第5章 信道编码

5.1 信道编码的基本概念

在有噪信道上传输信号时，所收到的数据不可避免地会出现差错，所以在数字通信、数据传输、图像传输、计算机网络等数字信息交换和传输中所遇到的一个主要问题是可靠性问题。不同的用户对可靠性的要求是大相径庭的，例如：对于普通电报，差错概率(误码率)在10^{-3}时是可以接受的；而对导弹运行轨道数据的传输，如此高的差错率将使导弹偏离预定的轨道，这显然是不允许的。在数字通信系统中要从多种途径来研究提高系统可靠性的方法。首先要进行合理基带信号设计、选择合适调制解调方式以及采用匹配滤波、均衡等，这是降低差错的根本措施，其目的是改善信道特性，减少差错。在此基础之上利用纠错编码技术对差错进行控制，可大大提高系统的抗干扰能力，降低误码率，这是提高系统可靠性的一项极为有效的措施，也是我们后面要研究和解决的问题。

中文教学录像

双语教学录像

视频

一个通信系统传输消息必须可靠、快速，如何合理解决可靠性和有效性这对矛盾是数字通信系统设计的一个关键问题。在香农之前，人们普遍认为在有噪信道进行信息传输时，要获得任意小的错误概率的唯一途径是不断减小传输速率，直至为零。1948年，香农在贝尔技术杂志上发表了奠基性文章《通信的数学理论》，标志着信息论与编码理论这一学科的创立。在文中香农提出了著名的有噪信道编码定理，该定理从理论上证明了通过信道编码技术可使信息传输率接近信道容量，同时系统的错误概率达到任意小，即实现可靠性和有效性的有机统一。编码定理及其证明虽然没有给出编码的具体设计方法，却指出了达到信道容量的编译码方法的方向和途径。此后，构造可逼近信道容量(香农限)的信道编码具体方法以及可实现的有效译码算法一直是信道编码理论和技术研究的中心任务，也是编码学者长期追求的目标。

5.1.1 信道编码的一般方法

纠错编码又称信道编码或差错控制编码，它涉及很多理论问题和数学知识，本节先介绍基本的纠错编码方法和概念。

设信源编码器输出的二元数字信息序列为(001010110001…)，序列中每个数字都是一个信息元素。为了适应信道的最佳传输而进行编码，首先需要对信息序列进行分组。一般是以截取相同长度的码元进行分组，每组长度为k(含有k个信息元)，这种序列一般称为**信息组**或**信息序列**，例如上面的信息序列以$k=1$分组为0(代表“雨”)、1(代表“晴”)。如果将这样的信息组直接送入信道传输，它是没有任何抗干扰能力的，因为任意信息组中任意元素出错都会变成另一个信息组，如信息元(0)出错，变成(1)，而它们代表着不同的信息组，因此原本发送的气象信息在接收端就会判断错误(“雨”变成“晴”)。可见不管k的大小如何，若直接传输信息组是无任何抗干扰能力的。

如果在各个信息组后按一定规律人为地添上一些数字，如上例，在$k=1$的信息组后再添上两位数字，使每一组的长度变为3，这样的各组序列称为**码字**。码字长度记为n，本例中$n=k+2$。每个码字的前一个码元为原来的信息组，称为**信息元**，它主要用来携带要传输的信息内容。后两个新添的码元称为**监督元**(或**校验元**)，其作用是利用添加规

则来监督传输是否出错。如果添监督元的规则为新添的每个监督元符号(0 或 1)与前面信息元符号(0 或 1)一样,那么这样的码字共有 $2^k=2^1=2$ 个,即{(000),(111)},它们组成了一个码字集合,称为(3,1)重复码,其每个码字分别代表一个不同的信息组。而在 3 位二进制序列(码组)中共有 $2^3=8$ 个,除以上 2 个作为码字外,还有 6 个未被选中,即这 6 个码组不在发送之列,称为**禁用码组**,而被编码选中的 n 重码字称为**许用码组**。接收端接收序列不在码字集合中,说明不是发送端所发出的码字,从而确定传输有错。因此这种变换后的码字就具有一定的抗干扰能力。以二元对称信道 BSC(0.01)为例,若直接传输信息元,接收端平均错误译码概率就等于信道的错误传输概率,即 $P_E=10^{-2}$,而采用(3,1)重复码后,只有当码字中 3 位码元都出错才会误判(将发端传递的"雨"消息误判成"晴"或相反),否则可以检出传输出错,此时系统错误译码概率 $P_E=p^3=10^{-6}$;当传输中每个码字仅 1 位出错,接收端还能根据编码规律进行自动纠错,可计算出此时接收端平均错误译码概率:

$$P_E=p^3+3(1-p)p^2\approx 3\times 10^{-4}$$

可见采用(3,1)重复码后,接收端无论采取检错方式还是纠错方式,系统的错误译码概率都会降低。

以上就是一种最简单的信道编码,在一位信息元之后添加了两位监督元,从而获得了抗干扰能力。一般来说,添加的监督元位数越多,码字的抗干扰能力就越强,不但能识别传输是否有错(检错),还可以根据编码规则确定哪一位出错(即纠错)。因此,纠错编码的一般方法可归纳为:在传输的信息码元之后按一定规律产生一些附加码元(图 5.1.1),经信道传输,在传输中若码字出现错误,接收端能利用编码规律发现码的内在相关性受到破坏,从而按一定译码规则自动纠正错误,降低误码率 P_E。

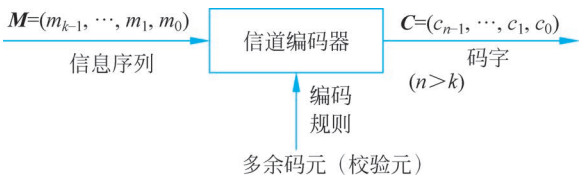


图 5.1.1　信道编码过程


视频

通常按以下方式对纠错码进行分类。

(1) 按照对信息元处理方法分为分组码与卷积码两大类。

分组码是把信源输出的信息序列,以 k 个码元划分为一段,通过编码器把这段 k 个信息元按一定规则产生 r 个校验(监督)元,输出码长为 $n=k+r$ 的一个码组。这种编码中每一码组的校验元仅与本组的信息元有关,而与别组无关。分组码用(n,k)表示,n 表示码长,k 表示信息位。

卷积码是把信源输出的信息序列,以 k_0 个(通常 $k_0<k$)码元分为一段,通过编码器输出码长为 n_0($n_0>k_0$)的码段,但是该码段的 n_0-k_0 个校验元不仅与本组的信息元有关,而且与其前 m 段的信息元有关,一般称 m 为编码存储,因此卷积码用(n_0,k_0,m)表示。

(2) 根据校验元与信息元之间的关系分为线性码与非线性码。

若校验元与信息元之间的关系是线性关系(满足线性叠加原理),则称为线性码;否则,称为非线性码。

目前实用的纠错码以线性码为主,且非线性码的分析比较困难,故本书仅讨论线性码。

(3) 按照所纠错误的类型可分为纠随机错误码、纠突发错误码以及既纠随机错误又纠突发错误码。

(4) 按照每个码元取值可分为二进制码与 q 进制码($q=p^m$,p 为素数,m 为正整数)。

(5) 按照对每个信息元保护能力是否相等可分为等保护纠错码与不等保护(UEP)纠错码。

除非特别说明,今后讨论的纠错码均指等保护能力的码。

此外,在分组码中按照码的结构特点又可分为循环码与非循环码。为了清晰起见,把上述分类用图 5.1.2 表示。

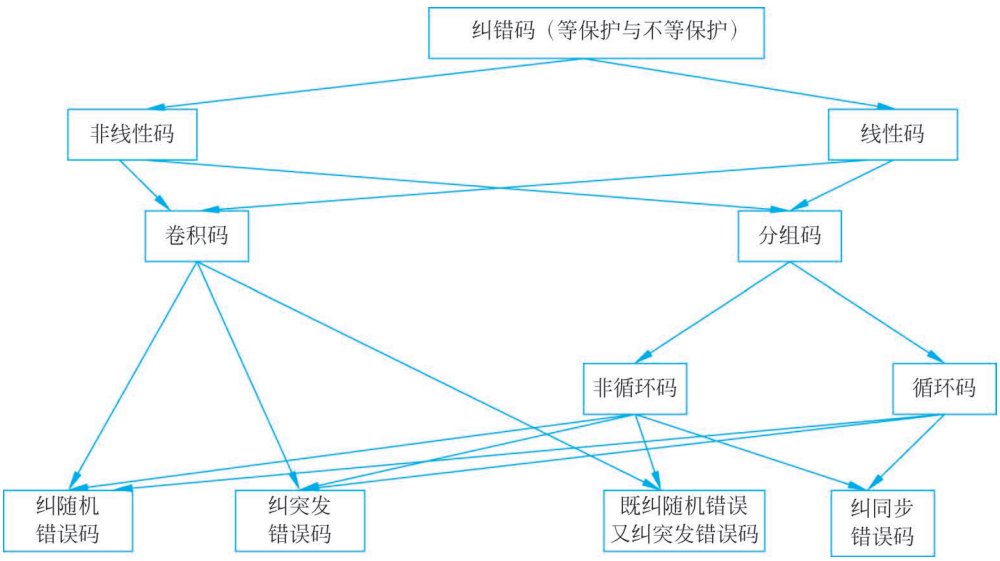


图 5.1.2 纠错码分类

为了今后学习的方便,除了上面讲到的基本概念外,还将介绍一些编码中常用的参数。

5.1.2 信道编码的基本参数

1. *码率*

一般把分组码记为(n,k)码,n 为编码输出的码字长度,k 为输入的信息组长度,在一个(n,k)码中,信息元位数 k 在码字长度 n 中所占的比重称为码率,用 R 表示。对于二元线性码,它可等效为编码效率,即

$$\eta = R = \frac{k}{n}$$

码率是衡量所编的分组码有效性的一个基本参数，码率越大，表明信息传输的效率越高。但对编码来说，每个码字中所加进的监督元越多，码字内的相关性越强，码字的纠错能力越强。而监督元本身并不携带信息，单纯从信息传输的角度来说是多余的。一般地说，码字中冗余度越高，纠错能力越强，可靠性越高，而此时码的效率则降低了。所以信道编码必须注意综合考虑有效性与可靠性的问题，在满足一定纠错能力要求的情况下，总是力求设计码率尽可能高的编码。

2. 汉明距离与重量

定义 5.1.1 一个码字 $\boldsymbol{C}$ 中非零码元的个数称为该码字的(汉明)重量，简称码重，记为 $W(\boldsymbol{C})$。

若码字 $\boldsymbol{C}$ 是一个二进制 n 重，则 $W(\boldsymbol{C})$ 就是该码字中“1”码的个数。例如：

若 $\boldsymbol{C}_1=(000)$，则 $W(\boldsymbol{C}_1)=0$；

若 $\boldsymbol{C}_2=(011)$，则 $W(\boldsymbol{C}_2)=2$；

若 $\boldsymbol{C}_3=(101)$，则 $W(\boldsymbol{C}_3)=2$；

若 $\boldsymbol{C}_4=(110)$，则 $W(\boldsymbol{C}_4)=2$。

定义 5.1.2 两个长度相同的不同码字 $\boldsymbol{C}$ 和 $\boldsymbol{C}'$ 中对应位码元不同的码元数目称为这两个码字间的汉明距离，简称码距(或距离)，记为 $d(\boldsymbol{C};\boldsymbol{C}')$。

例如上例中，$d(\boldsymbol{C}_2;\boldsymbol{C}_3)=2$。

在一个码集中，每个码字都有一个重量，每两个码字间都有一个码距，对于整个码集而言，还有以下两个定义。

定义 5.1.3 一个码集中非零码字的汉明重量的最小值称为该码的最小汉明重量，记为 $W_{\min}(\boldsymbol{C})$。

定义 5.1.4 一个码集中任两个码字间的汉明距离的最小值称为该码的最小汉明距离，记为 d_0 或 $d_{\min}$。

例如上例中的(3,2)码的最小汉明距离为 2。

对于二元编码，不难得出如下关系式：

$$d(\boldsymbol{C}_1;\boldsymbol{C}_2)=W(\boldsymbol{C}_1\oplus\boldsymbol{C}_2)$$

$$W(\boldsymbol{C}_1\oplus\boldsymbol{C}_2)=W(\boldsymbol{C}_1)+W(\boldsymbol{C}_2)-2W(\boldsymbol{C}_1\odot\boldsymbol{C}_2) \tag{5.1.1}$$

式中，“$\oplus$”是模 2 加法运算；“$\odot$”是模 2 乘法运算。

例如，$\boldsymbol{C}_1=(11001)$，$\boldsymbol{C}_2=(10111)$，则有 $\boldsymbol{C}_3=\boldsymbol{C}_1\oplus\boldsymbol{C}_2=(01110)$，$d(\boldsymbol{C}_1;\boldsymbol{C}_2)=W(\boldsymbol{C}_3)=3$；$\boldsymbol{C}_1\odot\boldsymbol{C}_2=(10001)$，$W(\boldsymbol{C}_3)=3+4-2\times2=3$。

一个 (n,k) 分组码共含有 2^k 个码字，每两个码字之间都有一个汉明距离 d，因此要计算其最小距离，需比较计算 $2^{k-1}(2^k-1)$ 次。当 k 较大时，计算量就很大。但对于 5.3 节将要介绍的 (n,k) 线性分组码具有以下特点：任意两个码字之和仍是线性分组码中的一个码字，因此两个码字之间的距离 $d(\boldsymbol{C}_l;\boldsymbol{C}_2)$ 必等于其中某一个码字 $\boldsymbol{C}_3=\boldsymbol{C}_1+\boldsymbol{C}_2$ 的重量。

定理 5.1.1 (n,k)线性分组码的最小距离等于非零码字的最小重量：

$$d_0=\min_{\boldsymbol{C},\boldsymbol{C}'\in(n,k)}\{d(\boldsymbol{C};\boldsymbol{C}')\}=\min_{\substack{\boldsymbol{C}_i\in(n,k)\\ \boldsymbol{C}_i\neq 0}}W(\boldsymbol{C}_i)$$

该定理的证明见定理 5.3.1。这样一来，(n,k)线性分组码的最小距离计算只需检查 2^k-1 个非零码字的重量即可。

此外，码的距离和重量还满足三角不等式的关系：

$$d(\boldsymbol{C}_1;\boldsymbol{C}_2)\leqslant d(\boldsymbol{C}_1;\boldsymbol{C}_3)+d(\boldsymbol{C}_3;\boldsymbol{C}_2)$$

$$W(\boldsymbol{C}_1+\boldsymbol{C}_2)\leqslant W(\boldsymbol{C}_1)+W(\boldsymbol{C}_2)$$

该性质在研究线性分组码的特性时常用到。一种码的 d_0 值是一个重要参数，它决定了该码的纠错、检错能力。d_0 越大，抗干扰能力越好。

3. 码的纠、检错能力


视频

定义 5.1.5 若一种码的任一码字在传输中出现了 e 个或 e 个以下的错误仍能自动发现，则称该码的检错能力为 e。

定义 5.1.6 若一种码的任一码字在传输中出现 t 个或 t 个以下的错误仍能自动纠正，则称该码的纠错能力为 t。

定义 5.1.7 若一种码的任一码字在传输中出现 t 个或 t 个以下的错误均能纠正，当出现多于 t 个而少于 $e+1$ 个错误$(e>t)$时此码能检出而不造成译码错误，则称该码能纠正 t 个错误同时检测 e 个错误。

(n,k)分组码的纠、检错能力与其最小汉明距离 d_0 有着密切的关系，一般有以下结论。

定理 5.1.2 若码的最小距离满足 $d_0\geqslant e+1$，则码的检错能力为 e。

定理 5.1.3 若码的最小距离满足 $d_0\geqslant 2t+1$，则码的纠错能力为 t。

定理 5.1.4 若码的最小距离满足 $d_0\geqslant e+t+1(e>t)$，则该码能纠正 t 个错误同时检测 e 个错误。

以上结论可以用图 5.1.3 所示的几何图加以说明。

图 5.1.3(a)中 $\boldsymbol{C}_1$ 表示某一码字，当误码不超过 e 个时，该码字的位置移动将不超过以它为圆心、以 e 为半径的圆(实际上是一个多维球)，即该圆代表着码字在传输中出现 e 个以下误码的所有码组的集合。若码的最小距离满足 $d_0\geqslant e+1$，则(n,k)分组码中除 $\boldsymbol{C}$ 这个码字外，其余码字均不在该圆中。这样当码字 $\boldsymbol{C}$ 在传输中出现 e 个以下误码时，接收码组必落在图 5.1.3(a)的圆内，而该圆内除 $\boldsymbol{C}_1$ 外均为禁用码组，从而可确定该接收码组有错。考虑到码字 $\boldsymbol{C}$ 的任意性，图 5.1.3(a)说明，当 $d_0\geqslant e+1$ 时任意码字传输误码在 e 个以下的接收码组均在以其发送码字为圆心、以 e 为半径的圆中，而不会和其他许用码组混淆，使接收端检出有错，即码的检错能力为 e。

图 5.1.3(b)中 $\boldsymbol{C}_1$、$\boldsymbol{C}_2$ 分别表示任意两个码字，当各自误码不超过 t 个时，发生误码后两码字的位置移动将各自不超过以 $\boldsymbol{C}_1$、$\boldsymbol{C}_2$ 为圆心、以 t 为半径的圆。若码的最小距离满足 $d_0\geqslant 2t+1$，则两圆不会相交(由图中可看出两圆至少有 1 位的差距)，设 $\boldsymbol{C}_1$ 传输出错误在 t 个以下变成 $\boldsymbol{C}_1'$，其距离

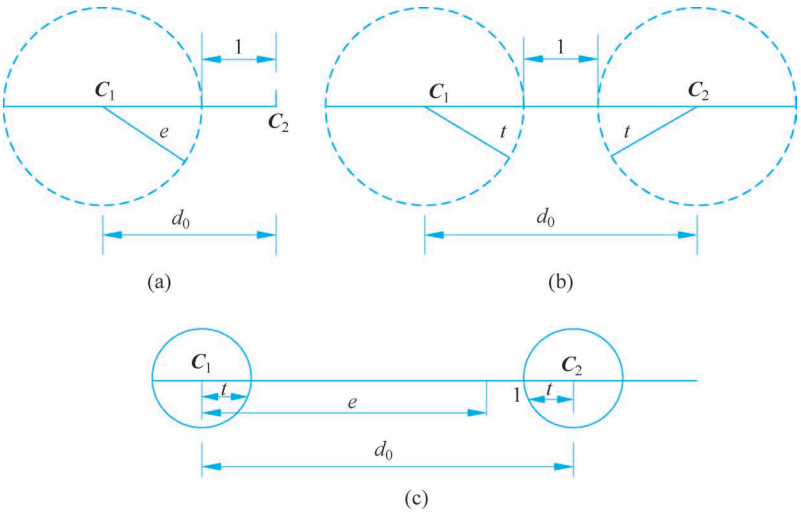


图 5.1.3　码距与检错和纠错能力的关系

$$d(\boldsymbol{C}_1;\boldsymbol{C}_1')\leqslant t$$

根据距离的三角不等式可得

$$d(\boldsymbol{C}_1';\boldsymbol{C}_2)\geqslant t$$

即

$$d(\boldsymbol{C}_1';\boldsymbol{C}_2)\geqslant d(\boldsymbol{C}_1;\boldsymbol{C}_1')$$

根据最大似然译码规则，将 $\boldsymbol{C}_1'$ 译为 $\boldsymbol{C}_1$ 从而纠正了 t 个以下的错误。

定理 5.1.4 中“能纠正 t 个错误同时检测 e 个错误”是指当误码不超过 t 个时系统能自动予以纠正，而当误码大于 t 个而小于 e 个时则不能纠正但能检测出来。该定理的关系由图 5.1.3(c)反映，其结论请读者自行证明。

以上三个定理是纠错编码理论中最重要的基本理论之一，它说明了码的最小距离 d_0 与其纠、检错能力的关系，从 d_0 中可反映码的性能强弱；反过来也可根据以上定理的逆定理设计满足纠检错能力要求的(n,k)分组码。

定理 5.1.5　对于任一(n,k)分组码，若要求：

(1) 码的检错能力为 e，则最小码距 $d_0\geqslant e+1$；

(2) 码的纠错能力为 t，则最小码距 $d_0\geqslant 2t+1$；

(3) 能纠 t 个误码同时检测 $e(e>t)$个误码，则最小码距 $d_0\geqslant t+e+1$。

5.1.3　信道译码规则

已知信道编码器的框图如图 5.1.1 信道编码过程所示，设任一个信息序列 M 是一个 k 位码元的序列，通过编码器按一定的规律(编码规则)产生若干监督元，形成一个长度为 n 的序列，即码字。每个信息序列将形成不同的码字与之对应，在二进制下，k 维序列共有 2^k 种组合，因此编码输出的码字集合共有 $Q=2^k$ 个码字，而二进制下的 n 重矢量共有 2^n 种。显然，编码输出的码字仅是所有二进制 n 重中的一部分，编码实际上就是从

这 2^n 种不同的 n 重矢量中按一定规律(编码规则)选出 2^k 个 n 重矢量(许用码字)代表 2^k 个不同的信源信息序列。

数字通信系统简化模型如图 5.1.4 所示,设发送码字是码长为 n 的序列 $\boldsymbol{C}$:$(c_{n-1},c_{n-2},\cdots,c_1,c_0)$,通过信道传输到达接收端的序列为 $\boldsymbol{R}$:$(r_{n-1},r_{n-2},\cdots,r_1,r_0)$,由于信道中存在干扰,$\boldsymbol{R}$ 序列中的某些码元可能与 $\boldsymbol{C}$ 序列中对应码元的值不同,即产生了错误。而二进制序列中的错误不外乎是 1 错成 0 或 0 错成 1,因此,如果把信道中的干扰也用二进制序列 $\boldsymbol{E}$:$(e_{n-1},e_{n-2},\cdots,e_1,e_0)$表示,则相应有错误的各位 e_i 取值为 1,无错的各位取值为 0,称 $\boldsymbol{E}$ 为信道的错误图样或错型,而 $\boldsymbol{R}$ 就是 n 维序列 $\boldsymbol{C}$ 与 $\boldsymbol{E}$ 模 2 相加的结果:

$$\boldsymbol{R}=\boldsymbol{C}\oplus\boldsymbol{E}$$

式中:"⊕"为模 2 加法。

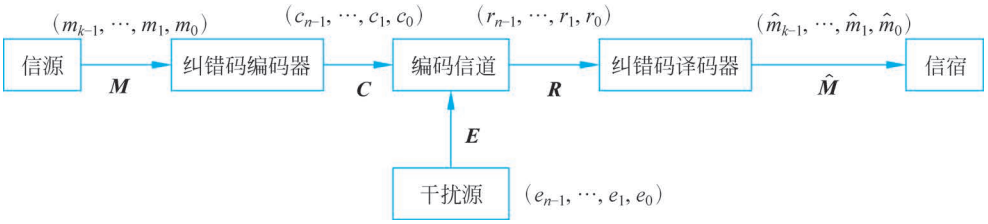


图 5.1.4　数字通信系统简化模型

经编码后产生的(n,k)码送信道传输,由于信道干扰的影响将不可避免地发生错误,这种错误有以下两种趋势。

(1) 许用码字变成禁用码组。这种错误一旦出现,由于接收到的码组不在编码器输出的码字集合中,译码时可以发现,所以这种错误模型是可检出的。

(2) 许用码字变成许用码字。即发端发生某一码字 $\boldsymbol{C}_i$ 经传输后错成码集中的另一码字 $\boldsymbol{C}_j$,这时收端无法确认是否出错,因此这是一种不可检出的错误模型。

可见,一个 n 重二进制码字 $\boldsymbol{C}$ 在传输中由于信道干扰的影响,到接收端可能变成 2^n 种 n 重矢量中的任何一个,译码的任务就是将所有可能的接收矢量进行分类,同属一类的接收矢量译为同一个许用码字,将 2^n 种不同的接收矢量还原为 2^k 个许用码字或信息序列。为了能在接收端确认发送的是何种码字,就需要建立一定的判决规则以获得最佳译码。信道编译码过程如图 5.1.5 所示。

一般来说,译码器要完成比编码器更为复杂的运算,译码器性能的好坏、速度的快慢往往决定了整个差错控制系统的性能和成本。译码正确与否的概率主要取决于所使用的码、信道特征及译码算法。对特定码类如何寻找译码错误概率小、译码速度快、设备简单的译码算法,是纠错编码理论中一个重要而实际的课题。下面讨论当码类和信道给定时,应采用什么样的算法使译码错误概率最小。


视频

由图 5.1.4 和图 5.1.5 可知,信道输出的 $\boldsymbol{R}$ 是一个二(或 q)进制序列,而译码器的输出是一个信息序列 $\boldsymbol{M}$ 的估值序列 $\hat{\boldsymbol{M}}$。

译码器的基本任务就是根据接收序列 $\boldsymbol{R}$ 和信道特征,按照一套译码规则,由接收序列 $\boldsymbol{R}$ 给出与发送的信息序列 $\boldsymbol{M}$ 最接近的估值序列 $\hat{\boldsymbol{M}}$。由于 $\boldsymbol{M}$ 与码字 $\boldsymbol{C}$ 之间存在一一

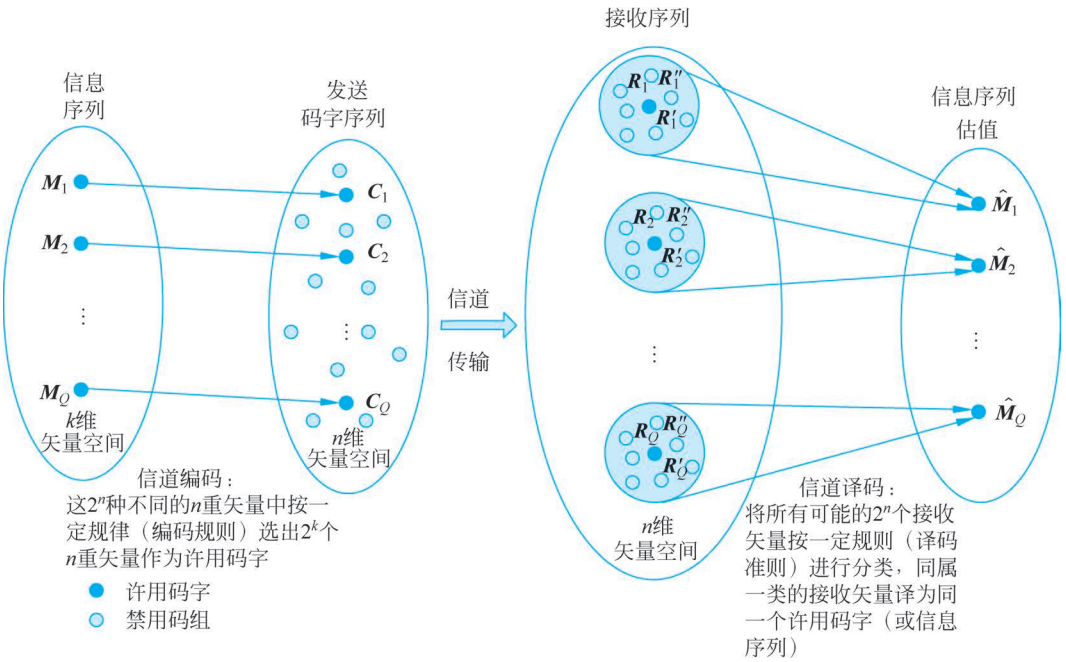


图 5.1.5　信道编译码过程

对应关系，所以这等价于译码器根据 $\boldsymbol{R}$ 产生一个 $\boldsymbol{C}$ 的估值序列 $\hat{\boldsymbol{C}}$。显然，当且仅当 $\hat{\boldsymbol{C}}=\boldsymbol{C}$ 时，$\hat{\boldsymbol{M}}=\boldsymbol{M}$，这时译码器正确译码。如果译码器输出的 $\hat{\boldsymbol{C}}\neq\boldsymbol{C}$，那么译码器产生了错误译码。产生错误译码的原因有两个：一是信道干扰很严重，超过了码本身的纠错能力；二是译码设备的故障（这点本书不予讨论）。

当给定接收序列 $\boldsymbol{R}$ 时，译码器的条件译码错误概率定义为

$$P(\boldsymbol{E}\mid\boldsymbol{R})=P(\hat{\boldsymbol{C}}\neq\boldsymbol{C}\mid\boldsymbol{R})$$

所以译码器的错误译码概率为

$$P_{\mathrm{E}}=\sum_{\boldsymbol{R}}P(\boldsymbol{E}\mid\boldsymbol{R})P(\boldsymbol{R})$$

$P(R)$ 是接收序列 $\boldsymbol{R}$ 的概率，与译码方法无关，所以译码错误概率最小的最佳译码规则是使

$$\min P_{\mathrm{E}}=\min_{\boldsymbol{R}}P(\boldsymbol{E}\mid\boldsymbol{R})=\min_{\boldsymbol{R}}P(\hat{\boldsymbol{C}}\neq\boldsymbol{C}\mid\boldsymbol{R})$$

$$=\min\left[1-\sum P(\boldsymbol{R})P(\hat{\boldsymbol{C}}=\boldsymbol{C}\mid\boldsymbol{R})\right]$$

进而得译码错误概率最小的最佳译码规则等价于使 $P(\hat{\boldsymbol{C}}=\boldsymbol{C}|\boldsymbol{R})$ 最大化。

因此，若译码器对输入的 $\boldsymbol{R}$ 能在 2^k 个码字中选择一个使 $P(\hat{\boldsymbol{C}}_i=\boldsymbol{C}|\boldsymbol{R})(i=1,2,\cdots,2^k)$ 最大的码字 $\boldsymbol{C}_i$ 作为 $\boldsymbol{C}$ 的估值序列 $\hat{\boldsymbol{C}}$，则这种译码规则一定使译码器输出错误概率最小，这种译码规则称为**最大后验概率译码**。

进一步，由贝叶斯公式

$$P(\boldsymbol{C}_i \mid \boldsymbol{R}) = \frac{P(\boldsymbol{C}_i)P(\boldsymbol{R} \mid \boldsymbol{C}_i)}{P(\boldsymbol{R})}$$

可知，若发端发送每个码字的概率 $P(\boldsymbol{C}_i)$ 均相同，且由于 $P(R)$ 与译码方法无关，则有

$$\max_{i=1,2,\cdots,2^k} P(\boldsymbol{C}_i \mid \boldsymbol{R}) \Rightarrow \max_{i=1,2,\cdots,2^k} P(\boldsymbol{R} \mid \boldsymbol{C}_i) \tag{5.1.2}$$

对离散无记忆信道(DMC)而言，有

$$P(\boldsymbol{R} \mid \boldsymbol{C}_i) = \prod_{j=1}^{n} P(r_j \mid c_{ij}) \tag{5.1.3}$$

式中

$$\boldsymbol{C}_i = (c_{i_1}, c_{i_2}, \cdots, c_{i_n}) \quad (i = 1, 2, \cdots, 2^k)$$

一个译码器的译码规则若能在 2^k 个码字 $\boldsymbol{C}$ 中选择某一个 $\boldsymbol{C}_i$ 并使式(5.1.2)最大，则这种译码规则称为**最大似然译码**(MLD)，$P(\boldsymbol{R}|\boldsymbol{C})$ 称为似然函数。由于 $\log_b x$ 与 x 是单调关系，因此式(5.1.2)与式(5.1.3)可写成

$$\max_{i=1,2,\cdots,2^k} \log_b P(\boldsymbol{R} \mid \boldsymbol{C}_i) = \max_{i=1,2,\cdots,2^k} \sum_{j=1}^{n} \log_b P(r_j \mid c_{ij}) \tag{5.1.4}$$

式中：$\log_b P(\boldsymbol{R}|\boldsymbol{C})$ 为对数似然函数或似然函数。

对于 DMC，当发端发送每一码字的概率 $P(\boldsymbol{C}_i)(i = 1, 2, \cdots, 2^k)$ 均相等时，MLD 是使译码错误概率最小的一种译码方法；否则，MLD 不是最佳的。因而，MLD 算法是一种准最佳的译码算法。在以后的讨论中，都认为 $P(\boldsymbol{C}_i)$ 均近似相等。

例 5.1.1 一个码由 00000，00111，11100 与 11011 四个码字组成。每个码字可用来表示 4 种可能的信息之一。可以计算出该码的最小距离 $d_0 = 3$，由定理 5.1 3 可知，它可纠正在任何位上出现的单个误码。同时注意到，码长为 5 的二进制码组共有 $2^5 = 32$ 种可能的序列，除了上述 4 个许用码字外，其余 28 个为禁用码组。为了对该码进行纠错处理，需将 28 种禁用码组的每一个与 4 种许用码字做“最邻近性”的比较。这种处理意味着要建立一个“译码表”，所以译码的本质就是对码组进行分类，即先将所有与每个许用码字有一位差错的各个可能接收序列列在该码字的下面，这样就得到表 5.1.1 中以虚线围起的部分。除了这一部分之外，应注意到尚有 8 个序列未被列入。这 8 个序列与每个码字至少差两位。但是，它们与上述序列不同，没有唯一的方法可把它们安排到表内。例如，既可将序列 10001 放在第 4 列，也可将它放在第 1 列。在译码过程中使用此表时，可将所接收序列与表内各列对照，当查到该序列时，将该列第一行的码字作为译码器的输出。

表 5.1.1 4 个码字的译码表

00000	11100	00111	11011
10000	01100	10111	01011
01000	10100	01111	10011
00100	11000	00011	11111
00010	11110	00101	11001

续表

00001	11101	00110	11010
10001	01101	10110	01010
10010	01110	10101	01001

用这种方式建立的表具有很大的优点。设信道误码率为 p_e，出现任何一种具有 i 个差错特定模式的概率是 $p_e^i(1-p_e)^{5-i}$。当 $p_e<1/2$，即信道的信噪比足够大时，可以看到：

$$(1-p_e)^5 > p_e(1-p_e)^4 > p_e^2-(1-p_e)^3 > \cdots$$

即不出错概率大于出错概率；一个特定的单个差错模式要比一个特定的两个(或多个)差错模式更容易出现。因此，译码器将所收到的一个特定码组译为在汉明距离上最邻近的一个码字时，实际上是选择了最可能发送的那个码字(设各个码字的发送机会相同)。这就是 MLD 的具体应用，实际上它就是根据接收序列 $\boldsymbol{R}$ 在 2^k 个码字集中寻找与 $\boldsymbol{R}$ 的汉明距离最小的码字 $\boldsymbol{C}_i$ 作为译码输出，因为它最可能是发送的码字。这种译码方法又称为**最小汉明距离译码**。执行这种译码规则的译码器称为**最大似然译码器**。在上述条件下，其序列差错概率最小。当用译码表进行译码时，为了实现最大似然译码，可用上述方法列表对码字进行分类。遗憾的是，译码表的大小随码组长度按指数关系增加，故对长码来说直接使用译码表是不切合实际的。但对说明分组码的某些重要性质而言，译码表仍是一种很有用的概念性工具。

总结而言，信道译码准则有以下几种。

最小错误概率译码准则：$F(\boldsymbol{R}_i)=\boldsymbol{C}^*=\arg\min P_E$

最大联合概率译码准则：$F(\boldsymbol{R}_i)=\boldsymbol{C}^*=\arg\max[P(\boldsymbol{C}^*\boldsymbol{R}_i)]$

最大后验概率译码准则：$F(\boldsymbol{R}_i)=\boldsymbol{C}^*=\arg\max[P(\boldsymbol{C}^*|\boldsymbol{R}_i)]$

最大似然概率译码准则：$F(\boldsymbol{R}_i)=\boldsymbol{C}^*=\arg\max[P(\boldsymbol{R}_i|\boldsymbol{C}^*)]$

最小距离译码准则：$F(\boldsymbol{R}_i)=\boldsymbol{C}^*=\arg\min[d(\boldsymbol{C}^*;\boldsymbol{R}_i)]$

可以证明，最大联合概率译码和最大后验概率译码等价，都属于最小错误概率译码，可以得到最小平均译码概率 P_E，是性能最佳的译码准则。但这几类译码准则依赖发送码字的先验概率 $P(\boldsymbol{C}_i)$，计算复杂度大。最大似然译码或最小距离译码仅依赖信道转移概率或码距，译码相对简单，但最大似然译码只有在输入等概时才能得到最小平均译码错误概率，最小距离译码在传输错误概率较小时，等价于最大似然译码。

中文教学录像

5.2 有噪信道编码定理

5.2.1 联合渐近等同分割性与联合典型序列

有噪信道编码定理又称香农第二定理，其基础依据是信道传输表现出联合渐近等同分割性(AEP)。

针对离散信源，AEP 定理和 ε 典型序列说明离散信源的无记忆扩展信源随着序列 $X_1,X_2,X_3,\cdots,X_n$ 的长度 n 无限加大，每个 ε 典型序列发生的概率约为 $2^{-nH(X)}$，这些

典型序列的总数约为 $2^{nH(X)}$，这些 ε 典型序列为高概率事件。类似的结论对于离散信道也成立。以下讨论离散信道$[X,P(y|x),Y]$的 n 次无记忆扩展信道$[X^n,P(\boldsymbol{y}|\boldsymbol{x}),Y^n]$，其中，$\boldsymbol{x}=(x_1,x_2,\cdots,x_n)(x_i\in X)$和 $\boldsymbol{y}=(y_1,y_2,\cdots,y_n)(y_i\in Y)$分别表示 n 次无记忆扩展信道的输入和输出的 n 长随机序列。满足

$$P(\boldsymbol{y}\mid\boldsymbol{x})=P(y_1y_2\cdots y_n\mid x_1x_2\cdots x_n)=\prod_{i=1}^{n}P(y_i\mid x_i)$$

它们对应的信息熵和联合熵分别为 $H(X)$、$H(Y)$和 $H(XY)$。

定义 5.2.1 设$(\boldsymbol{x},\boldsymbol{y})\in(X^n;Y^n)$是 n 长随机序列对，对于任意小的正数 $\varepsilon>0$，存在足够大的 n，若满足下列条件：

$$\begin{cases}\left|\dfrac{1}{n}\log P(\boldsymbol{x})+H(X)\right|<\varepsilon\\[2ex]\left|\dfrac{1}{n}\log P(\boldsymbol{y})+H(Y)\right|<\varepsilon\\[2ex]\left|\dfrac{1}{n}\log P(\boldsymbol{xy})+H(XY)\right|<\varepsilon\end{cases}\tag{5.2.1}$$

则称$(\boldsymbol{x},\boldsymbol{y})$为**联合$\boldsymbol{\varepsilon}$ 典型序列**。

$G_\varepsilon(X)$和 $G_\varepsilon(Y)$ 分别表示 X^n 和 Y^n 中的典型序列集，$G_\varepsilon(XY)$ 表示 X^nY^n 联合空间中的联合典型序列集。联合 ε 典型序列满足如下性质。

定理 5.2.1(联合渐近等同分割性) 对于任意小的 $\varepsilon>0,\delta>0$，当 n 足够大时，则有以下性质。

性质 1 典型序列 $\boldsymbol{x},\boldsymbol{y}$ 和联合典型序列$(\boldsymbol{x},\boldsymbol{y})$满足

$$\begin{cases}2^{-n[H(X)+\varepsilon]}\leqslant P(\boldsymbol{x})\leqslant 2^{-n[H(X)-\varepsilon]}\\2^{-n[H(Y)+\varepsilon]}\leqslant P(\boldsymbol{y})\leqslant 2^{-n[H(Y)-\varepsilon]}\\2^{-n[H(XY)+\varepsilon]}\leqslant P(\boldsymbol{xy})\leqslant 2^{-n[H(XY)-\varepsilon]}\end{cases}\tag{5.2.2}$$

性质 2 典型序列集 $G_\varepsilon(X)$，$G_\varepsilon(Y)$和联合典型序列集 $G_\varepsilon(XY)$满足

$$\begin{cases}P(G_\varepsilon(X))\geqslant 1-\delta\\P(G_\varepsilon(Y))\geqslant 1-\delta\\P(G_\varepsilon(XY))\geqslant 1-\delta\end{cases}\tag{5.2.3}$$

性质 3 以 N_G 表示典型序列集中元素个数，则 $G_\varepsilon(X)$，$G_\varepsilon(Y)$和 $G_\varepsilon(XY)$中序列个数满足

$$\begin{cases}(1-\delta)2^{n[H(X)-\varepsilon]}\leqslant N_{G_x}\leqslant 2^{n[H(X)+\varepsilon]}\\(1-\delta)2^{n[H(Y)-\varepsilon]}\leqslant N_{G_y}\leqslant 2^{n[H(Y)+\varepsilon]}\\(1-\delta)2^{n[H(XY)-\varepsilon]}\leqslant N_{G_{xy}}\leqslant 2^{n[H(XY)+\varepsilon]}\end{cases}\tag{5.2.4}$$

证明： 性质 1 是大数定律的直接结果，因$\dfrac{1}{n}\log\dfrac{1}{P(\boldsymbol{x})}$、$\dfrac{1}{n}\log\dfrac{1}{P(\boldsymbol{y})}$、$\dfrac{1}{n}\log\dfrac{1}{P(\boldsymbol{xy})}$分别依概率趋于其统计平均 $E\left[\log\dfrac{1}{P(x)}\right]=H(X)$、$E\left[\log\dfrac{1}{P(y)}\right]=H(Y)$以及

$E\left[\log\frac{1}{P(xy)}\right]=H(XY)$。再根据联合 ε 典型序列定义式(5.2.1)整理即得性质 1。

性质 2 的证明采用极限理论中 ε-δ 描述方法，对于任意小的正数 $\varepsilon>0,\delta>0$，存在正整数 n_0，使得对于所有 $n\geqslant n_0$，有

$$\begin{cases}P\left\{\left|\frac{1}{n}\log\frac{1}{P(\boldsymbol{x})}-H(X)\right|\leqslant\varepsilon\right\}\geqslant 1-\delta\\ P\left\{\left|\frac{1}{n}\log\frac{1}{P(\boldsymbol{y})}-H(Y)\right|\leqslant\varepsilon\right\}\geqslant 1-\delta\\ P\left\{\left|\frac{1}{n}\log\frac{1}{P(\boldsymbol{xy})}-H(XY)\right|\leqslant\varepsilon\right\}\geqslant 1-\delta\end{cases}$$

即得性质 2。

最后证明性质 3，结合概率满足完备性以及联合典型序列满足式(5.2.2)，可得

$$\begin{cases}1\geqslant\sum\limits_{\boldsymbol{x}\in G_\varepsilon(X)}P(\boldsymbol{x})>N_{G_\varepsilon(X)}2^{-n[H(X)+\delta]}\\ 1\geqslant\sum\limits_{\boldsymbol{y}\in G_\varepsilon(Y)}P(\boldsymbol{y})>N_{G_\varepsilon(Y)}2^{-n[H(Y)+\delta]}\\ 1\geqslant\sum\limits_{\boldsymbol{xy}\in G_\varepsilon(XY)}P(\boldsymbol{xy})>N_{G_\varepsilon(XY)}2^{-n[H(XY)+\delta]}\end{cases}$$

$$\begin{cases}N_{G_\varepsilon(X)}2^{-n[H(X)-\delta]}>\sum\limits_{\boldsymbol{x}\in G_\varepsilon(X)}P(\boldsymbol{x})>1-\varepsilon\\ N_{G_\varepsilon(Y)}2^{-n[H(Y)-\delta]}>\sum\limits_{\boldsymbol{y}\in G_\varepsilon(Y)}P(\boldsymbol{y})>1-\varepsilon\\ N_{G_\varepsilon(XY)}2^{-n[H(XY)-\delta]}>\sum\limits_{\boldsymbol{xy}\in G_\varepsilon(XY)}P(\boldsymbol{xy})>1-\varepsilon\end{cases}$$

两组公式整理即得性质 3。 [证毕]

定理 5.2.1 说明，在两个随机变量情况下，信源 X^n、Y^n 和联合集$(X^n\ Y^n)$也具有渐近等同分割性。联合 ε 典型序列对$(\boldsymbol{x},\boldsymbol{y})$是 n 次无记忆扩展联合空间中经常出现的高概率序列对。这些高概率序列对的出现概率几乎相等，即

$$P(\boldsymbol{x})\approx 2^{-nH(X)},\quad P(\boldsymbol{y})\approx 2^{-nH(Y)},\quad P(\boldsymbol{xy})\approx 2^{-nH(XY)}$$

并且与典型序列一样，联合典型序列对$(\boldsymbol{x},\boldsymbol{y})$的概率和也是趋于 1 的。即随着 n 的增大，典型序列和联合典型序列属于高概率事件，各自占据了 X^n、Y^n、(X^n,Y^n)的几乎全部概率组成。

那么联合 ε 典型序列对$(\boldsymbol{x},\boldsymbol{y})$是如何构成的？$X^n$ 是扩展信道的输入概率空间，Y^n 是扩展信道的输出概率空间。取 X^n 中典型序列 $\boldsymbol{x}$ 与 Y^n 中典型序列 $\boldsymbol{y}$ 配对，同时满足式(5.2.2)，则构成联合 ε 典型序列对$(\boldsymbol{x},\boldsymbol{y})$。典型序列 $\boldsymbol{x}$ 是输入端高概率出现的序列，典型序列 $\boldsymbol{y}$ 是输出端高概率出现的序列，而联合典型序列对$(\boldsymbol{x},\boldsymbol{y})$就是那些信道输入和输出之间密切关联、经常出现的序列对。

基于定理 5.2.1 中性质 3 可知，随着 n 的增大，典型序列和联合典型序列各自的数目大约为

$$N_{G_\varepsilon(X)} \approx 2^{nH(X)}, \quad N_{G_\varepsilon(Y)} \approx 2^{nH(Y)}, \quad N_{G_\varepsilon(XY)} \approx 2^{nH(XY)}$$

例 5.2.1 设信源为 $\begin{bmatrix} X \\ P(x) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0.9 & 0.1 \end{bmatrix}$，传输信道为 BSC(0.2)的 n 次扩展信道，其输入和输出构成联合典型序列对$(\boldsymbol{x},\boldsymbol{y})$。存在一个长度 $n=100$ 的联合典型序列对：

$\boldsymbol{x}$=1111111111000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000

$\boldsymbol{y}$=0011111111000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000011111111111111111111

可见，$\boldsymbol{x}$ 是一个 X^n 集合的典型序列(满足 $I(\boldsymbol{x})/100=H(X)$)，序列 $\boldsymbol{y}$ 中有 26 个“1”，74 个“0”，也是 Y^n 中的典型序列，而 $\boldsymbol{x}$ 与 $\boldsymbol{y}$ 有 20bit 不同，即序列对$(\boldsymbol{x},\boldsymbol{y})$正是该信道的一个典型序列对(符合信道典型翻转的概率)。

进一步，可以证明联合 ε 典型序列还具有如下性质。

定理 5.2.2 对于任意小的正数 $\varepsilon>0$，当 n 足够大时，有

性质 1

$$2^{-n[H(Y|X)+2\varepsilon]} \leqslant P(\boldsymbol{y} \mid \boldsymbol{x}) \leqslant 2^{-n[H(Y|X)-2\varepsilon]}$$

$$2^{-n[H(X|Y)+2\varepsilon]} \leqslant P(\boldsymbol{x} \mid \boldsymbol{y}) \leqslant 2^{-n[H(X|Y)-2\varepsilon]}$$

式中：$(\boldsymbol{x};\boldsymbol{y})\in G_\varepsilon(XY)$。

性质 2

令 $G_\varepsilon(X|\boldsymbol{y})=\{\boldsymbol{x}:(\boldsymbol{x},\boldsymbol{y})\in G_\varepsilon(XY)\}$，$G_\varepsilon(Y|\boldsymbol{x})=\{\boldsymbol{y}:(\boldsymbol{x},\boldsymbol{y})\in G_\varepsilon(XY)\}$，即 $G_\varepsilon(X|\boldsymbol{y})$ 表示在给定 ε 典型序列 $\boldsymbol{y}$ 条件下与 $\boldsymbol{y}$ 构成联合 ε 典型序列对的所有序列 $\boldsymbol{x}$ 的集合，$G_\varepsilon(Y|\boldsymbol{x})$表示在给定 ε 典型序列 $\boldsymbol{x}$ 条件下与 $\boldsymbol{x}$ 构成联合 ε 典型序列对的所有典型序列 $\boldsymbol{y}$ 的集合。则有

$$N_{G_\varepsilon(X|\boldsymbol{y})} \leqslant 2^{n[H(X|Y)+2\varepsilon]}$$

$$N_{G_\varepsilon(Y|\boldsymbol{x})} \leqslant 2^{n[H(Y|X)+2\varepsilon]}$$

证明：(1) 令$(\boldsymbol{x};\boldsymbol{y})\in G_\varepsilon(XY)$，对于任意 $\boldsymbol{x},\boldsymbol{y}$ 满足

$$P(\boldsymbol{y} \mid \boldsymbol{x}) = \frac{P(\boldsymbol{xy})}{P(\boldsymbol{x})}$$

根据式(5.2.2)可得

$$2^{-n[H(XY)+\varepsilon]} \times 2^{n[H(X)-\varepsilon]} \leqslant P(\boldsymbol{y} \mid \boldsymbol{x}) \leqslant 2^{-n[H(XY)-\varepsilon]} \times 2^{n[H(X)+\varepsilon]}$$

$$2^{-n[H(XY)-H(X)+2\varepsilon]} \leqslant P(\boldsymbol{y} \mid \boldsymbol{x}) \leqslant 2^{-n[H(XY)-H(X)-2\varepsilon]}$$

所以

$$2^{-n[H(Y|X)+2\varepsilon]} \leqslant P(\boldsymbol{y} \mid \boldsymbol{x}) \leqslant 2^{-n[H(Y|X)-2\varepsilon]}$$

同理，可得

$$2^{-n[H(X|Y)+2\varepsilon]} \leqslant P(\boldsymbol{x} \mid \boldsymbol{y}) \leqslant 2^{-n[H(X|Y)-2\varepsilon]}$$

(2) 假设序列 $\boldsymbol{y}\in G_\varepsilon(Y)$，因为

$$1 = \sum_{X^n} P(\boldsymbol{x} \mid \boldsymbol{y}) = \sum_{X^n} \frac{P(\boldsymbol{xy})}{P(\boldsymbol{y})} \geqslant \sum_{x\in G_\varepsilon(X|\boldsymbol{y})} \frac{P(\boldsymbol{xy})}{P(\boldsymbol{y})}$$

又因为 $\boldsymbol{y}$ 是典型序列，$\boldsymbol{x}$ 与 $\boldsymbol{y}$ 构成联合典型序列对，即$(\boldsymbol{x},\boldsymbol{y})\in G_\varepsilon(XY)$，根据式(5.2.2)可得

$$\begin{cases} P(\boldsymbol{xy}) > 2^{-n[H(XY)+\varepsilon]} \\ P(\boldsymbol{y}) < 2^{-n[H(Y)-\varepsilon]} \end{cases}$$

进而可得

$$1 \geqslant \sum_{x\in G_\varepsilon(X|\boldsymbol{y})} 2^{-n[H(XY)-H(Y)+2\varepsilon]} = N_{G_\varepsilon(X|\boldsymbol{y})}\, 2^{-n[H(X|Y)+2\varepsilon]}$$

移项后可得

$$N_{G_\varepsilon(X|\boldsymbol{y})} \leqslant 2^{n[H(X|Y)+2\varepsilon]}$$

同理，假设序列 $\boldsymbol{x}\in G_\varepsilon(X)$，因为

$$1 = \sum_{Y^n} P(\boldsymbol{y}\mid\boldsymbol{x}) = \sum_{Y^n} \frac{P(\boldsymbol{xy})}{P(\boldsymbol{x})} \geqslant \sum_{y\in G_\varepsilon(Y|\boldsymbol{x})} \frac{P(\boldsymbol{xy})}{P(\boldsymbol{x})}$$

又因为$(\boldsymbol{x},\boldsymbol{y})\in G_\varepsilon(XY)$，根据式(5.2.2)可得

$$\begin{cases} P(\boldsymbol{xy}) > 2^{-n[H(XY)+\varepsilon]} \\ P(\boldsymbol{x}) < 2^{-n[H(X)-\varepsilon]} \end{cases}$$

进而可得

$$1 \geqslant \sum_{\boldsymbol{y}\in G_\varepsilon(Y|\boldsymbol{x})} 2^{-n[H(XY)-H(X)+2\varepsilon]} = N_{G_\varepsilon(Y|\boldsymbol{x})}\, 2^{-n[H(Y|X)+2\varepsilon]}$$

移项后可得

$$N_{G_\varepsilon(Y|\boldsymbol{x})} \leqslant 2^{n[H(Y|X)+2\varepsilon]}$$

[证毕]

定理 5.2.3 若$(\boldsymbol{x}',\boldsymbol{y}')$是统计独立的随机联合典型序列对，并与 $P(\boldsymbol{xy})$有相同的边缘分布，即$(\boldsymbol{x}',\boldsymbol{y}')\sim P(\boldsymbol{x}')P(\boldsymbol{y}')=P(\boldsymbol{xy})$，且对于任意正数 $\delta>0$，当 n 足够大时，有

$$(1-\delta)2^{-N[I(X;Y)+3\delta]} \leqslant P\{(\boldsymbol{x}',\boldsymbol{y}')\in G_\varepsilon(XY)\} \leqslant 2^{-N[I(X;Y)-3\delta]}$$

证明： 因为$(\boldsymbol{x}';\boldsymbol{y}')$是统计独立且与 $P(\boldsymbol{xy})$有相同的边缘分布，即

$$P(\boldsymbol{x}'\boldsymbol{y}') = P(\boldsymbol{x})P(\boldsymbol{y}) = P(\boldsymbol{xy})$$

因此可得

$$\begin{aligned} P\{(\boldsymbol{x}';\boldsymbol{y}')\in G_\varepsilon(XY)\} &= \sum_{\boldsymbol{xy}\in G_\varepsilon(XY)} P(\boldsymbol{x})P(\boldsymbol{y}) \\ &\leqslant 2^{n[H(XY)+\varepsilon]} \times 2^{-n[H(X)-\varepsilon]} \times 2^{-n[H(Y)-\varepsilon]} \\ &= 2^{-n[H(X)+H(Y)-H(XY)-3\varepsilon]} = 2^{-n[I(X;Y)-3\varepsilon]} \end{aligned}$$

同理，当 n 足够大时，有

$$\begin{aligned} P\{(\boldsymbol{x}',\boldsymbol{y}')\in G_\varepsilon(XY)\} &= \sum_{\boldsymbol{xy}\in G_\varepsilon(XY)} P(\boldsymbol{x})P(\boldsymbol{y}) \\ &\geqslant (1-\delta)2^{n[H(XY)-\varepsilon]} \times 2^{-n[H(X)+\varepsilon]} \times 2^{-n[H(Y)+\varepsilon]} \\ &= (1-\delta)2^{-n[H(X)+H(Y)-H(XY)+3\varepsilon]} = (1-\delta)2^{-n[I(X;Y)+3\varepsilon]} \end{aligned}$$

[证毕]

可以用图形进一步描述上述联合 ε 典型序列的一系列结论。将联合向量空间 X^nY^n 中所有序列排成如图 5.2.1 所示的阵列。在联合向量空间中的全部序列对依据是否是典型序列 $\boldsymbol{x}$ 和典型序列 $\boldsymbol{y}$，可以划分为 4 部分。以 X^n 空间中的序列 $\boldsymbol{x}$ 为行，以 Y^n 空间中的序列 $\boldsymbol{y}$ 为列，分别将属于 ε 典型序列的排列在前面，即前面近似 $2^{nH(X)}$ 行为典型序列 $\boldsymbol{x}$，前面约 $2^{nH(Y)}$ 列为典型序列 $\boldsymbol{y}$。那么，只有位于阵列左上角部分的序列对 $(\boldsymbol{x},\boldsymbol{y})$ 属于联合 ε 典型序列的序列对，每个黑点对应一个联合典型序列对，共有 $2^{nH(XY)}$ 个黑点。根据定理 5.2.2 中(2)的结论，对于 X^n 中的每个 ε 典型序列 $\boldsymbol{x}_i$，约有 $2^{nH(Y|X)}$ 个 Y^n 中的 ε 典型序列 $\boldsymbol{y}_j$ 与之配对，构成联合典型序列对，即每行有 $2^{nH(Y|X)}$ 个黑点，构成集合 $G_\varepsilon(Y|\boldsymbol{x}_i)$；同理，对于 Y^n 中的每个 ε 典型序列 $\boldsymbol{y}_j$，约有 $2^{nH(X|Y)}$ 个 X^n 中的 ε 典型序列 $\boldsymbol{x}_i$ 与之配对，构成联合典型序列对，即每列有 $2^{nH(X|Y)}$ 个黑点，构成集合 $G_\varepsilon(X|\boldsymbol{y}_j)$。

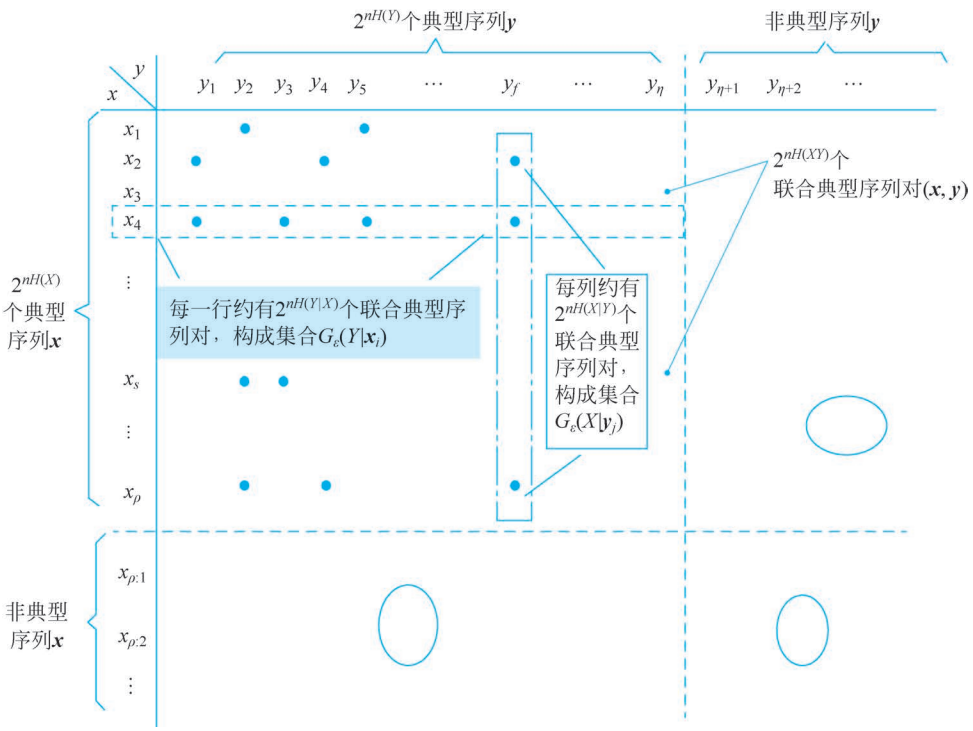


图 5.2.1　联合典型序列示意

又根据定理 5.2.3，随机选择序列对是统计独立的联合典型序列对的概率约为 $2^{-nI(X;Y)}$。也就是说，在 $2^{nH(XY)}$ 个联合典型序列对中，$\boldsymbol{x}$ 与 $\boldsymbol{y}$ 相互统计独立的序列对 $(\boldsymbol{x},\boldsymbol{y})$ 只占一部分，约为 $2^{nI(X;Y)}$。因此，对于某一典型序列 $\boldsymbol{y}_j$，可以认为与它统计独立的联合典型序列对可能有 $2^{nI(X;Y)}$ 个，这也提示我们在 $2^{nH(X)}$ 个 X 典型序列中有 $2^{nI(X;Y)}$ 个是可区分识别的典型序列 $\boldsymbol{x}_i$。

由前面讨论可知，在 n 次无记忆扩展信道的输入和输出空间之间，联合 ε 典型序列对 $(\boldsymbol{x},\boldsymbol{y})$ 是一些密切关联的序列对。也就是说，DMC 经过足够多次扩展后会呈现出传输极化的特性，即当某一输入典型序列 $\boldsymbol{x}_i$ 发送，转移到收端时，对应的接收序列高概率是

能与 $\boldsymbol{x}_i$ 构成联合 ε 典型序列对的那些典型序列 $\boldsymbol{y}_j$。

如果把全部 $2^{nH(X)}$ 个输入 ε 典型序列 $\boldsymbol{x}_i$ 都用于传送信息，每个 $\boldsymbol{x}_i$ 将会高概率地转移为能与之构成联合典型序列对的那 $2^{nH(Y|X)}$ 个典型序列 $\boldsymbol{y}_j$，势必会造成不同的 $\boldsymbol{x}_i$ 对应(转移为)相同的 $\boldsymbol{y}_j$，即图 5.2.1 左上角中同一列的黑点重复，这将会造成译码错误。为了避免这种情况，就要求 Y^n 中与每个发送序列配对的典型序列集合不相交，即从 X^n 中选择若干(不是全部)典型序列 $\boldsymbol{x}_i$，要求其在 Y^n 中的联合典型序列集 $G_\varepsilon(Y|\boldsymbol{x}_i)$ 不相交，如图 5.2.2 所示。根据定理 5.2.2 和定理 5.2.3，为使收端译码不出错，从编码角度而言，发送端需要从 $2^{nH(X)}$ 个 X 典型序列中选择一部分作为许用码字。选取原则是要求每个选出的许用码字 $\boldsymbol{x}_i$ 所配对的 $2^{nH(Y|X)}$ 个 Y 典型序列和其他许用码字所对应的 Y 典型序列没有交集。那么，为保证译码无错，发送端编码最多能够选出的码字数目 Q 个为多少？对于 X 典型序列 $\boldsymbol{x}_i$，传输后高概率转移为 $G_\varepsilon(Y|\boldsymbol{x}_i)$ 中的一个典型序列 $\boldsymbol{y}_j$，由于每个 $G_\varepsilon(Y|\boldsymbol{x}_i)$ 中的 Y 典型序列数目约为 $2^{nH(Y|X)}$，Y^n 中的典型序列集 $G_\varepsilon(Y)$ 的总数约为 $2^{nH(Y)}$，所以要求

$$2^{nH(Y|X)}Q \leqslant 2^{nH(Y)}$$

即在不发生译码错误的要求下能够用于传输信息的输入典型序列 $\boldsymbol{x}_i$ 的个数最多为

$$Q \leqslant \frac{2^{nH(Y)}}{2^{nH(Y|X)}} = 2^{nI(X;Y)}$$

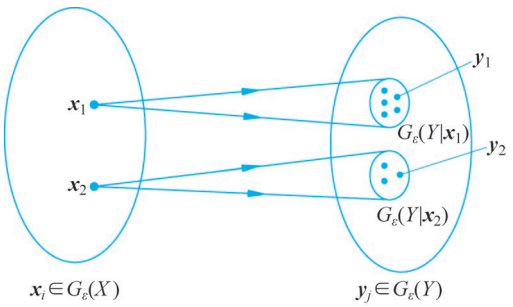


图 5.2.2　n 次扩展信道 x 与 y 的关系

信道编码需要从全部 $2^{nH(X)}$ 个 n 重矢量(输入典型序列)中选取 $Q(\leqslant 2^{nI(X;Y)})$ 个作为许用码字，用来传送消息，并保证这些输入典型序列所对应的输出典型序列彼此无重叠。定理 5.2.3 从理论上保证了在 $2^{nH(X)}$ 个 X 典型序列中可区分识别出 $2^{nI(X;Y)}$ 个典型序列，与它们配成联合典型序列对的 Y 典型序列集不重叠，相当于可以把输出典型序列集合划分 $2^{nI(X;Y)}$ 个不相交的子集，从而可保证译码无错，如图 5.2.2 所示。


视频

5.2.2　香农第二定理(有噪信道编码定理)

有噪信道编码定理又称为香农第二定理，是信息论的基本定理之一。

综合前面的分析，可以将信道编码理解为在编码时随机地选择输入端 ε 典型序列作为码字 $\boldsymbol{C}_i$，因为它们是在输入端 X^n 集中高概率出现的序列。n 次无记忆扩展信道传输

中具有联合渐近等同分割性，所发送的码字 $\boldsymbol{C}_i$ 与接收序列 $\boldsymbol{y}_j$ 构成联合典型序列对的概率很高，它们之间是高概率密切相关的。因此译码可采取联合典型序列译码准则，即在译码时接收端 Y^n 集中将接收序列 $\boldsymbol{y}_j$ 译成与它构成联合典型序列对的那个码字。

下面给出香农第二定理的完整描述和证明。

定理 5.2.4（有噪信道编码定理） 设离散无记忆信道$[X,P(y|x),Y]$的信道容量为 C。当信息传输率 $R<C$ 时，只要码长 n 足够长，总可以在输入 X 符号集中找到 $Q(=2^{nR})$ 个码字组成的一组码和相应的译码规则，使译码的平均错误概率任意小（$P_E\to 0$）。当 $R>C$ 时，无论码长 n 多长，都找不到一种满足 $Q=2^{nR}$ 的编码方案使得 $P_E\to 0$。

证明： 设输入端有 $Q=2^{nR}$ 个消息（信息序列），用$\{1,2,\cdots,Q\}=[1,Q]$ 表示消息的标号集。信道编码就是将 Q 个消息映射成 X^n 中不同的序列，即编码函数 $f:\{1,2,\cdots,Q=2^{nR}\}\to X^n$，选出许用码字 $\boldsymbol{C}_i=(x_{i_1}x_{i_2}\cdots x_{i_N})$，组成有 Q 个码字的码集（或称为**码书**）$\boldsymbol{C}=\{\boldsymbol{C}_0\boldsymbol{C}_1\cdots\boldsymbol{C}_{Q-1}\}$。现在采用随机编码：在 X^n 中以信道输入概率分布 $P(\boldsymbol{x})$ 随机地选取 $\boldsymbol{C}_i=(x_{i_1}x_{i_2}\cdots x_{i_N})$ 作为码字，设 x_i 统计独立，且满足

$$P(x_{i_k})=P(x)\quad(k=1,2,\cdots,n)$$

$$x_{i_k}\in X=\{x_1,x_2,\cdots,x_r\}$$

则码字 $\boldsymbol{C}_i$ 出现的概率为

$$P(\boldsymbol{C}_i)=\prod_{k=1}^{n}P(x_{i_k})$$

随机编码得到的某个码书 $\boldsymbol{C}$ 的出现概率为

$$P(\boldsymbol{C})=\prod_{i=0}^{Q-1}P(\boldsymbol{C}_i)=\prod_{i=0}^{Q-1}\prod_{k=1}^{n}P(x_{i_k})$$

码书 $\boldsymbol{C}$ 中每个码字都是以输入概率分布 $P(x)$，按同一规则随机选取的，因此 r 元随机编码可以得到的所有码书数目为

$$S=r^{nQ}=r^{n2^{nR}}$$

在接收端，接收到序列 $\boldsymbol{y}_j$ 后要译成发送的码字，相应的译码函数 $g,Y^n\to\{1,2,\cdots,Q=2^{nR}\}$ 即将 Y^n 集映射到 Q 个消息集中。

设从码书 $\boldsymbol{C}$ 中任选一个码字 $\boldsymbol{w}\in\boldsymbol{C}$ 被发送，相应接收到的序列为 $\boldsymbol{y}_j$，根据译码函数

$$g(\boldsymbol{y}_j)=\boldsymbol{w}\in\boldsymbol{C},\text{为正确译码}$$

$$g(\boldsymbol{y}_j)=\boldsymbol{w}'\in\boldsymbol{C},\boldsymbol{w}'\neq\boldsymbol{w},\text{为错误译码}$$

则发送消息 $\boldsymbol{w}$，在接收端译码产生的错误概率为

$$P_{ew}=P\{g(\boldsymbol{y}_j)\neq\boldsymbol{w}\mid\boldsymbol{C}_j=\boldsymbol{w}\},\quad\boldsymbol{w}\in\{\boldsymbol{C}_0,\boldsymbol{C}_1,\cdots,\boldsymbol{C}_{Q-1}\}$$

而该码书的平均错误概率

$$P_E(C)=\frac{1}{Q}\sum_{j=0}^{Q-1}P(g(\boldsymbol{y}_j)\neq\boldsymbol{w}\mid\boldsymbol{w})=\frac{1}{2^{nR}}\sum_{i=1}^{Q-1}P_{ew}\tag{5.2.5}$$

由 5.2.1 节分析可知，对于 n 次扩展信道传输具有联合渐近等同分割性，发送的码字 $\boldsymbol{C}_i$ 时，其接收序列很高概率是与 $\boldsymbol{C}_i$ 构成联合典型序列对的 Y 典型序列，所以可选择

联合典型序列译码准则作为译码规则。即对某接收序列 $\boldsymbol{y}_j$，若存在 $\boldsymbol{w}\in\boldsymbol{C}$，并且$(\boldsymbol{w},\boldsymbol{y}_j)\in G_\varepsilon(XY)$，即 $\boldsymbol{y}_j$ 与 $\boldsymbol{w}$ 构成联合典型序列对，则将 $\boldsymbol{y}_j$ 译成 $\boldsymbol{w}$，即 $g(\boldsymbol{y}_j)=\boldsymbol{w}\in\boldsymbol{C}$。不难看出联合典型序列译码准则本质就是最大似然译码准则。

因为随机编码选取出的码书 $\boldsymbol{C}$ 有很多种，因此为了计算随机编码总的平均错误概率，应基于式(5.2.4)的平均错误概率，对所有可能选取的码书进行统计平均，即

$$P_{\mathrm{E}}\triangleq P_{\mathrm{r}}(E)=\underset{\text{所有码书}}{E}[P_{\mathrm{E}}(\boldsymbol{C})]=\sum_{S}P(\boldsymbol{C})P_{\mathrm{E}}(\boldsymbol{C})=\frac{1}{2^{nR}}\sum_{i=1}^{Q-1}\sum_{S}P(\boldsymbol{C})P_{\mathrm{ew}}$$

另外，由随机编码的规则知，各码字生成方式一样，即选取的机会均等。因此，在所有码书上求平均的平均错误概率与发送码字 $\boldsymbol{w}$ 具体是哪个无关，即上式中 $\sum_{S}P(\boldsymbol{C})P_{\mathrm{ew}}$ 不依赖发送码字 $\boldsymbol{w}$。为不失一般性，可以先某一码字，如令 $\boldsymbol{w}=\boldsymbol{C}_0$ 统一考虑，即 $\sum_{S}P(\boldsymbol{C})P_{\mathrm{ew}}=P_{\mathrm{e}0}$，进而可得

$$\begin{aligned}P_{\mathrm{r}}(E)&=\frac{1}{2^{nR}}\sum_{i=1}^{Q-1}P_{\mathrm{e}0}=P_{\mathrm{e}0}\\&=P\{g(\boldsymbol{y}_j)\neq\boldsymbol{C}_0\mid\boldsymbol{x}_i=\boldsymbol{C}_0\}\end{aligned}$$

当采用联合典型序列译码准则时，产生译码错误情况包括以下两种。

(1) 接收矢量 $\boldsymbol{y}_j$ 与发送码字 $\boldsymbol{w}$ 不构成联合典型序列对，而可能是与其他许用码字 $\boldsymbol{w}'\neq\boldsymbol{w}$ 构成了联合典型序列对，即 $\boldsymbol{w}\in\boldsymbol{C},\boldsymbol{w}'\in\boldsymbol{C},\boldsymbol{w}'\neq\boldsymbol{w},(\boldsymbol{w},\boldsymbol{y}_j)\notin G_\varepsilon(XY)$而$(\boldsymbol{w}',\boldsymbol{y}_j)\in G_\varepsilon(XY)$。

(2) 接收矢量 $\boldsymbol{y}_j$ 既与发送码字 $\boldsymbol{w}$ 构成联合典型序列对，也与其他许用码字 $\boldsymbol{w}'\neq\boldsymbol{w}$ 构成联合典型序列对，即 $\boldsymbol{w}\in\boldsymbol{C},\boldsymbol{w}'\in\boldsymbol{C},\boldsymbol{w}'\neq\boldsymbol{w},(\boldsymbol{w},\boldsymbol{y}_j)\in G_\varepsilon(XY)$且$(\boldsymbol{w}',\boldsymbol{y}_j)\in G_\varepsilon(XY)$。

为方便计算，令事件 $E_i=\{(\boldsymbol{C}_i,\boldsymbol{y}_j)\in G_\varepsilon(XY)\}(i=0,1,\cdots,Q-1)$；表示第 i 个码字 $\boldsymbol{C}_i$ 与接收序列 $\boldsymbol{y}_j$ 构成联合 ε 典型序列对，事件 $\bar{E}_i$ 表示发送的第 i 个码字 $\boldsymbol{C}_i$ 与接收序列 $\boldsymbol{y}_j$ 不构成联合 ε 典型序列对，即事件 $\bar{E}_i=\{(\boldsymbol{C}_i,\boldsymbol{y}_j)\notin G_\varepsilon(XY)\}(i=0,1,\cdots,Q-1)$。所以有

$$\begin{aligned}P_{\mathrm{e}_0}&=P\{g(\boldsymbol{y}_j)\neq\boldsymbol{C}_0\mid\boldsymbol{x}_i=\boldsymbol{C}_0\}\\&=P(\bar{E}_0\cup E_1\cup\cdots\cup E_{Q-1})\end{aligned}$$

由集合论可得

$$P(\bigcup_k s_k)\leqslant\sum_k P(s_k)$$

故而有

$$P_{\mathrm{e}_0}\leqslant P(\bar{E}_0)+\sum_{i=1}^{Q-1}P(E_i)\tag{5.2.6}$$

根据式(5.2.3)可得

$$P(E_i)\geqslant 1-\delta$$

又

$$P(\bar{E}_0)=1-P(E_0)\leqslant\delta(n\text{ 足够大}) \tag{5.2.7}$$

而因为信道编码是随机选择码字的，所以码字 $\boldsymbol{C}_0$ 和 $\boldsymbol{C}_i$ 彼此是统计独立的，则序列 $\boldsymbol{y}_j$ 与 $\boldsymbol{C}_i(i\neq0)$也统计独立。由定理 5.2.3 可得

$$P(E_i)=\{P(\boldsymbol{C}_i,\boldsymbol{y}_j)\in G_\varepsilon(XY)\}\leqslant 2^{-N[I(X;Y)-3\delta]} \tag{5.2.8}$$

结合式(5.2.7)和式(5.2.8)，式(5.2.6)可写为

$$P_{e_0}\leqslant\delta+\sum_{i=1}^{Q-1}2^{-n[I(X;Y)-3\delta]}=\delta+(2^{nR}-1)2^{-n[I(X;Y)-3\delta]}$$

信道传递信息时，总希望传输率 R 尽可能大。在证明中可以选择 $P(\boldsymbol{x})$是达到信道容量的最佳输入概率分布 $P^*(\boldsymbol{x})$，于是可达条件 $R<I(X;Y)$成为 $R<C$。因此，选择任意小正数 δ 和ε，当 $R<C$ 并且 n 足够大时，P_{e_0} 为任意小。

$$P_{e_0}\leqslant\delta+2^{-n[C-R-3\delta]}-2^{-n[C-3\delta]}\leqslant\delta+2^{-n[C-R-3\delta]}\leqslant 2\delta$$

$P_r(E)$是对所有码书求统计平均，由此得当 n 足够大，$R<C$ 时，平均错误概率 $P_r(E)$为任意小。因此，至少存在一种码书($Q=2^{nR}$，n)，其错误概率小于或等于平均值 $P_r(E)$。由此证得定理。 [证毕]

总结：有噪信道编码定理告诉人们，只要信息传输率不超过信道容量，即 $R<C$，换句话说发送的码字数 $Q=2^{nR}$，只要码长 n 足够长，总可以在输入的符号集 X^n 中找到 Q 个许用码字和相应的译码规则，使得 $P_E\to0$，从而实现信息的可靠传输。

其证明方法的基本思路如下。

(1) 连续使用信道多次，即在 n 次无记忆扩展信道中讨论，当 n 足够大时，大数定律有效。

(2) 采取随机编码，在 X^n 中以输入概率分布 $P(\boldsymbol{x})$随机地选取符号序列，即经常出现的高概率典型序列作为码字。

(3) 采用联合典型序列译码准则，该准则等价于最大似然译码准则，即将接收序列译成传输模式中与最可能与之配对的那个码字。

(4) 考虑随机编码得到的所有码书结果，计算其平均错误概率，当 n 足够大时，此平均错误概率趋于零，由此证明得至少有一种好码存在。(定理的目标是证明存在一个码和相应的译码方法，具有很小的差错概率。但计算任意特定编码和译码系统的差错概率都是不容易的。香农的创新在于：不是构造一个好的编码和译码系统并计算其差错概率，而是计算所有码的平均分组差错概率，并证明这个平均值很小。那么一定存在某些码具有很小的差错概率。)

以上证明是基于离散无记忆信道的，已有文献证明香农第二定理对连续信道和有记忆信道同样成立。因此，无论是离散信道还是连续信道，其信道容量 C 是可靠通信的最大信息传输率。要想使信道中信息传输率大于信道容量而又无错误地传输是不可能的。

香农第二定理指出，当 $R<C$，n 足够大时，存在信道编码，使平均错误概率趋于零。那么 n 有限时，P_E 有可能趋于 0 吗？

已有研究证明，对于 DMC，P_E 趋于零的速度是与码长 n 呈指数关系。即当 $R<C$

时，平均错误概率为

$$P_E \leqslant \exp\{-nE_r(R)\} \tag{5.2.9}$$

式中，$E_r(R)$为随机编码指数，又称作 DMC 的可靠性函数，其表达式为

$$E_r(R) = \max_{0\leqslant\rho\leqslant 1}\max_{P(x)}\{E_0[\rho,P(x)]-\rho R\}$$

式中：ρ 为修正系数，$0\leqslant\rho\leqslant 1$。

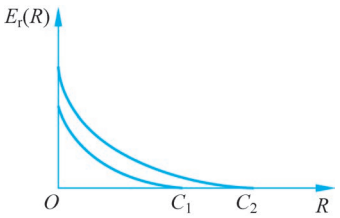


图 5.2.3　DMC 中 $E_r(R)$与 R 的关系

可见可靠性函数与输入概率分布有关。一般 $E_r(R)$与 R 的关系如图 5.2.3 所示，是一条∪型凸函数曲线。从图中可以看出，在 $R<C$ 的范围内 $E_r(R)>0$，式(5.2.9)表明 P_E 随 n 增大以指数趋于零。由此可知实际编码的码长 n 不需选择得很大。

任何信道的信道容量是一个明确的分界点，当取分界点以下的信息传输率时，P_E 以指数趋于零；当取分界点以上的信息传输率时，P_E 以指数趋于 1。因此，在任何信道中信道容量是可达的、最大的可靠信息传输率。

香农第二定理也只是一个存在定理，它说明错误概率趋于零的好码是存在的，但并没有给出具体的编码方法。从实用观点来看，定理似乎不令人满意。为了证明该定理，香农提出了随机编码的方法，这种编码并不具有工程可操作性。为了使大数定律有效，要求码长 n 足够长，但这样得到的码集很大，通过搜索得到好码很难实现，而且即使通过随机编码的方法找到了好码，这种码的码字也是毫无结构的，这就意味着在接收端只能通过查找表的方法进行译码。当码长 n 很大时，这种查找表占用的存储空间将会是难以承受的，也就难以实用和实现。尽管如此，信道编码定理仍然具有根本性的重要意义。它有助于指导各种通信系统的设计，有助于评价各种通信系统及编码的效率。为此，在香农 1948 年发表文章后，研究人员致力于研究实际信道中的各种易于实现的实际纠错编码方法，赋予码以各种形式的数学结构。这方面的研究非常活跃，出现了代数编码、卷积码、几何码等。至今已经发展有许多有趣和有效结构的码，并在实际通信系统中得到广泛应用。而且随着对信道编码研究的深入，关于随机编码的认识也更加深入，例如，1993 年 Turbo 码的出现，使我们发现随机编码理论不仅是分析和证明编码定理的主要方法，其思想在构造码上也发挥重要作用。近代研究反映出随机码思想的回归，许多性能优异的编码都有向随机编码方向靠近的趋势，如 Turbo 码中交织器的设计，LDPC 中稀疏校验矩阵的设计，这些为香农随机码理论的应用研究开启了新思路。


视频

5.3　线性分组码

线性分组码是纠错码中很重要的一类码，它也是讨论其他各类码的基础，这类码的原理虽然比较简单，但由此引入的一些概念非常重要，如码率、距离、重量等及其与纠检错能力的关系，以及码的生成矩阵 $\boldsymbol{G}$ 和一致校验矩阵 $\boldsymbol{H}$ 的表示及它们之间的关系，$\boldsymbol{H}$ 与纠错能力之间的关系等，这些概念也广泛地应用于其他各类码。

5.3.1 线性分组码的基本概念

5.3.1.1 线性分组码的定义

将信源所给出的二元信息序列首先分成等长的各个信息组，每组信息位长度为 k，记为

$$\boldsymbol{M}=(m_{k-1},m_{k-2},\cdots,m_1,m_0)$$

信息组 $\boldsymbol{M}$ 每一位上的信息元取 0 或 1，共有 2^k 种可能的取值。编码器根据某些规则将输入的信息序列编成码长为 n 的二元序列，即码字记为

$$\boldsymbol{C}=(c_{n-1},c_{n-2},\cdots,c_{n-k},c_{n-k-1},\cdots,c_1,c_0)$$

码字中每一位数字称为码元，取值为 0 或 1。如果码字的各监督元与信息元关系是线性的（可用一次线性方程来描述），那么这样的码称为**线性分组码**，记为**(n,k)码**。

例 5.3.1 (7,3)分组码，按以下的规则（校验方程）可得到 4 个校验元 $c_0c_1c_2c_3$：

$$\begin{cases} c_3=c_6+c_4 \\ c_2=c_6+c_5+c_4 \\ c_1=c_6+c_5 \\ c_0=c_5+c_4 \end{cases} \tag{5.3.1}$$

式中：c_6、c_5 和 c_4 是 3 个信息元。

由此可得到(7,3)分组码的 8 个码字。8 个信息组与 8 个码字的对应关系列于表 5.3.1 中。式(5.3.1)中的加均为模 2 加。由此方程看到，信息元与校验元满足线性关系，因此该(7,3)码是线性码。

表 5.3.1 例 5.3.1 的(7,3)码的码字与信息组对应关系

信 息 组	码 字
000	0000000
001	0011101
010	0100111
011	0111010
100	1001110
101	1010011
110	1101001
111	1110100

为了深入理解线性分组码的概念，将其与线性空间联系起来。由于每个码字都是一个长为 n 的（二进制）数组，因此可将每个码字看成一个二进制 n 重数组，进而看成二进制 n 维线性空间 $\boldsymbol{V}_n(F_2)$ 中的一个矢量。n 长的二进制数组共有 2^n 个，每个数组都称为一个二进制 n 重矢量。显然，所有 2^n 个 n 维数组将组成一个 n 维线性空间 $\boldsymbol{V}_n(F_2)$。

(n,k)分组码的 2^k 个 n 重就是这个 n 维线性空间的一个子集，若它能构成一个 k 维线性子空间，则它就是一个(n,k)线性分组码，这点可由上面的例子得到验证。

长为 7 的二进制 7 重共有 $2^7=128$ 个，显然这 128 个 7 重是 GF(2)上的一个 7 维线

性空间，而(7,3)码的 8 个码字，是从 128 个 7 重中按式(5.3.1)的规则挑出来的。可以验证，这 8 个码字对模 2 加法运算构成阿贝尔(Abel)群，即该码集是 7 维线性空间中的一个 3 维子空间。(n,k)线性分组码又可定义如下：

定义 5.3.1 二进制(n,k)线性分组码是 GF(2)域上的 n 维线性空间 $\boldsymbol{V}_n$ 中的一个 k 维子空间 $\boldsymbol{V}_{n,k}$。

因为线性空间在模 2 加运算下构成阿贝尔加群，所以又称线性分组码为群码。

从上面的讨论可知，线性分组码的编码问题就是如何从 n 维线性空间 $\boldsymbol{V}_n$ 中挑选出一个 k 维子空间 $\boldsymbol{V}_{n,k}$，而选择的规则完全由 $n-k$ 个校验方程决定。由于线性分组码对模 2 加满足封闭性，就为其最小距离的计算带来方便，具体有如下定理。

定理 5.3.1 一个(n,k)线性分组码中非零码字的最小重量等于码字集合 $\boldsymbol{C}$ 中的最小距离 d_0。

证明： 设有任两个码字 $\boldsymbol{C}_a,\boldsymbol{C}_b\in\boldsymbol{C}$。根据线性分组码性质，有 $\boldsymbol{C}_a+\boldsymbol{C}_b=\boldsymbol{C}_c\in\boldsymbol{C}$。而 $\boldsymbol{C}_c$ 的码重等于 $\boldsymbol{C}_a$ 与 $\boldsymbol{C}_b$ 的码距 $d(\boldsymbol{C}_a;\boldsymbol{C}_b)$，即

$$W(\boldsymbol{C}_c)=W(\boldsymbol{C}_a+\boldsymbol{C}_b)=d(\boldsymbol{C}_a;\boldsymbol{C}_b)$$

$\boldsymbol{C}_a$ 和 $\boldsymbol{C}_b$ 是 $\boldsymbol{C}$ 中任意两个码字，所以

$$W_{\min}(\boldsymbol{C}_a+\boldsymbol{C}_b)=W_{\min}(\boldsymbol{C}_c)=d_0 \qquad [证毕]$$

由表 5.3.1 中(7,3)码的 8 个码字可见，除全零码字外，其余 7 个码字最小重量 $W_{\min}=4$，而其中任二码字之间的最小距离 $d_0=4$。

根据线性分组码的封闭性不难得出如下结论。

定理 5.3.2 任何一个二元线性分组码中要么所有码字的重量都是偶数，要么偶数重量码字与奇数重量码字的数目相等。

证明： 令 $\boldsymbol{C}$ 表示任何一个线性二元分组码，$\boldsymbol{A}$ 表示所有偶数重量的码字集合，$\boldsymbol{B}$ 表示所有奇数重量的码字集合，显然 $\boldsymbol{C}=\boldsymbol{A}\cup\boldsymbol{B}$。根据线性分组码的特点，无论是 $\boldsymbol{A}$ 或 $\boldsymbol{B}$ 中码字两两相加，还是从 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 中各选一个码字相加，得到的和矢量仍为该线性分组码集中的码字。

若 $\boldsymbol{B}$ 为空集，则 $\boldsymbol{C}=\boldsymbol{A}$，这时所有码字都是偶数重量。若 $\boldsymbol{B}$ 不为空集，则至少有一个奇数重量的码字 $\boldsymbol{b}$，构成

$$\boldsymbol{b}+\boldsymbol{A}=\{\boldsymbol{b}+\boldsymbol{a}\mid \boldsymbol{a}\in\boldsymbol{A}\}$$

集合 $\boldsymbol{b}+\boldsymbol{A}$ 表示由集合 $\boldsymbol{A}$ 中的码字与 $\boldsymbol{b}$ 相加后所得的码字全体。由式 $W(\boldsymbol{C}_1\oplus\boldsymbol{C}_2)=W(\boldsymbol{C}_1)+W(\boldsymbol{C}_2)-2W(\boldsymbol{C}_1\odot\boldsymbol{C}_2)$可知，奇数重量的矢量与偶数重量矢量之和为奇数重量的矢量，因此 $\boldsymbol{A}+\boldsymbol{b}$ 中所有码字的重量为奇数，而且 $\boldsymbol{A}+\boldsymbol{b}$ 中的码字数与 $\boldsymbol{A}$ 中码字的数量相同(这也是陪集的性质)，可见偶数重量码字与奇数重量码字的数目相等。

下面再说明集合 $\boldsymbol{B}$ 中的所有码字都在集合 $\boldsymbol{A}+\boldsymbol{b}$ 中。设 $\boldsymbol{b}'$ 为 $\boldsymbol{B}$ 中任取的一个码字，$\boldsymbol{b}'$ 和 $\boldsymbol{b}$ 都是奇数重量，同样由式$(W(\boldsymbol{C}_1\oplus\boldsymbol{C}_2)=W(\boldsymbol{C}_1)+W(\boldsymbol{C}_2)-2W(\boldsymbol{C}_1\odot\boldsymbol{C}_2))$可知，奇数重量的矢量与奇数重量矢量之和为偶数重量的矢量，$\boldsymbol{B}$ 中码字两两相加的码字是偶数重量，所以存在一个 $\boldsymbol{a}\in\boldsymbol{A}$ 使 $\boldsymbol{b}'+\boldsymbol{b}=\boldsymbol{a}$，那么

$$\boldsymbol{b}'=\boldsymbol{a}-\boldsymbol{b}=\boldsymbol{a}+\boldsymbol{b}\in\boldsymbol{b}+\boldsymbol{A}$$

式中利用了“二元码加法与减法等价”的性质。可见，集合 $\boldsymbol{B}$ 中的所有码字都在集合 $\boldsymbol{A}+\boldsymbol{b}$ 中。即若 $\boldsymbol{B}$ 中有一个奇数重码字，则 $\boldsymbol{C}$ 中奇数重码字数目与偶数重码字数相等。

［证毕］

5.3.1.2 生成矩阵 G 与一致校验矩阵 H

(n,k)线性分组码的编码问题就是如何在 n 维线性空间 $\mathbf{V}_n(F_2)$中找出满足一定要求的由 2^k 个矢量组成的 k 维线性子空间 $\boldsymbol{C}$ 的问题。或者说，在满足给定条件(码的最小距离 d_0 或码率 R)下，如何根据已知的 k 个信息元求得 $n-k$ 个监督元。由于是线性码，这些监督元一定是由若干线性方程构成的一个方程组，若 $n-k$ 个监督元是独立的，则该方程组由 $n-k$ 个线性方程构成。

例 5.3.1 中的(7,3)码。若 c_6、c_5、c_4 代表 3 个信息元，c_3、c_2、c_1、c_0 代表 4 个监督元，可以由下列线性方程组建立它们之间的关系：

$$\begin{cases} 1\cdot c_3 = 1\cdot c_6 + 0\cdot c_5 + 1\cdot c_4 \\ 1\cdot c_2 = 1\cdot c_6 + 1\cdot c_5 + 1\cdot c_4 \\ 1\cdot c_1 = 1\cdot c_6 + 1\cdot c_5 + 0\cdot c_4 \\ 1\cdot c_0 = 0\cdot c_6 + 1\cdot c_5 + 1\cdot c_4 \end{cases} \tag{5.3.2}$$

上述运算均为模 2 加。在已知 c_6、c_5、c_4 后，可立即求出 c_3、c_2、c_1、c_0。例如，$\boldsymbol{M}=(101)$，即 $c_6=1$，$c_5=0$，$c_4=1$，代入式(5.3.2)，可得 4 个监督元为(0011)，进而得到码字为(1010011)。根据式(5.3.2)对不同信息组进行计算，即可得到全部(7,3)码字。可见式(5.3.2)对确定该(7,3)码是非常关键的。式(5.3.2)称为一致校验方程，它直接反映了码字中信息元与监督元之间的约束关系。不过，从编码的角度看，利用式(5.3.2)逐个计算得到全体码字是很麻烦的，还需进一步寻求其内在规律。

1. 生成矩阵

已知(n,k)线性分组码的 2^k 个码字组成 n 维矢量空间的一个 k 维子空间，而线性空间可由其基底张成，因此(n,k)线性分组码的 2^k 个码字完全可由 k 个独立的矢量所组成的基底张成。设 k 个矢量为

$$\begin{aligned} \boldsymbol{g}_1 &= (g_{11} \quad g_{12}\cdots g_{1k} \quad g_{1,k+1}\cdots g_{1n}) \\ \boldsymbol{g}_2 &= (g_{21} \quad g_{22}\cdots g_{2k} \quad g_{2,k+1}\cdots g_{2n}) \\ &\vdots \\ \boldsymbol{g}_k &= (g_{k1} \quad g_{k2}\cdots g_{kk} \quad g_{k,k+1}\cdots g_{kn}) \end{aligned}$$

将它们写成矩阵形式：

$$\boldsymbol{G} = \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1k} & g_{1,k+1} & \cdots & g_{1n} \\ g_{21} & g_{22} & \cdots & g_{2k} & g_{2,k+1} & \cdots & g_{2n} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ g_{k1} & g_{k2} & \cdots & g_{kk} & g_{k,k+1} & \cdots & g_{kn} \end{bmatrix} \tag{5.3.3}$$

(n,k)码中的任何码字均可由这组基底的线性组合生成，即

$$\boldsymbol{C}=\boldsymbol{MG}=(m_{k-1}\quad m_{k-2}\cdots\quad m_0)\begin{bmatrix}g_{11}&g_{12}&\cdots&g_{1n}\\g_{21}&g_{22}&\cdots&g_{2n}\\\vdots&\vdots&\ddots&\vdots\\g_{k1}&g_{k2}&\cdots&g_{kn}\end{bmatrix}\tag{5.3.4}$$

式中：$\boldsymbol{M}=(m_{k-1}\quad m_{k-2}\cdots\quad m_0)$是 k 个信息元组成的信息组。

这就是说，每给定一个信息组，通过式(5.3.4)可求得其相应的码字。故这个由 k 个线性无关矢量组成的基底所构成的 $k\times n$ 矩阵 $\boldsymbol{G}$ 称为(n,k)码的**生成矩阵**。

(7,3)码可以从表 5.3.1 中的 8 个码字中任意挑选出 $k=3$ 个线性无关的码字(1001110)、(0100111)和(0011101)作为码的一组基底，由它们组成 $\boldsymbol{G}$ 的行，可得

$$\boldsymbol{G}=\begin{bmatrix}1&0&0&1&1&1&0\\0&1&0&0&1&1&1\\0&0&1&1&1&0&1\end{bmatrix}$$

若信息组 $\boldsymbol{M}_i=(011)$，则相应的码字为

$$\boldsymbol{C}_i=(011)\begin{bmatrix}1&0&0&1&1&1&0\\0&1&0&0&1&1&1\\0&0&1&1&1&0&1\end{bmatrix}=(0111010)$$

它是 $\boldsymbol{G}$ 矩阵后两行相加的结果。

值得注意的是，线性空间(或子空间)的基底可以不止一组，因此作为码的生成矩阵 $\boldsymbol{G}$ 也可以不止一种形式。但不论哪种形式，它们都生成相同的线性空间(或子空间)，即生成同一个(n,k)线性分组码。

实际上，码的生成矩阵还可由其编码方程直接得出。例如，例 5.3.1 的(7,3)码可将编码方程改写为

$$\begin{aligned}c_6&=c_6\\c_5&=c_5\\c_4&=c_4\\c_3&=c_6+c_4\\c_2&=c_6+c_5+c_4\\c_1&=c_6+c_5\\c_0&=c_5+c_4\end{aligned}$$

写成矩阵形式，即

$$[c_6\ \ c_5\ \ c_4\ \ c_3\ \ c_2\ \ c_1\ \ c_0]=\begin{bmatrix}c_6\\c_5\\c_4\\c_3\\c_2\\c_1\\c_0\end{bmatrix}^{\mathrm{T}}=\begin{bmatrix}c_6&&&&&&\\&&c_5&&&&\\&&&&&&c_4\\c_6&&&&+&&c_4\\c_6&+&c_5&+&c_4&&\\c_6&+&c_5&&&&\\&&c_5&+&c_4&&\end{bmatrix}^{\mathrm{T}}$$

$$=[c_6\ \ c_5\ \ c_4]\begin{bmatrix}1&0&0&1&1&1&0\\0&1&0&0&1&1&1\\0&0&1&1&1&0&1\end{bmatrix}$$

$$=[c_6\ \ c_5\ \ c_4]\boldsymbol{G}$$

故(7,3)码的生成矩阵为

$$\boldsymbol{G}=\begin{bmatrix}1&0&0&1&1&1&0\\0&1&0&0&1&1&1\\0&0&1&1&1&0&1\end{bmatrix}$$

可见,生成矩阵可由编码方程的系数矩阵转置得到。

在线性分组码中经常用到一种特殊的结构,如上例(7,3)码的所有码字的前三位,都是与信息组相同,属于信息元,后四位是校验元。这种形式的码称为系统码。

定义 5.3.2 若信息组以不变的形式,在码字的任意 k 位中出现,则该码称为**系统码**;否则,称为**非系统码**。

目前最流行的有两种形式的系统码:一种是信息组排在码字$(c_{n-1},c_{n-2},\cdots,c_0)$的最左边 k 位,$c_{n-1},c_{n-2},\cdots,c_{n-k}$,如表 5.3.1 中所列出的码字就是这种形式;另一种是信息组被安置在码字的最右边 k 位,$c_{k-1},c_{k-2},\cdots,c_0$。

若采用码字左边 k 位(前 k 位)是信息位的系统码形式(今后均采用此形式),则式(5.3.3)所示的 $\boldsymbol{G}$ 矩阵左边 k 列应是一个 k 阶单位方阵 $\boldsymbol{I}_k$($g_{1,1}=g_{2,2}=\cdots=g_{k,k}=1$,其余元素均为 0)。因此,系统码的生成矩阵可表示成

$$\boldsymbol{G}_0=\begin{bmatrix}1&0&\cdots&0&g_{1,k+1}&\cdots&g_{1n}\\0&1&\cdots&0&g_{2,k+1}&\cdots&g_{2n}\\\vdots&\vdots&\ddots&\vdots&\vdots&\ddots&\vdots\\0&0&\cdots&1&g_{k,k+1}&\cdots&g_{kn}\end{bmatrix}=[\boldsymbol{I}_k\boldsymbol{P}]$$

式中:$\boldsymbol{P}$ 为 $k\times(n-k)$ 矩阵。

只有这种形式的生成矩阵才能生成(n,k)系统型线性分组码,即标准形式,因此,系统码的生成矩阵也是一个典型矩阵(或称标准阵)。考察典型矩阵,便于检查 $\boldsymbol{G}$ 的各行是否线性无关。如果 $\boldsymbol{G}$ 不具有标准型,虽能生成线性码,但码字不具备系统码的结构,此时可将 $\boldsymbol{G}$ 的非标准型经过行初等变换变成标准型 $\boldsymbol{G}_0$。由于系统码的编码与译码较非系统码简单,而且对分组码而言,系统码与非系统码的抗干扰能力完全等价,故若无特别声明,仅讨论系统码。

2. 一致校验矩阵

前面介绍过,编码问题是在给定的 d_0 或码率 R 下如何利用从已知的 k 个信息元求得 $r=n-k$ 个校验元。例 5.3.1 中的(7,3)码的 4 个检验元由式(5.3.2)所示的线性方程组决定。为了更好地说明信息元与校验元的关系,现将式(5.3.2)填补系数并移项,变换为

$$\begin{cases}1\cdot c_6+0\cdot c_5+1\cdot c_4+1\cdot c_3+0\cdot c_2+0\cdot c_1+0\cdot c_0=0\\1\cdot c_6+1\cdot c_5+1\cdot c_4+0\cdot c_3+1\cdot c_2+0\cdot c_1+0\cdot c_0=0\\1\cdot c_6+1\cdot c_5+0\cdot c_4+0\cdot c_3+0\cdot c_2+1\cdot c_1+0\cdot c_0=0\\0\cdot c_6+1\cdot c_5+1\cdot c_4+0\cdot c_3+0\cdot c_2+0\cdot c_1+1\cdot c_0=0\end{cases}$$

用矩阵表示这些线性方程：

$$\begin{bmatrix}1&0&1&1&0&0&0\\1&1&1&0&1&0&0\\1&1&0&0&0&1&0\\0&1&1&0&0&0&1\end{bmatrix}=\begin{bmatrix}c_6\\c_5\\c_4\\c_3\\c_2\\c_1\\c_0\end{bmatrix}=\begin{bmatrix}0\\0\\0\\0\end{bmatrix}=\mathbf{0}^{\mathrm{T}} \tag{5.3.5}$$

或

$$[c_6\ \ c_5\ \ c_4\ \ c_3\ \ c_2\ \ c_1\ \ c_0]\begin{bmatrix}1&1&1&0\\0&1&1&1\\1&1&0&1\\1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}=[0000]=\mathbf{0} \tag{5.3.6}$$

将上面方程的系数矩阵用 $\boldsymbol{H}$ 表示：

$$\boldsymbol{H}=\begin{bmatrix}1&0&1&1&0&0&0\\1&1&1&0&1&0&0\\1&1&0&0&0&1&0\\0&1&1&0&0&0&1\end{bmatrix}$$

式(5.3.5)或式(5.3.6)表明，$\boldsymbol{C}$ 中的各码元是满足由 $\boldsymbol{H}$ 所确定的 $r(r=n-k)$ 个线性方程的解，故 $\boldsymbol{C}$ 是一个码字；若 $\boldsymbol{C}$ 中码元组成一个码字，则一定满足由 $\boldsymbol{H}$ 所确定的 r 个线性方程。故 $\boldsymbol{C}$ 是式(5.3.5)或式(5.3.6)解的集合。显而易见，$\boldsymbol{H}$ 一定，便可由信息元求出校验元，编码问题迎刃而解；或者说，要解决编码问题，只要找到 $\boldsymbol{H}$ 即可。由于 $\boldsymbol{H}$ 矩阵是一致校验方程组的系数矩阵，它也反映了码字中的约束关系，故称其为**一致校验矩阵**。按 $\boldsymbol{H}$ 所确定的规则也可求出(n,k)码的所有码字。

一般而言，(n,k)线性码有 $r(r=n-k)$个校验元，若这些校验元是独立加入的，则一致校验方程必须有 r 个独立的线性方程。所以(n,k)线性码的 $\boldsymbol{H}$ 矩阵由 r 行和 n 列组成，可表示为

$$\boldsymbol{H}=\begin{bmatrix}h_{1,n-1}&h_{1,n-2}&\cdots&h_{10}\\h_{2,n-1}&h_{2,n-2}&\cdots&h_{20}\\\vdots&\vdots&\ddots&\vdots\\h_{r,n-1}&h_{r,n-2}&\cdots&h_{r0}\end{bmatrix}$$

这里 h_{ij} 中，i 代表行号，j 代表列号。因此，$\boldsymbol{H}$ 是一个 r 行 n 列矩阵。由 $\boldsymbol{H}$ 矩阵可建立码的 r 个线性方程：

$$\begin{bmatrix} h_{1,n-1} & h_{1,n-2} & \cdots & h_{10} \\ h_{2,n-1} & h_{2,n-2} & \cdots & h_{20} \\ \vdots & \vdots & \ddots & \vdots \\ h_{r,n-1} & h_{r,n-2} & \cdots & h_{r0} \end{bmatrix} \begin{bmatrix} c_{n-1} \\ c_{n-2} \\ \vdots \\ c_1 \\ c_0 \end{bmatrix} = \boldsymbol{0}^{\mathrm{T}}$$

简写为

$$\boldsymbol{H}\boldsymbol{C}^{\mathrm{T}} = \boldsymbol{0}^{\mathrm{T}} \tag{5.3.7}$$

或

$$\boldsymbol{C}\boldsymbol{H}^{\mathrm{T}} = \boldsymbol{0} \tag{5.3.8}$$

式中：$\boldsymbol{C}=[c_{n-1}\ c_{n-2}\cdots\ c_1\ c_0]$，$\boldsymbol{C}^{\mathrm{T}}$ 是 $\boldsymbol{C}$ 的转置；$\boldsymbol{0}$ 是一个全为 0 的 r 重。

综上所述，将 $\boldsymbol{H}$ 矩阵的特点归纳如下。

(1) $\boldsymbol{H}$ 矩阵的每一行代表一个线性方程的系数，它表示求一个校验元的线性方程。

(2) $\boldsymbol{H}$ 矩阵每一列代表此码元与哪几个校验方程有关。

(3) 由此 $\boldsymbol{H}$ 矩阵得到的 (n,k) 分组码的每个码字 $\boldsymbol{C}_i$ $(i=1,2,\cdots,2^k)$ 都必须满足由 $\boldsymbol{H}$ 矩阵行所确定的线性方程，即式(5.3.7)或式(5.3.8)。

(4) 若 (n,k) 码的 $r(r=n-k)$ 个校验元是独立的，则 $\boldsymbol{H}$ 矩阵必须有 r 行，且各行之间线性无关，即 $\boldsymbol{H}$ 矩阵的秩为 r。若将 $\boldsymbol{H}$ 的每一行看成一个矢量，则此 r 个矢量必然张成 n 维矢量空间中的一个 r 维子空间 $\boldsymbol{V}_{n,r}$。

(5) 考虑到生成矩阵 $\boldsymbol{G}$ 中的每一行及其线性组合都是 (n,k) 码中的一个码字，故有

$$\boldsymbol{G}\boldsymbol{H}^{\mathrm{T}} = \boldsymbol{0} \tag{5.3.9}$$

或

$$\boldsymbol{H}\boldsymbol{G}^{\mathrm{T}} = \boldsymbol{0}^{\mathrm{T}} \tag{5.3.10}$$

这说明由 $\boldsymbol{G}$ 和 $\boldsymbol{H}$ 的行生成的空间互为零空间。也就是说，$\boldsymbol{H}$ 矩阵的每一行与由 $\boldsymbol{G}$ 矩阵行生成的分组码中每个码字内积均为零，即 $\boldsymbol{G}$ 和 $\boldsymbol{H}$ 彼此正交。

(6) 由上面的例子不难看出，(7,3)码的 $\boldsymbol{H}$ 矩阵右边 4 行 4 列为一个 4 阶单位方阵，一般而言，系统型 (n,k) 线性分组码的 $\boldsymbol{H}$ 矩阵右边 r 列组成一个单位方阵 $\boldsymbol{I}_r$，故有

$$\boldsymbol{H} = [\boldsymbol{Q}\boldsymbol{I}_r]$$

式中：$\boldsymbol{Q}$ 为 $r\times k$ 矩阵。

这种形式的矩阵称为典型阵或标准阵，采用典型阵形式的 $\boldsymbol{H}$ 矩阵更易于检查各行是否线性无关。

(7) 由式(5.3.10)可得

$$[\boldsymbol{Q}\boldsymbol{I}_r][\boldsymbol{I}_k\boldsymbol{P}]^{\mathrm{T}} = [\boldsymbol{Q}\boldsymbol{I}_r]\begin{bmatrix} \boldsymbol{I}_k \\ \boldsymbol{P}^{\mathrm{T}} \end{bmatrix} = \boldsymbol{Q} + \boldsymbol{P}^{\mathrm{T}} = \boldsymbol{0}^{\mathrm{T}}$$

这说明

$$\boldsymbol{P}=\boldsymbol{Q}^{\mathrm{T}}$$

或

$$\boldsymbol{P}^{\mathrm{T}}=\boldsymbol{Q}$$

这就是说，$\boldsymbol{P}$ 的第一行就是 $\boldsymbol{Q}$ 的第一列，$\boldsymbol{P}$ 的第二行就是 $\boldsymbol{Q}$ 的第二列，……。因此，$\boldsymbol{H}$ 一定，$\boldsymbol{G}$ 也就一定；反之亦然。

3. 对偶码

(n,k)码是 n 维矢量空间中的一个 k 维子空间 $\boldsymbol{V}_{n,k}$，可由一组基底即 $\boldsymbol{G}$ 的行张成。由式(5.3.9)可以看出，由 $\boldsymbol{H}$ 矩阵的行所张成的 n 维矢量空间中的一个 r 维子空间 $\boldsymbol{V}_{n,r}$ 是(n,k)码空间 $\boldsymbol{V}_{n,k}$ 的一个零空间。由线性代数知，$\boldsymbol{V}_{n,r}$ 的零空间必是一个 k 维子空间，它正是(n,k)码的全体码字集合。若把(n,k)码的一致校验矩阵看成(n,r)码的生成矩阵，将(n,k)码的生成矩阵看成(n,r)码的一致校验矩阵，则称这两种码互为**对偶码**。相应地称 $\boldsymbol{V}_{n,k}$ 和 $\boldsymbol{V}_{n,r}$ 互为对偶空间。

例 5.3.2 求例 5.3.1 所述(7,3)码的对偶码。

显然，(7,3)码的对偶码应是(7,4)码，因此，(7,4)码的 $\boldsymbol{G}$ 矩阵就是(7,3)码的 $\boldsymbol{H}$ 矩阵：

$$\boldsymbol{G}_{(7,4)}=\boldsymbol{H}_{(7,3)}=\begin{bmatrix}1&0&1&1&0&0&0\\1&1&1&0&1&0&0\\1&1&0&0&0&1&0\\0&1&1&0&0&0&1\end{bmatrix}$$

由此可得出(7,4)码的码字如表 5.3.2 所示。

表 5.3.2 (7,4)线性分组码

信 息 元	码 字	信 息 元	码 字
0 0 0 0	0 0 0 0 0 0 0	1 0 0 0	1 0 0 0 1 0 1
0 0 0 1	0 0 0 1 0 1 1	1 0 0 1	1 0 0 1 1 1 0
0 0 1 0	0 0 1 0 1 1 0	1 0 1 0	1 0 1 0 0 1 1
0 0 1 1	0 0 1 1 1 0 1	1 0 1 1	1 0 1 1 0 0 0
0 1 0 0	0 1 0 0 1 1 1	1 1 0 0	1 1 0 0 0 1 0
0 1 0 1	0 1 0 1 1 0 0	1 1 0 1	1 1 0 1 0 0 1
0 1 1 0	0 1 1 0 0 0 1	1 1 1 0	1 1 1 0 1 0 0
0 1 1 1	0 1 1 1 0 1 0	1 1 1 1	1 1 1 1 1 1 1

若一个码的对偶码就是它自己，则称该码为**自对偶**。自对偶码必是$(2m,m)$形式的分组码，(2,1)重复码就是一个自对偶码。

中文教学录像

5.3.2 线性分组码的译码方法

视频

只要找到了 $\boldsymbol{H}$ 矩阵或 $\boldsymbol{G}$ 矩阵，便解决了编码问题。经编码后发送的码字，由于信道干扰可能出错，收方怎样发现或纠正错误？这就是译码要解决的问题。

设发送的码字 $\boldsymbol{C}=(c_{n-1},c_{n-2},\cdots,c_1,c_0)$，信道产生的错误图样 $\boldsymbol{E}=(e_{n-1},e_{n-2},\cdots,$

e_1,e_0)，而接收序列 $\boldsymbol{R}=(r_{n-1},r_{n-2},\cdots,r_1,r_0)$，那么 $\boldsymbol{R}=\boldsymbol{C}+\boldsymbol{E}$，即有 $r_i=c_i+e_i$，其中 $c_i,r_i,e_i\in$GF(2)。译码的任务就是要从 $\boldsymbol{R}$ 中求出 $\boldsymbol{E}$，从而得到码字估值 $\boldsymbol{C}=\boldsymbol{R}-\boldsymbol{E}$。

5.3.2.1 标准阵列译码


视频

标准阵列译码法是对线性分组码进行译码最一般的方法，这种方法的原理也是对解释线性分组码概念最直接的描述。

在5.1.3节中已介绍，(n,k)码中任一码字 $\boldsymbol{C}$ 在有噪信道上传输，接收矢量 $\boldsymbol{R}$ 可以是 n 维线性空间 $\boldsymbol{V}_n(F_2)$ 中任一矢量。收端译码有很多种，但其本质就是对码字进行分类，以便从接收矢量中确定发送码字的估值。

二元(n,k)码的 2^k 个码字集合是 n 维矢量空间的一个 k 维子空间。如果将整个 n 维矢量空间的 2^n 个矢量划分成 2^k 个子集 $\wp_1,\wp_2,\cdots,\wp_{2^k}$，且这些子集不相交，即彼此不含有公共的矢量，每个子集 $\wp_i$ 包含且仅包含一个码字 $\boldsymbol{C}_i(i=1,2,\cdots,2^k)$，从而建立一一对应的关系：

$$\boldsymbol{C}_1\leftrightarrow\wp_1,\quad \boldsymbol{C}_2\leftrightarrow\wp_2,\cdots,\quad \boldsymbol{C}_{2^k}\leftrightarrow\wp_{2^k}$$

当发送一个码字 $\boldsymbol{C}_i$，而接收字为 $\boldsymbol{R}_i$，则 $\boldsymbol{R}_i$ 必属于且仅属于这些子集之一。若 $\boldsymbol{R}_i$ 落入 $\wp_i$ 中，则译码器可判断发送码字是 $\boldsymbol{C}_i$。

这样做的风险是：如子集 $\wp_i$ 是对应原发送的码字，则译码正确；若 $\wp_i$ 并不对应原发送的码字，则译码错误。当然，在有扰信道找到一个绝对无误的译码方案是不可能的，但可以找到一种使译码错误概率最小的方案。那么怎样才能将 n 维矢量空间划分成符合上述要求的 2^k 个子集？最一般的方法是按下列方法制作一个表。先把 2^k 个码矢量置于第一行，并以零码矢 $\boldsymbol{C}_1=(0,0,0,\cdots,0)$为最左面的元素，在其余 2^n-2^k 个 n 重中选择一个重量最轻的 n 重 $\boldsymbol{E}_2$，并置 $\boldsymbol{E}_2$ 于零码矢 $\boldsymbol{C}_1$ 的下面，于是表的第二行是 $\boldsymbol{E}_2$ 和每个码矢 $\boldsymbol{C}_i$ 相加，并把 $\boldsymbol{E}_2+\boldsymbol{C}_i$ 置于 $\boldsymbol{C}_i$ 的下面即同一列，完成第二行。第三行是再从其余的 n 重中任选一个重量最轻的 n 重 $\boldsymbol{E}_3$ 置于 $\boldsymbol{C}_i$ 的下面(第三行第一列)，同理将 $\boldsymbol{E}_3+\boldsymbol{C}_i$ 置于 $\boldsymbol{C}_i$ 之下完成第三行，以此类推，一直到全部 n 重用完为止。于是就得到表5.3.3。

表5.3.3 标准阵译码表

码字	$\boldsymbol{C}_1$(陪集首)	$\boldsymbol{C}_2$	…	$\boldsymbol{C}_i$	…	$\boldsymbol{C}_{2^k}$
禁用码组	$\boldsymbol{E}_2$	$\boldsymbol{C}_2+\boldsymbol{E}_2$	…	$\boldsymbol{C}_i+\boldsymbol{E}_2$	…	$\boldsymbol{C}_{2^k}+\boldsymbol{E}_2$
	$\boldsymbol{E}_3$	$\boldsymbol{C}_2+\boldsymbol{E}_3$	…	$\boldsymbol{C}_i+\boldsymbol{E}_3$	…	$\boldsymbol{C}_{2^k}+\boldsymbol{E}_3$
	⋮	⋮	⋮	⋮	⋮	⋮
	$\boldsymbol{E}_{2^{n-k}}$	$\boldsymbol{C}_2+\boldsymbol{E}_{2^{n-k}}$	…	$\boldsymbol{C}_i+\boldsymbol{E}_{2^{n-k}}$	…	$\boldsymbol{C}_{2^k}+\boldsymbol{E}_{2^{n-k}}$

表5.3.3共有 2^{n-k} 行 2^k 列，其中每一列就是含有 $\boldsymbol{C}_i$ 的子集 $\wp_i$。从按照上述方法列出的表可以看出：表中同一行中没有两个 n 重是相同的，也没有一个 n 重出现在不同行中，所以所划分的子集 $\wp_i$ 之间是互不相交的，即每个 n 重在此表中仅出现一次，这个表称为线性分组码的标准阵列。译码表或简称**标准阵**，而每一行称为一个陪集，每一行最左边的那个 n 重 E_i 称为**陪集首**。而表的第一行即为(n,k)分组码的全体，又称**子群**。

收到的 n 重 R 落在某一列中，则译码器就译成相应于该列最上面的码字。因此，若

发送的码字码为 $\boldsymbol{C}_i$，收到的 $\boldsymbol{R}=\boldsymbol{C}_i+\boldsymbol{E}_j$（$1\leqslant j\leqslant 2^{n-k}$，$\boldsymbol{E}_1$ 是全 0 矢量），则能正确译码。若收到的 $\boldsymbol{R}=\boldsymbol{C}_l+\boldsymbol{E}_j$（$l\neq i$），则产生了错误译码。现在的问题是如何划分陪集，使译码错误概率最小？这最终取决于如何挑选陪集首。因为一个陪集的划分主要取决于子群，而子群就是 2^k 个码字，这已确定，因此余下的问题就是如何决定陪集首。

在信道误码率 $P<0.5$ 的 BSC 中，传输出错情况应为少数，产生 1 个错误的概率比产生 2 个错误的大，产生 2 个错误的比 3 个错误的大……也就是说，错误图样重量越小，产生的可能性越大。因此，译码器必须首先保证能正确纠正这种出现可能性最大的错误图样，即重量最轻的错误图样。这相当于在构造译码表时要求挑选重量最轻的 n 重为陪集首，放在标准阵中的第一列，而以全 0 码字作为子群的陪集首。这样得到的标准阵能使译码错误概率最小。由于这样安排的译码表使得 $\boldsymbol{C}_i+\boldsymbol{E}_j$ 与 $\boldsymbol{C}_i$ 的距离保证最小，因而也称为最小距离译码，在 BSC 下，它们等效于最大似然译码。

构造一般(n,k)码标准阵列的方法归纳如下。

(1) 将 $\boldsymbol{V}_{n,k}$ 的 2^k 个码字作第一行，全零矢量作其陪集首，即作为 $\boldsymbol{E}_1$。

(2) 在剩下的禁用码组中挑选重量最小的 n 重作第二行的陪集首，以 $\boldsymbol{E}_2$ 表示，以此求出 $\boldsymbol{C}_2+\boldsymbol{E}_2,\boldsymbol{C}_3+\boldsymbol{E}_2,\cdots,\boldsymbol{C}_{2^k}+\boldsymbol{E}_2$ 分别列于对应码字 $\boldsymbol{C}_i$ 所在列，从而构成第二行。

(3) 依方法(2)所述方法，直至将 2^n 个矢量划分完毕。

这样就得到如表 5.3.3 所示的标准阵列。

从标准阵列可以看出，陪集首的集合就是一个可纠正错误图样 $\boldsymbol{E}_i$ 的集合，而各码字所对应的列就是该码字的正确接收区。因此，在 BSC 下，二元线性分组码正确译码概率为

$$P_c=\sum_{j=0}^{2^r-1}p^{W(j)}(1-p)^{n-W(j)}=\sum_{i=0}^{n}A_i p^i(1-p)^{n-i}$$

式中：$r=n-k$；$W(j)$为第 j 个陪集首的重量；A_i 是重量为 i 的陪集首的个数；$\boldsymbol{A}=(A_0A_1\cdots A_n)$是陪集首的重量分布矢量；$p$ 为信道误码率。

从几何的角度理解，码的陪集划分相当于把 n 维线性空间 $\boldsymbol{V}_n$ 按照该线性分组码 $\boldsymbol{V}_{n,k}$ 划分。共有 2^k 个码字，相当于有 2^k 个互不相交的球，球的半径是陪集首中错误图样的最大码重，这 2^k 个球把整个线性空间 $\boldsymbol{V}_n$ 充满，所有禁用码组将分别落在不同的球中。当发送某个码字时，如果错误图样的影响使接收序列位于发送码字的球内，这相当于错误图样出现在陪集首，错误图样中“1”的个数很少，错误不是太严重，则接收端可以根据标准阵正确译码；如果错误图样的影响使接收码字序列落在其他许用码字的球内，这相当于错误很严重，错误图样中“1”的个数很多，错误图样不属于陪集首，则接收端根据标准阵译码就会产生译码错误。

结合这个几何解释可以分析线性分组码检错、纠错能力与最小码距 d_0 之间的关系。

首先，为了能检测出一个码字中发生的全部 e 个错误，要求 $d_0\geqslant e+1$。只有这样，该 e 个错误才不至于把一个许用码字变为另一个许用码字。其次，为了能纠正一个码字中发生的全部 t 个错误，要求 $d_0\geqslant 2t+1$，如图 5.3.1 所示。只有满足了这个条件才能保证这个 t 位错误引起的禁用码组仍位于以发送码字为球心的球内。最后，在一个码字中，为

了能纠正所有 t 个错误；并同时检测所有 e 个错误，要求 $d_0 \geqslant t+e+1(t<e)$。当误码数小于 t 时，能纠正所有 t 个错误；当误码数在 $(t, t+e+1]$ 时，能检测所有 e 个错误。这即是基于线性分组码对定理 5.1.2～定理 5.1.4 的解释。

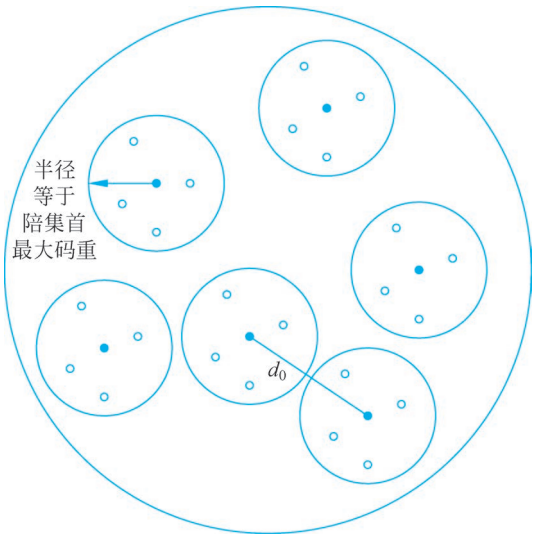


图 5.3.1　对线性分组码纠错能力的几何解释

例 5.3.3　以(6,3)码为例排列出它的标准阵列。对于(6,3)码，它的生成矩阵为

$$\boldsymbol{G}=\begin{pmatrix}1&0&0&0&1&1\\0&1&0&1&0&1\\0&0&1&1&1&0\end{pmatrix}$$

此码共有 8 个码字

信息组	码字
0 0 0	0 0 0 0 0 0
0 0 1	0 0 1 1 1 0
0 1 0	0 1 0 1 0 1
0 1 1	0 1 1 0 1 1
1 0 0	1 0 0 0 1 1
1 0 1	1 0 1 1 0 1
1 1 0	1 1 0 1 1 0
1 1 1	1 1 1 0 0 0

它的标准阵列如表 5.3.4 所示。

表 5.3.4　(6,3)码标准阵列

000000	001110	010101	100011	011011	101101	110110	111000
000001	001111	010100	100010	011010	101100	110111	111001
000010	001100	010111	100001	011001	101111	110100	111010
000100	001010	010001	100111	011111	101001	110010	111100

续表

001000	000110	011101	101011	010011	100101	111110	110000
010000	011110	000101	110011	001011	111101	100110	101000
100000	101110	110101	000011	111011	001101	010110	011000
001001	000111	011100	101010	010010	100100	111111	110001

由表 5.3.4 可以看到，用这种标准阵译码，需要把 2^n 个 n 重存储在译码器中。所以采用这种译码方法的译码器的复杂性随 n 指数增长，很不实用。简化查表的步骤需引入伴随式的概念。

视频

双语教学录像

5.3.2.2 伴随式译码

由于(n,k)码的任何一个码字 $\boldsymbol{C}$ 均满足式(5.3.9)或式(5.3.10)，可将接收矢量 $\boldsymbol{R}$ 用两式中的一个进行检验。若

$$\begin{aligned}\boldsymbol{R}\boldsymbol{H}^{\mathrm{T}}&=(\boldsymbol{C}+\boldsymbol{E})\boldsymbol{H}^{\mathrm{T}}=\boldsymbol{C}\boldsymbol{H}^{\mathrm{T}}+\boldsymbol{E}\boldsymbol{H}^{\mathrm{T}}\\&=\boldsymbol{E}\boldsymbol{H}^{\mathrm{T}}=\boldsymbol{0}\end{aligned}$$

则 $\boldsymbol{R}$ 满足校验关系，可认为它是一个码字；反之，则 $\boldsymbol{R}$ 有错。

定义 5.3.3 设(n,k)码的一致校验矩阵为 $\boldsymbol{H}$，$\boldsymbol{R}$ 是发送码字 $\boldsymbol{C}$ 的接收矢量，称

$$\boldsymbol{S}=\boldsymbol{R}\boldsymbol{H}^{\mathrm{T}}=\boldsymbol{E}\boldsymbol{H}^{\mathrm{T}} \tag{5.3.11}$$

为接收矢量 $\boldsymbol{R}$ 的**伴随式或校正子**。

显然，若 $\boldsymbol{E}=\boldsymbol{0}$，则 $\boldsymbol{S}=\boldsymbol{0}$，那么 $\boldsymbol{R}$ 就是 $\boldsymbol{C}$；若 $\boldsymbol{E}\neq\boldsymbol{0}$，则 $\boldsymbol{S}\neq\boldsymbol{0}$，如能从 $\boldsymbol{S}$ 得到 $\boldsymbol{E}$，则从 $\boldsymbol{C}=\boldsymbol{R}-\boldsymbol{E}$ 即可恢复发送的码字。可见，$\boldsymbol{S}$ 仅与 $\boldsymbol{E}$ 有关，它充分反映了信道干扰的情况，而与发送的是什么码字无关。

将(n,k)码的一致校验矩阵写成列矢量的形式：

$$\begin{aligned}\boldsymbol{H}&=[\boldsymbol{h}_{n-1}\quad\boldsymbol{h}_{n-2}\cdots\boldsymbol{h}_{n-i}\cdots\boldsymbol{h}_1\quad\boldsymbol{h}_0]\\&=\begin{bmatrix}h_{1,n-1}&h_{1,n-2}&\cdots&h_{10}\\h_{2,n-1}&h_{2,n-2}&\cdots&h_{20}\\\vdots&\vdots&\ddots&\vdots\\h_{r,n-1}&h_{r,n-2}&\cdots&h_{r0}\end{bmatrix}\end{aligned}$$

式中：$\boldsymbol{h}_{n-i}$ 对应 $\boldsymbol{H}$ 矩阵的第 i 列，它是一个 $r=n-k$ 重列矢量。

若码字传送发生 t 个错误，不失一般性，设码字的第 $i_1,i_2,\cdots,i_t$ 位有错误，则错误图样可表示成

$$\boldsymbol{E}=(0\cdots e_{i_1}\quad 0\cdots e_{i_2}\quad 0\cdots e_{i_t}\cdots 0)$$

那么伴随式

$$\boldsymbol{S}=\boldsymbol{E}\boldsymbol{H}^{\mathrm{T}}=[0\cdots e_{i_1}0\cdots e_{i_2}0\cdots e_{i_t}\cdots 0]\begin{pmatrix}\boldsymbol{h}_{n-1}\\\boldsymbol{h}_{n-2}\\\vdots\\\boldsymbol{h}_{n-i}\\\vdots\\\boldsymbol{h}_0\end{pmatrix}$$

$$=e_{i_1}\boldsymbol{h}_{n-i_1}+e_{i_2}\boldsymbol{h}_{n-i_2}+\cdots+e_{i_t}\boldsymbol{h}_{n-i_t} \tag{5.3.12}$$

这说明，$\boldsymbol{S}$ 是 $\boldsymbol{H}$ 矩阵中 $\boldsymbol{E}$ 不等于 0 的那几列 $\boldsymbol{h}_{n-i}$ 的线性组合。因为 $\boldsymbol{h}_{n-i}$ 是 r 重列矢量，所以 $\boldsymbol{S}$ 也是一个 r 重的矢量。当传输没有错误，即 $\boldsymbol{E}$ 的各位均为 0 时，$\boldsymbol{S}$ 是一个 r 重全零矢量。

注意，若 $\boldsymbol{E}$ 本身就是一个码字，即 $\boldsymbol{E}\in(n,k)$ 码，则此时计算 $\boldsymbol{S}$ 必须等于 0。此时的错误不能发现，也无法纠正，称为**不可检错误图样**。

伴随式在标准阵列中有如下性质。

定理 5.3.3 每个陪集全部 2^k 个矢量都有相同的伴随式，而不同陪集有不同的伴随式。

证明： 如果第 l 行陪集首为 $\boldsymbol{E}_l$（看成错误图样），那么第 l 行任意 n 重矢量的伴随式为

$$(\boldsymbol{E}_l+\boldsymbol{C}_i)\boldsymbol{H}^{\mathrm{T}}=\boldsymbol{S}_l \quad (i=2,3,\cdots,2^k)$$

$$\boldsymbol{S}_l=\boldsymbol{E}_l\boldsymbol{H}^{\mathrm{T}}+\boldsymbol{C}_i\boldsymbol{H}^{\mathrm{T}}=\boldsymbol{E}_l\boldsymbol{H}^{\mathrm{T}}$$

可见，l 陪集的伴随 $\boldsymbol{S}_l$ 与 $\boldsymbol{C}_i$ 无关，故同一陪集中的矢量其伴随式相同。若第 l 陪集和第 t 陪集（$l<t$）的伴随式是相同的，则有

$$\boldsymbol{S}_l=\boldsymbol{E}_l\boldsymbol{H}^{\mathrm{T}}$$

$$\boldsymbol{S}_t=\boldsymbol{E}_t\boldsymbol{H}^{\mathrm{T}}$$

$$\boldsymbol{S}_l+\boldsymbol{S}_t=(\boldsymbol{E}_l+\boldsymbol{E}_t)\boldsymbol{H}^{\mathrm{T}}=\boldsymbol{0}$$

式中：$\boldsymbol{E}_l+\boldsymbol{E}_t$ 是码字。

假如 $\boldsymbol{E}_l+\boldsymbol{E}_t=\boldsymbol{C}_i$，则有

$$\boldsymbol{E}_l=\boldsymbol{E}_t+\boldsymbol{C}_i$$

那么 $\boldsymbol{E}_l$ 在第 t 个陪集中。这个结论和标准阵列的构成原则矛盾，故不同陪集的伴随式不可能相同。 ［证毕］

可见，陪集首和伴随式有一一对应的关系，实际上这些陪集首也就代表着可纠正的错误图样。根据这一关系可把上述标准阵译码表进行简化，得到一个简化译码表。

例 5.3.4 如例 5.3.3 中的(6,3)码标准阵，可简化为表 5.3.5 所示的译码表。译码器收到 $\boldsymbol{R}$ 后，与 $\boldsymbol{H}$ 矩阵进行运算得到伴随式 $\boldsymbol{S}$，由 $\boldsymbol{S}$ 查表得到错误图样 $\hat{\boldsymbol{E}}$，从而译出码字 $\hat{\boldsymbol{C}}=\boldsymbol{R}-\hat{\boldsymbol{E}}$。因此，这种译码器中不必存储所有 2^n 个 n 重，而只存储错误图样 $\boldsymbol{E}$ 与 2^{n-k} 个 $(n-k)$ 重 $\boldsymbol{S}$。

表 5.3.5 (6,3)码简化译码表

错误图样	000000	100000	010000	001000	000100	000010	000001	001001
伴随式	000	011	101	110	100	010	001	111

综上所述，利用伴随式译码表译码的步骤如下。

（1）计算接收矢量 $\boldsymbol{R}$ 的伴随式 $\boldsymbol{S}=\boldsymbol{R}\boldsymbol{H}^{\mathrm{T}}$。

（2）根据计算出的伴随式找出对应的陪集首 $\hat{\boldsymbol{E}}$（这是根据 $\boldsymbol{S}$ 做出的 $\boldsymbol{E}$ 的估值）。

(3) 码字估值 $\hat{\boldsymbol{C}}=\boldsymbol{R}+\hat{\boldsymbol{E}}$ 被认为是发送码字，例如，设发送码字 $\boldsymbol{C}_4=(100011)$，接收矢量 $\boldsymbol{R}=(101011)$(实际错误图样为 $\boldsymbol{E}=(001000)$)，则 $\boldsymbol{R}$ 的伴随式为

$$\boldsymbol{S}=\boldsymbol{R}\boldsymbol{H}^{\mathrm{T}}=(101011)\begin{bmatrix}0&1&1&1&0&0\\1&0&1&0&1&0\\1&1&0&0&0&1\end{bmatrix}^{\mathrm{T}}=(110)$$

从表 5.3.5 中找到与 $\boldsymbol{S}=(110)$ 相对应的陪集首为(001000)，于是译码输出为

$$\boldsymbol{C}=(101011)+(001000)=(100011)$$

从表 5.3.4 可知，因为 $\hat{\boldsymbol{E}}$ 在陪集首中，所以译码正确。当信道错误图样 $\boldsymbol{E}=(101000)$时，同样的码字 $\boldsymbol{C}=(100011)$，经传输得 $\boldsymbol{R}=(001011)$，根据式(5.3.10)得 $\boldsymbol{S}=(101)$，再由表 5.3.5 求得 $\hat{\boldsymbol{E}}=(010000)$，这时认为 $\hat{\boldsymbol{C}}=(001011)+(010000)=(011011)\neq\boldsymbol{C}$，译码是错误的，原因是 $\boldsymbol{E}$ 不是标准阵列的陪集首。由表 5.3.4 可知，此矢量已在(010000)为陪集首的陪集中出现，不能作为陪集首，所以根据表 5.3.5 译码结果也是错的。原因是(6,3)码的 $d_{\min}=3$，它可纠正任何一位差错，而不能普遍地纠正两位差错。该码只能纠正一种有两位差错的信道错误图样，即

$$\boldsymbol{E}=(001001)$$

总之，线性码正确译码的充要条件为信道实际错误图样是标准阵列的陪集首，这个结论对任何线性分组码的译码方法都适用。

根据上述，线性分组码的译码器应如图 5.3.2 所示。

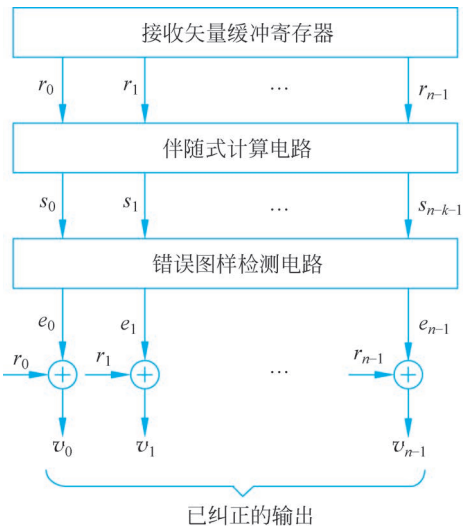


图 5.3.2　线性分组码的译码器

简化译码表虽然把译码器的存储容量降低了很多，但由于(n,k)分组码的 n、k 通常比较大，即使用这种简化译码表，译码器的复杂性还是很高的。例如，一个[100,70]分组码，共有 $2^{30}\approx10^9$ 个伴随式及错误图样，译码器要存储如此多的图样和 $n-k$ 重是不太可能的。因此，在线性分组码理论中，如何寻找简化译码器是最中心的研究课题之一。为

了寻找更加简单的比较实用的译码方法，仅有线性特性是不够的，还需要附加一些其他特性，如循环特性。这就是 5.4 节要介绍的循环码。

5.3.3 线性分组码的纠错能力

5.3.3.1 一致校验矩阵与纠错能力

由 5.1.2 节的介绍可知，从一个码的最小距离中得出该码的纠检错能力，线性分组码同样满足定理 5.1.2～定理 5.1.5 的结论。进一步考察线性分组码的一致校验矩阵，会发现它与纠错能力之间的关系。

定理 5.3.4 (n,k)线性分组码具有最小距离 d 的充分必要条件是 $\boldsymbol{H}$ 矩阵中任意 $d-1$ 列线性无关，至少有一种 d 列线性相关。

证明： 设一致校验矩阵为 $\boldsymbol{H}$，用 $\boldsymbol{h}_j(j=1,2,\cdots,n)$表示 $\boldsymbol{H}$ 矩阵中 n 个列矢量，即

$$\boldsymbol{H}=\begin{bmatrix} h_{1n-1} & h_{1n-2} & \cdots & h_{10} \\ \vdots & \vdots & \ddots & \vdots \\ h_{rn-1} & h_{rn-2} & \cdots & h_{r0} \end{bmatrix}_{r\times n}=[\boldsymbol{h}_{n-1} \quad \boldsymbol{h}_{n-2}\cdots\boldsymbol{h}_0]$$

则对于每个码字 $\boldsymbol{C}=(c_{n-1} \quad c_{n-2}\cdots c_0)$，满足校验方程

$$\boldsymbol{C}\boldsymbol{H}^{\mathrm{T}}=0$$

该方程可以写为

$$\sum_{i=0}^{n-1}c_i\boldsymbol{h}_i^{\mathrm{T}}=0$$

上式说明码字 $\boldsymbol{C}$ 中非零的码元位对应的 $\boldsymbol{H}$ 矩阵那些列的线性组合为 0。于是，若一个二元码字重量为 w，则 $\boldsymbol{H}$ 矩阵中与该码字中 w 个取值为“1”码元对应的列矢量线性相关。

一个二元线性码，如果它的最小汉明距离为 d，即该码的最小汉明重量为 d，则它的校验矩阵 $\boldsymbol{H}$ 中有 d 列矢量线性相关，而任意 $d-1$ 个列矢量是线性独立的；否则，就存在一个重量小于 d 的矢量 $\boldsymbol{C}'$，使 $\boldsymbol{C}'\boldsymbol{H}^{\mathrm{T}}=0$。这与码的最小重量为 d 相矛盾。

[证毕]

因为一致校验矩阵 $\boldsymbol{H}$ 与最小距离 d 有明确的关系，所以研究中往往以一致校验矩阵 $\boldsymbol{H}$ 定义一个线性分组码，LDPC 就是如此。

从式(5.3.12)中也可以看出，伴随式 $\boldsymbol{S}$ 是错误图样 $\boldsymbol{E}$ 中不为 0 的码元位所对应的 $\boldsymbol{H}$ 矩阵的列的线性组合。若一个(n,k)码要能纠正所有单个错误，则由所有单个错误的错误图样确定的 $\boldsymbol{S}$ 均不相同且不等于 0，这意味着，$\boldsymbol{H}$ 矩阵各列不为 0 且各不相同。那么，一个(n,k)码的 $\boldsymbol{H}$ 矩阵为怎样的形式才能纠正小于等于 t 个错误？这就必须要求小于或等于 t 个错误的所有可能组合的错误图样，都必须有不同的伴随式与之对应。因此，若有

$$\boldsymbol{E}_1=(0\cdots0 \quad e_{i_1}\cdots e_{i_2}\cdots e_{i_t}0\cdots0)\neq(0\cdots0 \quad e'_{i_1}\cdots e'_{i_2}\cdots e'_{i_t}0\cdots0)=\boldsymbol{E}_2$$

则要求

$$e_{i_1}\boldsymbol{h}_{n-i_1}+e_{i_2}\boldsymbol{h}_{n-i_2}+\cdots+e_{i_t}\boldsymbol{h}_{n-i_t}\neq e'_{i_1}\boldsymbol{h}'_{n-i_1}+e'_{i_2}\boldsymbol{h}'_{n-i_2}+\cdots+e'_{i_t}\boldsymbol{h}'_{n-i_t}$$

$$e_{i_1}\boldsymbol{h}_{n-i_1}+e_{i_2}\boldsymbol{h}_{n-i_2}+\cdots+e_{i_t}\boldsymbol{h}_{n-i_t}+e'_{i_1}\boldsymbol{h}'_{n-i_1}+e'_{i_2}\boldsymbol{h}'_{n-i_2}+\cdots+e'_{i_t}\boldsymbol{h}'_{n-i_t}\neq 0$$

这说明，(n,k)码要纠正小于或等于 t 个错误，其 $\boldsymbol{H}$ 矩阵中任意 $2t$ 列须线性无关。由此得如下定理。

定理 5.3.5 任一(n,k)线性分组码若要纠正小于或等于 t 个错误，其充要条件是 $\boldsymbol{H}$ 矩阵中任何 $2t$ 列线性无关。

定理 5.3.4 和定理 5.3.5 是构造任何类型线性分组码的基础，由此不难看出：

(1) 根据这些定理，由 $\boldsymbol{H}$ 列的相关性就可以直接知道码的纠错、检错能力。

(2) 为了构造最小距离 $d \geqslant e+1$(为检测不大于 e 个错误)或 $d \geqslant 2t+1$(为纠正不大于 t 个错误)的线性分组码，其充要条件是要求 $\boldsymbol{H}$ 中任意 $d-1$(或 $2t$)列线性无关。例如，要构造最小距离为 3 的码，则要求 $\boldsymbol{H}$ 任意 $3-1=2$ 列线性无关。对于二元域上的码，要求 $\boldsymbol{H}$ 当且仅当满足无相同的列和无全 0 的列，就可纠正所有单个错误。

(3) 因为交换 $\boldsymbol{H}$ 矩阵的各列不会影响码的最小距离，所以所有列矢量相同但排列位置不同的 $\boldsymbol{H}$ 矩阵所对应的分组码，在纠错能力和码率上是等价的。


视频

5.3.3.2 码纠错能力限

基于上述定理不难得出几个关于码的纠错能力界限的结论。

定理 5.3.6 **(Singleton 限)**任一线性分组码的最小距离(或最小重量)d_0 均满足

$$d_0 \leqslant n-k+1$$

证明： 任何一个线性码可以变换成等价的系统码，则该码中至少有一个码字重量不大于 $n-k+1$。因为这个码中至少存在一个码字，它的信息位仅有一个“1”，校验位最多可能有 $n-k$ 个“1”，所以这个码字的最小汉明重量不大于 $n-k+1$，因此这个线性码的最小汉明距离 $d_0 \leqslant n-k+1$。

[证毕]

满足 $d_0=n-k+1$ 的线性分组码称为**极大最小距离可分(Maximum Distance Separable，MDS)码**。在同样 n、k 下，由于 d_0 最大，因此纠错能力更强，所以设计这种码是编码理论中人们感兴趣的一个课题。

下面一个定理将告诉我们，在已知信息位 k 的条件下，如何去确定监督位 $r=n-k$，即确定码长，才能满足对纠错能力 t 的要求。这个问题在设计一个码组时是很重要的。

定理 5.3.7(汉明限) 若 $\boldsymbol{C}$ 是 k 维 n 重二元码，当已知 k 时，要使 $\boldsymbol{C}$ 能纠正 t 个错，则必须有不少于 r 个校验位，并且使 r 满足

$$2^r-1 \geqslant \sum_{i=1}^{t} C_n^i$$

证明： 假设 $\boldsymbol{C}$ 的一致校验矩阵为 $\boldsymbol{H}$，伴随式为

$$\boldsymbol{S}=\boldsymbol{R}\boldsymbol{H}^{\mathrm{T}}, \quad \boldsymbol{R} \in V_n(F_2)$$

$\boldsymbol{C}$ 是 $k \times n$ 的，则 $\boldsymbol{H}$ 是 $r=n-k$ 行 n 列，且在 $\boldsymbol{S}$ 的 2^r 种状态除去全零状态，其余还有 2^r-1 种非零状态。

另外，在一个 n 重矢量中，它们的错误图样也是 n 重。错误图样用矢量 $\boldsymbol{E}$ 代表，可计算 $W(\boldsymbol{E})=t$ 的错误图样个数，有

$$W(\boldsymbol{E})=1 \text{ 的个数为 } C_n^1$$

$$W(\boldsymbol{E})=2 \text{ 的个数为 } C_n^2$$
$$W(\boldsymbol{E})=3 \text{ 的个数为 } C_n^3$$
$$\vdots$$
$$W(\boldsymbol{E})=t \text{ 的个数为 } C_n^t$$

则错误码元不大于 t 个的错误图样共有 $\sum_{i=1}^{t} C_n^i$ 种。

若要能使 $\boldsymbol{C}$ 纠正 t 个错误，$\boldsymbol{S}$ 的状态数要大于错误图样总数，只有满足以上条件才能建立起伴随式与错误图样的一一对应关系。因此有

$$2^r-1 \geqslant \sum_{i=1}^{t} C_n^i \tag{5.3.13}$$

当满足 $2^r-1=\sum_{i=1}^{t} C_n^i$ 时，这样的码称为**完备码**。这种码的校验元得到了最充分的利用。而式(5.3.13)给出的界限称为**汉明限**，它也可改写为

$$n-k \geqslant \log_2\left(\sum_{i=1}^{t} C_n^i\right)$$

上式给出在已知 n、k 和 t 时所需要的监督位数。该式又称为 Hamming 不等式。

迄今为止，已找到的完备码有 Hamming 码，(23,12)非本原 BCH 码(又称 Golay 码)及三进制的(11,6)码。

5.3.3.3 示例：Hamming 码


中文教学录像


视频

前面曾多次提到汉明距离、汉明重量等术语，都是为了纪念对纠错编码作出杰出贡献的科学家汉明而命名的。Hamming 码的命名当然更直接，这种码是由汉明在 1950 年首先提出的。它有以下特征：

码　　长：　$n=2^m-1$
信息位数：　$k=2^m-m-1$
监督码位：　$r=n-k=m$
最小距离：　$d=3$
纠错能力：　$t=1$

这里 m 为大于或等于 2 的正整数，给定 m 后，即可构造出具体的(n,k)Hamming 码。这可以从建立一致校验矩阵着手。我们已经知道，$\boldsymbol{H}$ 矩阵的列数就是码长 n，行数等于 m。如 $m=3$，就可计算出 $n=7$，$k=4$，因而是(7,4)线性码。其 $\boldsymbol{H}$ 矩阵正是用 $2^r-1=7$ 个非零 3 重作列矢量构成的，如下所示：

$$\boldsymbol{H}=\begin{bmatrix}0&0&0&1&1&1&1\\0&1&1&0&0&1&1\\1&0&1&0&1&0&1\end{bmatrix}$$

这时 $\boldsymbol{H}$ 矩阵的对应列正好是十进制数 1～7 的二进制表示，对于纠 1 位差错来说，其伴随式的值就等于对应的 $\boldsymbol{H}$ 的列矢量，即错误位置。所以这种形式的 $\boldsymbol{H}$ 矩阵构成的码很便

于纠错，但这是非系统的(7，4)Hamming 码的一致校验矩阵。如果要得到系统码，可调整各列次序来实现：

$$\boldsymbol{H}_0=\begin{bmatrix}1&1&1&0&1&0&0\\0&1&1&1&0&1&0\\1&1&0&1&0&0&1\end{bmatrix}=[\boldsymbol{Q}\quad \boldsymbol{I}_3] \tag{5.3.14}$$

有了 $\boldsymbol{H}_0$，按照式(5.3.14)就可得到系统码的校验位，如其相应的生成矩阵为

$$\boldsymbol{G}_0=[\boldsymbol{I}_4\quad \boldsymbol{Q}^{\mathrm{T}}]=\begin{bmatrix}1&0&0&0&1&0&1\\0&1&0&0&1&1&1\\0&0&1&0&1&1&0\\0&0&0&1&0&1&1\end{bmatrix}$$

Hamming 码的译码方法如 5.3.2 节中所述，可以采用计算伴随式，然后确定错误图样并加以纠正的方法。

值得一提的是，(7，4)Hamming 码的 $\boldsymbol{H}$ 矩阵并非只有以上两种。原则上讲，(n,k) Hamming 码的一致校验矩阵有 n 列 m 行，它的 n 列分别由除了全 0 之外的 m 位码组构成，每个码组只在某列中出现一次。而 $\boldsymbol{H}$ 矩阵各列的次序是可变的。

容易证明，Hamming 码实际上是 $t=1$ 的完备码。

Hamming 码如果再加上一位对所有码元都进行校验的监督位，则监督码元由 m 增至 $m+1$，信息位不变，码长由 2^m-1 增至 2^m，通常把这种 $(2^m,2^m-1-m)$ 码称为**扩展 Hamming 码**。扩展 Hamming 码的最小码距增加为 4，能纠正 1 位错误同时检测 2 位错误，简称纠 1 检 2 错码。例如，(7，4)Hamming 码可变成(8，4)扩展 Hamming 码(又称增余 Hamming 码)。(8，4)码的 $\boldsymbol{H}$ 矩阵如下：

$$\boldsymbol{H}_{(8,4)}=\begin{bmatrix}1&1&1&1&1&1&1&1\\1&1&1&0&1&0&0&0\\0&1&1&1&0&1&0&0\\1&1&0&1&0&0&1&0\end{bmatrix}$$

它的第一行为全 1 行，最后一列的列矢量为 $[1000]^{\mathrm{T}}$，它的作用是使第 8 位成为偶校验位，而前 7 位码元同(7，4)码。这种 $\boldsymbol{H}$ 矩阵任何 3 列都是线性独立的，而只有 4 列才能线性相关，按照定理 5.3.4 可知它的 $d_{\min}=4$，可实现纠 1 检 2 错。

5.3.4 线性分组码的派生与组合

5.3.4.1 由一个已知码的派生

视频

视频

人们往往需要从一个已知的(n,k)线性码出发来构造一个新的线性分组码，使得某些参数能符合实际的需要。5.3.1.2 节中介绍的对偶码就是其中的一种方法。下面再介绍几种对已知码的 $\boldsymbol{G}$ 或 $\boldsymbol{H}$ 矩阵进行适当修正和组合，以构造新码的方法。这些方法虽然很简单，但很实用，而且也是以后构造各种复合码的基础。

1. *扩展码*

设 $\boldsymbol{C}$ 是一个(n,k,d)线性分组码，它的码字有奇数重量也有偶数重量。若对每个码

字 $\boldsymbol{C}=(c_{n-1},c_{n-2}\cdots c_1,c_0)$ 增加一个全校验位 c'_0，满足以下校验关系：

$$c_{n-1}+\cdots+c_0+c'_0=0$$

这样得到的新码 $\boldsymbol{C}'=[\boldsymbol{C}|c'_0]$，是一个 $(n+1,k)$ 线性码。若原来的一致校验矩阵为 $\boldsymbol{H}$，则 $\boldsymbol{C}'$ 的一致校验矩阵为

$$\boldsymbol{H}'=\begin{bmatrix}1\cdots1 & 1\\ & 0\\ \boldsymbol{H} & \vdots\\ & 0\end{bmatrix}$$

由 $\boldsymbol{H}$ 矩阵列的相关性与码的最小距离的关系可得，若原码 $\boldsymbol{C}$ 的最小距离 d 是奇数，则 $\boldsymbol{C}'$ 的最小距离是偶数 $d+1$。例如，前面介绍的 $(8,4)$ 扩展 Hamming 码即是由 $(7,4)$ Hamming 码扩展得到。

2. 凿孔码(删余码)

与扩展码(增加一位校验位)相对应的是凿孔码，它把 (n,k) 线性分组码 $\boldsymbol{C}$ 中码字的一些校验位删除(或者说进行凿孔)，从而得到一个新的线性码 $\boldsymbol{C}'$，这时码字长度会减少，但信息位数目不变，因此码率会更高。

凿孔码一般是在设计好一个线性分组码后，希望提升码率时采取的派生方法，在卷积码和 Turbo 码中也经常使用(一般会设计一个凿孔图案)。其中一个典型情况是**删余码**，即在原码基础上删除一位校验元。删余码的最小距离可能比原码小 1。

3. 除删码(增余删信码)

在原码的基础上增加一些校验元，删去一些信息元，保持码长不改变，但明显码率会降低。一种典型的增余删信码是**除删码**，它是在原 (n,k) 码的基础上增加一位全校验元，删去一位信息元，得到 $(n,k-1)$ 线性分组码，其一致校验矩阵 $\boldsymbol{H}'$ 是在原码的 $\boldsymbol{H}$ 矩阵下加一个全 1 行。因为有全 1 行，意味着所有码字的重量皆为偶数。这种除删码过程是在原码中挑选所有偶数重量的码字组成的新码，码字数目将少了一半。如果原来码字最小距离为奇数，则新的除删码的最小距离将增加 1，成为偶数。

4. 增广码(增信删余码)

增广码 $\boldsymbol{C}^a$ 是在原码 $\boldsymbol{C}$ 的基础上，增加一个信息元，删去一个校验元得到的。因此，新码的码长与原码相同，但信息位长度加 1。若二元 (n,k,d) 线性码中不包含分量为全“1”的码字，则可以把这个全“1”矢量 $\mathbf{1}$ 加入码 $\boldsymbol{C}$ 中，同时把全 1 码字与 $\boldsymbol{C}$ 中每个码字之和添加到原来码中，这样就构成了一个增广码 $\boldsymbol{C}^a$，即 $\boldsymbol{C}^a=\boldsymbol{C}\cup(\boldsymbol{I}+\boldsymbol{C})$，则其生成矩阵之间的关系为

$$\boldsymbol{G}^{(a)}=\begin{bmatrix}1\cdots1\\ \boldsymbol{G}\end{bmatrix}$$

可见，$\boldsymbol{C}^a$ 是一个 $(n,k+1,d^a)$ 的线性分组码，其中

$$d^a=\min\{d,n-\max(W(\boldsymbol{C}_i))\},\quad \boldsymbol{C}_i\in\boldsymbol{C}$$

于是，增广码的最小距离一般是减小的。

5. 缩短码

在某些情况下，已设计好的编码不能满足实际应用对码长的要求，可以对原码进行缩短。缩短码去除了原来线性分组码中前 i 个信息位，具体做法是把原来(n,k)线性分组码 $\boldsymbol{C}$ 中前 i 位$(c_{n-1}\sim c_{n-i})$为 0 的码字选取出来，再把这 i 个信息位删除掉，组成一个新的子集，这样构成的是一个$(n-i,k-i)$缩短码。由于缩短码是 k 维空间 $\boldsymbol{V}_{n,k}$(码 $\boldsymbol{C}$ 的全体码字集合)中取前 i 位均为 0 的码字组成的一个子集，显然该子集是 $\boldsymbol{V}_{n,k}$ 空间中的一个 $k-i$ 维的子空间 $\boldsymbol{V}_{n,k-i}$。因为线性分组码的 $\boldsymbol{H}$ 矩阵列对应码字每一位码元的校验约束关系，因此缩短码的一致校验矩阵 $\boldsymbol{H}'$是删除原码的 $\boldsymbol{H}$ 矩阵前 i 列，而其生成矩阵 $\boldsymbol{G}'$则是去掉原码生成矩阵 $\boldsymbol{G}$ 中前 i 行前 i 列，可见缩短码的最小距离不会比原码的小，但码率会有所降低，即新码的码率

$$\boldsymbol{R}'=\frac{k-i}{n-i}<\frac{k}{n}$$

6. 延长码(增信码)

与缩短码相对应的是延长码。首先在原(n,k)线性码 $\boldsymbol{C}$ 的基础上通过增加一个全 **1** 码字来增广这个码集，然后增加一个全校验位来扩展它，这样构成一个$(n+1,k+1)$分组码。

图 5.3.3 中以 Hamming 码$(2^m-1,2^m-1-m,3)$为例说明构成新码的 6 种方法及相互关系。

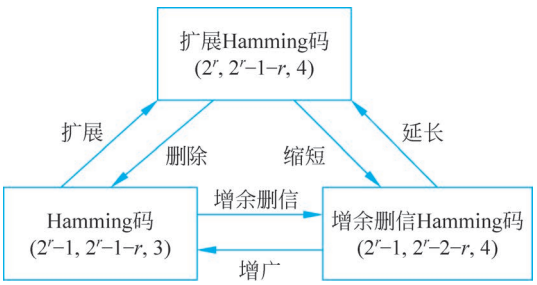


图 5.3.3　Hamming 码的各类派生码之间的关系图

5.3.4.2　乘积码

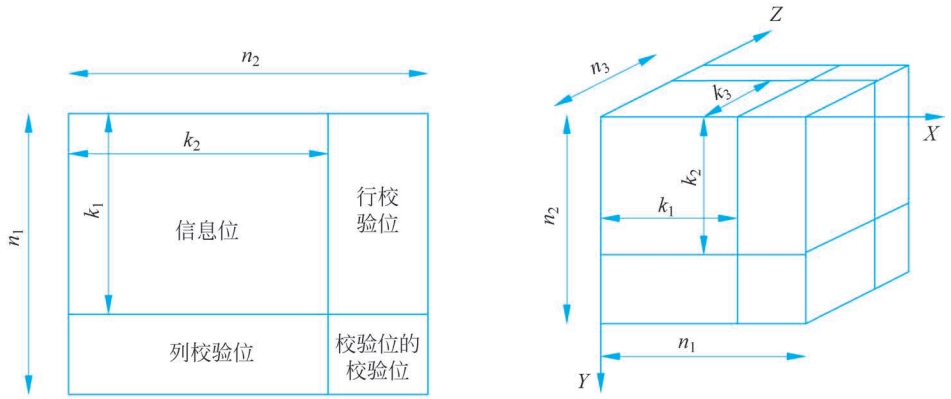
1994 年，Pyndiah 提出 Turbo 乘积码(Turbo Product Code，TPC)及其单输入单输出(SISO)迭代译码方案，该码可以做到高码率、近信道容量时仍可保持较好性能。另外，TPC 对于带宽应用是很具吸引力的，与串行 TCM-RS 码比较，TPC 能得到 0.85dB 的提高；另外，TPC 可以很容易地将最小距离保持在 16、36 或更大，从而避免“错误平层”效应；此外，TPC 的译码时延也可以通过将多个子译码器并行而大幅度减小。

由于 TPC 编码的优异性能，它在传统的国际卫星通信和区域卫星通信系统中有着广泛的应用前景。事实证明，TPC 技术功能强大且具有灵活性，可为各行业的用户和卫星运营商带来明显的效益。另外，TPC 增强的编码增益和带宽效率可以明显降低转发器成本，这一技术也可用来解决许多其他问题，如特小天线的通量密度降低问题。它同时

可应用于微波、高清电视(HDTV)、光纤、数字视频广播(DVB)等。2001 年 9 月 IEEE 公布了宽带无线接入系统的空中无线接口草案 802.16 中 TPC 被推荐为信道编码方式。

1. TPC 的构造

TPC 用两个或两个以上的分组码构造较长的码,从而提高码的性能。以二维 TPC 为例,假设有两个线性分组码分别为 $\boldsymbol{C}_1(n_1,k_1,d_1)$、$\boldsymbol{C}_2(n_2,k_2,d_2)$,其中,$n_i$、$k_i$、$d_i(i=1,2)$分别代表码长、信息位长、最小汉明距离。信息位以 $k_1\times k_2$ 的形式放入行列阵列中,先用码 $\boldsymbol{C}_2(n_2,k_2,d_2)$ 对每行进行编码(共 k_1 行),再用码 $\boldsymbol{C}_1(n_1,k_1,d_1)$对每列进行编码(共 n_2 列),如图 5.3.4(a)所示。此二维 TPC 可写成$(n_1,k_1,d_1)\times(n_2,k_2,d_2)$,其码率为$(k_1\times k_2)/(n_1\times n_2)$。



(a) 二维TPC　　(b) 三维TPC

图 5.3.4　TPC 编码结构图

同样的方法可构造三维立体 TPC。如图 5.3.4(b)所示,首先构造 k_3 个二维 TPC$(n_1,k_1,d_1)\times(n_2,k_2,d_2)$,然后在 Z 方向上进行 $\boldsymbol{C}_3(n_3,k_3,d_3)$线性分组编码,这就构成了三维立体 TPC$(n_1,k_1,d_1)\times(n_2,k_2,d_2)\times(n_3,k_3,d_3)$。其他多维的 TPC 也以类似的步骤来构造。对于多维 TPC(假设为 m 维,$m\geqslant 2$),其最小距离和码率分别为

$$d=d_1\times d_2\times d_3\times\cdots\times d_m$$

$$R=\frac{k_1}{n_1}\times\frac{k_2}{n_2}\times\frac{k_3}{n_3}\times\cdots\times\frac{k_m}{n_m}$$

作为 TPC 各维上的分量码通常为 Hamming 码、扩展 Hamming 码、奇偶校验码、BCH 码和扩展 BCH 码,RS 码和扩展 RS 码等线性分组码。目前国际上多采用前三种码型,其编译码相对较简单。TPC 除分量码码型可变外,还可以进行一定的变型:先对各维信息位缩短后再编码,可以得到缩短型 TPC;在对角线上加入奇偶校验形成“超轴”,可以得到增强型 TPC,这为 TPC 提供了更多的灵活选择。在 TPC 中各维上分量码的不同,可得到不同的编码码率,这对 IEEE 802.16 宽带无线接入网中码率自适应是非常重要的。

2. TPC 迭代译码算法

若采用最大似然译码方法,性能可以达到最优,但随着码长的增加,复杂度指数倍增长。在性能和复杂性两方面综合考虑,可采用基于 Chase2 的次优译码算法。

算法描述如下。

假设发送端码字 $\boldsymbol{E}=(e_1,\cdots,e_i,\cdots,e_n)$，$e_i\in\{+1,-1\}$，经过高斯白噪声信道(AWGN)，信道样值序列 $\boldsymbol{G}=(g_1,\cdots,g_i,\cdots,g_n)$，标准方差为$\boldsymbol{\sigma}$；接收端序列 $\boldsymbol{R}=\boldsymbol{E}+\boldsymbol{G}$，其中 $\boldsymbol{R}=(r_1,\cdots,r_i,\cdots,r_n)$。先对 $\boldsymbol{R}$ 进行硬判决，得到序列 $\boldsymbol{Y}=(y_1,\cdots,y_i,\cdots,y_n)$，$y_i=0.5(1+\mathrm{syn}(r_i))$，$y_i\in\{0,1\}$，再进行三步 chase2 译码算法。

步骤一：利用$|r_i|$值确定 $\boldsymbol{Y}$ 序列中可信度最小的 $p=4$ 个位置。

步骤二：在最小可信度位置上分别取“0”“1”，其他位置取“0”，可得到 $q=2^p$ 个测试图样 T^q。

步骤三：生成测试序列 Z^q，$Z^q=\boldsymbol{Y}\oplus T^q$。对于 Z^q 进行代数译码得到码集 C^q。

随后采用欧几里得距离最小原则在码集 C^q 中寻找最佳码字 $\boldsymbol{D}$，并同时寻找竞争码字 $\boldsymbol{C}$(仅当前位与 $\boldsymbol{D}$ 不同)。按下式计算外信息参与下次迭代：

$$w_j=\left(\frac{|\boldsymbol{R}-\boldsymbol{C}|^2-|\boldsymbol{R}-\boldsymbol{D}|^2}{4}\right)d_j-r_j \tag{5.3.15}$$

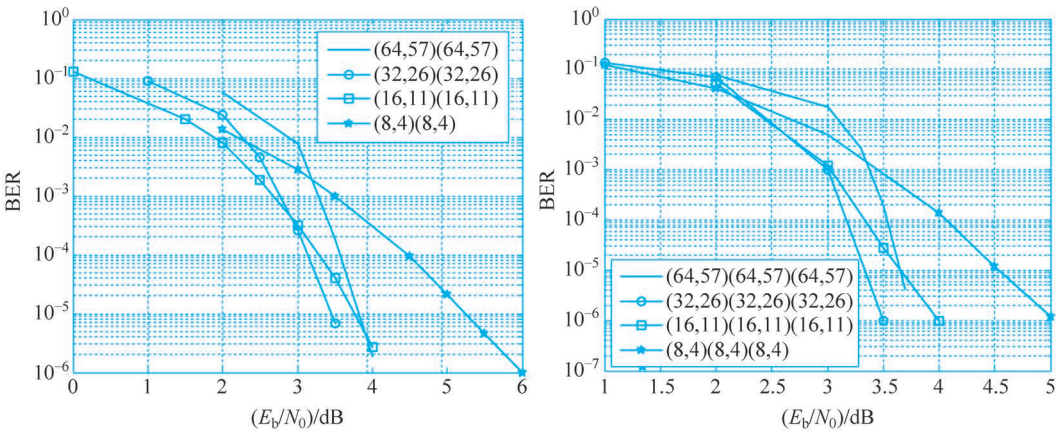
式中：$d_j(j=1,2,\cdots,n,d_j\in D)$为最佳码字的元素，$w_j$ 为外信息。

若竞争码字无法寻找到，则使用可信度因子 β 来近似计算，即

$$w_j=\beta d_j-r_j$$

3. TPC 的性能分析

TPC 采用基于 Chase2 的次优译码算法，性能与码率、码型、信噪比、译码迭代次数有关。一般来说，迭代次数越大，性能越好，但译码时间越长。当迭代次数大于一定的值(大于 6)时，性能改善不明显。下面以扩展 Hamming 码为分量码，迭代次数为 6 次，BPSK 调制，AWGN 下进行计算机仿真。



(a) 二维　　(b) 三维

图 5.3.5　TPC 性能图

从图 5.3.5(a)中可以看出，(32,26)(32,26)码性能优势明显，在 BER$=10^{-5}$，与(8,4)(8,4)比较，编码增益多出近 2dB；与(16,11)(16,11)和(64,57)(64,57)码比较，多出约 0.5dB。从趋势上看，(32,26)(32,26)码具有最为陡峭的“瀑布区”。图 5.3.5(b)的三维码性能曲线中“瀑布区”较二维更为陡峭。

不同调制方式下性能比较也有差异，图 5.3.6 给出(32,26)(32,26)码在不同调制方式下的性能曲线。

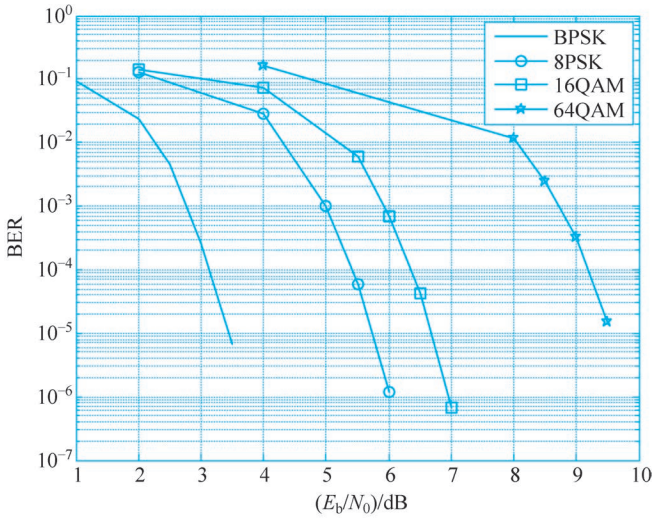


图 5.3.6 (32,26)(32,26)码不同调制方式下的性能

在 IEEE 802.16 标准中采用的码型如表 5.3.6 所示。

表 5.3.6 IEEE 802.16 标准中的 TPC

TPC	码率	有效载荷大小
(39,32)(39,32)(下行链路)	0.673	1024bit
(53,46)(51,44)(下行链路)	0.749	3136bit
(30,24)(25,9)(上行链路)	0.608	456bit

目前，TPC 技术相对较为成熟，一些芯片公司已有相应的芯片产品，这就进一步推动了其应用范围的扩展。在学术上，TPC 技术的研究主要集中在其应用领域的革新。

5.3.4.3 级联码

信道编码定理指出，随着码长 n 的增加，译码器错误概率按指数接近 0。因此，性能较好的码码长一般较长。但是，随着码长的增加，在一个码组中要求纠错的数目相应增加，译码器的复杂性和计算量也相应增加以致难以实现。为了解决性能与设备复杂性的矛盾，1966 年 Forney 提出了级联码的概念，把编制长码的过程分几级完成，通常分两级。

一种典型的方式如图 5.3.7 所示，假定在内信道上使用是码 $\boldsymbol{C}_1$，称为内码，在外信道上使用的码 $\boldsymbol{C}_2$，称为外码。所以这种码是(n_1,n_2,k_1,k_2)码，其中 n_1、k_1 和 n_2、k_2 分别是码 $\boldsymbol{C}_1$ 和 $\boldsymbol{C}_2$ 的长度和信息位数。而且，一般 $\boldsymbol{C}_2$ 采用(n_2,k_2)的多进制码，而 $\boldsymbol{C}_1$ 采用(n_1,k_1)的二元码。若内码和外码的最小距离分别为 d_1 和 d_2，则它们级联后的最小距离至少为 d_1d_2。这种单级级联码已广泛应用于通信和数据存储系统中。为获得较高的可靠性并减小译码复杂度，内码一般较短，使用软判决译码算法进行译码；非二进制的外码一般较长，使用代数译码方法进行译码。编码时，先将 k_2k_1 个信息数字分成 k_2 个 k_1 重，这 k_2 个 k_1 重按 $\boldsymbol{C}_1$ 码进行编码，将每个 k_1 重转换成 n_1 重。译码时，先对 $\boldsymbol{C}_1$ 码译

码，再对 $\boldsymbol{C}_2$ 码译码。这种码如果遇上少量的随机错误，内码 $\boldsymbol{C}_1$ 就可以纠正；如果遇到较长的突发错误，内码则无能为力，则由纠密集型突发错误很强的外码纠正。另一种做法是，内码作检错码，外码作纠错码，使译码过程简化。若还要提高纠正突发错误能力，则可将交织技术用于级联码。

图 5.3.7　采用级联码的通信系统

将内码编码器、信道与内码译码器的组合称为超信道。将外码与内码编码器的组合称为超编码器，将外与内译码器的组合称为超译码器。可以看到，所得级联码码字的总长度 $N=n_1n_2$ bit；其编码效率 $\eta_r=\eta_1\eta_2=k_1k_2/n_1n_2$。虽然，码字的总长度为 N，但由级联概念所提出的结构可以分别用两个长度为 n_1 和 n_2 的译码器来完成译码运算。故在相同的总差错率下这种方法比采用单级编码时所需的设备复杂程度要少得多。选用各种 RS 码作外码是最为适宜的，因为它们是极大最小距离码（$d=n-k+1$），并易于实现。在二级编码方案中 RS 码是级联码中外码 $\boldsymbol{C}_2$ 所常用的，而作为内码 $\boldsymbol{C}_1$ 可以采用不同的线性分组码，如正交码、循环码等，当然也可以采用卷积码作为内码。为了进一步提高抗随机错误和突发错误的能力，还可以采用多级编码方案，同时交织码也可以结合到具体的多级编码方案中。多级编码方案的缺点是编码器相当复杂，同时增长了译码时延，在某些场合并不适合。

美国航空航天局（NASA）的跟踪和数据中继卫星系统（Tracking Data Relay Satellite System，TDRSS）中使用的差错控制编码方案采用级联卷积码编码系统。在该系统中，GF(2^8)上的(255，233，33)RS 码被用作外码，由多项式 $g_1(D)=1+D+D^3+D^4+D^6$ 和 $g_2(D)=1+D^3+D^4+D^5+D^6$ 生成的状态数为 64、编码效率为 1/2 的卷积码被用作内码。卷积内码的自由距离 $d_{free}=10$。系统的整体编码效率为 0.437。该级联卷积编码系统性能如图 5.3.8 所示。从图 5.3.8 中可以看出，该级联卷积编码系统在信噪比 2.53dB 处，误码率达到 10^{-6}，可获得近 8.5dB 增益。

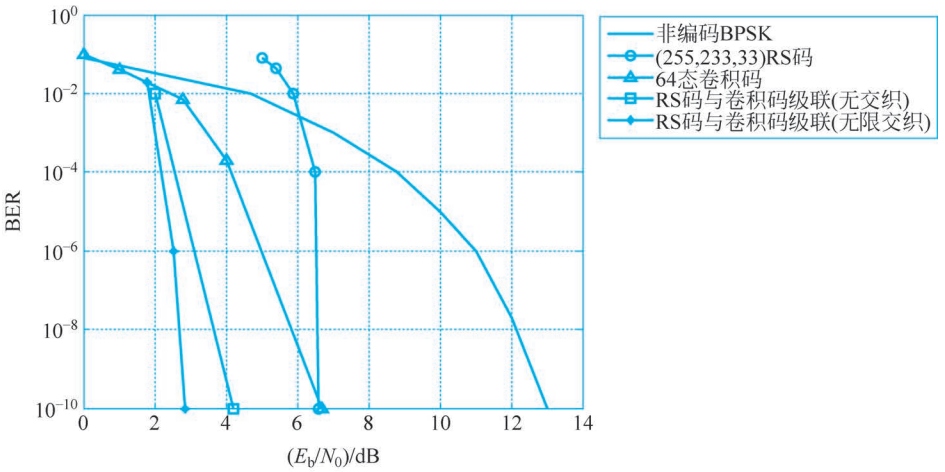


图 5.3.8　NASA 的 TDRSS 中使用的差错控制编码方案的误码性能

同样，在微小卫星的链路中，采用国际空间数据系统咨询委员会(Consultative Committee on Space Data Systems，CCSDS)推荐的级联码(卷积码＋RS 码)，使用级联码是为了以少于单个编码操作所需的整体实现的复杂度，来获得较低的错误概率。

5.4 循环码的基本原理

循环码是线性分组码的一个重要子类，也是目前研究得最成熟的一类码。它有许多特殊的代数结构，这些性质有助于按照所要求的纠错能力系统地构造这类码，并且简化译码方法。循环码还有易于实现的特点，很容易用带反馈的移位寄存器实现，且性能较好，不但可用于纠正独立的随机错误，而且可以用于纠正突发错误。因此，目前在实际差错控制系统中所用的线性分组码大部分是循环码。

5.4.1 基本概念

5.4.1.1 循环码的定义

什么是循环码，它究竟与一般的(n,k)线性分组码有何不同？我们先看一个例子。

例 5.4.1 (7,4)Hamming 码 $\boldsymbol{C}$ 的生成矩阵为

$$\boldsymbol{G}=\begin{bmatrix}1&0&0&0&1&0&1\\0&1&0&0&1&1&1\\0&0&1&0&1&1&0\\0&0&0&1&0&1&1\end{bmatrix}$$

由此可得到所有的 $2^4=16$ 个码字是(1000101)、(0001011)、(0010110)、(0101100)、(1011000)、(0110001)、(1100010)、(0100111)、(1001110)、(0011101)、(0111010)、(1110100)、(1101001)、(1010011)、(1111111)、(0000000)。

由这些码字看出，如果 $\boldsymbol{C}_i$ 是 $\boldsymbol{C}$ 的码字，则它向左(或右)循环移位一次所得到的，也是 $\boldsymbol{C}$ 的码字。具有这种循环移位特性的线性分组码称为循环码。

把 $\boldsymbol{C}$ 码的任一码字中的 7 个码元排成一个圆环，如图 5.4.1(a)～(d)所示。可以看到，从圆环的任一码元开始，按顺时针方向移动，得到的 7 重数组都是该码的一个码字，这就是循环码名字的由来。

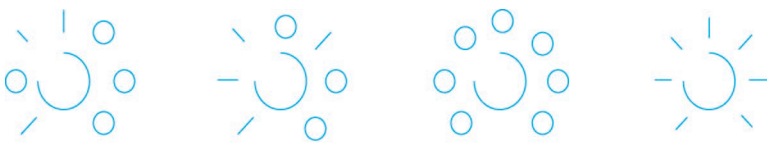


图 5.4.1 (7,4)循环码的码字循环移位示意图

(n,k)线性分组码是 n 维线性空间 $\boldsymbol{V}_n$ 中的一个 k 维子空间 $\boldsymbol{V}_{n,k}$。在循环码中，该子空间中的元素(n 重)具有循环移位特性，称为**循环子空间**。

定义 5.4.1 在任一个 GF(q)(q 为素数或素数幂)上的 n 维线性空间 $\boldsymbol{V}_n$ 中，一个 n 重子空间 $\boldsymbol{V}_{n,k}\in\boldsymbol{V}_n$，若对任何一个 $\boldsymbol{C}_i=(c_{n-1},c_{n-2},\cdots,c_0)\in\boldsymbol{V}_{n,k}$，恒有 $\boldsymbol{C}'_i=(c_{n-2}\cdots$

$c_0 c_{n-1}) \in \boldsymbol{V}_{n,k}$，则称 $\boldsymbol{V}_{n,k}$ 是**循环子空间**或**循环码**。

可见 GF(q)上的循环码是具有循环移位特性的线性分组码。

5.4.1.2 循环码的多项式描述

为了用代数理论研究循环码，可将码组用多项式来表示，称为码多项式。设许用码组

$$\boldsymbol{C}=(c_{n-1}\ c_{n-2}\cdots c_1\ c_0)$$

对应的码多项式可表示为

$$C(x)=c_{n-1}\ x^{n-1}+c_{n-2}\ x^{n-2}+\cdots+c_1\ x+c_0 \tag{5.4.1}$$

其中 $c_i \in \mathrm{GF}(2)$，则它们之间建立了一一对应关系，上述多项式也称为码字多项式，其中多项式的系数就是码字各分量的值，x 为一个任意实变量，其幂次 i 代表该分量所在位置。

由循环码特性可知，若 $\boldsymbol{C}=(c_{n-1}c_{n-2}\cdots c_1\ c_0)$是循环码的一个码字，则 $\boldsymbol{C}^{(1)}=(c_{n-2}\cdots c_0\ c_{n-1})$也是该循环码的一个码字，它的码多项式为

$$C^{(1)}(x)=c_{n-2}\ x^{n-1}+\cdots+c_0\ x+c_{n-1}$$

与式(5.4.1)比较可知

$$C^{(1)}(x)\equiv x\ C(x)\ \bmod(x^n+1)$$

同样，$xC^{(1)}(x)$对应的码字 $\boldsymbol{C}^{(2)}$ 相当于将码字 $\boldsymbol{C}^{(1)}$ 左移一位，即码字 $\boldsymbol{C}$ 左移两位，由此可得

$$\begin{aligned}C^{(2)}(x)&\equiv c_{n-3}\ x^{n-1}+\cdots+c_0\ x^2+c_{n-1}\ x+c_{n-2}\\&\equiv x\ C^{(1)}(x)\ \bmod(x^n+1)\\&\equiv x^2\ C(x)\ \bmod(x^n+1)\end{aligned}$$

以此类推，不难得出循环左移 i 位时，有

$$C^{(i)}(x)\equiv x^i\ C(x)\ \bmod(x^n+1)\quad(i=0,1,\cdots,n-1)$$

可见，$x^iC(x)$在模 x^n+1 下的余式对应着将码字 $\boldsymbol{C}$ 左移 i 位的码字 $\boldsymbol{C}^{(i)}$。

定理 5.4.1 若 $C(x)$是 n 长循环码中的一个码多项式，则 $x^iC(x)$按模 x^n+1 运算的余式必为循环码中另一码多项式。

为简便起见，上述中的 $\bmod(x^n+1)$在码多项式的表示中不一定写出，而通常用类似式(5.4.1)表示。

5.4.1.3 生成多项式

观察循环码的所有码多项式不难发现，除全 0 码外，它存在着一个特殊的多项式，这个多项式在循环码的构成中具有十分重要的意义，它就是该码的最低次多项式。以下几个定理是关于它的特性的。

定理 5.4.2 一个二进制中(n,k)循环码中有唯一的非零最低次多项式 $g(x)$，且其常数项为 1。

证明： 设 $g(x)$是码中次数最低的非零码多项式，具有如下形式：

$$g(x)=x^r+g_{r-1}\ x^{r-1}+\cdots+g_1\ x+g_0$$

若 $g(x)$不唯一，则必存在另一个次数最低的码多项式，例如 $g'(x)=x^r+g'_{r-1}x^{r-1}+\cdots+g'_1x+g'_0$，因为循环码是线性分组码，所以 $g(x)+g'(x)=(g_{r-1}+g'_{r-1})x^{r-1}+\cdots+(g_1+g'_1)x+(g_0+g'_0)$是一个次数小于 r 的码多项式。若 $g(x)+g'(x)\neq 0$，则 $g(x)+g'(x)$是一个次数小于最低次数 r 的非零码多项式，这显然与 r 是最低次数相矛盾。因此，必有 $g(x)+g'(x)=0$，即 $g(x)=g'(x)$，也即 $g(x)$是唯一的。

再证 $g_0=1$。

若 $g_0=0$，则有 $g(x)=x^r+g_{r-1}x^{r-1}+\cdots+g_2x^2+g_1x=x(x^{r-1}+g_{r-1}x^{r-2}+\cdots+g_2x+g_1)$，因为 $g(x)$是码多项式，则将其对应的码字右移一位后，得到一非零码多项式

$$x^{r-1}+g_{r-1}x^{r-2}+\cdots+g_2x+g_1$$

也是循环码的码多项式，而它的次数小于 r，这与 $g(x)$是次数最低的非零码多项式的假设相矛盾，故 $g_0=1$。 [证毕]

上例(7,4)循环码，只有一个最低次多项式 x^3+x+1，而码中所有码多项式都是它的倍式，即由 x^3+x+1 可生成所有(7,4)循环码，把它称为(7,4)循环码的生成多项式。

定义 5.4.2 若一个码的所有码多项式都是多项式 $g(x)$的倍式，则称 $g(x)$生成该码，且称 $g(x)$为该码的生成多项式，所对应的码字称为**生成子**或**生成子序列**。

定理 5.4.3 GF(2)上的(n,k)循环码中，存在有唯一的 $n-k$ 次首 1 多项式

$$g(x)=x^{n-k}+g_{n-k-1}x^{n-k-1}+\cdots+g_1x+g_0$$

使得每一码多项式 $C(x)$都是 $g(x)$的倍式，且每一小于或等于 $n-1$ 次的 $g(x)$的倍式一定是码多项式。

下面定理给出了循环码的生成多项式 $g(x)$应满足的条件。

定理 5.4.4 设 $g(x)$是(n,k)循环码$[C(x)]$中的一个次数最低的多项式$(g(x)\neq 0)$，则该循环码由 $g(x)$生成，并且 $g(x)\mid(x^n+1)$。

综上所述，可得出生成多项式 $g(x)$的性质：$g(x)$是循环码的码多项式中的一个唯一的最低次多项式，它具有首 1 末 1 的形式。该码集中任一码多项式都是它的倍式，它本身必是多项式 x^n+1 的一个$(n-k)$次的因式，由它可生成 2^k 个码字的循环码。

从以上讨论中，可得到以下重要结论。

(1) 在二元域 GF(2)上找一个(n,k)循环码，就是找一个能除尽 x^n+1 的 $n-k$ 次首 1 多项式 $g(x)$，为了寻找生成多项式，必须对 x^n+1 进行因式分解，这可用计算机来完成。

对于某些 n 值 x^n+1 只有很少的几个因式，因而码长为 n 的循环码不多。仅对于很少的几个 n 值才有较多的因式，在一些参考书上已将因式分解列成表格，有兴趣的读者可查阅有关书籍。

(2) 若 $C(x)$是(n,k)码的一个码多项式，则 $g(x)$一定除尽 $C(x)$。若 $g(x)\mid C(x)$，则次数小于或等于 $n-1$ 的 $C(x)$必是码的码多项式。也就是说若 $C(x)$是码多项式，则

$$C(x)\equiv 0 \mod g(x)$$

上述所有结论虽然都在 GF(2)上讨论的，但可以推广到 GF(q)上。

例 5.4.2 GF(2)上多项式 $x^7+1=(x+1)(x^3+x+1)(x^3+x^2+1)$，构造一个(7,3)循环码。

要构造一个(7,3)循环码,就是在 x^7+1 中找一个 $n-k=4$ 次的因式 $g(x)$,作为码的生成多项式,由它的一切倍式就组成了(7,3)循环码。若选 $g(x)=(x^3+x+1)(x+1)=x^4+x^3+x^2+1$,则(7,3)循环码的码多项式与码字列于表 5.4.1 中。由该表可知,该码的 8 个码字可由 $g(x)$、$xg(x)$、$x^2g(x)$的线性组合产生出来,而且这三个码多项式是线性无关的,它们构成一组基底。所以生成的循环子空间(循环码)是一个三维子空间 $\boldsymbol{V}_{7,3}$,对应一个(7,3)循环码。

表 5.4.1　$g(x)=x^4+x^3+x^2+1$ 生成的(7,3)循环码

码 多 项 式	码　　字
$g(x)=x^4+x^3+x^2+1$	(0011101)
$xg(x)=x^5+x^4+x^3+x$	(0111010)
$x^2g(x)=x^6+x^5+x^4+x^2$	(1110100)
$(1+x^2)g(x)=x^6+x^5+x^3+1$	(1101001)
$(1+x+x^2)g(x)=x^6+x^4+x+1$	(1010011)
$(1+x)g(x)=x^5+x^2+x+1$	(0100111)
$(x+x^2)g(x)=x^6+x^3+x^2+x$	(1001110)
$0g(x)=0$	(0000000)

在 $x^7+1=(x+1)(x^3+x+1)(x^3+x^2+1)$中,若选 $g(x)=(x+1)(x^3+x^2+1)=x^4+x^2+x+1$,则生成另一个循环码。同理,在 x^7+1 的因式中,若选 $g(x)=x^3+x+1$ 或 $g(x)=x^3+x^2+1$,则可构造出两个不同的(7,4)循环码,若选 $g(x)=(x^3+x+1)(x^3+x^2+1)$,则可构造出一个(7,1)循环码,它就是重复码。由此可知,只要知道了 x^n+1 的因式分解式,用它的各个因式的乘积,便能得到很多个不同的循环码。

循环码可由其生成多项式确定,一般用八进制数字表示 $g(x)$。例如,八进制数 13 的二进制表示为 001011,代表 $g(x)=x^3+x+1$。表 5.4.2 列出了几种不同长度下循环 Hamming 码的生成多项式。

表 5.4.2　循环 Hamming 码的生成多项式

m	$n=2^m-1$	$k=2^m-m-1$	$g(x)$
3	7	4	x^3+x+1(13)
4	15	11	x^4+x+1(23)
5	31	26	x^5+x^2+1(45)
6	63	57	x^6+x+1(103)
7	127	120	x^7+x^3+1(211)
8	255	247	$x^8+x^4+x^3+x^2+1$(435)
9	511	502	x^9+x^4+1(1021)
10	1023	1013	$x^{10}+x^3+1$(2011)
11	2047	2036	$x^{11}+x^2+1$(4005)
12	4095	4083	$x^{12}+x^6+x^4+x+1$(10123)

5.4.1.4　循环码的生成矩阵和一致校验矩阵

循环码的生成矩阵可以很容易地由生成多项式得到。由于 $g(x)$为 $n-k$ 阶多项式,

以与此相对应的码字作为生成矩阵中的一行，则 $g(x),x^2g(x),\cdots,x^{k-1}g(x)$ 等多项式必定是线性无关的。把这 k 个多项式相对应的码字作为各行构成的矩阵即为生成矩阵，由各行的线性组合可以得到 2^k 个循环码字。所以循环码的生成矩阵 $\boldsymbol{G}$ 可用以下方法得到。设

$$\begin{aligned}
&g(x)=g_{n-k}x^{n-k}+g_{n-k-1}x^{n-k-1}+\cdots+g_1x+g_0\\
&xg(x)=g_{n-k}x^{n-k+1}+g_{n-k-1}x^{n-k}+\cdots+g_1x^2+g_0x\\
&\vdots\\
&x^{k-1}g(x)=g_{n-k}x^{n-1}+g_{n-k-1}x^{n-2}+\cdots+g_1x^k+g_0x^{k-1}
\end{aligned}$$

则码的生成矩阵以多项式形式表示为

$$\boldsymbol{G}(x)=\begin{bmatrix}x^{k-1}g(x)\\x^{k-2}g(x)\\\vdots\\g(x)\end{bmatrix} \tag{5.4.2}$$

取其系数即得相应的生成矩阵为

$$\boldsymbol{G}=\begin{bmatrix}
g_{n-k} & g_{n-k-1} & \cdots & g_1 & g_0 & \overbrace{0 & 0 & \cdots & 0}^{k-1}\\
0 & g_{n-k} & g_{n-k-1} & \cdots & g_1 & g_0 & 0 & \cdots & 0\\
\vdots & \vdots & & & & & & & \\
\underbrace{0 & \cdots & 0}_{k-1} & \underbrace{g_{n-k} & & g_{n-k-1}}_{n-k+1} & & \cdots & g_1 & g_0
\end{bmatrix}$$

由式(5.3.3)可知，输入信息组为 $(m_{k-1},\cdots,m_0)$ 时，相应的码多项式为

$$\begin{aligned}
C(x)&=(m_{k-1},m_{k-2},\cdots,m_0)\boldsymbol{G}(x)\\
&=(m_{k-1}x^{k-1}+m_{k-2}x^{k-2}+\cdots+m_0)g(x)
\end{aligned}$$

这表明，所有码多项式一定是 $g(x)$ 的倍式。

由式(5.4.2)所示生成矩阵得到的循环码并非系统码。在系统码中码的最左 k 位是信息码元，随后是 $n-k$ 位校验码元。这相当于码多项式 $C(x)$ 的第 $n-1$ 次至 $n-k$ 次的系数是信息位。其余的是校验位

$$\begin{aligned}
C(x)&=m_{k-1}x^{n-1}+\cdots+m_0x^{n-k}+r_{n-k-1}x^{n-k-1}+\cdots+r_0\\
&=m(x)x^{n-k}+r(x)\equiv 0,\bmod g(x)
\end{aligned} \tag{5.4.3}$$

式中：$m(x)=m_{k-1}x^{k-1}+\cdots+m_1x+m_0$ 是信息多项式；$r(x)=r_{n-k-1}x^{n-k-1}+\cdots+r_1x+r_0$ 是校验元多项式，它的系数 $(r_{n-k-1},\cdots,r_1,r_0)$ 就是信息组 $(m_{k-1},\cdots,m_1,m_0)$ 的校验元。由式(5.4.3)可知

$$-r(x)=-C(x)+m(x)x^{n-k}=m(x)x^{n-k},\bmod g(x) \tag{5.4.4}$$

而 $-r(x)$ 是 $r(x)$ 中的每一个系数取加法逆元，在 GF(2) 中加法和减法等效，即

$$-r_i=r_i,\quad -r(x)=r(x)$$

由上式可知，构造系统循环码时，只需将信息码多项式升 $n-k$ 阶(乘以 x^{n-k})，然后以

$g(x)$为模，所得余式$r(x)$的系数即为校验元。因此，系统循环码的编码过程就变成用除法求余的问题。

系统码的生成矩阵必为典型形式$\boldsymbol{G}=[\boldsymbol{I}_k \quad \boldsymbol{P}]$，与单位矩阵$\boldsymbol{I}_k$每行对应的信息多项式为

$$m_i(x)=m_i x^{k-i}=x^{k-i} \quad (i=1,2,\cdots,k)$$

由式(5.4.4)可得相应的校验多项式为

$$r_i(x)\equiv x^{k-i}\,x^{n-k}\equiv x^{n-i} \bmod g(x) \quad (i=1,2,\cdots,k) \tag{5.4.5}$$

由此得到生成矩阵中每行的码多项式为

$$C_i(x)=x^{n-i}+r_i(x) \quad (i=1,2,\cdots,k)$$

因此，系统循环码生成矩阵多项式的一般表示为

$$\boldsymbol{G}(x)=\begin{bmatrix}C_1(x)\\C_2(x)\\\vdots\\C_k(x)\end{bmatrix}=\begin{bmatrix}x^{n-1}+r_1(x)\\x^{n-2}+r_2(x)\\\vdots\\x^{n-k}+r_k(x)\end{bmatrix} \tag{5.4.6}$$

例 5.4.3 已知(7,4)系统码的生成多项式为$g(x)=x^3+x^2+1$，求生成矩阵。

解： 由式(5.4.5)可得

$$\begin{aligned}
r_1(x) &\equiv x^6\equiv x^2+x && \bmod g(x)\\
r_2(x) &\equiv x^5\equiv x+1 && \bmod g(x)\\
r_3(x) &\equiv x^4\equiv x^2+x+1 && \bmod g(x)\\
r_4(x) &\equiv x^3\equiv x^2+1 && \bmod g(x)
\end{aligned}$$

因此，生成矩阵多项式表示为

$$\boldsymbol{G}(x)=\begin{bmatrix}x^6+x^2+x\\x^5+x+1\\x^4+x^2+x+1\\x^3+x^2+1\end{bmatrix}$$

由多项式系数得到的生成矩阵为

$$\boldsymbol{G}=\left[\begin{array}{cccc:ccc}1&0&0&0&1&1&0\\0&1&0&0&0&1&1\\0&0&1&0&1&1&1\\0&0&0&1&1&0&1\end{array}\right]=[\boldsymbol{I}_4 \quad \boldsymbol{P}]$$

由于$g(x)$能除尽x^n+1，因此有

$$\begin{aligned}
x^n+1&=g(x)h(x)\\
&=(g_{n-k}\,x^{n-k}+\cdots+g_1x+g_0)(h_kx^k+\cdots+h_1x+h_0)
\end{aligned}$$

由此可推出循环码的一致校验矩阵，即

$$\boldsymbol{H}=\begin{bmatrix} h_0 & h_1 & \cdots & h_k & & & 0 \\ & h_0 & h_1 & \cdots & h_k & & \\ & & \ddots & & \ddots & & \ddots \\ 0 & & & h_0 & h_1 & \cdots & h_k \end{bmatrix}$$

（$n-k-1$ 个零；$k+1$ 个 h）

它完全由 $h(x)$ 的系数决定，故称 $h(x)$ 是循环码的校验多项式。

可以验证，有 $\boldsymbol{G}\boldsymbol{H}^{\mathrm{T}}=0$。式中：$\boldsymbol{0}$ 是一个 $k\times(n-k)$ 零矩阵。

同理，以上 $\boldsymbol{H}$ 矩阵是非标准型的，若要得到标准型 $\boldsymbol{H}$ 矩阵，可利用 $\boldsymbol{H}$ 与 $\boldsymbol{G}$ 的关系。也可由式(5.4.6)得出

$$\boldsymbol{H}=[\boldsymbol{P}^{\mathrm{T}} \vdots \boldsymbol{I}_{n-k}]=[r_1^{\mathrm{T}} r_2^{\mathrm{T}} \cdots r_k^{\mathrm{T}} \boldsymbol{I}_{n-k}]$$

另外，可定义 $h(x)$ 的互反多项式为

$$h^*(x)=x^k h(x^{-1})=h_0 x^k+h_1 x^{k-1}+\cdots+h_k$$

可见，$\boldsymbol{H}$ 矩阵可由下述的多项式矩阵的系数构成，即由 $h(x)$ 的互反多项式 $h^*(x)$ 循环移位得到的 r 组互不相关的多项式系数矢量构成。

仿照线性分组码，称 $\boldsymbol{H}$ 为循环码的一致校验矩阵：

$$\boldsymbol{H}(x)=\begin{pmatrix} x_{r-1}h^*(x) \\ \vdots \\ x_2 h^*(x) \\ x_1 h^*(x) \\ h^*(x) \end{pmatrix}$$

定义一个矩阵是生成矩阵还是校验矩阵，主要是看它们在编码过程中所起的作用。由于 $\boldsymbol{H}$ 矩阵与 $\boldsymbol{G}$ 矩阵彼此正交，所以两者的作用可以互换。若 $g(x)$ 生成一(n,k)循环码，那么 $h^*(x)$ 可生成$(n,n-k)$循环码，$h(x)$ 也可作为生成多项式得到一$(n,n-k)$循环码。

定义 5.4.3 以 $g(x)$ 作为生成多项式生成的(n,k)循环码和以 $h^*(x)$ 作为生成多项式生成的$(n,n-k)$循环码互为**对偶码**，而以 $g(x)$ 作为生成多项式生成的(n,k)循环码和以 $h(x)$ 作为生成多项生成的$(n,n-k)$循环码互为**等效对偶码**。

由 $\boldsymbol{G}$ 和 $\boldsymbol{H}$ 的正交性可以证明对偶码是互为正交的。而等效对偶码相互不满足正交性。

5.4.2 循环码的编码及其实现

一旦确定了循环码的生成多项式 $g(x)$，就完全确定了码。循环码的每个码多项式 $C(x)=g(x)m(x)$ 都是 $g(x)$ 的倍式。对系统码来说，就是已知信息多项式 $m(x)$ 求 $m(x)x^{n-k}$ 被 $g(x)$ 除以后的余式 $r(x)$。所以，循环码的编码器就是 $m(x)$ 乘 $g(x)$ 的乘法器，或者是 $g(x)$ 除法电路。另外，循环码的译码实际上也是用 $g(x)$ 去除接收多项式

中文教学录像

视频

$R(x)$，检测余式结果，因此，多项式乘法及除法是编译码的基本运算。本节首先介绍作为编译码电路核心的多项式除法电路，这里主要针对二进制编译码；然后讨论编码电路，对于多进制循环码，即 GF(q)上循环码的电路可以此类推。

5.4.2.1 多项式除法运算电路

设 GF(2)上两个多项式为

$$g(x)=g_r x^r+g_{r-1}x^{r-1}+\cdots+g_1 x+g_0,\quad g_r=1$$

$$A(x)=a_k x^k+a_{k-1}x^{k-1}+\cdots+a_1 x+a_0,\quad k\geqslant r$$

用 $g(x)$去除任意多项式 $A(x)$的电路即为 $g(x)$除法电路，如图 5.4.2 所示。

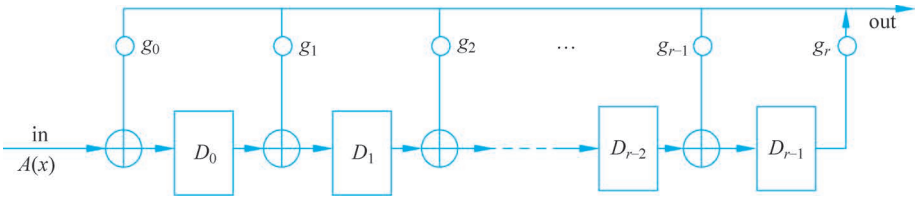


图 5.4.2 $g(x)$除法电路

可以证明，用 r 次多项式 $g(x)$去除任意 k 次多项式 $A(x)$，经 $k+1$ 拍各移位寄存器的存数即为余式，电路输出商比输入序列固定延迟 r 拍。

例 5.4.4 设被除式 $A(x)$与除式 $B(x)$都是 GF(2)上的多项式，且

$$A(x)=x^4+x^3+1,\quad B(x)=x^3+x+1$$

完成除以 $B(x)=x^3+x+1$ 的电路示于图 5.4.3 中。GF(2)上多项式的系数仅取 0 和 1，系数为 1 的逆元仍是 1，相加和相减相同，所以当 $b_i\neq 0$ 时，b_i^{-1} 与 $-b_i$ 均为一直线。

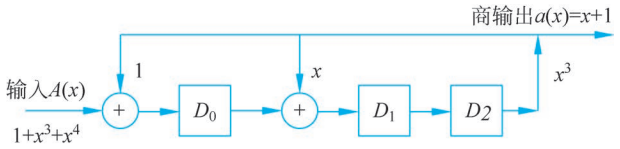


图 5.4.3 除 $B(x)=x^3+x+1$ 的电路

完成上述两个多项式相除的长除法算式如下：

$$x^4+x^3+1=(x+1)(x^3+x+1)+x^2$$

这里商为 $x+1$，余式为 x^2。表 5.4.3 给出了图 5.4.3 电路除 x^4+x^3+1 的运算过程，$r+1=4$ 次移位后得到商 x 项的系数，$k+1=5$ 次移位后，完成了整个除法运算，在移位寄存器中保存的数(001)，代表余式(x^0 x^1 x^2)的系数。

表 5.4.3 $B(x)$除 x^4+x^3+1 的运算过程

节拍	输入	移位寄存器内容			输出
		$D_0(x^0)$	$D_1(x^1)$	$D_2(x^2)$	
0	0	0	0	0	0
1	1(x^4)	1	0	0	0

续表

节拍	输入	移位寄存器内容			输出
		$D_0(x^0)$	$D_1(x^1)$	$D_2(x^2)$	
2	$1(x^3)$	1	1	0	0
3	$0(x^2)$	0	1	1	0
4	$0(x)$	1	1	1	$1(x)$
5	$1(x^0)$	0	0	1	$1(x^0)$
		余式			商式

5.4.2.2 循环码编码器

1. $n-k$ 级编码器

利用生成多项式 $g(x)$实现编码是编码电路的常用方法。若已知信息位为 k，纠错能力为 t，则可以按循环码的性质设计一套循环码。

首先可以根据式

$$2^r-1\geqslant\sum_{i=1}^{t}C_n^i$$

求出所需要的 n，$r=n-k$。求出 n 以后，再从 x^n+1 的因式中找出 $g(x)$生成多项式。该多项式最高次数为 $n-k$，记为$\partial^0 g(x)=n-k$，由 $g(x)$生成的码 $\boldsymbol{C}$ 就是满足要求的循环码。

现在的问题是给定 $g(x)$以后，在电路上如何实现编码？下面研究这个问题。

$n-k$ 级编码器有两种：一是 $g(x)$的乘法电路；二是 $g(x)$的除法电路。前者主要利用方程式 $C(x)=m(x)g(x)$进行编码，但这样编出的码为非系统码；后者是系统码编码器中常用的电路。这里只介绍系统码的编码电路。

设从信源输入编码器的 k 位信息组多项式

$$m(x)=m_{k-1}x^{k-1}+\cdots+m_1x+m_0$$

若要编出系统码的码字，则由式(5.4.3)和式(5.4.4)可知

$$C(x)=m(x)x^{n-k}+r(x)$$

$$r(x)\equiv m(x)x^{n-k}\mod g(x)$$

系统码的编码器就是信息组 $m(x)$乘 x^{n-k}，然后用 $g(x)$除，求余式 $r(x)$的电路。

下面以二进制(7,4)Hamming 码为例说明。设码的生成多项式 $g(x)=x^3+x+1$，其系统码编码器示于图 5.4.4。

编码过程如下。

(1) 三级移位寄存器初态全为 0，门 1 断开，门 2 接通。信息组以高位先入的次序送入电路，一方面经或门输出，另一方面送入 $g(x)$除法电路右端，这相应于完成 $x^{n-k}m(x)$的除法运算。

(2) 4 次移位后，信息组全部通过或门输出，它就是系统码码字的前 4 个信息元，与此同时它也全部进入 $g(x)$电路，完成除法。此时在移位寄存器中的存数就是余式 $r(x)$

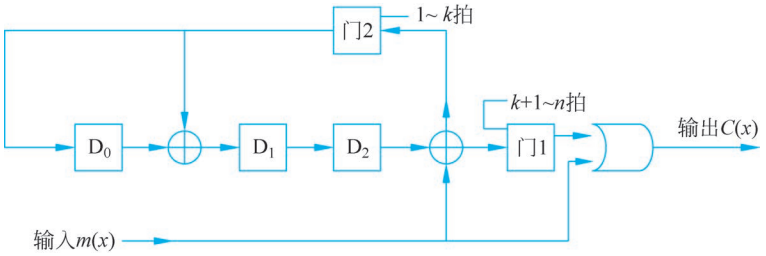


图 5.4.4 (7,4)Hamming 码三级除法编码器

的系数，也就是码字的校验元(c_2,c_1,c_0)。

(3) 门 1 接通，门 2 断开，再经 3 次移位后，移存器中的校验元(c_2,c_1,c_0)跟在信息组后面，形成一个码字($c_6=m_3$,$c_5=m_2$,$c_4=m_1$,$c_3=m_0$,c_2,c_1,c_0)从编码器输出。

(4) 门 1 断开，门 2 接通，送入第二组信息组，重复上述过程。

表 5.4.4 列出该编码器的工作过程。输入信息组是(1001)，7 次移位后输出端得到了已编好的码字(1001110)。

表 5.4.4 (7,4)Hamming 码编码的工作过程

节拍	信息组输入	移位寄存器内容			输出
		$D_0(x^0)$	$D_1(x^1)$	$D_2(x^2)$	
0		0	0	0	
1	1	1	1	0	1
2	0	0	1	1	0
3	0	1	1	1	0
4	1	0	1	1	1
5		0	0	1	1
6		0	0	0	1
7		0	0	0	0

校验元

2. k 级编码器

编码问题就是已知信息元 c_{n-1},c_{n-2},…,c_{n-k}，如何唯一求出校验位 c_{n-k-1},…,c_0，上述提到的编码方式是利用生成多项式来确定校验位，那么另一种编码方法则是用校验多项式来确定校验位。

k 级编码器是根据校验多项式

$$h(x)=h_k x^k+\cdots+h_1 x+h_0$$

构造的。其电路如图 5.4.5 所示。

如果移位寄存器初始状态从右至左是 $c_{n-1}\sim c_{n-k}$ 的 k 个信息元，经过 n 拍就能输出 $c_{n-1}\sim c_0$ 全部码元完成编码。此电路需要 k 级移位寄存器。

一般来说，当 $k>r$ 时，使用第一种 $n-k$ 级编码器较好($g(x)$编码)；当 $k<r$ 时，使用第二种 k 级编码器较好($h(x)$编码)。按上述条件设计的编码器可使用较少的移位寄存器。

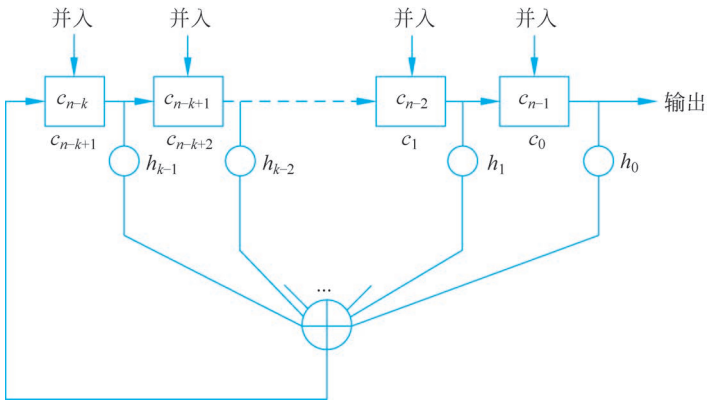


图 5.4.5　循环码的 k 级编码

5.4.3　循环码译码及其实现


中文教学录像

当一个码矢量通过噪声信道传送时，会遇到噪声干扰而产生错码，即在接收端所收到的矢量 $\boldsymbol{R}$ 可能与发送端码字不同。已经分析过 $\boldsymbol{R}$ 与 $\boldsymbol{C}$ 有如下关系：

$$\boldsymbol{R}=\boldsymbol{C}+\boldsymbol{E} \tag{5.4.7}$$

式中：$\boldsymbol{E}$ 为错型矢量。


视频

当然，式(5.4.7)也可以写成多项式的关系：

$$R(x)=C(x)+E(x) \tag{5.4.8}$$

式中

$$R(x)=r_{n-1}x^{n-1}+r_{n-2}x^{n-2}+\cdots+r_1x+r_0$$

$$C(x)=c_{n-1}x^{n-1}+c_{n-2}x^{n-2}+\cdots+c_1x+c_0$$

$$E(x)=e_{n-1}x^{n-1}+e_{n-2}x^{n-2}+\cdots+e_1x+e_0$$

从式(5.4.7)、式(5.4.8)中可以看到接收矢量含有码字矢量的信息和错误图样的信息。译码就是研究各种译码方法，选择和设计适当的译码电路，用于对 $\boldsymbol{R}$ 进行检错或纠错。

对于一组确定的循环码来说，从码字本身的代数结构来讲它的性质是确定的。即它的检错纠错能力是确定的。实际应用中由于采用不同的译码方法，实际达到的检、纠错能力并不等于码字所具有的能力。因此，衡量一种译码方法的优劣不单考虑检、纠错能力(当然这是最重要的指标)，还要考虑它的复杂程度和计算速度。下面就循环码译码的几种主要方法进行研究。

5.4.3.1　伴随式的计算

设发送的码字 $\boldsymbol{C}=(c_{n-1},c_{n-2},\cdots,c_1,c_0)$，其码字多项式 $C(x)=(c_{n-1}x^{n-1}+\cdots+c_1x+c_0)$(以后不再严格区分码字与码多项式)，信道产生的错误图样 $\boldsymbol{E}=(e_{n-1},e_{n-2},\cdots,e_1,e_0)$，译码器收到的 n 重

$$\begin{aligned}\boldsymbol{R}&=\boldsymbol{C}+\boldsymbol{E}\\&=(c_{n-1}+e_{n-1},c_{n-2}+e_{n-2},\cdots,c_1+e_1,c_0+e_0)\end{aligned}$$

$$=(r_{n-1},r_{n-2},\cdots,r_1,r_0),\quad r_i=c_i+e_i$$

或

$$\begin{aligned}R(x)&=C(x)+E(x)\\&=(r_{n-1}x^{n-1}+\cdots+r_1x+r_0),\quad r_i=c_i+e_i\end{aligned}$$

相应的伴随式为

$$\boldsymbol{S}=\boldsymbol{R}\boldsymbol{H}^{\mathrm{T}}=(\boldsymbol{C}+\boldsymbol{E})\boldsymbol{H}^{\mathrm{T}}=\boldsymbol{E}\boldsymbol{H}^{\mathrm{T}}$$

由此可知，伴随式 $\boldsymbol{S}$ 仅与错误图样有关，而与发送的码字无关，由它可计算出错误图样 $\boldsymbol{E}$。

设(n,k)循环码的生成多项式为 $g(x)$，且 $x^n+1=g(x)h(x)$，$\partial^0 g(x)=n-k$。该码的一致校验矩阵为

$$\boldsymbol{H}=[\tilde{\boldsymbol{x}}^{n-1\ \mathrm{T}}\quad\tilde{\boldsymbol{x}}^{n-2\ \mathrm{T}}\quad\cdots\quad\tilde{\boldsymbol{x}}^{\mathrm{T}}\quad\tilde{\boldsymbol{1}}^{\mathrm{T}}]\bmod g(x)$$

式中：$\tilde{\boldsymbol{x}}^i\equiv\boldsymbol{x}^i \bmod g(x)(i=0,1,\cdots,n-1)$。

所以

$$\boldsymbol{S}=\boldsymbol{R}\boldsymbol{H}^{\mathrm{T}}=(r_{n-1},r_{n-2},\cdots,r_1,r_0)\begin{bmatrix}\tilde{\boldsymbol{x}}^{n-1}\\\tilde{\boldsymbol{x}}^{n-2}\\\vdots\\\tilde{\boldsymbol{x}}^{1}\\\tilde{\boldsymbol{x}}^{0}\end{bmatrix}$$

$$=(c_{n-1},\cdots,c_1,c_0)\begin{bmatrix}\tilde{\boldsymbol{x}}^{n-1}\\\tilde{\boldsymbol{x}}^{n-2}\\\vdots\\\tilde{\boldsymbol{x}}^{1}\\\tilde{\boldsymbol{x}}^{0}\end{bmatrix}+(e_{n-1},\cdots,e_1,e_0)\begin{bmatrix}\tilde{\boldsymbol{x}}^{n-1}\\\tilde{\boldsymbol{x}}^{n-2}\\\vdots\\\tilde{\boldsymbol{x}}^{1}\\\tilde{\boldsymbol{x}}^{0}\end{bmatrix}$$

由此式可知相应的多项式表示为

$$S(x)\equiv C(x)+E(x)\equiv R(x)\equiv E(x)\bmod g(x)\tag{5.4.9}$$

根据欧几里得除法，有

$$R(x)=R_g(x)+g(x)q(x)$$

$$E(x)=E_g(x)+g(x)q_1(x)$$

因此式(5.4.9)又可表示为

$$S(x)=R_g(x)=E_g(x)\tag{5.4.10}$$

式中：$R_g(x)$、$E_g(x)$分别是 $R(x)$、$E(x)$被 $g(x)$除后所得的余式。两式表明循环码的伴随式计算电路就是一个 $g(x)$除法电路，伴随式 $S(x)$就是 $g(x)$除 $R(x)$后所得的余式。如果接收的矢量 $R(x)$没有错误，$E(x)=0$，则 $S(x)=0$；否则，$S(x)\neq0$(在码的检错能力以内)。因此，循环码的检错电路非常简单，就是一个 $g(x)$除法电路。收到 $R(x)$后送入 $g(x)$除法电路运算，若最后得到的余式为 0，则说明 $E(x)=0$，接收到的 $R(x)$就是一个码字；若不为 0，则说明接收到的 $R(x)$不是码字。

由式(5.4.9)或式(5.4.10)看出，若$\partial^0 E(x)<\partial^0 g(x)=n-k$，或 $E^{(i)}(x)=x^iE(x)$，

$\partial^0 E_1(x)<\partial^0 g(x)$，则 $S(x)\neq 0(\bmod\ g(x))$。这说明(n,k)循环码至多能检测长度等于 $n-k$ 的突发错误，以及检测使 $S(x)\equiv E(x)\neq 0(\bmod\ g(x))$的所有错误图样 $E(x)$。

用 $g(x)$除法电路计算伴随式的电路(伴随式计算电路)有如下几个很重要的特点。

定理 5.4.5 若 $S(x)$是 $R(x)$的伴随式，则 $R(x)$的循环移位 $x\ R(x)$(在模 x^n+1运算下)的伴随式 $S_1(x)$，是 $S(x)$在伴随式计算电路中无输入时(自发运算)右移一位的结果，即

$$S_1(x)\equiv xS(x)\bmod g(x)$$

证明： 由伴随式定义可知，$xR(x)$的伴随式为

$$\begin{aligned}S_1(x)&\equiv xR(x)\bmod g(x)\\&=x[R_g(x)+g(x)q(x)]\bmod g(x)\\&=xR_g(x)\bmod g(x)\end{aligned}$$

由式(5.4.10)可知

$$xS(x)=[xR_g(x)+xq(x)g(x)]\bmod g(x)$$

两式相减可得

$$xS(x)-S_1(x)=xq(x)g(x)\equiv 0\ \bmod g(x)$$

因此

$$S_1(x)\equiv xS(x)\ \bmod g(x)$$

5.4.3.2 循环码的通用译码算法

循环码属于线性分组码，其基本的译码方法也是伴随式译码，只不过由于其具有循环移位特性，可以多项式进行表示和计算，故而循环码的伴随式译码是基于多项式计算，具体过程如下。

(1) 计算接收多项式 $R(x)$对应的伴随式 $S(x)$。

(2) 根据伴随式 $S(x)$查表寻找对应的错误图样多项式(陪集首项)。

(3) 把接收多项式和错误图样多项式相加得到完成纠错的码字多项式。

循环码的通用译码器如图 5.4.6 所示。

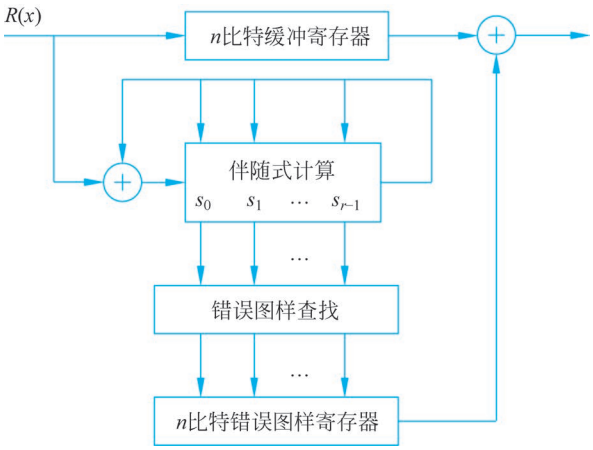


图 5.4.6 循环码的通用译码器

译码之前首先把寄存器清零，接着接收多项式 $R(x)$ 从高位到低位依次输入到“n 比特缓冲寄存器”，把接收矢量保存起来；同时接收多项式 $R(x)$ 输入到伴随式计算电路，这是一个除法电路。当 $R(x)$ 全部进入伴随式计算电路后，在移位寄存器中存放的就是相应的伴随式。用 $r=n-k$(bit)的伴随式作为地址去查找 n(bit)的错误图样，把错误图样放在 n 比特错误形式寄存器中；然后 n 比特缓冲寄存器中存放的接收矢量与 n(bit)错误图样同步输出并相加，达到纠错的目的。

伴随式计算电路和缓冲寄存电路都比较容易实现，困难的是从伴随式去查找错误图样。对于一个 (n,k) 循环码来说伴随式长度为 $(n-k)$，地址数目为 2^{n-k}，当 n 和 k 很大时，无法实现这样查表，所以要利用循环码的代数特征来简化查表复杂性。

5.4.3.3 梅吉特译码器

定理 5.4.6 $x^jR(x)$ 的伴随式 $S_j(x)\equiv x^jS(x)\bmod g(x)$，$(j=0,1,\cdots,n-1)$。而任意多项式 $a(x)$ 乘 $R(x)$ 所对应的伴随式

$$S_a(x)\equiv a(x)S(x)\bmod g(x)$$

伴随式计算电路的这些性质在循环码的译码运算中非常有用。若 $C(x)$ 是循环码 $\boldsymbol{C}$ 的一个码字，则 $xC(x)\in\boldsymbol{C}$。因此，若 $S(x)\equiv E(x)\bmod g(x)$ 是 $R(x)=C(x)+E(x)$ 的伴随式，则 $xS(x)\equiv xE(x)\bmod g(x)$ 就是 $xR(x)=xC(x)+xE(x)$ 的伴随式。也就是说，若 $E(x)$ 是一个可纠正的错误图样(陪集首)，则 $E(x)$ 的循环移位 $xE(x)$ 也是一个可纠正的错误图样。一般来说 $x^jE(x)(1\leqslant j\leqslant n-1)$ 也是可纠正的错误图样。可以根据这种循环关系划分错误图样，把任一特定的错误图样及其所有循环移位作为一类。例如，可将错误图样 100…0，0100…0，0010…0，00…01 作为一类，并将第一位开头为非零的错误图样 100…0 作为此类错误图样的代表。这时，对于二进制码，若码要纠正小于或等于 t 个错误，则错误图样代表共有

$$N_1=\sum_{j=1}^{t}C_{n-1}^{j-1}(\text{个}) \tag{5.4.11}$$

译码时，只要知道这些代表错误图样的伴随式，该类中其他错误图样的伴随式都可由此代表图样伴随式在伴随式计算电路中得到。这样，就使得循环码译码器的错误图样识别电路大为简化，由原来识别 N_2 个图样减少到 N_1 个。其中

$$N_2=\sum_{j=1}^{t}C_n^j(\text{个}) \tag{5.4.12}$$

例如，$n=63$，$t=4$，由式(5.4.11)和式(5.4.12)计算译码器所需识别的错误图样个数如表 5.4.5 所示。

表 5.4.5 N_1，N_2 比较

t	1	2	3	4
N_1	1	63	1954	39774
N_2	63	2016	41727	637382

一般来说，纠错码译码设备的复杂性主要取决于伴随式找出错误图样的识别电路或

组合逻辑电路的复杂性。由表 5.4.5 可知，虽然循环码识别错误图样的个数比一般非循环码大为减少，但随着 n、t 的加大，需要识别的错误代表个数 N_1 仍增加很快，以致难以实现。因此，利用组合逻辑电路识别错误图样代表的方法仅适合于 n、t 较小的情况。若 n、t 都较大，则必须利用循环码的其他特点寻找更为简单和巧妙的译码方法，这就是纠错码理论研究和实际应用中引人注目的问题之一。

例 5.4.5 二进制(7,4)循环 Hamming 码，它的 $g(x)=x^3+x+1$，相应的校验矩阵为

$$\boldsymbol{H}=[\tilde{x}^{6^T}\ \tilde{x}^{5^T}\ \tilde{x}^{4^T}\ \tilde{x}^{3^T}\ \tilde{x}^{2^T}\ \tilde{x}^{1^T}\ \tilde{x}^{0^T}]\bmod g(x)=\begin{bmatrix}1&1&1&0&1&0&0\\0&1&1&1&0&1&0\\1&1&0&1&0&0&1\end{bmatrix}$$

可见，该码的 $d_0=3$，可纠 $t=1$ 位错码。由式(5.4.11)知，构造此译码器的错误图样识别电路时，只要识别一个图样 $\boldsymbol{E}_6=(1000000)$ 就够了，该图样的伴随式就是 $\boldsymbol{H}$ 的第一列(101)。而 E_6 错误图样的识别电路就是一个检测伴随式是否为(101)的电路。由此可得如图 5.4.7 所示的译码电路。图中的伴随式计算电路就是一个 $g(x)=x^3+x+1$ 的除法电路，而有 3 个输入端的与门和反相器，组成了识别(101)的伴随式识别器。译码器的译码过程如表 5.4.6。

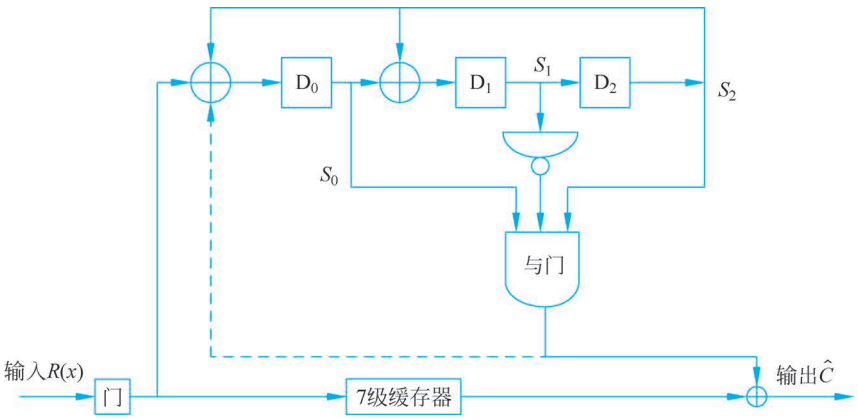


图 5.4.7 (7,4)循环 Hamming 码译码器

表 5.4.6 图 5.4.7 译码器的译码过程

节拍	输入 $\boldsymbol{R}(x)$	伴随式计算电路			与门输出	缓存输出	译码器输出
		$\boldsymbol{D}_0$	$\boldsymbol{D}_1$	$\boldsymbol{D}_2$			
0		0	0	0			
1	$1(x^6)$	1	0	0			
2	$0(x^5)$	0	1	0			
3	$0(x^4)$	0	0	1			
4	$0(x^3)$	1	1	0			
5	$0(x^2)$	0	1	1			
6	$1(x)$	0	1	1			
7	$1(x^0)$	0	1	1			

续表

节拍	输入 $R(x)$	伴随式计算电路			与门输出	缓存输出	译码器输出
		D_0	D_1	D_2			
8		1	1	1		1	1
9		1	0	1		0	0
10		0	0	0	1	0	1
11		0	0	0		0	0
12		0	0	0		0	0
13		0	0	0		1	1
14		0	0	0		1	1

具体过程如下。

(1) 开始译码时“门”接通,移位寄存器内容全为 0。收到的 $R(x)=r_6x^6+\cdots+r_0$,以高次项系数(r_6)至低次项系数的次序,一方面送入 7 级缓存器,另一方面送入 $g(x)$除法电路计算伴随式。7 次移位后,$R(x)$的系数全部存入缓存器,$g(x)$电路也得到了伴随式 $S_0(x)$,此时“门”断开,禁止输入。

(2) 若 $S(x)\equiv 1+x^2\equiv x^6 \pmod{g(x)}$,说明 $E(x)=x^6$,r_6 位有错,伴随式计算($g(x)$除法器)电路中的 D_0、D_1、D_2 存储的值是(101),这就是 $S_0(x)=1+x^2$ 的系数。D_1 的 0 经反相后成了 1,与门的 3 个输入端全为 1,呈打开状态。这时译码器继续移位,r_6 从缓存器输出,与门也输出一个信号“1”与 r_6 相加,使 r_6 由原来的 1 变成 0,或由 0 变成 1,纠正了 r_6 的错误:$r_6+1=c_6+e_6+1=c_6+1+1=c_6$,得到原来发送的码元。此时与门的纠错信号“1”也反馈到伴随式计算电路输入端(图 5.4.7 虚线所示),对伴随式进行修正,以消去该错误对伴随式的影响。

由于

$$R(x)=r_{n-1}x^{n-1}+\cdots+r_1x+r_0$$

相应的伴随式是 $S_0(x)$。纠错后 $R(x)$成为

$$R'_1(x)=(r_{n-1}+1)x^{n-1}+\cdots+r_1x+r_0$$

与 $R'_1(x)$相应的伴随式

$$S'_1(x)\equiv S_0(x)+x^{n-1} \bmod g(x)$$

因为纠错是在第 $n+1$ 次移位进行的,所以 $R'_1(x)$成为

$$R_1(x)=xR'_1(x)\equiv r_{n-2}x^{n-1}+\cdots+r_0x+r_{n-1}+1 \bmod (x^n-1)$$

相应的伴随式

$$S_1(x)\equiv xS'_1(x)=xS_0(x)+x^n\equiv xS_0(x)+1 \bmod g(x)$$

由于 $S_1(x)$是 $xR'_1(x)$的伴随式,而 $xS_0(x)$是 $xR(x)$的伴随式,也就是 $xE(x)$的伴随式,因此为了得到真正的 $xR(x)$的伴随式,就必须从 $S_1(x)$中消去“1”,也就是在伴随式计算电路输入端加 1。

例如,第 7 次移位后,若 $S_0=1+x^2$,说明 r_6 有错,第 8 次移位时对 r_6 进行纠错,纠错信号“1”输入到 $g(x)$电路输入端,结果使 $g(x)$移位寄存器中的内容成为(000),消除

了 e_6 的影响。

(3) 若 $E(x)=x^5$，则 $S_0(x)\equiv x^5\equiv x^2+x+1 \bmod g(x)$，此时与门不打开，说明 r_6 正确。这时伴随式计算电路和缓存器各移位一次，r_6 输出，r_5 移到缓存器最右一级，伴随式计算电路得到的伴随式为

$$S_1(x)\equiv xS_0(x)\equiv xE(x)\equiv x^2+1 \bmod g(x)$$

因此再移动一次，与门输出的纠正信号“1”正好与缓存器输出的 $r_5=c_5+1$ 相加，得到了 c_5，从而完成了纠错。若 r_5 不错，则重复上述过程一直到译完一个码字为止。

已知 $R(x)=x^6+x+1$，$E(x)=x^4$。由表 5.4.6 可知，到第 10 个节拍，与门输出一个“1”纠正 r_4，最后译码器输出码字(1010011)。

从上述译码过程可知，译一组码共需 14(2n)个节拍，仅当第一组的 $R(x)$ 移出 7 级缓存器后才能接收第二组的 $R(x)$。为了使译码连续，必须再加一个伴随式计算电路，如图 5.4.8 所示。开始工作时，所有移存器的存数全为 0，门 1 接通、门 2 断开。当 $k=4$ 次移位后，4 级缓存器接收了前面的 4 个信息位(对系统码而言)，此时门 1 断开，并使 4 级缓存器停止移位。再移动 $n-k=3$ 次后，$g(x)$除法电路得到了伴随式 $S_0(x)$，此时门 2 接通，把上边 $g(x)$除法电路得到的伴随式送到下面的伴随式计算电路中，随即门 2 断开，且上边 $g(x)$除法电路立即清除为 0。门 1 再次接通，4 级缓存器一边送出第一组的信息，一边接收第二组 $R(x)$的前 k 位信息组。与此同时，上边伴随式计算电路计算第二组 $R(x)$的伴随式，而下边伴随式计算电路通过自发运算对第一组 $R(x)$中的信息元进行纠错。

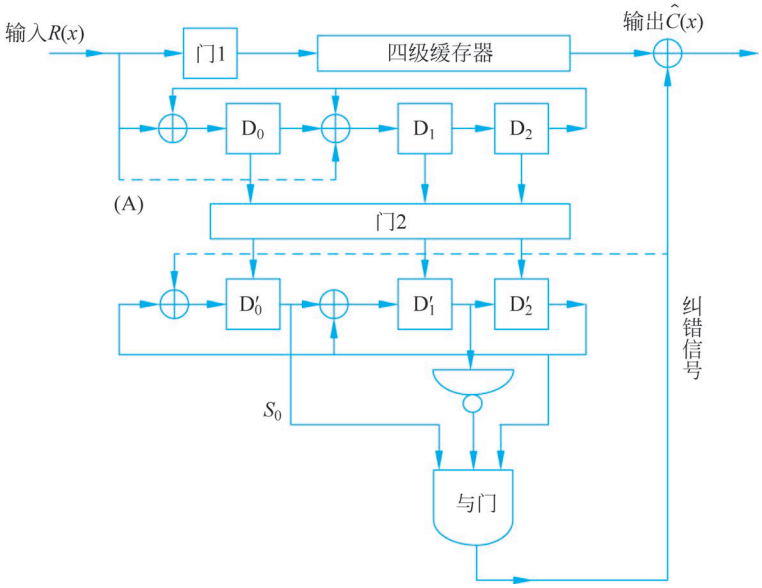


图 5.4.8 (7,4)码完整译码器

显然，上述译码电路仅适应于系统码。若为非系统码，k 级缓存器必须变成 n 级，且需要从已纠错过的 $\hat{C}(x)$中取出 k 个信息元 $\hat{m}(x)$。对非系统码而言，由 $C(x)=m(x)g(x)$ 可知，$\hat{m}(x)=\hat{C}(x)g^{-1}(x)$。说明译码器输出 $\hat{C}(x)$后，把 $\hat{C}(x)$再通过 $g(x)$除法电路，

所得的商就是最终所需的估值信息组 $\hat{m}(x)$。

由上述讨论可得出系统循环码的一般译码器，如图 5.4.9 所示，这种译码器也称梅吉特(Meggitt)译码器，它的复杂性由组合逻辑电路决定。

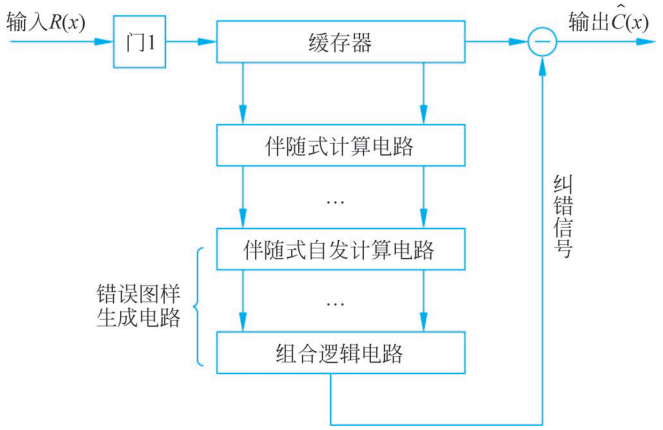


图 5.4.9　循环码的梅吉特译码器

下面，我们来分析缩短循环码的编译码问题。先以例 5.4.5 说明缩短循环码如何产生。

例 5.4.6　试构造一个(6,3)循环码。

需要构造一个(6,3)循环码，即 $n=6, k=3, r=3$。不难发现，找不到一个三次多项式 $g(x)$满足 $g(x)|(x^6+1)$。但 $g(x)=x^3+x+1$ 时，$g(x)|(x^7+1)$，故可先构成(7,4)码，而后去掉一位信息元，共有 $2^{4-1}=8$ 个码字，便得(6,3)码。具体做法是将(7,4)循环码中第一位为零的码字取出，去掉第一个零，即组成了(6,3)缩短码的全部码字：

$$
\begin{array}{cccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0, & 1\ 0\ 0\ 0\ 1\ 0\ 1, \\
0 & 0 & 0 & 1 & 0 & 1 & 1, & 1\ 0\ 0\ 1\ 1\ 1\ 0, \\
0 & 0 & 1 & 0 & 1 & 1 & 0, & 1\ 0\ 1\ 0\ 0\ 1\ 1, \\
0 & 0 & 1 & 1 & 1 & 0 & 1, & 1\ 0\ 1\ 1\ 0\ 0\ 0, \\
0 & 1 & 0 & 0 & 1 & 1 & 1, & 1\ 1\ 0\ 0\ 0\ 1\ 0, \\
0 & 1 & 0 & 1 & 1 & 0 & 0, & 1\ 1\ 0\ 1\ 0\ 0\ 1, \\
0 & 1 & 1 & 0 & 0 & 0 & 1, & 1\ 1\ 1\ 0\ 1\ 0\ 0, \\
0 & 1 & 1 & 1 & 0 & 1 & 0, & 1\ 1\ 1\ 1\ 1\ 1\ 1,
\end{array}
$$

注意，缩短循环码已不再具有循环移位的特点，不过它的每个码字多项式仍是原(n,k)码生成多项式 $g(x)$的倍式。$g(x)|(x^n+1)$，但 $g(x)$不能整除 $x^{n-i}+1$。

$(n-i,k-i)$缩短循环码的生成矩阵可以通过将原(n,k)码生成矩阵去掉前 i 行 i 列得到，而其一致校验矩阵则通过将原(n,k)码一致校验矩阵去掉前 i 列得到。例如，(6,3)码的生成矩阵和一致校验矩阵分别为

$$
\boldsymbol{G}_{7,4}=\begin{bmatrix}1&0&0&0&1&0&1\\0&1&0&0&1&1&1\\0&0&1&0&1&1&0\\0&0&0&1&0&1&1\end{bmatrix}
\xrightarrow{\text{去掉第一行、第一列}}
\boldsymbol{G}_{6,3}=\begin{bmatrix}1&0&0&1&1&1\\0&1&0&1&1&0\\0&0&1&0&1&1\end{bmatrix}
$$

$$\boldsymbol{H}_{7,4}=\begin{bmatrix}1&1&1&0&1&0&0\\0&1&1&1&0&1&0\\1&1&0&1&0&0&1\end{bmatrix}\xrightarrow{\text{去掉第一列}}\boldsymbol{H}_{6,3}=\begin{bmatrix}1&1&0&1&0&0\\1&1&1&0&1&0\\1&0&1&0&0&1\end{bmatrix}$$

尽管缩短循环码的码字已不再具有循环特性，但这并不影响其编、译码的简单实现，它仅需对原(n,k)循环码的编、译码稍作修正。

缩短循环码的编码器仍与原来循环码的编码器一样(因为去掉前i个为零的信息元，并不影响监督位的计算)，只是操作的总节拍少了i拍。译码时，只要在每个接收码组前加i个零，原循环码的译码器就可用来译缩短循环码。但为了节省资源也可不加i个零，而对伴随式寄存器的反馈连接进行修正。由于缩短了i位，相当于信息位也提前了i位，故需自动乘以x^i，并可用$R_{g(x)}[x^i]$电路实现。伴随式计算电路的输入应改为按下式的计算结果方式接入：

$$R_{g(x)}[x^i]=f(x)=x^k+\cdots+x^m+x^l$$

式中：$0\leqslant k<\cdots<m<l\leqslant r-1$，$r$为$g(x)$的次数，即接收码组$R$应从$s_k,\cdots,s_m,s_l$各级输入端同时接入。这时，伴随式计算电路的状态将为

$$S'_i(x)=R_{g(x)}[f(x)k(x)]$$

而

$$f(x)=x^i+g(x)q(x)$$

$$f(x)R(x)=x^iR(x)+g(x)q(x)k(x)$$

故

$$S'_i(x)=Rg(x)[x^iR(x)]=Rg(x)[x^iS(x)]=x^iS(x)=S_i(x)$$

这说明缩短i位的$(n-i,k-i)$码，除法运算可以提前i拍完成。经$n-i$拍后的伴随式状态$S'_i(x)$等于R从S_0输入端接入的情况下移位运算i拍后的状态$S_i(x)$。因此，如果将接收码组R按$f(x)$的方式接入伴随式计算电路，同时将缓冲寄存器改为$n-i$级，那么原循环码的一般译码电路就可改成$(n-i,k-i)$缩短循环码译码电路。例如，(15，11)循环 Hamming 码缩短 5 位便得到了(10,6)码，其生成多项式为$g(x)=x^3+x+1$，$f(x)=R_{g(x)}[x^5]=x^2+x$，图 5.4.10 是(10,6)缩短循环码的译码电路。

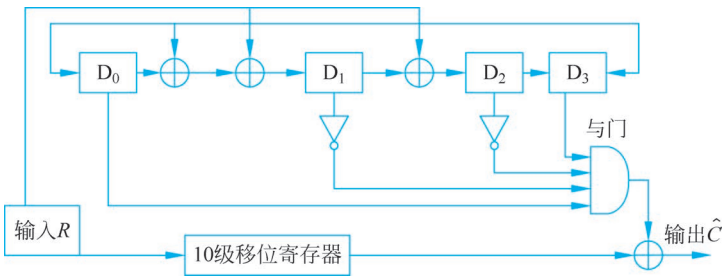


图 5.4.10 (10,6)缩短循环码的译码电路

总之，缩短循环码是在原循环码中选前i个信息位为 0 的码字组成。由于缩短的是信息元，缩短循环码的校验元数目与原循环码相同，因此缩短码的汉明距离和纠错能力不会低于原循环码，甚至会比原循环码更大些。

缩短循环码的译码器可在原(n,k)循环码译码器基础上做如下修正后使用。

(1) k 级缓存器改为$k-i$ 级(或 n 级缓存器改为$n-i$ 级)。

(2) 为了与(1)的改动相适应,$R(x)$应自动乘以 x^i,再输入伴随式计算电路。

例 5.4.7 试基于(7,4)循环 Hamming 码构造一个(7,3)删信码。

删信是信息位向校验位的变相转化,也就是说,保持长度 n 不变的情况下,维数 k 减小,奇偶校验符号的个数 $n-k$ 增加。如果循环码的最小码距为奇数,生成子多项式乘以 $x+1$ 会产生删信码、$d_{\min}$ 增加 1 的效果。从表 5.4.2 中可查到(7,4)循环 Hamming 码的生成多项式为

$$g(x)=x^3+x+1$$

则其(7,3)删信码的生成多项式为

$$g(x)(x+1)=x^4+x^3+x^2+1$$

新的生成子的阶数增加了 1,使得校验位数也增加了。然而,对于任意的 n 值,$x+1$ 是 x^n+1 的因子,这样对于原始的 n 值来讲,新的生成子仍是 x^n+1 的因子,因而码长不变。

新码中的任意码字都由原始码乘以 $x+1$(也就是说,向左移位,然后与原始码相加)得到的码字构成。所得的结果一定是码重为偶数,因为相加的两个序列具有相同码重,模 2 加法运算不会将总体偶校验转变成奇校验。以码序列 1000101 为例,它向左移位后与其自身相加,即

$$1000101+0001011=1001110$$

相加的每个序列的码重都是 3,但是加法运算导致码字中的两个 1 被取消,剩下一个重量为 4 的码字。

假设原始码的最小码距值为奇数,因此包含了奇数码重的码字,删除的码字就是原始码中码重为偶数的码字。术语“删信”的产生是因为所有码重为奇数的码字都被删除。结果就是最小码距成为某个偶数值。

在本例中,生成子 x^3+x+1 被删除,成为 $x^4+x^3+x^2+1$,新的生成子码重为 4,显然,新的 $d_{\min}$ 不能大于 4。由于最小码距必然由它的原始值 3 增加到了某个偶数值,因此现在这个值是 4。在其他删信 Hamming 码中,生成子的码重可能更高,但是仍然可以看到该码中包含码重为 4 的码字,所以删信操作后 $d_{\min}=4$。

删信码可以按照以新的生成子为基础的一般方法进行解码,但由于码字也是原始 Hamming 码的码字,因而可以根据以原始 Hamming 码生成子为除数和以 $x+1$ 为除数两种情况来形成两个伴随式,如图 5.4.11 所示。如果两个伴随式都是 0,那么说明无误码。如果两个伴随式都是非零的,那么可以假设存在单个比特的误码,并按照一般方法用汉明伴随式电路来对其纠错。如果一个伴随式是零,另外一个是非零,那么说明存在着不可纠正的误码。这种方法有助于对不可纠正的误码进行检测。

存在以下 3 种情况。

(1) 接收序列 0110010(单个错误),伴随式为 101 和 1(可纠正的误码)。

(2) 接收序列 0110011(两个误码),伴随式为 110 和 0(不可纠正的误码)。

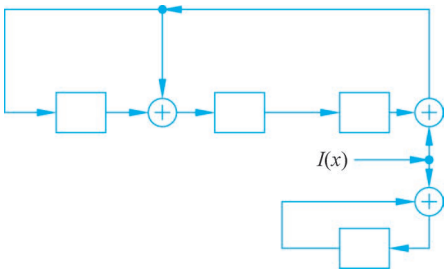


图 5.4.11　删信码伴随式的形成

(3) 接收序列 1011000(三个误码),伴随式为 000 和 1(不可纠正的误码)。

在第一种情况中,对第一个伴随式移位一次得到 001、010、100,表明误码在第 3 比特。

5.4.3.4　捕错译码

循环码译码器的复杂性主要取决于由伴随式确定错误图样的组合电路的简单与否,上节所述的梅吉特译码器对于纠错能力较小的循环码,这种组合电路可以做得比较简单,如循环 Hamming 码。对于纠大量错误的译码而言,梅吉特译码器中由伴随式求错误图样电路将很复杂。此外,对于某些码型,采用捕错译码方法也能够用较简单的组合逻辑电路实现译码,它特别适用于纠突发错误码、纠单个随机错误码,以及某些低码率和码长较短、纠错能力较弱的码。这种方法的错误图样生成电路更为简单,因而特别受到工程技术人员的欢迎,但这种译码方法对长码或高纠错能力的码来说效果还不理想。

捕错译码的基本原理如下。

设码字 $C(x)$ 是某个纠 $t(t\leqslant n-k)$ 个错误的系统型(n,k)循环码的码字多项式,接收矢量多项式 $R(x)=C(x)+E(x)$,通过译码电路可得伴随式为

$$S(x)\equiv E(x) \bmod g(x)$$

记 $E(x)=e_{n-1}x^{n-1}+\cdots+e_{n-k}x^{n-k}+e_{n-k-1}x^{n-k-1}+\cdots+e_0$,$E_{\mathrm{I}}(x)=e_{n-1}x^{n-1}+\cdots+e_{n-k}x^{n-k}$ 为信息位错误图样,$E_{\mathrm{P}}(x)=e_{n-k-1}x^{n-k-1}+\cdots+e_0$ 为校验位错误图样,则有

$$S(x)\triangleq E_{\mathrm{I}}(x)+E_{\mathrm{P}}(x) \bmod g(x)$$

若传输中恰好使一个码字的所有 t 个以内的错误都集中于校验位段,信息位无错,则有 $E_{\mathrm{I}}(x)=0$,$E(x)=E_{\mathrm{P}}(x)$,再考虑到 $g(x)$ 是 $n-k$ 次的,所以有伴随式 $S(x)=E_{\mathrm{P}}(x) \bmod g(x)=E_{\mathrm{P}}(x)$,这说明错误图样的后 $r(r=n-k)$ 位就是伴随式 $S(x)$ 的系数,而其前 k 位为 0,这样只要译码器算出了伴随式 $S(x)$,就意味着得到错误图样。纠错就只需用接收序列与伴随式右端对齐相减(二元码也可相加)。然而很遗憾,在实际情况中 t 个错误不会只出现在校验位上。所幸的是对于循环码,由于其循环移位特性使伴随式的“循环性”等价于错误图样在“循环”,因此只要传输中的错误集中在码字的任意 r 位码元段内,就可以通过伴随式计算电路自发运算若干次,将错误图样中的所有错误集中“捕获”到码字的后 r 位码元段,将此时的伴随式与移位后的接收矢量相减,完成纠错。

对于纠 t 个错误的循环码来说,必须使 t 个错误能连续地出现在 $n-k$ 位以内,这等

价于要求有连续 k 位码元无错，或错误图样中连续 t 位的值为 0。由于 t 个错误均匀分布在 n 个码元中是最难满足连续 t 位无错这一要求，因此 (n,k) 循环码可以用捕错译码的条件如下：

$$k < n/t \quad 或 \quad R < 1/t$$

在译码过程中，如何判断经过循环移位，错误已全部集中在接收矢量的最低次的 $n-k$ 位以内？对 (n,k) 循环码来说，下面定理给出了判断准则。

定理 5.4.7 对于纠正 t 个错误的 (n,k) 循环码，捕错译码过程中已把 t 个错误集中在矢量 $R^{(i)}(x)$ 的最低次 $n-k$ 位以内的充要条件是此时的伴随式重量满足小于或等于 t，即

$$W(S_i(x)) \leqslant t$$

证明： 若经过 i 次循环移位后错误已集中在 $n-k$ 位低次位码元段以内，则

$$S_i(x) = x^i E(x) = E^{(i)}(x) = E_{\mathrm{P}}^{(i)}(x) \bmod g(x)$$

$$S_i(x) = E_{\mathrm{P}}^{(i)}(x)$$

$$W(S_i(x)) = W(E_{\mathrm{P}}^{(i)}(x))$$

因为码的纠错能力为 t，若错误图样 $E(x)$ 是一个可纠正的错误图样，则其中“1”码元的数目不大于 t，即 $W(E_{\mathrm{P}}^{(i)}(x)) \leqslant t$，因而，当错误图样 $E(x)$ 经循环移位 i 次，错误已集中在 $n-k$ 位低次位码元段以内时，对应 $E^{(i)}(x)$ 的重量也必小于或等于 t，所以

$$W(S_i(x)) = W(E^{(i)}(x)) \leqslant t$$

反之，若 $W(S_i(x)) \leqslant t$，则错误一定集中在 $n-k$ 位低次位码元段内。设错误没有集中在该段以内，则

$$\partial^0 E^{(i)}(x) \geqslant \partial^0 g(x)$$

由此

$$E^{(i)}(x) = q(x)g(x) + S_i(x)$$

$$E^{(i)}(x) - S_i(x) = q(x)g(x) = C^{(i)}(x)$$

$C^{(i)}(x)$ 是 $g(x)$ 的倍式，由循环码性质可知它必是 (n,k) 循环码中的一个码字。因此

$$W(E^{(i)}(x) + (-S_i(x))) = W(C^{(i)}(x)) \geqslant d = 2t+1$$

由码重三角不等式可知

$$W(E^{(i)}(x)) + W(-S_i(x)) \geqslant W(E^{(i)}(x) + (-S_i(x))) \geqslant 2t+1$$

因为 $W(E^{(i)}(x)) \leqslant t$，所以 $W(-S_i(x)) \geqslant 2t+1-t = t+1 > t$

由于 $W(-S_i(x)) = W(S_i(x))$，因此上式与假设 $W(S_i(x)) \leqslant t$ 相矛盾。因而错误没有集中在 $n-k$ 低次位以内的反证法假设不能成立，故错误集中在 $n-k$ 低次位内。

［证毕］

捕错译码的基本流程如下。

(1) 计算接收多项式 $R(x)$ 对应的伴随式 $S(x)$。

(2) 计算 $S(x)$ 的重量 $W(S(x))$，并判断是否满足 $W(S_i(x)) \leqslant t$。若满足，说明错误已被集中捕获至 $n-k$ 低次位以内，则令 $E^{(i)}(x) = S(x)$，$R^{(i)}(x) = R(x)$，进入步

骤(3)；反之，把 $R(x)$ 和相应的伴随式 $S(x)$ 一起同时在译码器的各自移位寄存器中移位 i 次，使这些错误全部集中到 $x^iR(x)$ 的后 $n-k$ 位上。

(3) $C^{(i)}(x)=R^{(i)}(x)+E^{(i)}(x)$。

(4) 将已纠错的码字 $C^{(i)}(x)$ 再循环移位 $n-i$ 次，恢复到原来的接收矢量中各码元的顺序关系，得到码字估值 $C(x)$。

5.4.3.5 大数逻辑译码

循环码译码器的复杂性主要取决于码的代数结构，梅吉特译码和捕错译码仅对部分循环码复杂度较低，但对绝大多数纠多个随机错误的循环码来说没有如此简单的译码方法。大数逻辑译码是 1954 年里德(Reed)在译里德-穆勒(Reed-Muller，RM)码时提出的一种较简单的译码方法，尽管对于一般码长而言，能用大数逻辑方法译码的大数逻辑可译码，其纠错能力和码率都稍次于有相同参数的其他码，如 BCH 码，但由于它的译码算法和设备均比相应的 BCH 码译码方法要简单，因此在实际中有一定的应用。

通过以下例子介绍二进制循环码的大数逻辑译码的基本思想。

例 5.4.8 (7,3,4)增余删信 Hamming 码的生成多项式 $g(x)=(x+1)(x^3+x+1)=x^4+x^3+x^2+1$，一致校验矩阵为

$$\boldsymbol{H}=\begin{bmatrix}1011000\\1110100\\1100010\\0110001\end{bmatrix}$$

接收 n 重矢量 $\boldsymbol{R}=\boldsymbol{C}+\boldsymbol{E}$，发送码字和错误图样分别为 $\boldsymbol{C}=(c_6,\cdots,c_1,c_0)$ 和 $\boldsymbol{E}=(e_6,\cdots,e_1,e_0)$，相应的伴随式为

$$\boldsymbol{S}^{\mathrm{T}}=\begin{pmatrix}s_3\\s_2\\s_1\\s_0\end{pmatrix}=\boldsymbol{H}\boldsymbol{E}^{\mathrm{T}}=\begin{bmatrix}1011000\\1110100\\1100010\\0110001\end{bmatrix}\begin{pmatrix}e_6\\e_5\\e_4\\e_3\\e_2\\e_1\\e_0\end{pmatrix}$$

对伴随式的分量 s_3、s_2 和 s_1 进行线性组合，也就是对 $\boldsymbol{H}$ 矩阵的行进行线性组合，得到以下一组新的校验方程组：

$$\begin{cases}A_1=s_3=e_6+e_4+e_3\\A_2=s_1=e_6+e_5+e_1\\A_3=s_2+s_0=e_6+e_2+e_0\end{cases}\tag{5.4.13}$$

该方程组所对应的校验矩阵为

$$\boldsymbol{H}_G=\begin{bmatrix}1011000\\1100010\\1000101\end{bmatrix}$$

由校验方程的定义可知，$\boldsymbol{H}_G$ 的行向量是 $\boldsymbol{H}$ 的行向量的线性组合，因为 $\boldsymbol{C}\boldsymbol{H}^{\mathrm{T}}=0$，即许用码字 $\boldsymbol{C}$ 与一致校验矩阵 $\boldsymbol{H}$ 的各行正交，所以同样有许用码字 $\boldsymbol{C}$ 与校验矩阵 $\boldsymbol{H}_G$ 的各行正交，即 $\boldsymbol{C}\boldsymbol{H}_G{}^{\mathrm{T}}=0$。

另外，观察校验方程组(5.4.13)可知，每个方程式中均包含 e_6，而 $e_5,e_4,\cdots,e_0$ 仅出现在一个方程式中。具有这样特点的校验方程称为正交于 e_6 的**正交校验方程**(或正交监督方程、正交一致校验和式)，该方程组的系数矩阵 $\boldsymbol{H}_G$ 称为**正交校验矩阵**，e_6 称为**正交码元位**。

进一步观察例 5.4.8 中校验方程组(5.4.13)可以发现如下规律。

(1) 若错误图样发生一个错误，且在正交码元位上，即 $e_6=1$，则 $A_1=1,A_2=1,A_3=1$。

(2) 若错误图样发生一个错误，且不在正交码元位上，即 $e_6=0$，某个 $e_i=1(i=0,1,2,3,4,5)$，则 A_1、A_2、A_3 有且只有一个为 1。

(3) 若错误图样发生两个错误，一个在正交码元位上，一个不在正交码元位上，即 $e_6=1$，某个 $e_i=1(i=0,1,2,3,4,5)$，则在 A_1、A_2、A_3 中有两个 1、一个 0。

(4) 若错误图样发生两个错误，都不在正交码元位上，则 A_1、A_2、A_3 或者都为 0，或者两个为 1、一个为 0。

不难分析该(7,3)码最小距离为 4，其纠错能力为 1，结合上述规律可知，当传输中只发生 1 位错误(码的纠错能力之内)时，可以根据和式 A_i 中取值为 1 的个数情况对正交位 r_6 的传输情况进行判断和纠错。这就是大数逻辑译码的基本思想。

对于循环码不必分别建立正交于每个码元位的正交一致校验方程组，而只需根据其循环移位特性，在对 r_6 进行正交校验后，仍基于式(5.4.13)，依次将 $r_5\sim r_0$ 循环移位至正交位，按照同样的方法进行校验。这种根据正交方程中取值为 1 的个数的多少进行译码的方法称为**大数逻辑译码**。由式(5.4.13)可见，正交和式 A_i 可以表示为伴随式各分量的线性组合，也可以表示为接收矢量(错误图样)各分量的线性组合，因此大数逻辑译码器可分为Ⅰ型和Ⅱ型，它们没有本质的区别。

可以证明，有 J 个正交校验和式的码能纠正 $t\leqslant\lfloor J/2\rfloor$ 个错误。当传输错误发生在正交位时，正交一致校验方程组中将有大于 $\lfloor J/2\rfloor$ 个和式取值为 1，因此可依据此对每位码元进行正交校验。例 5.4.8 中正交一致校验方程组仅对 e_6(即 x^6)码元位正交，因此只要用一次大数逻辑判决就能完成对该位的译码，故称之为**一步大数逻辑译码**。一步大数逻辑译码的方法虽然简单，但是此类码不多，且纠错能力也不高。为了提高大数逻辑译码的应用范围和改善码的纠错能力，可把对某一码元位正交的概念扩展至对某一码元位置集合正交，并逐次进行大数逻辑译码。这种需要 L 步大数逻辑译码才能最后译出一个码元的方法称为***L* 步大数逻辑译码**。

5.4.4 BCH 码与 RS 码

5.4.4.1 BCH 码

BCH 码是最重要的一类循环码，BCH 码分别由 Hocquenghem 在 1959 年，以及

Bose 和 Chaudhuri 在 1960 年独立提出。1960 年 Peterson 对二进制 BCH 码提出一种有效的译码方法。从那时起编码领域的学者对 BCH 码的译码进行了深入研究。由于这些算法均需要应用许多复杂的代数理论知识，特别是有限域上的多项式知识，在此不做深入介绍，仅对 BCH 码的概念、构造及译码做简单介绍。

BCH 码是一类纠正多个随机错误的循环码，它的参数可以在较大范围内变化，选用灵活，适用性强。最为常用的二元 BCH 码是本原 BCH 码，其参数及其关系式：

$$\begin{aligned} &\text{码字长度：} && n=2^m-1 \\ &\text{校验位数目：} && n-k\leqslant mt \\ &\text{最小距离：} && d\geqslant 2t+1 \end{aligned} \tag{5.4.14}$$

式中：m 为正整数，一般 $m\geqslant 3$，纠错位数 $t<(2^m-1)/2$。

BCH 码的构造是利用循环码生成多项式 $g(x)$ 之根来研究的，其生成多项式可以按如下流程得到。

令 α 是 GF(2^m)的本原元，考虑 α 的连续幂序列

$$\alpha,\alpha^2,\alpha^3,\alpha^4,\cdots,\alpha^{2t}$$

令 $m_i(t)$ 是以 α^i 为根的最小多项式，则满足式(5.4.14)所列参数要求的 BCH 码生成多项式为

$$g(x)=\mathrm{LCM}[m_1(x),m_2(x),m_3(x),\cdots,m_{2t}(x)]$$

式中：LCM 为最小公倍式。

利用共轭元具有相同最小多项式的特点，则生成多项式可以写成

$$g(x)=\mathrm{LCM}[m_1(x),m_3(x),\cdots,m_{2t-1}(x)]$$

表 5.4.7 给出较小码长的 BCH 码相关参数和生成多项式。表中：n 表示码长；k 表示信息位长度；t 表示码的纠错能力；生成多项式 $g(x)$ 栏下的数字表示其系数，如表中生成多项式为 3551 时，该多项式序列的二进制表示为(11101101001)，则其生成多项式为 $g(x)=x^{10}+x^9+x^8+x^6+x^5+x^3+1$，构成一个能纠 2 个错误的(31,21)BCH 码。

表 5.4.7　BCH 码的参数和生成多项式

n	k	t	$g(x)$(八进制)
15	7	2	721
15	5	3	2461
31	21	2	3551
31	16	3	107657
31	11	5	5423325
63	51	2	12471
63	45	3	1701317
63	39	4	166623567
63	30	6	157464165547
127	113	2	41567
127	106	3	11554743
255	239	2	267543
255	231	3	156720665

例 5.4.9 在 GF(2^4)上构造长度为 $2^n-1=15$ 的纠错能力为 t 的二元本原 BCH 码。

解：GF(2^4)如表 5.4.8 所示。

表 5.4.8 $n=4, p(x)=x^4+x+1$ 的 GF(2^4)的所有元素

α 的幂	α 的多项式	α 的幂	α 的多项式
0	0	α^7	$\alpha^3+\alpha+1$
1	1	α^8	α^2+1
α	α	α^9	$\alpha^3+\alpha$
α^2	α^2	α^{10}	$\alpha^2+\alpha+1$
α^3	α^3	α^{11}	$\alpha^3+\alpha^2+\alpha$
α^4	$\alpha+1$	α^{12}	$\alpha^3+\alpha^2+\alpha+1$
α^5	$\alpha^2+\alpha$	α^{13}	$\alpha^3+\alpha^2+1$
α^6	$\alpha^3+\alpha^2$	α^{14}	α^3+1

(1) 设 $t=1$。由定义该码的生成多项式以 GF(2^4)本原元 α 和 α^2 为根。考虑到共轭元素有相同最小多项式，所以生成多项式为

$$g(x)=\mathrm{LCM}[m_1(x),m_3(x),\cdots,m_{2t-1}(x)]=m_1(x)$$
$$=(x+\alpha)(x+\alpha^2)(x+\alpha^4)(x+\alpha^8)=x^4+x+1$$

该生成多项式系数矢量的八进制表示为“23”，它正是(15,11)Hamming 码的生成多项式，它有 4 位校验位。该码的设计距离 $d=2t+1=3$，可以求出该码的实际最小距离也是 3，设计距离等于最小距离。

(2) 长度为 $2^4-1=15$，能纠 $t=2$ 位错误的 BCH 码的生成多项式以 α、α^2、α^3、α^4 为根(α 为 GF(2^4)本原元)。考虑到 α、α^2、α^4、α^8 和 α^3、α^6、α^9、α^{12} 为两组共轭根，所以生成多项式为

$$g(x)=\mathrm{LCM}[m_1(x),m_3(x),\cdots,m_{2t-1}(x)]=m_1(x)m_3(x)$$
$$=(x+\alpha)(x+\alpha^2)(x+\alpha^4)(x+\alpha^8)(x+\alpha^3)(x+\alpha^6)(x+\alpha^9)(x+\alpha^{12})$$
$$=(x^4+x+1)(x^4+x^3+x^2+x+1)$$
$$=x^8+x^7+x^6+x^4+1$$

这个生成多项式的八进制表示为“721”，它生成一个(15,7)BCH 码，有 8 位校验位，能纠正任意两位错误。这个码的设计距离 $d=2t+1=5$，可以求出这个码的实际最小距离也是 5。

(3) 对于可以纠正 $t=3$ 个错误的 BCH 码，有

$$g(x)=\mathrm{LCM}[m_1(x),m_3(x),\cdots,m_{2t-1}(x)]=m_1(x)m_3(x)m_5(x)$$
$$=(x+\alpha)(x+\alpha^2)(x+\alpha^4)(x+\alpha^8)(x+\alpha^3)(x+\alpha^6)(x+\alpha^9)(x+\alpha^{12})(x+\alpha^5)$$
$$(x+\alpha^{10})$$
$$=(x^4+x+1)(x^4+x^3+x^2+x+1)(x^2+x+1)$$
$$=x^{10}+x^8+x^5+x^4+x^2+x+1$$

可见，$r=\partial^0 g(x)=10$，这是一个(15,5)BCH 码，该码的设计距离 $d=2t+1=7$，与其实际

最小距离相同。

(4) 当 $t=4$ 时,有

$$g(x)=\mathrm{LCM}[m_1(x),m_3(x),\cdots,m_{2t-1}(x)]=m_1(x)m_3(x)m_5(x)m_7(x)$$
$$=(x^4+x+1)(x^4+x^3+x^2+x+1)(x^2+x+1)(x^4+x^3+1)$$
$$=x^{14}+x^{13}+x^{12}+x^{11}+x^{10}+x^9+x^8+x^7+x^6+x^5+x^4+x^3+x^2+x+1$$

可见,该码为(15,1)BCH 码,即(15,1)重复码。这个码只有两个码字,全 0 码字和全 1 码字,所以它的最小距离为 15,然而这个码的设计距离 $d=2t+1=9$。此时,实际最小距离大于设计距离。这个码实际可以纠正 7 个随机错误。

(5) 如果继续用上面的方法构造 t 为 5、6、7 的 BCH 码,将会发现得到的都是和上面同样的生成多项式:

$$g(x)=x^{14}+x^{13}+x^{12}+x^{11}+x^{10}+x^9+x^8+x^7+x^6+x^5+x^4+x^3+x^2+x+1$$

因为(15,1)BCH 码的纠错能力覆盖了 $t=4\sim7$。可见按 BCH 码构造的生成多项式实际可达纠错能力大于或等于设计能力 t。

(6) 如果要构造 $t=8$ 的 BCH 码,就需要域元素 α^{15} 的最小多项式 $m_{15}(x)$,这已经超出了扩域 GF(2^4)的范畴,需要更大的扩域才能做到。可见,纠错能力更强的 BCH 码需要用域元素更多的扩域来构造。

对于 BCH 码来说,也可以生成多项式的根定义一致校验矩阵 $\boldsymbol{H}$。如果码字多项式 $C(x)$有一个根β,即 $C(\beta)=0$,也就是

$$c_{n-1}\beta^{n-1}+c_{n-2}\beta^{n-2}+\cdots+c_1\beta+c_0=0$$

将上式写为矩阵形式,有

$$\boldsymbol{C}\begin{bmatrix}\beta^{n-1}\\ \vdots\\ \beta\\ 1\end{bmatrix}=0$$

式中:$\boldsymbol{C}=(c_{n-1},c_{n-2},\cdots,c_1,c_0)$是码字矢量。

因此,如果$\beta_1,\beta_2,\cdots,\beta_j$ 都是$C(x)$的根,则

$$\boldsymbol{C}\begin{bmatrix}\beta_1^{n-1} & \beta_2^{n-1} & \cdots & \beta_j^{n-1}\\ \vdots & \vdots & \ddots & \vdots\\ \beta_1 & \beta_2 & \cdots & \beta_j\\ 1 & 1 & 1 & 1\end{bmatrix}=0$$

由校验矩阵 $\boldsymbol{H}$ 的定义可得

$$\boldsymbol{C}\boldsymbol{H}^{\mathrm{T}}=0$$

因此可认为校验矩阵 $\boldsymbol{H}$ 为

$$\begin{bmatrix}(\beta_1^{n-1})^{\mathrm{T}} & \cdots & (\beta_1)^{\mathrm{T}} & \mathbf{1}^{\mathrm{T}}\\ (\beta_2^{n-1})^{\mathrm{T}} & \cdots & (\beta_2)^{\mathrm{T}} & \mathbf{1}^{\mathrm{T}}\\ \vdots & \ddots & \vdots & \vdots\\ (\beta_j^{n-1})^{\mathrm{T}} & \cdots & (\beta_j)^{\mathrm{T}} & \mathbf{1}^{\mathrm{T}}\end{bmatrix}$$

对于(15,11)Hamming 码来说，若 α 为 GF(2^4)上本原元，则有

$$\boldsymbol{H}=[(\boldsymbol{\alpha}^{14})^{\mathrm{T}},\cdots,(\boldsymbol{\alpha}^{2})^{\mathrm{T}},(\boldsymbol{\alpha})^{\mathrm{T}},\mathbf{1}^{\mathrm{T}}]$$

本原元素$\boldsymbol{\alpha}$ 的各次幂正好是这个域的全部非零元。由表 5.4.8 可得相应的校验矩阵为

$$\boldsymbol{H}=\begin{bmatrix}111101011001000\\011110101100100\\001111010110010\\111010110010001\end{bmatrix}$$

同样，对于生成多项式为 $g(x)=x^{10}+x^8+x^5+x^4+x^2+x+1$ 的(15,7)BCH 码，生成多项式的根中只有$\boldsymbol{\alpha}$、$\boldsymbol{\alpha}^3$ 是独立的，所以校验矩阵为

$$\boldsymbol{H}=\begin{bmatrix}(\boldsymbol{\alpha}^{n-1})^{\mathrm{T}} & \cdots & (\boldsymbol{\alpha}^{2})^{\mathrm{T}} & (\boldsymbol{\alpha})^{\mathrm{T}} & \mathbf{1}^{\mathrm{T}}\\(\boldsymbol{\alpha}^{3n-3})^{\mathrm{T}} & \cdots & (\boldsymbol{\alpha}^{6})^{\mathrm{T}} & (\boldsymbol{\alpha}^{3})^{\mathrm{T}} & \mathbf{1}^{\mathrm{T}}\end{bmatrix}$$

下面介绍 BCH 码的经典译码算法原理，是一种扩展 Peterson 算法（也称 Gorenstein-Zierler 算法）。

在对 BCH 码进行译码时，如果发生 $v\leqslant t$ 个错误，那么一般要找出这 v 个错误的位置，同时要求出这些错误的值（错误大小）。一般可以从伴随式求得错误位置多项式。迭代算法的译码过程就是利用 BCH 码生成多项式的特点确定这两个信息的过程。当然，在二元码中因为出错位的取值总等于 1，所以只需求出错误位置。

对于一个码长为 n 的 BCH 码，其纠错能力为 t，考虑其错误图样多项式（简称错误多项式）：

$$E(x)=e_{n-1}x^{n-1}+e_{n-2}x^{n-2}+\cdots+e_1x+e_0$$

系数 $e_{n-i}(i=1\sim n-1)$表示错误图样中相应码元的取值，最多有 t 个非零系数。假设实际发生了 $v(0\leqslant v\leqslant t)$个错误，这些错误发生在位置 $j_1,j_2,\cdots,j_v$，则可以把错误多项式改写为

$$E(x)=e_{j_v}x^{j_v}+\cdots+e_{j_2}x^{j_2}+e_{j_1}x^{j_1}$$

$$E(x)=Y_vx^{j_v}+\cdots+Y_2x^{j_2}+Y_1x^{j_1}$$

式中：Y_l 为第 l 个错误的取值。

根据 BCH 码的定义，可把伴随式定义为接收矢量多项式在各个域元素处的值，如在域元素 α 处，伴随式为

$$\boldsymbol{S}=\begin{bmatrix}s_1\\s_2\\\vdots\\s_{2t}\end{bmatrix}^{\mathrm{T}}$$

式中

$$\begin{aligned}s_i&=E(\alpha^i)=R(\alpha^i)\\&=\sum_{l=1}^{v}Y_l(\alpha^i)^{j_l}=\sum_{l=1}^{v}Y_l(\alpha^{j_l})^i\quad(i=1,2,\cdots,2t)\end{aligned}\tag{5.4.15}$$

令

$$\alpha^{j_l} \triangleq X_l$$

则有

$$s_i = \sum_{l=1}^{v} Y_l X_l^i \quad (i = 1, 2, \cdots, 2t)$$

式中：Y_l 表示错误大小；X_l 表示错误位置。

于是上式展开为

$$\begin{cases} s_1 = Y_1 X_1 + Y_2 X_2 + \cdots + Y_v X_v \\ s_2 = Y_1 X_1^2 + Y_2 X_2^2 + \cdots + Y_v X_v^2 \\ \vdots \\ s_{2t} = Y_1 X_1^{2t} + Y_2 X_2^{2t} + \cdots + Y_v X_v^{2t} \end{cases} \tag{5.4.16}$$

这是一个有 $2v$ 个未知参量的方程组，包含错误大小 $Y_1, Y_2, \cdots, Y_v$，错误位置 $X_1, X_2, \cdots, X_v$，再定义错误位置多项式为

$$\sigma(x) = (1 - xX_1)(1 - xX_2)\cdots(1 - xX_t)$$

$$\triangleq 1 + \sigma_1 x + \sigma_2 x^2 + \cdots + \sigma_t x^t$$

也就是说，错误位置的倒数 x_l^{-1} 是这个多项式的零点值。应用错误位置多项式可以把上面的方程组改写成矩阵形式：

$$\begin{bmatrix} s_1 & s_2 & \cdots & s_v \\ s_2 & s_3 & \cdots & s_{v+1} \\ \vdots & \vdots & \ddots & \vdots \\ s_v & s_{v+1} & \cdots & s_{2v-1} \end{bmatrix} \begin{bmatrix} \sigma_v \\ \sigma_{v-1} \\ \vdots \\ \sigma_1 \end{bmatrix} = - \begin{bmatrix} s_{v+1} \\ s_{v+2} \\ \vdots \\ s_{2v} \end{bmatrix} \tag{5.4.17}$$

通过解这个方程组就可以求出错误位置多项式的所有系数 σ_i 的值。在解方程组过程中需要求伴随式矩阵的逆，所以要求伴随式矩阵是非奇异的。可以证明，当存在 v 个错误时，伴随式矩阵是非奇异的。由此可以从伴随式矩阵的奇异性中确定实际发生错误的个数 v。

BCH 码迭代法译码步骤具体如下。

(1) 设 $v = t$，计算伴随式矩阵的行列式 $\det(\boldsymbol{M})$，其中，

$$\boldsymbol{M} = \begin{bmatrix} s_1 & s_2 & \cdots & s_v \\ s_2 & s_3 & \cdots & s_{v+1} \\ \vdots & \vdots & \ddots & \vdots \\ s_v & s_{v+1} & \cdots & s_{2v-1} \end{bmatrix}$$

若 $\det(\boldsymbol{M}) = 0$，则说明伴随式矩阵是奇异的，接收码字中没有发生 t 个错误。

(2) 对矩阵 $\boldsymbol{M}$ 做降维处理，设 $v = t - 1$，重新计算 $\det(\boldsymbol{M})$。重复这个过程直到 $\det(\boldsymbol{M}) \neq 0$，这时的 v 值就是实际发生错误的个数。

(3) 根据找到的 v 值，解方程式(5.4.17)，得到错误位置多项式的所有系数 σ_i 的值。

(4) 解方程 $\sigma(x)=0$,得到它的所有零点。这里可采用钱氏搜索(Chien search)算法进行穷举搜索,即把所有域元素 α 逐一代入 $\sigma(x)$ 中进行检测,看方程 $\sigma(x)=0$ 是否成立。得到它的所有零点后,其倒数也就是

$$\sigma(x)=(1-xX_1)(1-xX_2)\cdots(1-xX_t)$$

式中:$X_1,X_2,\cdots,X_v$ 为错误位置。

对于二进制码来说,因为错误值只能是 1,所以译码过程到这里就结束了。对于非二进制码,要求错误值 $Y_1,Y_2,\cdots,Y_v$,再回到方程式(5.4.17),现在其中的错误位置 X_1,$X_2,\cdots,X_v$ 是已知的,解这个线性方程式,就可以得到所有错误值。

例 5.4.10 对于例 5.4.9 中构造的(15,7)BCH 码,设收到矢量 $\boldsymbol{R}=(100010111000101)$。将其以多项式表示,根据式(5.4.15)计算出其伴随式(运算在 GF(2^4)上进行)为 $s_1=\alpha^{14}$,$s_2=s_1^2=\alpha^{13}$,$s_3=\alpha^{13}$,$s_4=s_2^2=\alpha^{11}$。代入方程式(5.4.17)可得

$$\begin{aligned}\sigma(x)&=1+\alpha^{14}x+\alpha^2x^2\\&=(1-x\alpha^{-3})(1-x\alpha^{-14})\end{aligned}$$

试探可得 α^3 和 α^{10} 为错误位置多项式的根,故第 12 位和第 5 位出错,于是对应的码字 $C=(100000111001101)$。

由上面的介绍可知,决定 BCH 译码器复杂度和速度的主要因素是求错误位置多项式 $\sigma(x)$。1966 年伯利坎普(Berlekamp)提出了迭代译码算法,1969 年梅西(Massey)进行了改进,这种译码算法称为 BM 迭代译码算法,它大大节省了计算量,加快了译码速度,很好地解决了 BCH 码译码的实用问题。之后研究人员不断提出新的译码算法,如欧几里得算法、连分式算法、频域译码等。迭代算法的优点是使用的硬件资源相对较少,而且对错误位数不敏感,当错误位数较多时,迭代译码算法所使用的硬件资源与位数少的情况相差无几。另外,迭代译码算法的运算速度与码的纠错能力呈线性关系,当纠错能力大时,迭代次数也会相应增多。

5.4.4.2 RS 码

RS(Reed-Solomon)码首先由里德(Reed)和索罗门(Solomon)于 1960 年构造,是一类具有很强纠错能力的非二进制 BCH 码,近年来在许多通信系统中获得了应用。RS 码的码元符号取自有限域 GF(q),它的生成多项式的根也是 GF(q)中的本原元,所以它的符号域和根域相同。由于 RS 码是以每符号 m bit 进行的多元符号编码,在编码方法上与二元(n,k)循环码不同。分组块长 $n=2^m-1$ 的码字比特数为 $m(2^m-1)$bit,当 $m=1$ 时就是二元编码。RS 码一般常用 $m=8$bit,这类 RS 码具有很大应用价值。能纠正 t 个错误的 RS 码具有如下参数:

码字长度: $n=q-1$(符号)

校验位数据: $n-k=2t$(符号)

最小距离: $d=2t+1$(符号)

由 5.3.3 节介绍的码的性能限可知,线性码的最大可能的最小距离为校验位数目加 1,这就是 Singleton 限界,RS 码正好达到此限。因此,RS 码是一种 MDS 码,在同等码率

下其纠错能力最强。

若取 $q=2^m$，则 RS 码的码元符号取自 GF(2^m)，码长 $n=2^m-1$。一个能纠正 t 位符号错误的 RS 码的生成多项式可表示为

$$g(x)=(x+\alpha)(x+\alpha^2)(x+\alpha^3)\cdots(x+\alpha^{2t})$$

式中：α 为 GF(2^m)上的本原元。

由于 RS 码实质是一类非二进制 BCH 码，因此其译码算法可以沿用 BCH 译码算法，只是它在纠错时除了要确定错误位置 Y_i 外，还要求出相应的错误值 X_i。

RS 码在深空通信、移动通信、军用通信、光纤通信、磁盘阵列及光盘纠错等方面得到了广泛应用。例如，RS(255,223)码已成为美国国家航空航天局和欧洲空间站的深空通信系统中的标准信道编码，RS(31,15)码是军用通信中的首选信道编码，RS-PC(182,172)(208,192)成为数字光盘(DVD)系统的纠错标准。

例 5.4.11 一个符号取自 GF(2^3)，长度 $n=2^3-1=7$，能纠正 2 个错误的 RS 码的生成多项式为

$$g(x)=(x+\alpha)(x+\alpha^2)(x+\alpha^3)(x+\alpha^4)$$

根据表 5.4.9 所示的 GF(2^3)域元素，可得

$$g(x)=x^4+\alpha^3x^3+x^2+\alpha x+\alpha^3$$

式中：α 为本原多项式 x^3+x+1 的根。这个 RS 码的校验位长 $n-k=4$，因而是(7,3) RS 码。

假设该(7,3)RS 码经传输，在第 2 位出现值为 α 的错误，第 6 位出现 α^2 的错误，则错误多项式为

$$E(x)=\alpha^2x^6+\alpha x^2$$

因此伴随矢量 $\boldsymbol{S}=(s_1,s_2,s_3,s_4)$(运算在 GF($2^3$)上进行)。为

$$s_1=E(\alpha)=\alpha^2\alpha^6+\alpha\alpha^2=1$$

$$s_2=E(\alpha^2)=\alpha^2\alpha^{12}+\alpha\alpha^4=\alpha^4$$

$$s_3=E(\alpha^3)=\alpha^2\alpha^{18}+\alpha\alpha^6=\alpha^2$$

$$s_4=E(\alpha^4)=\alpha^2\alpha^{24}+\alpha\alpha^8=\alpha^3$$

表 5.4.9 $n=3, p(x)=x^3+x+1$ 的 GF(2^3)的所有元素

α 的幂	α 的多项式	3 重表示式
0	0	000
1	1	001
α	α	010
α^2	α^2	100
α^3	$\alpha+1$	011
α^4	$\alpha^2+\alpha$	110
α^5	$\alpha^2+\alpha+1$	111
α^6	α^2+1	101

于是，错误位置多项式 $\sigma(x)$ 的系数 σ_1、σ_2 是下面方程组的解：

$$\begin{bmatrix} 1 & \alpha^4 \\ \alpha^4 & \alpha^2 \end{bmatrix} \begin{bmatrix} \sigma_2 \\ \sigma_1 \end{bmatrix} = \begin{bmatrix} \alpha^2 \\ \alpha^3 \end{bmatrix}$$

由此得错误位置多项式为

$$\sigma(x) = \sigma_2 x^2 + \sigma_1 x + 1 = \alpha x^2 + x + 1$$

用钱氏搜索算法进行穷举搜索，把有限域 $GF(2^3)$ 的元素逐个代入 $\sigma(x)=\alpha x^2+x+1$ 试探，发现仅当 $x=\alpha$ 和 $x=\alpha^5$ 时错误位置多项式 $\sigma(x)=\alpha x^2+x+1$ 为零。所以错误位置为 $i_1=7-5=2$ 和 $i_2=7-1=6$，这正是所设的错误位置。

设 $\beta_1=\alpha^{i_1}=\alpha^2$，$\beta_2=\alpha^{i_2}=\alpha^6$，于是错误值方程为

$$\begin{bmatrix} \beta_1 & \beta_2 \\ \beta_1^2 & \beta_2^2 \end{bmatrix} \begin{bmatrix} Y_2 \\ Y_1 \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \end{bmatrix}$$

解此方程可得

$$Y_2 = \frac{\alpha^{12}+\alpha^{10}}{\alpha^{14}+\alpha^{10}} = \alpha$$

$$Y_1 = \frac{\alpha^6+\alpha^4}{\alpha} = \alpha^2$$

于是第 2 位出现值为 α、第 6 位出现值为 α^2 的错误，这正是题设所假设的错误值。

中文教学录像

5.5 卷积码

卷积码自 1955 年由 Elias 发明以来，从 20 世纪 70 年代开始被广泛应用于无线通信、深空通信及广播通信中，普及的原因是使用最大似然序列译码器进行译码时实现相对简单，以及与 Reed-Solomon 码级联时表现出来的优异性能。从 20 世纪 90 年代初开始，由于在 Turbo(或者类 Turbo)码中级联多个卷积码所表现出来的有效性，这一类码又重新得到重视。本节从码的定义与表述、译码算法以及应用等方面介绍卷积码。

5.5.1 卷积码的定义

卷积码是具有十分独特代数结构的线性码，虽然它们可以在基于分组的情况下使用，但是它们的编码器更多地被描述为面向数据流的形式。首先介绍四状态、码率为 1/2 的卷积码，它的编码器如图 5.5.1 所示。每一个信息比特进入编码器都会生成相应的两个编码比特，即 $k=1$，$n=2$；编码器中含有 $m=2$ 个二进制记忆单元(寄存器)，其状态数目为 $2^m=4$，编码约束度为 $m+1=3$。编码器的上下两部分像是运算在 GF(2)域上的两个离散时间有限冲激响应滤波器。顶部滤波器的冲激响应是 $\boldsymbol{g}^{(1)}=[1\quad 1\quad 1]$，而底部滤波器的冲激响应是 $\boldsymbol{g}^{(2)}=[1\quad 0\quad 1]$。从这样的角度看，编码器的输出 $\boldsymbol{c}^{(1)}$ 是输入 $\boldsymbol{u}$ 和冲激响应 $\boldsymbol{g}^{(1)}$ 的卷积，编码器的输出 $\boldsymbol{c}^{(2)}$ 也是如此，这就是卷积码的起源。卷积码通常用 (n,k,m)，则图 5.5.1 表示(2,1,2)卷积码。这样的话，对于 $j=1,2$，有

$$\boldsymbol{c}^{(j)} = \boldsymbol{u} \circledast \boldsymbol{g}^{(j)}$$

其中，⊛表示离散卷积，所有的运算都是模之和。

而且，时域里的卷积运算对应变换域里的乘积运算，这个方程还可以重新写成

$$c^{(j)}(D)=u(D)g^{(j)}(D)$$

其中，D 的多项式系数是对应向量的元素，所以 $g^{(1)}(D)=1+D+D^2$，$g^{(2)}(D)=1+D^2$。D 在编码的文献里被广泛使用，它等价于单位离散时间延迟算子 z^{-1}。

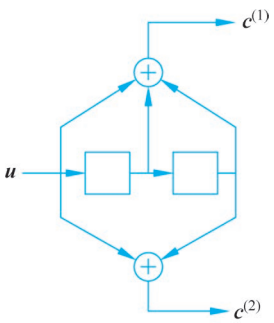


图 5.5.1 一个四状态、码率为 1/2 的卷积码编码器

上述编码过程可以描述为如下形式：

$$[c^{(1)}(D)\quad c^{(2)}(D)]=u(D)[g^{(1)}(D)\quad g^{(2)}(D)]$$
$$=u(D)\boldsymbol{G}(D)$$

式中：$\boldsymbol{G}(D)=[g^{(1)}(D)\quad g^{(2)}(D)]$是该码的生成矩阵，为 1×2 阶矩阵；$g^{(j)}(D)$为生成多项式。

对于长为 l 的输入数据（$u(D)$的次数是 $l-1$），输出数据的长度为 $2l+4$。为了使编码器状态归零，需使用额外的两个 0，从而产生了 4 个额外编码比特。因此，实际上码率是 $l/(2l+4)$，当 l 很大时码率接近 1/2。由于信息数据和码字之间具有简单而高度结构化的关系，卷积码具有较低的编码复杂度和译码复杂度。

5.5.2 卷积码的表示


双语教学录像

5.5.2.1 代数描述

定义一个二元卷积码 $\boldsymbol{C}$，码字 $\boldsymbol{c}(D)\in\boldsymbol{C}$ 有如下的形式：

$$\boldsymbol{c}(D)=[c^{(1)}(D)\quad c^{(2)}(D)\quad\cdots\quad c^{(n)}(D)]$$

式中：$c^{(j)}(D)=\sum_{i=r}^{\infty}c_i^{(j)}D^i$。

每一个 $\boldsymbol{c}(D)\in\boldsymbol{C}$ 可以写成这个基向量$\{\boldsymbol{g}_i(D)\}_{i=1}^{k}$ 的线性组合：

$$\boldsymbol{c}(D)=\sum_{i=1}^{k}u^{(i)}(D)\boldsymbol{g}_i(D)$$

式中：k 是$\boldsymbol{C}$ 的维数。

与线性分组码类似，上式可以重写为

$$\boldsymbol{c}(D)=\boldsymbol{u}(D)\boldsymbol{G}(D)\tag{5.5.1}$$

这里 $\boldsymbol{u}(D)=[u^{(1)}(D)\quad u^{(2)}(D)\quad\cdots\quad u^{(k)}(D)]$，基向量 $\boldsymbol{g}_i(D)=[g_i^{(1)}(D)\quad g_i^{(2)}(D)\quad\cdots\quad g_i^{(n)}(D)]$构成了 $k\times n$ 生成矩阵$\boldsymbol{G}(D)$的行，所以 $\boldsymbol{G}(D)$中第 i 行第 j 列的元素是 $g_i^{(j)}(D)$。

$\boldsymbol{G}(D)$表示一般的生成矩阵，它可以是系统或者非系统形式。存在一个$(n-k)\times n$校验矩阵 $\boldsymbol{H}(D)$，使得

$$\boldsymbol{c}(D)\boldsymbol{H}^{\mathrm{T}}(D)=\boldsymbol{0}$$

同时

$$\boldsymbol{G}(D)\boldsymbol{H}^{\mathrm{T}}(D)=\boldsymbol{0}$$

在前面的两个等式中的 $\boldsymbol{0}$ 分别表示 $1\times(n-k)$零向量和 $k\times(n-k)$零矩阵。对于线性分组码，$\boldsymbol{G}(D)$和 $\boldsymbol{H}(D)$都有系统形式，即为 $\boldsymbol{G}_0(D)=[\boldsymbol{I}|\boldsymbol{P}(D)]$和 $\boldsymbol{H}_0(D)=[\boldsymbol{P}^{\mathrm{T}}(D)|\boldsymbol{I}]$，$\boldsymbol{G}_0(D)$是由 $\boldsymbol{G}(D)$通过行变换得到。卷积码也有类似的情况，通过例 5.5.1 来描述。

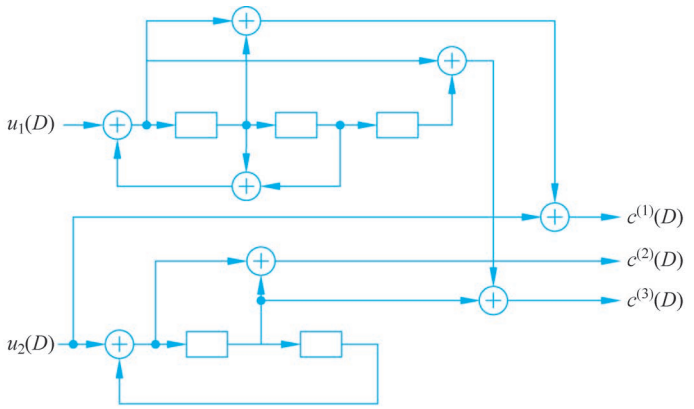
例 5.5.1 一个码率为 2/3 的卷积码的生成矩阵如下：

$$\boldsymbol{G}(D)=\begin{bmatrix}\dfrac{1+D}{1+D+D^2} & 0 & 1+D\\ 1 & \dfrac{1}{1+D} & \dfrac{D}{1+D^2}\end{bmatrix} \tag{5.5.2}$$

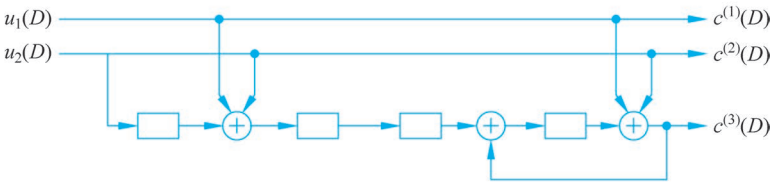
如将矩阵的第一行乘以$(1+D+D^2)/(1+D)$，把新得到的第一行加到第二行，把新得到的第二行乘以$(1+D)$，就可以得到系统形式

$$\boldsymbol{G}_0(D)=\begin{bmatrix}1 & 0 & 1+D+D^2\\ 0 & 1 & \dfrac{1+D^3+D^4}{1+D}\end{bmatrix} \tag{5.5.3}$$

基于式(5.5.2)、式(5.5.3)的编码器如图 5.5.2 所示。



(a) 2/3码率卷积码$\boldsymbol{G}(D)$的编码器



(b) 2/3码率卷积码$\boldsymbol{G}_0(D)$的编码器

图 5.5.2 例 5.5.1 中给出的 2/3 码率卷积码的编码器实现

由子矩阵 $\boldsymbol{P}(D)$($\boldsymbol{G}_0(D)$的最右一列)可得

$$\boldsymbol{H}_0(D)=\begin{bmatrix}1+D+D^2 & \dfrac{1+D^3+D^4}{1+D} & 1\end{bmatrix}$$

这是因为 $\boldsymbol{H}_0(D)=[\boldsymbol{P}^{\mathrm{T}}(D)|\boldsymbol{I}]$。$\boldsymbol{G}(D)$中元素分母的最小公倍式是$(1+D+D^2)(1+$

D^2),所以可得一个多项式形式的生成矩阵,用 $\boldsymbol{G}_{\mathrm{p}}(D)$来表示:

$$\boldsymbol{G}_{\mathrm{p}}(D)=\begin{bmatrix}(1+D)^3 & 0 & (1+D+D^2)(1+D)^3\\ (1+D+D^2)(1+D^2) & (1+D+D^2)(1+D) & D(1+D+D^2)\end{bmatrix}$$

将 $\boldsymbol{H}_0(D)$中的每个元素乘以 $1+D$,可得

$$\boldsymbol{H}_{\mathrm{p}}(D)=[1+D^3 \quad 1+D^3+D^4 \quad 1+D]$$

任意可实现的卷积码都必定存在一个多项式形式的生成矩阵和校验矩阵。需要强调的是,$\boldsymbol{G}(D)$、$\boldsymbol{G}_0(D)$和 $\boldsymbol{G}_{\mathrm{p}}(D)$都生成同一个码,即它们的行张成相同的子空间。类似线性分组码,卷积码的生成矩阵也可以写成码字的形式。

例 5.5.2 码率为 1/2、生成矩阵是 $\boldsymbol{G}(D)=[g^{(1)}(D) \quad g^{(2)}(D)]=[1+D+D^2 \quad 1+D^2]$的卷积码,$\boldsymbol{g}^{(1)}=[g_0^{(1)}\,g_1^{(1)}\,g_2^{(1)}]=[1 \quad 1 \quad 1]$,$\boldsymbol{g}^{(2)}=[g_0^{(2)}\,g_1^{(2)}\,g_2^{(2)}]=[1 \quad 0 \quad 1]$。对应复用码字的生成矩阵可写成如下形式:

$$\boldsymbol{G}'=\begin{bmatrix}g_0^{(1)}g_0^{(2)} & g_1^{(1)}g_1^{(2)} & g_2^{(1)}g_2^{(2)} & & \\ & g_0^{(1)}g_0^{(2)} & g_1^{(1)}g_1^{(2)} & g_2^{(1)}g_2^{(2)} & \\ & & g_0^{(1)}g_0^{(2)} & g_1^{(1)}g_1^{(2)} & g_2^{(1)}g_2^{(2)} \\ & & \ddots & \ddots & \ddots\end{bmatrix}$$

$$=\begin{bmatrix}11 & 10 & 11 & & \\ & 11 & 10 & 11 & \\ & 11 & 10 & 11 & \\ & & \ddots & \ddots & \ddots\end{bmatrix}$$

与分组码不同,这是一个半无限长的生成矩阵。

一般来说,卷积码的参数 n 和 k 的典型值都非常小。如 $k=1,2,3$ 和 $n=2,3,4$,典型的码率如 1/2、1/3、1/4、2/3 和 3/4,其中 1/2 是最常见的码率。更高的码率一般通过降低码率的卷积码进行**凿孔**实现,即将编码器输出的比特进行周期性地删除而不传输。例如,为了通过将 1/2 码率的卷积码进行凿孔来得到 2/3 码率的卷积码,只需要在每四个编码输出比特中删除一个。由于每组的四个码字比特是由两个信息比特产生的,但是四个里面只有三个被传输,于是就实现了码率 2/3。因为编译码器的复杂度和性能,码率接近 1(如 0.95)的卷积码在实际中是不存在的,不管是否使用凿孔。卷积码译码器的复杂度正比于 2^{μ},这里 μ 是编码器的记忆长度,即编码器记忆单元的个数。典型的 μ 值在 10 以下,工业标准中码率为 1/2 卷积码对应的 $\mu=6$,其生成多项式是 $g^{(1)}(D)=1+D+D^2+D^3+D^6$ 和 $g^{(2)}(D)=1+D^2+D^3+D^5+D^6$。常见的做法是用八进制数表示这些多项式,上述生成多项式可以表示成 $g_{\mathrm{oct}}^{(1)}=117$ 和 $g_{\mathrm{oct}}^{(2)}=155$。

5.5.2.2 图表示

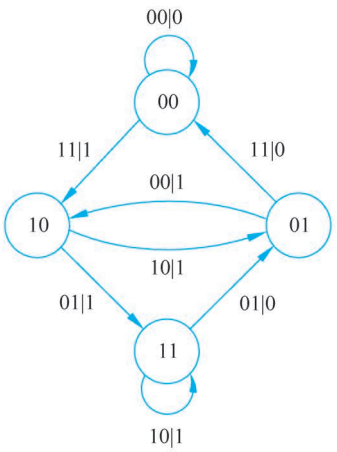
卷积码的代数描述有利于编码器的实现和分类。本节介绍的图表示方法则有利于卷积码的译码、设计、性能分析等。

下面以 1/2 码率的卷积码为例进行描述,其生成矩阵 $\boldsymbol{G}(D)=[1+D+D^2 \quad 1+D^2]$,编码器如图 5.5.1 所示。从图及状态的定义中可以得到状态转移表(表 5.5.1),按

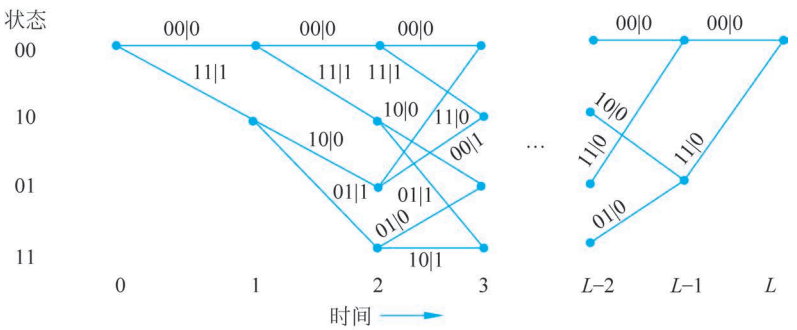
表可以画出这个码的有限状态转移图，如图 5.5.3(a)所示。状态图对于通过分析确定卷积码的距离谱很有用，而且可以直接导出码的网格图，网格图是在水平方向上加上离散时间维度的状态图(图 5.5.3(b))，编码器从 00 状态开始。值得注意的是，在 i 和 $i+1(i\geqslant 2)$节之间的网格，本质上是状态图的完全复制。同时，任意给定长度的码序列表，可以通过追溯网格图上同样长度的所有可能路径，并在每条路径上截取标记每个分支(或边)的码符号得到。

表 5.5.1　状态转移表

输入 u_i	当前状态 $u_{i-1}u_{i-2}$	下一状态 u_iu_{i-1}	输出 $c_i^{(1)}c_i^{(2)}$
0	00	00	00
1	00	10	11
0	01	00	11
1	01	10	00
0	10	01	10
1	10	11	01
0	11	01	01
1	11	11	10



(a) 有限状态图



(b) 编码器的网格图

图 5.5.3　有限状态转移图和编码器的网格图

考虑基于网格图译码的时候，将会发现，除非网格图的最后一个状态是已知的，否则得到的末尾几个译码比特（大概 5μ 比特）将有些不可靠。为了解决这种情况，系统设计者通常会将网格图终止在一个已知的状态里。很明显，这需要在信息序列后面再添加 μ 比特，这样做增加了编码开销（或者说降低了码率）。而一种能够保持网格图终止优势，同时可避免降低编码效率的办法是咬尾。对于咬尾卷积码，用信息序列的最后 μ 比特初始化编码器的状态，使得编码器的初始状态和最终状态都是一样的。

5.5.3 卷积码的译码

目前卷积码有大数逻辑译码器、序列译码器、Viterbi 译码器和 BCJR 译码器，算法根据实际应用选择。通过 Viterbi 算法实现的最大似然序列译码器（MLSD）可最小化码字序列的错误概率，通过 BCJR 算法实现的逐比特最大化后验概率（MAP）译码器则可最小化信息误码率。一般来讲，这两种译码器的性能特性不管是从比特错误概率还是码字错误概率来讲都是相当接近的。但 BCJR 算法的译码复杂度约为 Viterbi 算法的 3 倍。本节主要讨论 Viterbi 译码算法。

我们关注的是二元对称信道（BSC）和二进制输入加性高斯白噪声信道（BI-AWGNC）。为了满足统一的要求，对于两种信道用 x_i 表示第 i 个输入，用 y_i 表示第 i 个信道输出。给定信道输入 $x_i=c_i\in\{0,1\}$ 和信道输出 $y_i\in\{0,1\}$，BSC 的信道转移概率为

$$P(y_i\neq x\mid x_i=x)=\varepsilon$$
$$P(y_i=x\mid x_i=x)=1-\varepsilon$$

式中：ε 为交叉概率。

对于 BI-AWGNC，码字比特映射到信道输入是 $x_i=(-1)^{c_i}\in\{\pm1\}$。BI-AWGNC 的信道转移概率密度函数（PDF）为

$$p(y_i\mid x_i)=\frac{1}{\sqrt{2\pi}\sigma}\exp[-(y_i-x_i)^2/(2\sigma^2)]$$

式中：σ^2 为零均值的高斯噪声采样 n_i 的方差。

信道将 n_i 直接加到传输值 x_i 上（因此，$y_i=x_i+n_i$）。首先考虑 BSC，它的 ML 判决是

$$\hat{\boldsymbol{c}}=\arg\max_{\boldsymbol{c}}P(\boldsymbol{y}\mid\boldsymbol{x})$$

可以化简成

$$\hat{\boldsymbol{c}}=\arg\min_{\boldsymbol{c}}d_{\mathrm{H}}(\boldsymbol{y},\boldsymbol{x})$$

式中：$d_{\mathrm{H}}(\boldsymbol{y},\boldsymbol{x})$ 表示 $\boldsymbol{y}$ 和 $\boldsymbol{x}$ 之间的汉明距离（注意 $\boldsymbol{x}=\boldsymbol{c}$）。

BI-AWGNC 的 ML 判决是

$$\hat{\boldsymbol{c}}=\arg\max_{\boldsymbol{c}}p(\boldsymbol{y}\mid\boldsymbol{x})$$

可以简化成

$$\hat{\boldsymbol{c}}=\arg\min_{\boldsymbol{c}}d_{E}(\boldsymbol{y},\boldsymbol{x})$$

式中：$d_{\mathrm{E}}(\boldsymbol{y},\boldsymbol{x})$ 表示 $\boldsymbol{y}$ 和 $\boldsymbol{x}$ 之间的欧几里得距离（注意 $\boldsymbol{x}=(-1)^{\boldsymbol{c}}$）。

因此，对于 BSC（BI-AWGNC），MLSD 将会选择在汉明（欧几里得）距离的意义下最

接近信道输出 $\boldsymbol{y}$ 的码字序列 $\boldsymbol{c}$。从理论上来说，由于网格图列举了所有的码字序列，因此可以用穷搜索的办法在网格图上寻找最接近 $\boldsymbol{y}$ 的序列。L 条网格分支的计算：

$$\Gamma_L=\sum_{l=1}^{L}\lambda_l \tag{5.5.4}$$

式中：λ_l 为第 l 个分支度量，且有

$$\lambda_l=\sum_{j=1}^{n}d_{\mathrm{H}}(y_l^{(j)},x_l^{(j)})=\sum_{j=1}^{n}y_l^{(j)}\oplus x_l^{(j)}(\mathrm{BSC}) \tag{5.5.5}$$

$$\lambda_l=\sum_{j=1}^{n}d_E^2(y_l^{(j)},x_l^{(j)})=\sum_{j=1}^{n}(y_l^{(j)}-x_l^{(j)})^2(\mathrm{BI\text{-}AWGNC}) \tag{5.5.6}$$

在 BSC 下，$x_l^{(j)}=c_l^{(j)}$，而在 BI-AWGNC 下，$x_l^{(j)}=(-1)^{c_l^{(j)}}$。式(5.5.5)中的度量称为汉明度量，式(5.5.6)中的度量称为欧几里得度量。

很明显，$\arg\min_{\boldsymbol{c}} d_{\mathrm{H}}(\boldsymbol{y},\boldsymbol{x})$ 和 $\arg\min_{\boldsymbol{c}} d_{\mathrm{E}}(\boldsymbol{y},\boldsymbol{x})$ 的计算量是非常庞大的，这是因为对于码率为 $k/(k+1)$ 的码来说，网格图上 L 节意味着有 2^{kL} 个码字序列。但是，Viterbi 根据以下观察发现，在不损失性能的情况下这个复杂度可以降低到 $O(2^{\mu})$。考虑在图 5.5.2(b)中以[00 00 00]和[11 10 11]开始的两个码字序列，它们对应的输入序列是[0 0 0…]和[1 0 0…]。观察到对应这两个序列的网格路径在状态 0/时间 0 开始岔开，而在经过三个分支后(或者等效地说，经过网格图中的三节后)又汇聚到状态 0。经过三节后汇合的重要性在于此后的网格中这两条路径有相同的扩展。因此，如果这两条路径其中一条在 $l=3$ 时有着较大的累积度量，那么这条路径加上扩展以后也会拥有较大的累积度量。因此，从长远角度考虑可以把 $l=3$ 时度量较大的路径移除。这条被保留的路径称为幸存路径。这样的做法将会应用到网格中所有汇聚路径，以及在 $l>3$ 时每个网格节点拥有多于两条汇聚路径的网格中。这就引出了以下基于 MLSD 的 Viterbi 算法。

算法 5.5.1　Viterbi 算法

定义

(1) $\lambda_l(s',s)$是从 $l-1$ 时刻状态 s'转移到 l 时刻状态 s 的分支度量，这里 $s',s\in\{0,1,\cdots,2^{\mu}-1\}$。

(2) $\Gamma_{l-1}(s')$是在 $l-1$ 时刻幸存状态 s'的累积度量，即幸存路径的分支度量和。

(3) $\Gamma_l(s',s)$是从 $l-1$ 时刻状态 s'延展到 l 时刻状态 s 路径的暂定累积度量，$\Gamma_l(s',s)=\Gamma_{l-1}(s')+\lambda_l(s',s)$。

加-比-选(Add-Compare-Select，ACS)迭代

初始化　设定 $\Gamma_0(0)=0$ 和 $\Gamma_0(s')=-\infty$，对于所有 $s'\in\{1,\cdots,2^{\mu}-1\}$(编码器、网格被初始化为状态 0)。

for $l=1$ to L

(1) 计算可能的分支度量 $\lambda_l(s',s)$。

(2) 对于在 $l-1$ 时刻的每个状态 s'和 l 时刻从状态 s'出发所有可能到达的状态 s，计算从状态 s'延展到状态 s 路径的暂定累积度量 $\Gamma_l(s',s)=\Gamma_{l-1}(s')+\lambda_l(s',s)$。

(3) 对于在时刻 l 的每个状态 s，从度量 $\Gamma_l(s',s)$选出并记录下具有最小度量的路径。这时状态 s 的累积度量是 $\Gamma_l(s)=\min_{s'}\{\Gamma_l(s',s)\}$

end

下面讨论几种选择最大似然码序列的方法。

面向分组的方法 1：假设一个信息序列长为 kL bit，在经过 L 次加-比-选迭代后，选择具有最优累积度量的路径(若相同，则任意确定)，L 在这里必须足够大，从而能充分发挥这个码的全部优势，一般有 $L>50\mu$。

面向分组的方法 2：假设一个非递归卷积码，在信息序列后面加上 μ 个 0，使得网格在信息序列的末端可以回归 0 状态。在最后一次 ACS 迭代后，最大似然路径即是回归到状态 0 的幸存路径。同样，这里的 L 也必须足够大。

面向流的方法 1：这个方法利用了幸存路径都会有共同的“尾巴”这个特点。在第 l 次 ACS 迭代后，译码器将会沿着任意的幸存路径回溯 δ 个分支，然后将网格第 $l-\delta$ 节分支上的标记作为信息比特输出。这是一个很有效的滑动窗口译码器，其顺着网格的长度滑动，在 δ 节内存储和处理信息。译码延迟 δ 的值通常直接使用记忆长度的 4 倍或者 5 倍。然而，通过计算机仿真来确定这个值会更好，因为这个值是根据具体的码来确定的。也就是说，可以在码的网格图上使用计算机搜索算法，从在 0 状态/0 时刻发散于全零路径、后来再聚合的最小重量码字序列中确定所需要的最大路径长度。δ 可以设定成比这个最大路径长度稍微大一点的值。

面向流的方法 2：在第 l 次 ACS 迭代后，选择具有最优累积度量的网格路径，然后回溯 δ 个分支，选择网格第 $l-\delta$ 节中分支上所标记的比特作为对应的输出判决。

针对 BI-AWGNG 的欧几里得距离度量(式 5.5.6)可以做如下简化：

$$\begin{aligned}\arg\min_{\boldsymbol{c}} d_{\mathrm{E}}^2(\boldsymbol{y},\boldsymbol{x}) &= \arg\min_{\boldsymbol{c}} \sum_{l=1}^{L}\sum_{j=1}^{n}(y_l^{(j)}-x_l^{(j)})^2 \\ &= \arg\min_{\boldsymbol{c}} \sum_{l=1}^{L}\sum_{j=1}^{n}[(y_l^{(j)})^2+(x_l^{(j)})^2-2y_l^{(j)}x_l^{(j)}] \\ &= \arg\max_{\boldsymbol{c}} \sum_{l=1}^{L}\sum_{j=1}^{n} y_l^{(j)}x_l^{(j)}\end{aligned}$$

最后一行是因为第二行的平方项是独立于 $\boldsymbol{c}$ 的。因此，式(5.5.6)中的 AWGN 分支度量可以用下面的相关度量代替：

$$\lambda_l=\sum_{j=1}^{n} y_l^{(j)}x_l^{(j)} \tag{5.5.7}$$

以上式子在 BI-AWGNC 下成立。

图 5.5.4 展示了一个 BSC 上应用面向分组判决方法 1 的 Viterbi 译码例子。这里 $\boldsymbol{y}=[00,01,00,01]$，非幸存路径用“×”标示，累积度量在汇聚分支附近用粗体写出。例子中不存在彻底的最小距离路径，任一距离为 2 的路径与 ML 路径一样。

图 5.5.5 展示了一个 BI-AWGNC 上应用面向分组判决方法 1 和相关度量的 Viterbi 译码的例子。这里 $\boldsymbol{y}=[-0.7,-0.5,-0.8,-0.6,-1.1,+0.4,+0.9,+0.8]$，非幸存路径用“×”标示，累积度量在汇聚分支附近用粗体写出。ML 路径就是累积度量为 3.8 的路径。

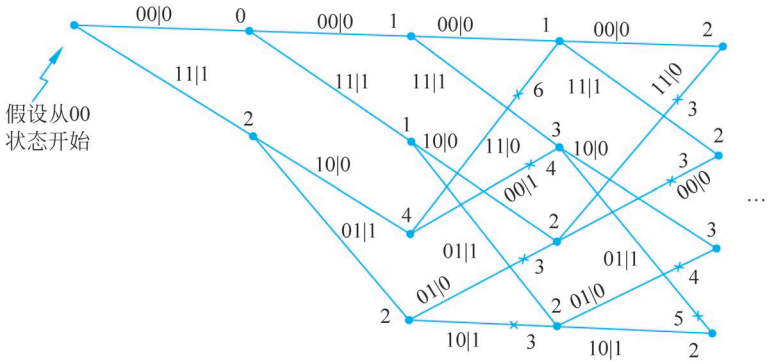


图 5.5.4　图 5.5.3 中的码在 BSC 上使用 Viterbi 译码的例子

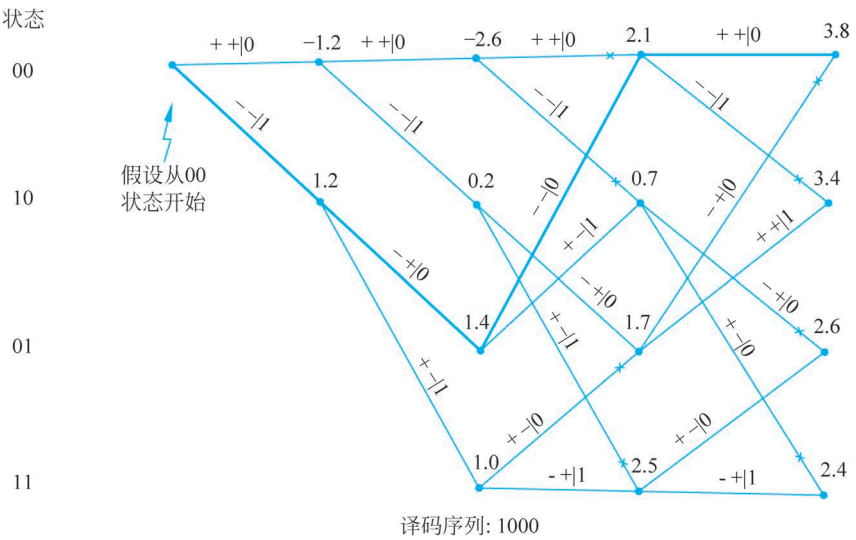


图 5.5.5　图 5.5.3 中的码在 BI-AWGNC 上使用 Viterbi 译码的例子

5.5.4　卷积码的应用

5.5.4.1　IEEE 802.16 中的卷积码

IEEE 802.16 工作组专门开发宽带固定无线技术标准，IEEE 802.16 标准的颁布推动了宽带无线接入的发展，是点对多点宽带固定无线接入系统的权威规范。在该标准的物理层规范中采用正交频分复用（OFDM）调制，信道编码分为扰码、前向纠错（FEC）和交织三步。

FEC 包括级联的卷积码（内码）和 RS 码（外码），编码过程是首先数据以块的形式经过 RS 编码器，然后经过 0 截止的卷积码编码器。RS 码采用 GF(2^8)中的(255，239，8)码，编码生成多项式为

$$g(x)=(x+\alpha)(x+\alpha^2)\cdots(x+\alpha^{2t}),\quad \alpha=02\text{HEX}$$

域生成多项式为 $g(x)=x^8+x^4+x^3+x^2+1$。码字可以缩短或打孔，以适应不同大小

的数据块和不同的纠错能力。每个 RS 数据块进行卷积编码，采用(2,1,7)卷积码，生成多项式为 $G=[171,133]$，如图 5.5.6 所示。

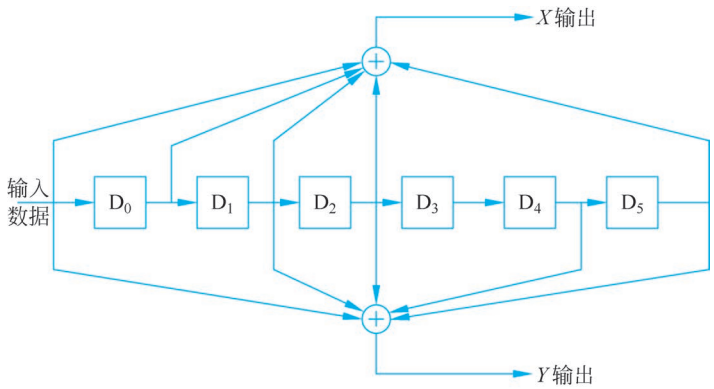


图 5.5.6　码率为 1/2 的卷积编码器

表 5.5.2 列出了为实现不同编码码率而使用的打孔模式和串联顺序，表中的“1”表示发送比特，“0”表示删除比特。

表 5.5.2　内卷积编码及配置

	码　率			
Rate	1/2	2/3	3/4	5/6
d_{free}	10	6	5	4
X	1	10	101	10101
Y	1	11	110	11010
XY	X_1Y_1	$X_1Y_1Y_2$	$X_1Y_1Y_2X_3$	$X_1Y_1Y_2X_3Y_4X_5$

不同调制方式的块长度和码率见表 5.5.3。

表 5.5.3　不同调制方式的块长度和码率

调制方式	未编码块长度/字节	编码块长度/字节	总码率	RS 码率	CC 码率
BPSK	12	24	1/2	(12,12,0)	1/2
QPSK	24	48	1/2	(32,24,4)	2/3
QPSK	36	48	3/4	(40,36,2)	5/6
16QAM	48	96	1/2	(64,48,8)	2/3
16QAM	72	96	3/4	(80,72,4)	5/6
64QAM	96	144	2/3	(108,96,6)	3/4
64QAM	108	144	3/4	(120,108,6)	5/6

5.5.4.2　CCSDS 中的卷积码

国际空间数据系统咨询委员会(CCSDS)标准是国际上广泛应用于深空通信领域的通信标准，该标准集合了目前在深空遥控、测控和通信标准等方面的最新研究成果，至今已发布了涵盖物理层调制解调、信道编译码、数据链路层协议和图像压缩等领域的众多标准，并为世界众多学术机构认可。CCSDS 建议书最初针对民用目的而开发，因其具有

卓越性能、良好经济效益和广泛适应性等特点，近年来越来越受到世界各国军事组织的关注，已逐步被世界各国应用于军用航天任务。

CCSDS 建立了包括卷积码、RS 码、级联码、Turbo 码和 LDPC 码等信道码在内的一系列信道码标准。这些高增益信道编码通过优异的编译码算法设计获得可观的信道增益，有效降低信号解调阈值，进而抵消信号在长距离空间传播过程中带来的能量损失。其中，CCSDS 标准推荐采用约束长度为 7bit、码率为 1/2 的卷积码，通过打孔抽取可得到 2/3、3/4、5/6 和 7/8，共 4 种码率。1/2 编码器结构如图 5.5.7 所示，卷积码抽取参数如表 5.5.4 所示。

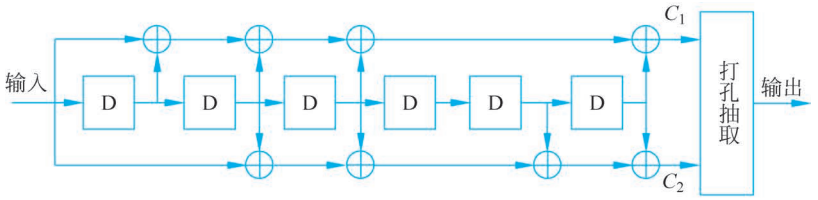


图 5.5.7　1/2 编码器结构

表 5.5.4　卷积码抽取参数

抽取样式	码　率	输　出
C_1：10 C_2：11	2/3	$C_1(1)C_2(1)C_2(2)\cdots$
C_1：101 C_2：110	3/4	$C_1(1)C_2(1)C_2(2)C_1(3)\cdots$
C_1：10101 C_2：11010	5/6	$C_1(1)C_2(1)C_2(2)C_1(3)C_2(4)C_1(5)\cdots$
C_1：1000101 C_2：1111010	7/8	$C_1(1)C_2(1)C_2(2)C_2(3)C_2(4)C_1(5)C_2(6)C_1(7)\cdots$

5.5.4.3　移动通信中的卷积码

WCDMA 中有三种方式用于专用物理信道的编码，分别为卷积码（码率为 1/2 或 1/3）、Turbo 码（只有 1/3 码率）和没有信道编码。卷积编码主要用于误码率为 10^{-3} 级别的业务，典型的有传统的话音业务和低速信令的传输。

WCDMA 中定义的卷积编码器如图 5.5.8 所示，其约束长度 $K=9$，码率为 1/3 或 1/2。码率为 1/3 的编码器输出顺序为 Output0，Output1，Output2，Output0，Output1，Output 2，Output 0，…，Output2。而码率为 1/2 的编码器输出顺序为 Output 0，Output 1，Output 0，Output 1，Output 0，…，Output 1。移位寄存器的初始状态应为全“0”。因此，在编码前需要将 8 个 0 值尾比特添加到码块，以清零编码器。

从编码理论的角度考虑，卷积码性能主要与其编码效率、约束长度及自由距离有关。编码效率越小，编码冗余度就越大，纠错性能越好，占用带宽也越大；约束长度越大，其纠错性能越好，但译码复杂度的指数增加，$K=9$ 是一个比较折中的方案。最后，在相同的编码效率和约束长度下，自由距离越大越好。$K=9$，编码效率为 1/2、1/3 的卷积码的最

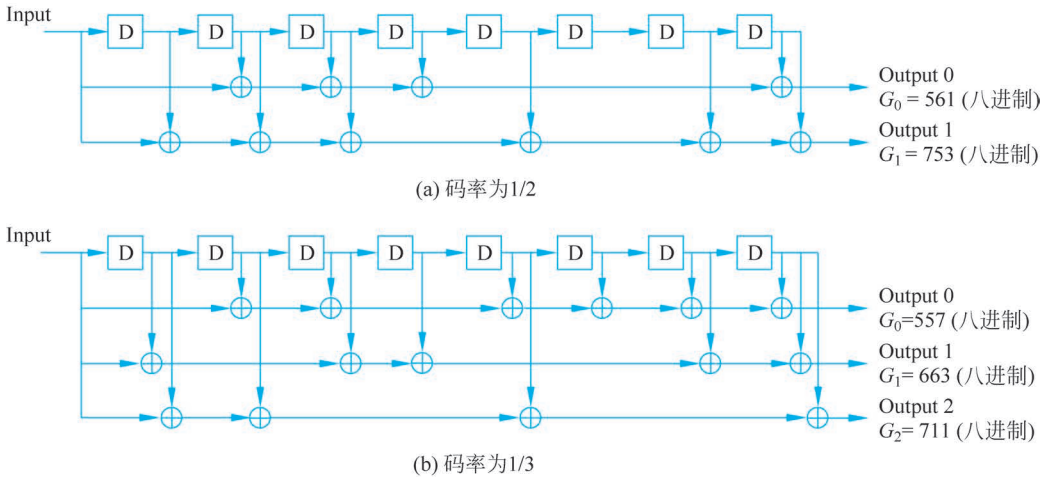


图 5.5.8　码率为 1/2 和 1/3 的卷积编码器结构

大自由距离为 12 和 18，其生成多项式分别为$(561)_8$、$(753)_8$ 和$(557)_8$、$(663)_8$、$(711)_8$。

图 5.5.9 给出了 LTE 系统中的咬尾卷积码编码器结构，其中寄存器的个数 $m=6$，码率 $R=1/3$，编码器输入的信息比特流为 $\boldsymbol{u}$，输出为$(\boldsymbol{c}^{(0)},\boldsymbol{c}^{(1)},\boldsymbol{c}^{(2)})$。

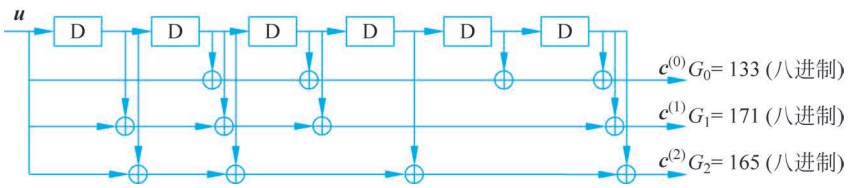


图 5.5.9　LTE 系统中的咬尾卷积码编码器结构

咬尾卷积码已应用于 4G 中的 LTE 系统中，并且成为 5G 中超高可靠与低时延通信(URLLC)场景中的备选编码方案。LTE 系统中的咬尾卷积码直接用信息序列的最后 6 个比特将编码寄存器初始化。这样全部编码后寄存器的结束状态将和编码前的起始状态相同，这样的编码方式没有传输额外的比特，从而提高了编码效率。

5.5.4.4　卫星导航系统中的卷积码

全球卫星导航系统(Global Navigation Satellite System，GNSS)一般由空间星座、地面监控和用户设备三部分构成。空间星座是指 GNSS 中的导航卫星，该部分的作用是进行数据观测，并将接收的导航信息进行一定的处理后将其发送给地球用户。地面监控部分是整个 GNSS 能否正常运行的保障，它主要的作用是对空间星座部分的卫星进行观测与跟踪，校对卫星的时钟、轨道等误差，使整个系统调整到正常运行状态。用户设备部分的功能是接收导航信息，并将接收到的信息进行各项处理(如数字化、功率放大、频率变换等)，从而得到有效的用户自身的时间、速度和位置等参数，以实现定位。导航电文是 GNSS 信号最主要的组成部分，它是导航卫星发送给接收机(用户)的描述卫星的各项运行状态参数的一段数据码信息。

导航电文的纠错编码作为导航电文设计的重要组成部分，为了保证 GNSS 导航电文

传输的可靠性，越来越多的编译码方法运用到卫星导航系统中。目前 GNSS 中所涉及的电文编译码方法有 Hamming 码，CRC 码、BCH 码、卷积码、LDPC 和交织编码等。

在全球定位系统（GPS）及伽利略（Galileo）系统的导航电文中，均采用了（2，1，6）卷积码，其编码结构如图 5.5.10 所示。Galileo 系统与 GPS 卷积码参数相同，区别在于 GPS 为无卷尾的卷积码，Galileo 为有卷尾卷积码，即 Galileo 系统的卷积码在编码的尾部要添加“0”（卷尾），使得卷积码从全“0”状态开始，最终编码结束状态又回到全“0”状态。

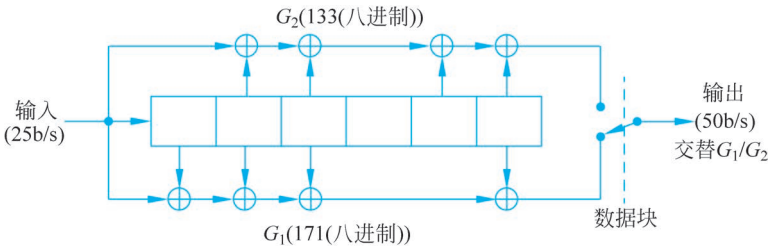


图 5.5.10　GPS（Galileo 系统）中的卷积码结构

小结

纠错能力与最小距离 d_0 的关系：

对于（n，k）分组码，一般有以下结论。

若码的最小距离满足 $d_0 \geqslant e+1$，则码的检错能力为 e。

若码的最小距离满足 $d_0 \geqslant 2t+1$，则码的纠错能力为 t。

若码的最小距离满足 $d_0 \geqslant e+t+1(e>t)$，则该码能纠正 t 个错误同时检测 e 个错误。

联合 ε 典型序列：

设 $(\boldsymbol{x},\boldsymbol{y}) \in (X^n; Y^n)$ 是 n 长随机序列对，对于任意小的正数 $\varepsilon>0$，存在足够大的 n，若满足下列条件，则称 $(\boldsymbol{x},\boldsymbol{y})$ 为**联合 ε 典型序列**。

$$\left|\frac{1}{n}\log P(\boldsymbol{x})+H(X)\right|<\varepsilon$$

$$\left|\frac{1}{n}\log P(\boldsymbol{y})+H(Y)\right|<\varepsilon$$

$$\left|\frac{1}{n}\log P(\boldsymbol{xy})+H(XY)\right|<\varepsilon$$

联合渐近等同分割性：

对于任意小的 $\varepsilon>0$，$\delta>0$，当 n 足够大时，则有以下性质。

典型序列 $\boldsymbol{x}$、$\boldsymbol{y}$ 和联合典型序列 $(\boldsymbol{x},\boldsymbol{y})$ 满足

$$2^{-n[H(X)+\varepsilon]} \leqslant P(\boldsymbol{x}) \leqslant 2^{-n[H(X)-\varepsilon]}$$

$$2^{-n[H(Y)+\varepsilon]} \leqslant P(\boldsymbol{y}) \leqslant 2^{-n[H(Y)-\varepsilon]}$$

$$2^{-n[H(XY)+\varepsilon]} \leqslant P(\boldsymbol{xy}) \leqslant 2^{-n[H(XY)-\varepsilon]}$$

典型序列集 $G_\varepsilon(X)$、$G_\varepsilon(Y)$和联合典型序列集 $G_\varepsilon(XY)$满足

$$P(G_\varepsilon(X)) \geqslant 1-\delta$$

$$P(G_\varepsilon(Y)) \geqslant 1-\delta$$

$$P(G_\varepsilon(XY)) \geqslant 1-\delta$$

以 N_G 表示典型序列集中元素个数，则 $G_\varepsilon(X)$、$G_\varepsilon(Y)$和 $G_\varepsilon(XY)$中序列个数满足

$$(1-\delta)2^{n[H(X)-\varepsilon]} \leqslant N_{G_x} \leqslant 2^{n[H(X)+\varepsilon]}$$

$$(1-\delta)2^{n[H(Y)-\varepsilon]} \leqslant N_{G_y} \leqslant 2^{n[H(Y)+\varepsilon]}$$

$$(1-\delta)2^{n[H(XY)-\varepsilon]} \leqslant N_{G_{xy}} \leqslant 2^{n[H(XY)+\varepsilon]}$$

有噪信道编码定理：有噪信道的信道容量为 C，当信息传输率 $R<C$，只要码长 n 足够长，总可以在输入 X^n 符号集中找出一组码$(Q=2^{nR}, n)$和相应的译码规则，使译码错误概率任意小。

线性分组码：

GF(2)域上的 n 维线性空间 $\boldsymbol{V}_n$ 中的一个 k 维子空间 $\boldsymbol{V}_{n,k}$。

一致校验矩阵 $\boldsymbol{H}$($r\times n$ 矩阵)满足

$$\boldsymbol{HC}^{\mathrm{T}}=\boldsymbol{0}^{\mathrm{T}}, \quad \boldsymbol{CH}^{\mathrm{T}}=\boldsymbol{0}$$

生成矩阵 $\boldsymbol{G}$($k\times n$ 矩阵)满足

$$\boldsymbol{HG}^{\mathrm{T}}=\boldsymbol{0}^{\mathrm{T}}, \quad \boldsymbol{GH}^{\mathrm{T}}=\boldsymbol{0}$$

标准生成矩阵和标准校验矩阵

$$\boldsymbol{G}_0=[\boldsymbol{I}_k\boldsymbol{P}], \quad \boldsymbol{H}_0=[\boldsymbol{QI}_r], \text{且 } \boldsymbol{P}=\boldsymbol{Q}^{\mathrm{T}}$$

标准阵列译码表见表 5.3.3。

伴随式：

$$\boldsymbol{S}=\boldsymbol{EH}^{\mathrm{T}}=\boldsymbol{RH}^{\mathrm{T}}$$

Hamming 码：

码　　长：	$n=2^m-1$
信息位数：	$k=2^m-m-1$
监督码位：	$r=n-k=m$
最小距离：	$d=3$
纠错能力：	$t=1$

基本码限：

Singleton 限：任一线性分组码的最小距离(或最小重量)d_0 均满足 $d_0\leqslant n-k+1$。

Hamming 限：若 $\boldsymbol{C}$ 是 k 维 n 重二元码，已知 k 时，要使 $\boldsymbol{C}$ 能纠正 t 个错，则必须有不少于 r 个校验位，并且使 r 满足 $2^r-1\geqslant\sum_{i=1}^{t}\mathrm{C}_n^i$。

循环码：

在任一个 GF(q)(q 为素数或素数幂)上的 n 维线性空间 $\boldsymbol{V}_n$ 中，一个 n 重子空间 $\boldsymbol{V}_{n,k}\in\boldsymbol{V}_n$，若对任何一个 $\boldsymbol{C}_i=(c_{n-1},c_{n-2},\cdots,c_0)\in\boldsymbol{V}_{n,k}$，恒有 $\boldsymbol{C}_i'=(c_{n-2}\cdots c_0c_{n-1})\in\boldsymbol{V}_{n,k}$，则称 $\boldsymbol{V}_{n,k}$ 是**循环子空间**或**循环码**。

如果一个码的所有码多项式都是多项式 $g(x)$ 的倍式，则称 $g(x)$ 为生成该码，且称 $g(x)$ 为该码的**生成多项式**，所对应的码字称为**生成子**或**生成子序列**。

BCH 码：

常用的二元 BCH 码是本原 BCH 码，其参数及其关系式：

$$\begin{aligned}&\text{码字长度：}\quad n=2^m-1\\&\text{校验位数：}\quad n-k\leqslant mt\\&\text{最小距离：}\quad d\geqslant 2t+1\end{aligned}$$

其中，m 为正整数，一般 $m\geqslant 3$，纠错位数 $t<(2^m-1)/2$。

RS 码：

RS 码是一类具有很强纠错能力的非二进制 BCH 码，码元符号取自有限域 GF(q)能纠正 t 个错误的 RS 码具有如下参数：

$$\begin{aligned}&\text{码字长度：}\quad n=q-1(\text{符号})\\&\text{校验位数：}\quad n-k=2t(\text{符号})\\&\text{最小距离：}\quad d=2t+1(\text{符号})\end{aligned}$$

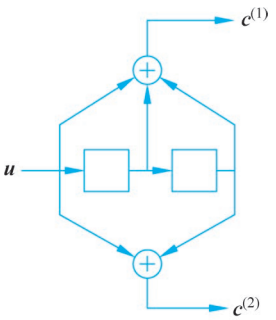
卷积码的定义：编码器的输出 $\boldsymbol{c}^{(j)}$ 是输入 $\boldsymbol{u}$ 和冲激响应 $\boldsymbol{g}^{(j)}$ 的卷积，即有

$$\boldsymbol{c}^{(j)}=\boldsymbol{u}\circledast\boldsymbol{g}^{(j)}$$

时域里的卷积运算对应变换域里的乘积运算，上述方程还可以重新写成

$$c^{(j)}(D)=u(D)g^{(j)}(D)$$

其中，D 的多项式系数是对应向量的元素，所以 $g^{(1)}(D)=1+D+D^2$，$g^{(2)}(D)=1+D^2$，等价于单位离散时间延迟算子 z^{-1}。



卷积码的代数描述：对于一个二元卷积码 $\boldsymbol{C}$，码字 $\boldsymbol{c}(D)\in\boldsymbol{C}$ 有如下形式，即

$$\boldsymbol{c}(D)=[c^{(1)}(D)\quad c^{(2)}(D)\cdots c^{(n)}(D)]$$

式中：$c^{(j)}(D)=\sum_{i=r}^{\infty}c_i^{(j)}D^i$。

其编码过程可写为

$$\boldsymbol{c}(D)=\boldsymbol{u}(D)\boldsymbol{G}(D)$$

其中，$\boldsymbol{u}(D)=[u^{(1)}(D)\quad u^{(2)}(D)\cdots u^{(k)}(D)]$，基向量 $\boldsymbol{g}_i(D)=[g_i^{(1)}(D)\quad g_i^{(2)}(D)\cdots g_i^{(n)}(D)]$构成了 $k\times n$ 的生成矩阵 $\boldsymbol{G}(D)$的行，即 $\boldsymbol{G}(D)$中第 i 行第 j 列的元素是 $g_i^{(j)}(D)$。

最大似然序列译码(MLSD)：

对于 BSC(BI-AWGNC)，MLSD 将会选择在汉明(欧几里得)距离的意义下最接近信道输出 $\boldsymbol{y}$ 的码字序列 $\boldsymbol{c}$。从理论上来说，可以利用穷搜索的办法在网格图上寻找最接近 $\boldsymbol{y}$ 的序列。

维特比(Viterbi)译码：基于幸存路径，开展加-比-选(Add-Compare Select，ACS)迭代的译码算法(算法 5.5.1)。

习题

5.1 在一个 $p=0.05$ 的 BSC 上，使用长度 $n=5$ 的分组码，并希望接收端分组错误概率小于 10^{-4}，最大可能的码率为多少？

5.2 已知一个(7，4)码的生成矩阵为

$$\boldsymbol{G}_0=\begin{bmatrix}1&0&0&0&1&1&1\\0&1&0&0&1&0&1\\0&0&1&0&0&1&1\\0&0&0&1&1&1&0\end{bmatrix}$$

(1) 求该码的全部码字。

(2) 求该码的一致校验矩阵 $\boldsymbol{H}_0$。

(3) 作出标准阵列译码表。

5.3 一个(8，4)系统码，其信息序列为$(m_3m_2m_1m_0)$，码字序列为$(c_7c_6c_5c_4c_3c_2c_1c_0)$，它的校验方程为

$$\begin{cases}c_3=m_3+m_1+m_0\\c_2=m_3+m_2+m_0\\c_1=m_2+m_1+m_0\\c_0=m_3+m_2+m_1\end{cases}$$

求该码的生成矩阵 $\boldsymbol{G}_0$ 和一致校验矩阵 $\boldsymbol{H}_0$，并证明该码最小重量为 4。

5.4 令 $\boldsymbol{H}$ 为某一(n,k)线性分组码的一致校验矩阵，其码的最小重量 d_0 为奇数，现构造一个新码，其一致校验矩阵为

$$\boldsymbol{H}'=\begin{bmatrix}\boldsymbol{H}&&&0\\&&&\vdots\\&&&0\\1&\cdots&1&1\end{bmatrix}$$

试证明：(1) 新码是一个$(n+1,k)$码。

(2) 新码中每个码字重量为偶数。

(3) 新码的最小重量为d_0+1。

5.5 对于一个码长为15的线性码，若允许纠正2个随机错误，需多少个不同的伴随式？至少要多少位校验元？

5.6 令$\boldsymbol{C}_1$是最小距离为d_1、生成矩阵为$\boldsymbol{G}_1=[\boldsymbol{P}_1 \vdots \boldsymbol{I}_k]$的$(n_1,k)$线性系统码，$\boldsymbol{C}_2$是最小距离为$d_2$、生成矩阵$\boldsymbol{G}_2=[\boldsymbol{P}_2 \mid \boldsymbol{I}_k]$的$(n_2,k)$线性系统码。研究具有下述一致校验矩阵的线性码。

$$\boldsymbol{H}=\begin{bmatrix} & \vdots & \boldsymbol{P}_1^{\mathrm{T}} \\ \boldsymbol{I}_{n_1+n_2-k} & \vdots & \boldsymbol{I}_k \\ & \vdots & \boldsymbol{P}_2^{\mathrm{T}} \end{bmatrix}$$

试求：(1) 码长及信息位长度。

(2) 证明此码的最小距离至少为d_1+d_2。

5.7 研究一(n,k)线性码$\boldsymbol{C}$，其生成矩阵$\boldsymbol{G}$不包括零列。将$\boldsymbol{C}$的所有码矢排列成$2^k\times n$的阵，试证明：(1)阵中不含有零列。

(2) 阵的每一列由2^{k-1}个0和2^{k-1}个1组成。

(3) 在特定分量上为0的所有码矢构成$\boldsymbol{C}$的一个子空间，这个子空间的维数有多大？

5.8 证明(n,k)线性码的最小距离d_0满足下述不等式：

$$d_0 \leqslant \frac{n\cdot 2^{k-1}}{2^k-1}$$

提示：利用题5.7(2)的结果，上述界限称为普洛金(Plotkin)限。

5.9 已知(7,4)码的全部码字为：0000000，0001011，0010110，0011101，0100111，0101100，0110001，0111010，1000101，1010011，1011000，1100010，1101001，1110100，1111111，1001110。

(1) 该码是否为循环码？为什么？

(2) 写出该码的生成多项式$g(x)$及标准型的生成矩阵$\boldsymbol{G}_0$。

(3) 写出标准型的一致校验矩阵$\boldsymbol{H}_0$。

5.10 证明$x^{10}+x^8+x^5+x^4+x^2+x+1$为(15,5)循环码的生成多项式，并写出信息多项式为$M(x)=x^4+x+1$时的码多项式(按系统码的形式)。

5.11 一个(n,k)循环码，其生成多项式为$g(x)$。假设n为奇数，且$x+1$不是$g(x)$的因式，试证全1码组是其中的一个码字；若$(x+1)$是$g(x)$的一个因式，证明全1的n重不是码字；若n是偶数，证明全1的n重是一个码字。

5.12 已知$g_1(x)=x^3+x^2+1$，$g_2(x)=x^3+x+1$，$g_3(x)=x+1$，试分别讨论：

(1) $g(x)=g_1(x)g_2(x)$。

(2) $g(x)=g_3(x)g_2(x)$。

两种情况下，由 $g(x)$ 生成的 7 位循环码能检测出哪些类型的单个错误和突发错误？

5.13 令(15,11)循环码的生成多项式为 $g(x)=x^4+x+1$。

(1) 求此码的一致校验多项式。

(2) 求此码的标准型的生成矩阵和一致校验矩阵。

(3) 讨论其纠错能力。

(4) 若信息序列多项式为 $M(x)=x^{10}+x^8+1$，求其编码后的系统型码字。

(5) 求接收码组 $R(x)=x^{14}+x^4+x+1$ 的伴随式，并列表说明 $R(x)$ 的译码过程。

5.14 令 $g(x)$ 是一个长为 n 的二元循环码的生成多项式。

(1) 若 $g(x)$ 中有 $(x+1)$ 因子，证明此码不含有奇数重量码字。

(2) 若 n 是 $g(x)$ 除尽 x^n+1 的最小整数，且 $n\geqslant 3$，证明此码的重量至少为 3。

5.15 令 $\boldsymbol{C}_1$ 和 $\boldsymbol{C}_2$ 分别是 $g_1(x)$ 和 $g_2(x)$ 生成的两个长为 n 的循环码，其最小距离分别为 d_1，d_2，证明既属于 $\boldsymbol{C}_l$ 码又属于 $\boldsymbol{C}_2$ 码的公共码多项式，形成了另一循环码 $\boldsymbol{C}_3$，确定 $\boldsymbol{C}_3$ 码的生成多项式，试讨论 $\boldsymbol{C}_3$ 码的最小距离。

5.16 扩展 Hamming 码扩展后的一致校验矩阵 $\boldsymbol{H}'$ 与原码 $\boldsymbol{H}$ 矩阵关系如下：

$$\boldsymbol{H}'=\begin{array}{c|c} \begin{matrix}0\\0\\\vdots\\0\end{matrix} & \boldsymbol{H} \\ \hline \begin{matrix}1&1&1&1&\cdots&1\end{matrix} & \end{array}$$

同样，删信 Hamming 码的一致校验矩阵 $\boldsymbol{H}''$ 与原码 $\boldsymbol{H}$ 矩阵关系如下

$$\boldsymbol{H}''=\begin{array}{|c|} \hline \boldsymbol{H} \\ \hline \begin{matrix}1&1&\cdots&1\end{matrix} \\ \hline \end{array}$$

计算这两类码的维数，证明 $d_{\min}=4$；试说明扩展和删信 Hamming 码可以纠正单个错误同时检测 2 个错误。

5.17 下列码中哪些与 BCH 码的规则一致？

(1) $(32,21)d_{\min}=5$

(2) $(63,45)d_{\min}=7$

(3) $(63,36)d_{\min}=11$

(4) $(127,103)d_{\min}=7$

5.18 给出长度为 15 的可纠正 3 个错误的 BCH 码的生成子多项式。假设 $\mathrm{GF}(2^4)$ 是用本原多项式 x^4+x^3+1 构造的。

5.19 构造一个 $\mathrm{GF}(2^4)$ 上 $n=15$ 的纠正 2 个错误的 RS 码，找出它的生成多项式和 k，并对接收矢量 $\boldsymbol{R}(x)=\alpha x^3+\alpha^{11}x^7$ 进行译码。

5.20 一个 $\mathrm{GF}(2^m)$ 上的纠正 t 个错误的 RS 码，若它的生成多项式为

$$g(x)=(x+\alpha)(x+\alpha^2)\cdots(x+\alpha^{2t})$$

其中 $\alpha\in GF(2^m)$ 是一个本原域元素，证明该码最小距离为 $2t+1$。

5.21 设(2,1,3)二元卷积码的生成多项式矩阵 $\boldsymbol{G}(D)=[1+D^2+D^3 \quad 1+D+D^2+D^3]$。

(1) 画出编码电路。

(2) 画出 $L=4$ 的信息序列的网格图。

(3) 若通过转移概率 $p=0.01$ 的 BSC 传送，收到的序列为 11 10 00 01 10 00，求译码序列。

5.22 考虑码率为 2/3 的卷积码，其校验矩阵为

$$\begin{aligned}\boldsymbol{H}(D)&=[h_2(D) \quad h_1(D) \quad h_0(D)]\\&=[1+D \quad 1+D^2 \quad 1+D+D^2]\end{aligned}$$

证明这个码的其中一个生成矩阵为

$$\boldsymbol{G}(D)=\begin{bmatrix}h_1(D) & h_2(D) & 0\\ 0 & h_0(D) & h_1(D)\end{bmatrix}$$

5.23 假设在 BI-AWGNC 下，仿真码率为 2/3 的卷积码的 Viterbi 译码，其校验矩阵如下：

$$\begin{aligned}\boldsymbol{H}(D)&=[h_2(D) \quad h_1(D) \quad h_0(D)]\\&=[1+D \quad 1+D^2 \quad 1+D+D^2]\end{aligned}$$

画出 P_b 为 $10^{-1}\sim10^{-6}$ 的误码率。