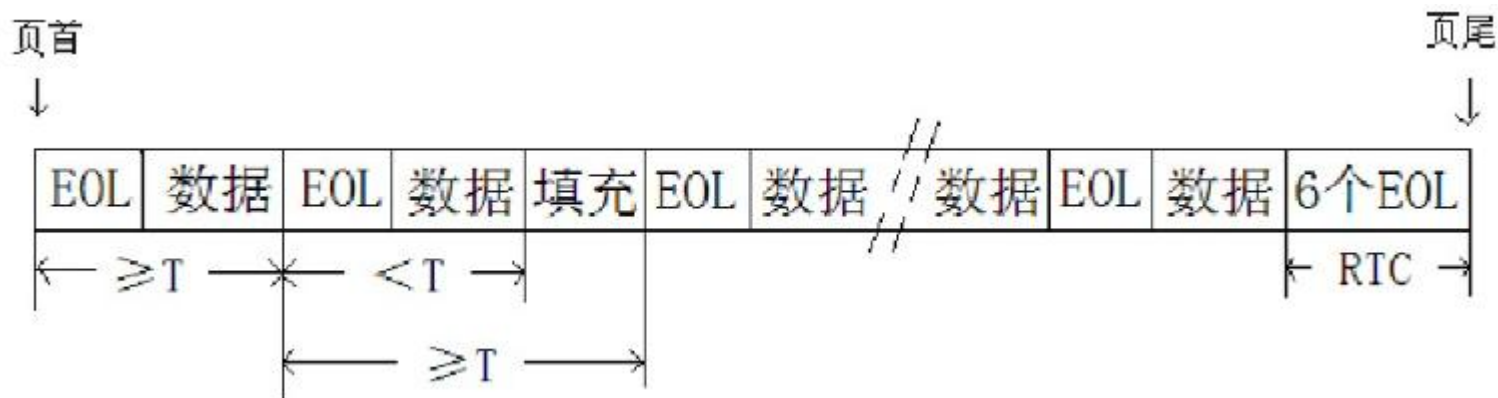
0000001001101 | 00000010111

■ MH编码方法：

4. 规定每行都从白游程开始，若实际出现黑游程开始的话，则在行首加上零长度白游程码字，每行结束用一个结束码（EOL）；
5. 每行恢复成1728个像素，否则有错；
6. 每页文件开始第一个数据前加一个结束码。
每页尾连续使用6个结束码表示结尾；

■ MH编码方法：

7. 为了传输时实现同步操作，规定T为每编码行的最小传输时间。一般规定T最小为20ms，最大为5s。若编码行传输时间小于T，则在结束码之前填以足够的“0”码元（称填充码）。



例：有一页传真文件其中某一扫描线上的像素点如图所示。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。

75个白	5个黑	9个白	18个黑	1621个白
------	-----	-----	------	--------

(1) 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，答案就是**57 bit**。

(3) 压缩前数据总比特：

75+5+9+18+1621=1728 bit。

所以数据压缩比：1728:57=30.316:1

75个白	5个黑	9个白	18个黑	1621个白
------	-----	-----	------	--------

根据编码的3个规则，参考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。所以答案为：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：规则(1)。结尾码为9（白）对应的10100。即为答案。
- 18个黑：规则(1)。结尾码为18（黑）对应的0000001000。即为答案。
- 1621个白：规则(2)。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。所以答案为：0100110100010111。
- EOL：规则(3)。同步码，查表可得为0000000000001。

MH编码的特点

- 1、MHC的编码表是根据一组典型文件的游程概率分布的统计平均值而构造出来的。
- 2、MHC的码字构成是由组合基干码和结尾码的组合实现的，这样一来，码表大大缩减。
- 3、MHC具有一定的检错能力，两个EOL码之间的数据经过译码恢复后应为1728个像素，否则认为出错。

第5章小结

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

$$\sum_{i=1}^n m^{-K_i} \leq 1$$

编码效率

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

第五章 小结

无失真信源编码定理（香农第一定理）

– 定长编码定理

$$\frac{K_L}{L} \log m \geq H_L(\mathbf{X}) + \varepsilon$$

– 变长编码定理

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