04104-106通信原理II第八讲

April 17, 2007

1 编码问题

线性分组码是分组码的一种。就一般来说，任何一种将k比特信息$\mathbf{u}$映射到n比特码字$\mathbf{c}$ 的做法就是一种编码设计，比如将00、01、10、11 分别映射到1110、1011 、0100 、1001 就构成了一种(4,2)分组码。

对于任意的(n,k)分组码，实现编码器的一种通用方法是查表法：用一个存储器存储所有2^k个码字，设计地址线宽是$k$bit，数据线宽是$n$bit。然后用$\mathbf{u}$作为地址线来读存储器。这种方法需要的存储量随k指数增长，是$2^k \times n$bit。

采用线性分组码时，编码器是向量和矩阵相乘：$\mathbf{c} = \mathbf{u}G$。特别对于系统码，$\mathbf{c} = (\mathbf{u}, \mathbf{p}) = \mathbf{u}(I, Q)$，只需要实现矩阵乘法$\mathbf{u}Q$。实现这样的乘法需要的是一些异或门，门数不超过$k(n-k)$个，硬件复杂度大大低于查表法。

2 译码

发送某个码字$\mathbf{c} = (c_{n-1}, c_{n-2}, \cdots, c_0) \in C$，经过BSC信道后成为$\mathbf{y}$。信道可能会使$\mathbf{c}$中的某些比特出错。如果某个比特发生错误，相当于这个比特加上了1。因此信道输出可以写成

$$\mathbf{y} = \mathbf{c} + \mathbf{e} \tag{1}$$

其中的$\mathbf{e} = (e_{n-1}, e_{n-2}, \cdots, e_0)$是错误图样。如果$e_i = 1$就表示$c_i$发生了错误。

接收端看到的是$\mathbf{y}$，它想知道的是$\mathbf{c}$。因为接收端不知道$\mathbf{e}$，所以发送的是什么没有确定的答案。在收端看来，最可能的发送码字是C中离$\mathbf{y}$最近的码字。因此译码器的通用设计可以是：把全部码字C存储在一个存储器中，用$\mathbf{y}$逐个和C中的码字进行比较，每次比较包括的操作是：从C中读出一个码字，与$\mathbf{y}$ 异或，数出异或结果中1的个数（汉明重量）。通过这些比较找出最近的那个码字作为判决结果。这就是ML译码。

对于线性分组码，我们也可以避免这种通用的方法，同时等达到相同的效果。能做到这一点的关键是因为线性分组码的码字一定满足

$$H\mathbf{c}^T = \mathbf{0} \tag{2}$$

其中$\mathbf{0}$是一个长为$r=n-k$的全零列向量，H是一个$r\times n$的矩阵，称为监督矩阵或者校验矩阵（parity check matrix）。

式(2)能够成立的原因是这样的。对于任意的线性分组码，我们总能把它转换成系统码的形式。故此不妨考虑系统码，$\mathbf{c}=(\mathbf{u},\mathbf{p})$，$G=(I,Q)$，因此$\mathbf{p}=\mathbf{u}Q$。对于GF(2)上的运算，若$a=b$则$a+b=0$，因此$\mathbf{u}Q+\mathbf{p}=\mathbf{0}$，写成矩阵形式就是$(\mathbf{u},\mathbf{p})\begin{pmatrix} Q \\ I \end{pmatrix}=\mathbf{0}$，转置后就是$(Q^T,I)\mathbf{c}^T=\mathbf{0}$，即(2)。

由于G的各行线性无关，所以方阵Q是满秩的，所以H的各行线性无关。

接收端已知H，用接收到的$\mathbf{y}$去乘H能得到一个长为$r=n-k$的向量$\mathbf{s}=\mathbf{y}H^T$，这个向量叫伴随式(syndrome)。根据式(1)和式(2)可得

$$H\mathbf{e}^T=\mathbf{s}^T \tag{3}$$

说明信道中发生的错误图样是线性方程组(3)的解，只要解出这个线性方程组，就知道了$\mathbf{e}$，再与$\mathbf{y}$相加，结果就是$\mathbf{(}c)+\mathbf{e}+\mathbf{e}=\mathbf{c}$。不过，式(3)有$r=n-k$个方程，而$\mathbf{e}$有$n$个变量，因此方程组(3)有多解。任意给定$\mathbf{e}$中的$k$个比特，就能得到一个解，这样我们将得到$2^k$种可能的错误图样。接收端不可能知道真正发生的错误是哪一个，只能猜。如果猜错也没办法。根据ML检测的精神，应该猜码重最小的解（即错误个数最少者）。

实际的译码电路实现时，倒不需要每收到一个$\mathbf{y}$，就忙着去解方程组(3)，再选择码重最小的错误图样。这个过程可以事先做，把结果存起来用就行了。比如(7,4)码，伴随式$\mathbf{s}=(s_2s_1s_0)$有3个比特，共8种。事先对于每一种$\mathbf{s}$，解方程组(3)，再选出码重最小的（如果最小的有多个，就随便选一个），称这个为“可纠正错误图样”（意思是：如果信道中实际发生的错误图样确实是它的话，就搞定了。否则只好译错）。然后把每个$\mathbf{s}$纪录下来。电路实现的构成包括三部分：(1)计算$\mathbf{s}$，这是一个矩阵乘法；(2)查表，即用$\mathbf{s}$为输入，输出相应的可纠正错误图样，一般可化为适当的逻辑电路；(3)将$\mathbf{y}$与所认为的错误图样$\hat{\mathbf{e}}$相加得到译码结果$\hat{\mathbf{c}}$。图9.2.4是一个例子。

3 H

两个向量的内积为零称为正交。比如$\mathbf{a}=(01100)$和$\mathbf{b}$=(01111)的内积是$\mathbf{a}\mathbf{b}^T=0\times0+1\times1+1\times1+0\times1+0\times1=0$，故此它们是正交的。式(2)表明$H$的每一行都和$C$中的任何一个码字正交。

将H进行初等行变换得到H'，那么H'的任何一行是H的行的线性组合，因此任何一个$\mathbf{c}\in C$也必然和H'正交，即方程组(2)可以改写成$H'\mathbf{c}^T=\mathbf{0}$。表明给定$G$时（即给定了$C$），校验矩阵并不唯一。即便给定的$G$是系统码的生成矩阵，$H$也同样不唯一。不过，如果要求$H$具有“典型形式”$H=(P,I)$，则结果是唯一的。

H有$k' = n - k$行，各行线性无关，因此它也符合成为生成矩阵的条件。若以$G' = H$为生成矩阵，将得到一个(n, k')线性分组码，称为原码的对偶码。若C'是对偶码的码字集合，则$\forall \mathbf{c}' \in C', \mathbf{c} \in C$，有$\mathbf{c}'\mathbf{c}^T = 0$。即对偶码和原码正交。

注意到$H\mathbf{c}^T$的结果是H的某些列之和，这些列的位置对应$\mathbf{c}$中1的位置。若$\mathbf{c}$的码重是d，则标明H一定有d列，其和为0，即H一定有d列是线性相关的。若$\mathbf{c} \neq \mathbf{0}$，则其码重最小是$d_{min}$，因此$H$的任意$d_{min} - 1$列必然是线性无关的。利用这个特性，可以根据$H$来求解码的最小码距。如果我们发现$H$的任何$d$列线性无关，则$d_{min} \geq d + 1$。对于式(9.2.12)给出的$H$，任何两列都不相同，因此$d_{min} \geq 2 + 1 = 3$。再注意到任何1列都不可能是其它两列之和，因此$d_{min} \geq 4$。再注意到左起第1列是第4、5、6列之和，因此得知该码的$d_{min} = 4$，它可保证纠1位错。如果发生2位错，有些或许能纠，但不保证能纠所有2位错。

4 汉明码

不论是否线性，任意(n, k)码的码字集合C都是在2^n个n长的向量中挑出了2^k个。假设这个码能保证纠t个错，那么C中任何两个码字之间的距离至少是$2t+1$。如果以任意一个$\mathbf{c} \in C$为球心做一个球，让它囊括GF(2^n)中所有距离$\mathbf{c}$不超过t的向量，那么这个球内的点数是$V = \sum_{i=0}^{t} C_n^i$。对C中的所有码字都做这样的球，那么它们必然不相交。这些球大小相同，它们所包括的总点数是$2^k V$，这个数值必然不会超过2^n，因此有如下不等式

$$\sum_{i=0}^{t} C_n^i \leq 2^{n-k} \tag{4}$$

给定n, k，若t_{max}是满足上述不等式的最大t值，则表明我们不可能设计出一个二进制分组码，它的纠错能力竟然比t_{max} 还大。这个上界叫Hamming界。

能使(4)等式成立的码叫完备码。能在$t = 1$的条件下等式成立的就是汉明码。此时$1 + n = 2^{n-k}$。若校验位个数是$r = n - k$，则码长是$n = 2^r - 1$，信息位个数是$k = n - r = 2^r - 1 - r$。最小码距是$d_{min} = 2t + 1 = 3$。因此其H的任意两列线性无关（即任意两列不相同）。Hamming码的校验矩阵很容易写出：将r个比特的所有可能结果写出，除去全零的一个，把剩下的作为H的各个列即可。

顺便指出：我们把生成矩阵写成G是因为Generator，把校验矩阵写成H是因为Hamming。