Channel Coding

Name:温嘉颖

Tutor:韩国军

目录

第一章 绪论

1.1 数字通信系统与信道模型

通信系统的基本目的在于将信息由信源高效、可靠、有时还需安全地传送到信宿。有扰通信信道中的噪声会不可避免地对传输信息产生不同程度的干扰，从而可能降低通信可靠性。而衡量通信信道的指标主要有 3 个，分别是**误码率、信道容量和平均功率**，所以通信系统设计的核心问题就是在存在随机噪声的信道中如何克服干扰，减小信息传输的差错，同时又不降低信息传输的效率，即如何解决系统的有效性与可靠性之间的矛盾。一般地，通信系统的可靠性用误码率（BER）来衡量，其有效性则用信息传输速率 R 比特/信道符号来衡量。早期的人们普遍认为：通信系统的可靠性与有效性之间是一对不可调和的矛盾，一方的改善总是以牺牲另一方为代价，并指出当功率受限时，在有扰通信信道上实现任意小错误概率的信息传输的唯一途径就是把信息传输速率降低至零。Shannon 信息和编码理论的奠基性论文《通信的数学理论》于 1948 年发表之后，改变了这一观点。他首次阐明了在有扰信道上实现可靠通信的方法，**他指出，在信道传输速率 R 小于或等于信道容量 C 时，即可以通过信道编码方式实现可靠通信**，从此信道编码的技术研究方向变得明确，并在接下来的几十年时间内，各种不同的编码方案被研究出来，而且这些编码方案的性能逐渐接近香浓最佳极限。根据 Shannon 的信息理论，数字通信系统的基本组成如下图所示。

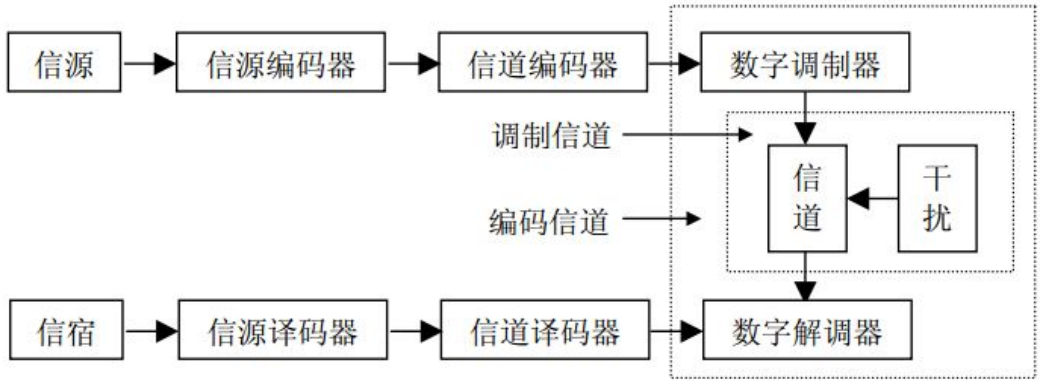


图 1.1 香农信道模型

1.2 香农三大定理

香农三大定理是信息论的基础理论。香农三大定理是存在性定理，虽然并没有提供具体的编码实现方法，但为通信信息的研究指明了方向。香农第一定理是可变长无失真信源编码定理。香农第二定理是有噪信道编码定理。香农第三定理是保失真度准则下的有失真信源编码定理。

香农第一定理(可变长无失真信源编码定理)

设信源 S 的熵 H(S),无噪离散信道的信道容量为 C,于是,信源的输出可以进行这样的编码,使得信道上传输的平均速率为每秒(C/H(S)-a)个信源符号.其中 a 可以是任意小的正数,

要使传输的平均速率大于(C/H(S))是不可能的。香农第一定理又称为无失真信源编码定理或变长码信源编码定理。香农第一定理的意义：将原始信源符号转化为新的码符号，使码符号尽量服从等概分布，从而每个码符号所携带的信息量达到最大，进而可以用尽量少的码符号传输信源信息。

香农第二定理(有噪信道编码定理)

有噪信道编码定理。当信道的信息传输率不超过信道容量时，采用合适的信道编码方法可以实现任意高的传输可靠性,但若信息传输率超过了信道容量,就不可能实现可靠的传输。

设某信道有 r 个输入符号，s 个输出符号，信道容量为 C，当信道的信息传输率 R<C，码长 N 足够长时，总可以在输入的集合中(含有 r^N 个长度为 N 的码符号序列)，找到 M（(M<=2^(N(C-a)))，a 为任意小的正数)个码字，分别代表 M 个等可能性的消息，组成一个码以及相应的译码规则,使信道输出端的最小平均错误译码概率 $P_{\min}$ 达到任意小。

香农公式：

$$C = B\log_2(1+\frac{S}{N})$$

其中 C 是信道容量，B 是信道带宽，S 是平均信号功率,N 是平均噪声功率，信噪比（S/N）通常用分贝（dB）表示，分贝数=10×log10（S/N ）

香农第三定理(保失真度准则下的有失真信源编码定理)

保真度准则下的信源编码定理，或称有损信源编码定理。只要码长足够长，总可以找到一种信源编码，使编码后的信息传输率略大于率失真函数，而码的平均失真度不大于给定的允许失真度，即 D'<=D.

设 R(D)为一离散无记忆信源的信息率失真函数，并且选定有限的失真函数，对于任意允许平均失真度 D>=0，和任意小的 a>0，以及任意足够长的码长 N，则一定存在一种信源编码 W，其码字个数为 M<=EXP{N[R(D)+a]}，而编码后码的平均失真度 D'(W)<=D+a。

Shannon 在信道编码定理的证明中引用了三个基本条件，即：
(1)采用随机的编码方式；
(2)码字长度趋近于无穷大；
(3)译码采用最佳的最大似然译码。

一般可将信道编译码器所使用的纠错码从性能上分为坏码和好码。所谓坏码是指只有将码率降至零才能使误码率为任意小的编码方式；而好码又可以分为当误码率任意小时，码率逼近信道容量限的非常好码和码率可达到的非零最大值小于信道容量限的一般好码。虽然 Shannon 指出一个随机选择的码以很高的概率为好码，但随机码的最大似然译码的复杂度

往往与码长呈指数关系，即在误码率随码长趋于无穷而趋向于零的同时，译码复杂度以指数增长，而在实际应用中，只能够使用以码长多项式的时间和空间复杂度内完成编译码的纠错码，因而尽管一般的随机码是好码，但不能看作是实用码。自信道编码定理提出以来，如何构造一个逼近信道容量限的实用好码成了众多研究学者竞相研究的课题，并逐渐形成信息论的一个重要分支——信道编码理论。五十多年来，人们构造实用好码的探索基本上是按照 Shannon 所引用的基本条件的后两条为主线展开的。虽然从理论上讲，除了目前已知的码外，几乎所有的码都是好码，但到目前为止，构造出真正意义上的实用好码还有较长的距离。虽然如此，通过众多学者，特别是有关数学和信息论学术界的研究人员五十多年的共同努力，目前已经取得了很多成果。

第二章 信道编码

2.1 纠错编码

纠错码能够检错或纠错，主要是靠码字之间有较大的差别。这可用码字之间的汉明距离 d(x，y)来衡量。它的定义为码字 x 与 y 之间的对应位取不同值的码元个数。一种纠错码的最小距离 d 定义为该种码中任两个码字之间的距离的最小值。**一种码要能发现 e 个错误,它的最小距离 d 应不小于 e+1。若要能纠正 t 个错误，则 d 应不小于 2t+1。**一个码字中非零码元的个数，称为此码字的汉明重量。一种码中非零码字的重量的最小值，称为该码的最小重量。对线性码来说，一种码的最小重量与其最小距离在数值上是相等的。

纠错码从构造方法上可分为分组码（Block Codes）和卷积码（Convolutional Codes）两大部分。在分组码方面，第一个分组码是 1950 年发现的能纠正单个错误的 Hamming 码；在整个 50 年代，基于代数理论又发现了多个短码长的分组码，如 1954 年 Golay 发现的 Golay 码以及 Reed 和 Muller 发现的 RM 码，Prange 在 1957 年发现的循环码等。最有意义的是 Bose 和 Ray-Chaudhuri 在 1960 年及 Hocuenghem 在 1959 年分别独立发现的能纠正多个错误的 BCH 码，以及 Reed 和 Solomon 在 1960 发现的非二进制 RS 码。实际上，BCH 码可以看作是某个 RS 码的子域子码，而 RS 码又可以看作是 BCH 码的特例。其后发现的分组码主要有 1970 年的 Goppa 码和 1982 年发现的代数几何码。在所有这些分组码中，除了 Goppa 码和代数几何码中存在个别达到 GV 限的渐进好码外，其它均不是好码。分组码的译码主要采用基于代数的硬判决译码。

2.2 有限(Galois)域

有限域理论在编码理论研究中起着特别重要的作用。一个元素个数有限的域称为有限域，或者伽罗华域、（Galois field）,有限域中元素的个数为一个素数，记为 GF(p)，p 为素数。有限域中运算满足交换律结合律和分配率。可以将 GF(p) 延伸为一个含有 p^m 个元素的域，称为 GF(p) 的扩展域，表示为 $GF(p^m)$,m 是一个非零正整数。并且有 GF(p)是 $GF(p^m)$ 的子集。

二进制域 GF(2) 是扩展域 $GF(2^m)$ 的一个子域，除了 0 和 1 外，在扩展域中还有特殊的元素，用一个新的符号表示，比如 a， $GF(2^m)$ 中任何非零元素都可以由 a 的幂次表示，

有限元素的集合$GF(2^m)$，**只能**含有2^m个元素，并且对乘法封闭，其约束条件为：

$$a^{(2^m-1)}+1=0$$

根据这个多项式的限制条件，任何幂次等于或超过2^m-1的域元素都可以降为幂次小于2^m-1的元素：

$$a^{(2^m+n)}=a^{(2^m-1)}a^{n+1}=a^{n+1}$$

故，$GF(2^m)$的元素可以表示为：

$$GF(2^m)=\{o,a,^0,a^1,a^2,.....,a^{2m-2}\}$$

有限域的本原多项式：因为这些函数用来定义有限域$GF(2^m)$，一个多项式是本原多项式的充要条件：一个 m 阶（最高次幂）的不可约多项式 f(x)，如果 f(x) 整除x^n+1的最小正整数 n 满足2^m-1，则该多项式是本原的。

$GF(p^m)$中，在模 p(x)运算下的扩域上，x 所表示的元素是本原元。

例如: 用本原多项式$p(x)=1+x+x^3$ 来构造$GF(2^3)$，设 GF(8)上的本原元为 a, 通过将 a 的幂模 p(a)得到$GF(2^3)$上的所有元素。即$1+a+a^3=0$ 得到$a^3=1+a$ 这意味着a^3可以表示为更低阶 a 项的加权和。再有：

$$a^4=a*a^3=a(1+a)=a+a^2$$
$$a^5=a*a^4=a(a+a^2)=a^2+a^3=1+a+a^2$$
$$a^6=a*a^5=a(1+a+a^2)=1+a^2$$
$$a^7=a*a^6=a(1+a^2)=a+a^3=a^0$$

所以，有限域$GF(2^3)$的 8 个元素为$\{0,a^0,a^1,a^2,a^3,a^4,a^5,a^6\}$

由基本代数学理论我们知道，对于幂次为 m 的多项式必然有 m 个根。对于这个例子，p(x)=0 有 3 个根，由于这 3 个根不可能位于与 p(x)系数相同的有限域中，而是位于扩展域$GF(2^3)$中。用扩展域的元素 a 来定义多项式 p(x)的根，可写成如下形式：p(a)=0 把这 8 个

元素带入到式中可以得到 $p(x)=1+x+x^3$ 的 3 个根是 a,a^2,a^4

如果 GF(p)上的所有元素(除 0 外)都可表示为某元素 a 的幂，则 a 称为 GF(p)上的本原元 。

定理：设 $b_1,b_2,.....b_{p-1}$ 为 GF(p)上的非零域元素，则 $x^{p-1}+1=(x+b_1)(x+b_2)...(x+b_{p-1})$

例子：考虑 $GF(2)$ 和它的扩展域 $GF(2^3)$ 。这里 p=2,m=3,对 x^7+1 进行分解

$x^7+1=(x+1)(x^3+x+1)(x^3+x^2+1)$ 因为 $GF(2^3)$ 中非零元素为

$\{1,a,a+1,a^2,a^2+1,a^2+a,a^2+a+1\}$，因此可以写为

$$x^7+1=(x+1)(x+a)(x+a+1)(x+a^2)(x+a^2+1)(x+a^2+a)(x+a^2+a)$$
$$=(x+1)[(x+a)(x+a^2)(x+a^2+a)][(x+a+1)(x+a^2+1)(x+a^2+a+1)]$$

在 $GF(2^3)$ 上有：

$$x^3+x\ +1=(x+a)(x+a^2)(x+a^2+a)$$

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

从极小多项式可得对应的根和对应的幂。

2.3 格雷(Gray)码

格雷码（Gray code）是由贝尔实验室的 Frank Gray 在 1940 年提出，用于在 PCM（脉冲编码调变）方法传送讯号时防止出错，并于 1953 年三月十七日取得美国专利。格雷码是一个数列集合，相邻两数间只有一个位元改变，为无权数码，且格雷码的顺序不是唯一的。

传统的二进制系统例如数字 3 的表示法为 011，要切换为邻近的数字 4，也就是 100 时，装置中的三个位元都得要转换，因此于未完全转换的过程时装置会经历短暂的，010,001,101,110,111 等其中数种状态，也就是代表着 2、1、5、6、7，因此此种数字编码方法于邻近数字转换时有比较大的误差可能范围。葛雷码的发明即是用来将误差之可能性缩减至最小，编码的方式定义为每个邻近数字都只相差一个位元，因此也称为最小差异码，可以使装置做数字步进时只更动最少的位元数以提高稳定性。 数字 0～7 的编码比较如下：

十进制	葛雷码	二进制
0	000	000
1	001	001
2	011	010

3	010	011
4	110	100
5	111	101
6	101	110
7	100	111

二进制数转格雷码：

（假设以二进制为0的值做为格雷码的0）
G：格雷码 B：二进制码
G(N) = (B(n)/2) XOR B(n)

2位元格雷码	3位元格雷码	4位元格雷码	4位元2进制原始码
00	000	0000	0000
01	001	0001	0001
11	011	0011	0010
10	010	0010	0011
	110	0110	0100
	111	0111	0101
	101	0101	0110
	100	0100	0111
		1100	1000
		1101	1001
		1111	1010
		1110	1011
		1010	1100
		1011	1101
		1001	1110
		1000	1111

格雷码转二进制数：

二进制码第 n 位 = 二进制码第（n+1）位+格雷码第 n 位。因为二进制码和格雷码皆有相同位数，所以二进制码可从最高位的左边位元取 0，以进行计算。（注：遇到 1+1 时结果视为 0）例如： 格雷码 0111，为 4 位数，所以其所转为之二进制码也必为 4 位数，因此可取转成之二进制码第五位为 0，即 0 b3 b2 b1 b0。

0+0=0，所以 b3=0

0+1=1，所以 b2=1

1+1 取 0，所以 b1=0

0+1 取 1，所以 b0=1

因此所转换为之二进制码为 0101

2.4 RM 码

在 Golay 码提出之后最主要的一类分组码就是 Reed-Muller 码。它是 Muller 在 1954 年提出的，此后 Reed 在 Muller 提出的分组码的基础上得到了一种新的分组码，称为 Reed-Muller 码，简记为 RM 码。在 1969 年到 1977 年之间，RM 码在火星探测方面得到了极为广泛的应用。即使在今天，RM 码也具有很大的研究价值，其快速的译码算法非常适合于光纤通信系统。

在 3GPP 的 LTE(long term evolution)系统中信道质量指示(channel quality indicator ,CQI)与混合自动重传请求应答(hybrid automatic repeat request acknowledgement ,HARQ-ACK)均采用了 RM 编码方式，但与经典 RM 码又有很大差异，3GPP LTE 协议上采用了基于 RM 码的超码编码方式。

2.4.1 RM 编码

一个 RM 编码用 R(r,m)表示，码的码长为 2m ,对于 R(r,m)，任意两个码字之间的距离是 2m-r 。根据差错编码控制理论，如果一种编码具有纠 e 个错的能力，那么这个码的任意两个码字之间的最小距离就要大于 2e。

以 m=3 为例子，m=3 表示码长为 8，这个码有四个基本的基，

即为：

x0=[1 1 1 1 1 1 1 1];

x1=[1 1 1 1 0 0 0 0];

x2=[1 1 0 0 1 1 0 0];

x3=[1 0 1 0 1 0 1 0];

那么 R(0,3)的码表示 0 级的线性组合，也就是码的生成多项式就为 x0,信息就是 1 比特。

R(1,3)表示 1 级的线性组合，这时码的生成多项式（生成矩阵）就为：

[x0;x1;x2;x3] = 1 1 1 1 1 1 1 1
1 1 1 1 0 0 0 0
1 1 0 0 1 1 0 0
1 0 1 0 1 0 1 0

信息就是 4 个比特。

R(2,3)表示的是 2 级的线性组合，这时的生成多项式就是：

[x0;x1;x2;x3;x1x2;x1x3;x2x3]

式中 x1x2 表示向量 x1 和 x2 向量相乘，也就是按位相与。

信息就是 7 个比特。

R(3,3)表示的是 3 级的线性组合，这时的生成多项式是：

[x0;x1;x2;x3;x1x2;x1x3;x2x3;x1x2x3]

$$[x_0;x_1;x_2;x_3;x_1x_2;x_1x_3;x_2x_3;x_1x_2x_3]$$

信息就是 8 个比特。

从上面就可以归纳出 RM(r,m)的生成多项式包含的列一共有：

k=1+C(m,1)+C(m,2)+...+C(m,r);

C(m,r)表示 m 个元素你 r 个元素相组合得到的组合个数。

Reed-Muller 码的编码非常简单。长度为(n, k)的 Reed-Muller 码的编码方法是将生成矩阵 M_{N*K} 与输入数据向量 x_{k*1} 按矩阵相乘，对每个元素对 2 取模，即可以得到编码数据向量。

2.4.2 RM 的译码

下面介绍一个直观的译码算法。这个算法的基本思想是：验证编码矩阵里的每一行即采用大数逻辑决定这一行在编码的时候是否被采用。

详细的译码算法：

第一步：对于 R(r,m)编码矩阵里的每一行，找到其 2m-r 个特征矢量，然后将每个特征矢量与这个码字进行点乘（GF（2）域上的矢量内积）。

第二步：观察点乘的结果里 0 和 1 的个数，如果 0 的个数多，则对应一行的系数定为 0，否则定为 1.

第三步：前两步是针对生成矩阵除了第一行的任意一行做的，做完之后将每一行与得到的系数相乘，然后将这些矢量相加得到新的一个矢量，然后将这个矢量与接收到的码字相加得到一个新的矢量，然后判断这个矢量里 0 和 1 的个数，如果 0 多，则表明第一行的系数是 0，否则表明第一行的系数是 1。这些系数即是译码的结果。

那么，每一行的监督矩阵如何得到：

对于每一行定义一个集合 E，这个集合包含的元素是这一行里所没有包含的所有 xi ,举一个例子，前面的 R(1,3)的最后一行是 x3,那么这一行对应的集合 E 就是{x1,x2}，对于 R(2,3)最后一行是 x2x3，那么这个集合就是{x1}。

特征多项式是集合里的每个元素或者它的补码的线性组合（一个组合只能出现它自己或者它的补码）。

补码的概念：多项式的各个元素去反得到的码，x1' 是 x1 的补码。

举个例子，对于 R(1,3)的生成多项式的最后一行，得到集合 E 为{x1,x2},那么它的特征多项式就是：x1x2,x1x2' ,x1'x2,x1' x2'

2.4.3 大数逻辑译码算法

大数逻辑译码算法(Majority Logic Algorithm)的核心问题是如何找到一组或多组正交于某一错误数字集合的校验和式，该算法通常用于译几类使用欧氏几何、射影几何和组合数学构造的大数逻辑可译码，利用其生成(或校验)矩阵(或多项式)的特点很容易得到其正交校验和式组。Reed 算法也算是其中的一种算法。

2.5 BCH 码

BCH 码是一类最重要的循环码，能纠正多个随机错误，它是 1959 年由 Bose、Chaudhuri 及 Hocquenghem 各自独立发现的二元线性循环码，人们用他们的名字字头命名为 BCH 码。一般情况下，我们所做的只是构造一个码，然后计算它的最小距离，从而估计出它的纠错能力，而在 BCH 码中，我们将采用另外一种方法：先说明我们希望它能纠错的个数，然后构造这种码。

若循环码的生成多项式具有如下形式：$g(x) = LCM[m_1(x), m_3(x), ...m_{2t-1}(x)]$　其中

LCM 表示最小公倍式，t 为纠错个数，$m_i(x)$ 为素多项式，则由此生成的循环码称为 BCH 码，其最小码距 $d \geq d_0 = 2t+1$ （d_0 称为设计码距)，它能纠正 t 个随机独立差错。BCH 码的码长 $n = 2^m - 1$（本元 BCH 码）或是 $n = 2^m - 1$ 的因子（非本元 BCH 码）。

2.5.1 BCH 码编码

对一个分组长度 $n = p^m - 1$、确定可纠 t 个错误的 BCH 码的生成多项式的步骤：

1. 选取一个次数为 m 的素多项式并构造 $GF(P^m)$

2. 求 a_i ,i=0,1,2,…n-2 的极小多项式 $f_i(x)$

3. 可纠 t 个错误的码的生成多项式为

$$g(x) = LCM[f_1(x), f_2(x), ...f_{2t}(x)]$$

用这种方法设计的码至少能纠 t 个错误，在很多情况下，这些码能纠多于 t 个错误!! 因此 d=2t+1 称为码的设计距离，其最小距离 d*≥ 2t+1。**注意：一旦确定了 n 和 t，我们便可以确定 BCH 码的生成多项式。**

如果希望纠 2 个错误，且 n=15。则查表可得其生成多项式为：

$$\begin{aligned} g(x) &= LCM[f_1(x), f_2(x), f_3(x), f_4(x)] \\ &= LCM[(x^4+x+1), (x^4+x+1), (x^4+x^3+x^2+x+1), (x^4+x+1)] \\ &= (x^4+x+1)(x^4+x^3+x^2+x+1) \\ &= x^8+x^7+x^6+x^4+1 \end{aligned}$$

因为 deg g(x)=n-k=8，所以 k=7,于是我们得到纠 2 个错误的 BCH(15,7)码的生成多项式。该码的设计距离为 d=2t+1=5，可以计算该码的实际最小距离 d*也是 5。

2.5.1 BCH 码译码

根据生成多项式，可以构造出快速的硬件编码器，而对于 BCH 码的译码，由于它是循环码的一个子类，任何对循环码的标准译码过程都适用于 BCH 码。

流行的主要针对 BCH 码译码的有 Peterson Gorenstein Zierler 算法 Berlekamp-Massey 算法。

2.6 RS(Reed-Solomon)码

RS 码又称里所码，即 Reed-solomon codes，是一种低速率的**前向纠错**的信道编码，对由校正过采样数据所产生的多项式有效。编码过程首先在多个点上对这些多项式求冗余，然后将其传输或者存储。对多项式的这种超出必要值的采样使得多项式超定（过限定）。当接收器正确的收到足够的点后，它就可以恢复原来的多项式，即使接收到的多项式上有很多点被噪声干扰失真。

RS(Reed-Solomon)码是一类纠错能力很强的特殊的非二进制 BCH 码。对于任选正整数 S 可构造一个相应的码长为 n=qS-1 的 q 进制 BCH 码，而 q 作为某个素数的幂。当 S=1，q>2 时所建立的码长 n=q-1 的 q 进制 BCH 码，称它为 RS 码。当 $q=2^m$ (m>1)，其码元符号取自于 $GF(2^m)$ 的二进制 RS 码可用来纠正突发差错，它是最常用的 RS 码。

2.6.1 RS 码编码

在 RS 码中，输入信号分成（N*m）比特一组，每组包括 N 个符号，每个符号由 m 个比特组成，在 $GF(2^m)$ 中，符号 RS(N,K) 表示:

m 表示符号的大小，m=8 表示符号由 8 位二进制数组成
N 表示码块长度
K=N-K=2t 表示校验码的符号数
t 表示能够纠正的错误数目

RS 码不但可以纠正随机错误，突发错误，以及二者的结合，而且可以构造其他码类，如级联码，同时它具有极低的未探测差错率，这意味着与它配合使用的译码器能可靠地指出是否正确地校正码字，因此 RS 码成为很重要的一种纠错码。

例如，（255，223）RS 码表示码块长度共 255 个符号，其中信息代码的长度为 223，检测码有 32 个检验符号，在这由 255 个符号组成的码块中，可以纠正在这个码块中出现的 16 个错误符号，但不能纠正 17 个或以上的错误符号。

RS 码属于循环码的一种，他的编码过程就是用他的信息多项式除以检验生成多项式求出校验位的过程，也就是计算信息码符多项式除以校验码生成多项式之后的余数。

对于一个信息码符多项式，RS 校验码生成多项式的一般形式为：

$$G(x)=(x-a^0)(x-a^1).....(x-a^{k_0+k-1})$$

K0 是偏移量，通常取 K0=0 或 K0=1.而（N-K）大于等于 2t

例子：设在 $GF(2^3)$ 域中的元素，假设（7,3）RS 码中的 3 个信息符号为 m_2， m_1， m_0，信息码符多项式为 $M(x)=m_2x^2+m_1x+m_0x$ ，并假设 RS 的 4 个校验符号：Q_3,Q_2,Q_1,Q_0，则 $x^{N-K}/G(x)=M(x)x/G(x)$ 的剩余多项式 R(x) 为

$$R(x)=Q_3x^3+Q_2x^2+Q_1x+Q_0$$

如 K0=0 ， t=2 ， 可以得出 RS 校验码生成多项式就为

$$G(x)=(x-a^0)(x-a^1)(x-a^2)(x-a^3)$$

显然,g(x)的全部根为它的系数 $a^1,a^2,a^3,.....a^{2t}$, 是 $GF(2^m)$ 的元素.由 g(x)生成的码是(n,n-2t)循环码.它包括所有那些能被 g(x)除尽,系数是 $GF(2^m)$ 的元素次数不大于 n-1 的

多项式.RS 码的编译码是基于一组码元而不是单独的 0 或 1,这也是 RS 码纠错能力特别强的原因,并且这种特点使得 RS 码特别适合处理突发成片的错误。

由于 RS 码是循环码的一种,因此,它的编码方法与一般循环码的编码方法完全一致.用信息码多项式 $m_{k-1}x^{k-1}+m_{k-2}x^{k-2}+.....m_0$ 升 x^{n-k} 位后去除生成多项式 g(x),所得余式 r(x)为监督（校验)多项式,将监督多项式置于升 x^{n-k} 位的信息多项式之后,就形成 RS 码.因此 RS 码的编码变成用除法求余的过程,由于纠 v 个符号错误的生成多项式为

$$g(x)=\prod_{i=1}^{2t}(x+a^i)=\sum_{i=0}^{2t}g_i x^i$$

a 是有限域 $GF(2^m)$ 的元素.设输入信息码为 m(x),编码后的码组为 c(x),则

$$c(x)=x^{n-k}m(x)+x^{n-k}m(x)\bmod g(x)$$

2.6.2 RS 码译码

差错控制编码中译码的好坏是应用的关键,而译码算法通常比编码复杂.相比之下,RS 码的译码算法又要比其它译码复杂得多,这是因为 RS 码是一种非二元循环码,它不再具备特征为 2 的域的运算等性质.并且 RS 码的译码不仅要找出错误位置,而且还要找出对应错误位置的错误大小,因此增加了译码的难度.RS 码的译码也是从计算接收码字的伴随式入手。

2.7 戈雷(Golay)码

对于一些部分非本原 BCH 码的列表，Golay 码就是(23,12)码。由表可查出，其生成多项式 $(5343)_8=101011100011$ 即：

$$g_1(x)=x^{11}+x^9+x^7+x^6+x^5+x+1$$

或

$$g_2(x)=x^{11}+x^{10}+x^6+x^5+x^4+x^2+1$$

它们都是 $x^{23}+1$ 的因式，即

$$x^{23}+1=(x+1)g_1(x)g_2(x)$$

其最小码距为 7，可纠正不大于 3 个的随机错误。**Golay 码是一个完备码**。

2.8 循环码

循环码是线性分组码的一种，所以它具有线性分组码的一般特性，此外还具有循环性。**循环码的编码和解码设备都不太复杂，且检(纠)错能力强。它不但可以检测随机的错误，还可以检错突发的错误。**（n,k）循环码可以检测长为 n-k 或更短的任何突发错误，包括首尾相接突发错误。

循环码是一种**无权码**，循环码编排的特点是相邻两个数码之间符合卡诺图中的邻接条件，即相邻两个数码之间只有一位码元不同，码元就是组成数码的单元。符合这个特点的有多种方案，但循环码只能是表中的那种。循环码的优点是没有瞬时错误，因为在数码变换过程中，在速度上会有快有慢，中间经过其它一些数码形式，称它们为瞬时错误。这在某些数字系统中是不允许的，为此希望相邻两个数码之间仅有一位码元不同，即满足邻接条件，这样就不会产生瞬时错误。循环码就是这样一种编码，它可以在卡诺图中依次循环得到。循环码又称格雷码（ Grey Code ）。

2.8.1 循环码编码

设码长为 n 的循环码表示为：$(a_{n-1},a_{n-2},...a_2,a_1,a_0)$，则码多项式为：

$$T(x)=a_{n-1}x^{n-1}+a_{n-2}x^{n-2}+...+a_ix^i+...a_1x^1+a_0$$

(n,k)循环码的生成多项式写为 g(x),它是(n,k)循环码码集中唯一的，幂次为 n-k 的码多项式。例如：（7,3）循环码的生成多项式：

分解因式：$x^7+1=(x+1)(x^3+x^2+1)(x^3+x+1)$，可以取一次和任个 3 次的成绩作为多项式，即，$g_1(x)=(x+1)(x^3+x^2+1)$ 或 $g_2(x)=(x+1)(x^3+x+1)$

2.8.2 循环码译码

纠错码的译码是该编码能否得到实际应用的关键所在。译码器往往比编码较难实现，对于纠错能力强的纠错码更复杂。根据不同的纠错或检错目的，循环码译码器可分为用于纠错目的和用于检错目的的循环码译码器。 通常，将接收到的循环码组进行除法运算，如果除尽，则说明正确传输；如果未除尽，则在寄存器中的内容就是错误图样，根据错误图样可以确定一种逻辑，来确定差错的位置，从而达到纠错的目的。

循环码译码可按以下三个步骤进行：

（1）有接收到的 y(x)计算伴随式 s(x);

（2）根据伴随式 s(x)找出对应的估值错误图样 $e(\hat{x})$；

（3）计算 $c(\hat{x})=y(x)+e(\hat{x})$ ，得到估计码字 $c(\hat{x})$ 。若 $c(\hat{x})=e(\hat{x})$ 则译 码正确，否则译码错误。

具体地说，译码程序为：初始化 -> 由 R(x)确定 S(x) , $S(x)=R(x)\bmod g(x)$ -> S(x) 查表确定图样 E(x) -> 纠错 $C(x)=E(x)+R(x)$ -> 存储 C(x) 。

2.9 级联码

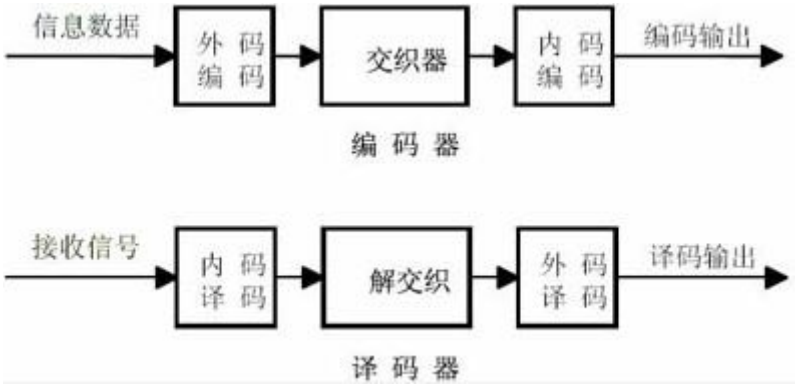
级联码(concatenated code)：要想进一步提高编码的性能，必须加长编码。对于线

性分组码就是加长 n，对卷积码就是加长 K。但很快我们就会陷于复杂度不可接受的窘境。为了解决这个问题，**级联码把两个编码以串联或者并联的方式结合在一起**，这两个码（称为成员码或分量码）的复杂度在可接受的范围内，它们整体构成了一个更强大的编码。新一代高性能编码如 LDPC、Turbo 码等都是级联码的例子。对于这些码，直接进行全局的 ML 译码（ML 译码算法本质就是求解线性方程组）是行不通的（复杂度过高），因此最关键的技术问题是如何达到最佳或近似最佳的译码。目前人们所想到的方法是迭带形式的概率译码，它能可接近最佳译码。采用迭代译码的级联码的性能几近香农极限。这样的级联一般需要在两级之间加一个交织器。级联码最简单的例子是 RA 码，它把重复码的编码结果交织后通过一个差分编码器。**Turbo 码是两个卷积码级联。LDPC 本质上是重复码级联了许多的偶校验码。**

从信息论角度看，要逼近信道容量，则：随机编码，长度足够长。但各种实际的编码都谈不上随机，否则根本无法译码，由于硬件复杂度的限制，长度是很有限的。**多次编码看成一个整体编码，就是级联码，级联码进一步降低残余误码率，提高较低信噪比性能，利用短码构造一个低复杂度的长码**。

2.9.1 串行级联码

串行级联码(Serial Concatenation Code)：由两个编码串联构成一个级联码。主要由内码，外码，和交织器组成。

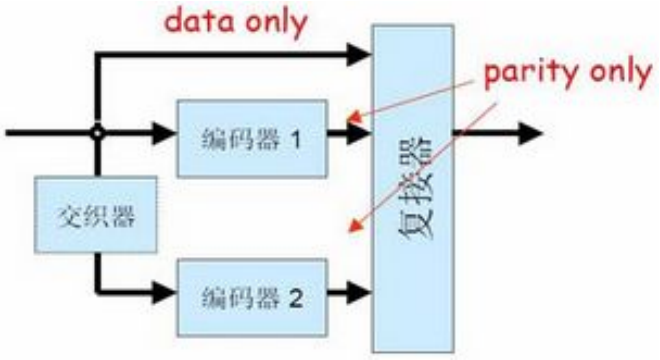


$$R = R_1 \times R_2 = (k_1 / n_1) \times (k_2 / n_2)$$

串行级联码的特点：1）大大提高了纠错能力，来源于编码效率的降低，误码率可以做到非常低。2）有误差扩散问题，出现两级还不如一级 3）编码效率降低，离容量极限还有相当距离。常见的串行级联码有 RS(外码)+卷积码(内码)

2.9.2 并行级联码

并行级联码(Parallel Concatenation Code):几个编码器具有相同的输入，数据来自不同编码器的校验位，码率 $R = k/(n_1 + n_2 + k)$



并行级联码——Turbo 卷积码（TCC）

- 译码在解码器 1 和解码器 2 之间进行迭代
- 编码器一般使用卷积码
- 迭代译码机制类似涡轮机(Turbo)的反馈工作原理
- Turbo 码的发现更新了编码理论研究中的一些概念和方法

2.9.3 交织器

交织器通常是对输入的原始信息序列进行随机置换后从前向后读出。交织器的作用是：一、可以产生长码。二、使两个 RSC（循环系统卷积码 recurrence system code）编码器的输入不相关，编码过程趋于独立。**交织使编码产生随机度，使码随机化、均匀化，起着对码重量整形的作用，直接影响 Turbo 码的性能**。在译码端，对于某一个子译码器来说不可纠正的错误事件，交织后在另一个译码器被打散，成为可纠正差错。

交织方式主要有规则交织，不规则交织和随机交织 3 种。通常规则交织即行写列读，效果不好。随机交织指交织格式是随机分配的，是理论上性能最好的交织方式，但是由于要将整个交织信息位置信息传送给译码器，降低了编码效率。实际应用中一般采用不规则交织，这是一种伪随机交织方式，对每一编码块采用固定的交织方式，但块与块之间交织器结构不一样。往往为了获得高的编码增益对交织器的长度提出要求。在无线移动通信系统对时延要求较高，因此采用交织长度为 400 左右的伪随机短交织器。

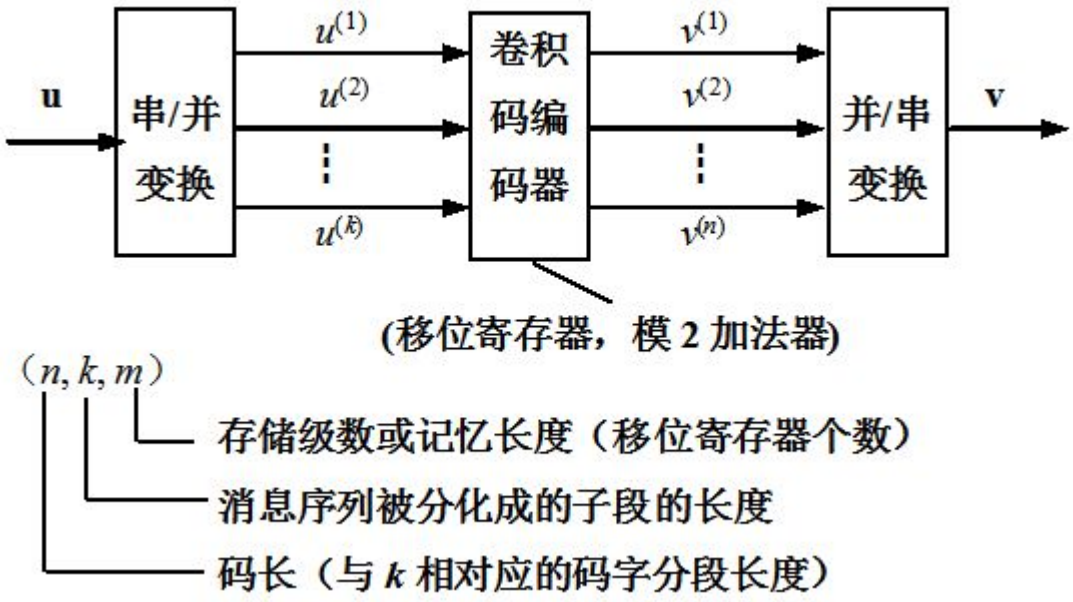
第三章 卷积码

3.1 卷积码的基本概念

卷积码是 1955 年由埃利斯(Elias)提出的，它以上面的分组码不同，分组码编码时，本组中的 n-k 个校验元仅与本组的 k 个信息元有关，而与其他各组码无关，分组译码时亦是。而在卷积编码中，本组的 n_0-k_0 个校验元不仅与本组的 k_0 个信息元有关，而且还与以前各时刻输入至编码器的信息组有关，同样，译码时亦是。此外，卷积码中每组的信息位 k_0 和码长 n_0 通常比分组码的 k 和 n 要小。

由于在卷积码的编码过程中，充分利用了各组之间的相关性，且 k_0 和 n_0 也较小，因此，在于分组码同样的码率 R 和设备复杂性条件下，无论从理论上还是实际上均已证明卷积码的性能至少不比卷积码差。

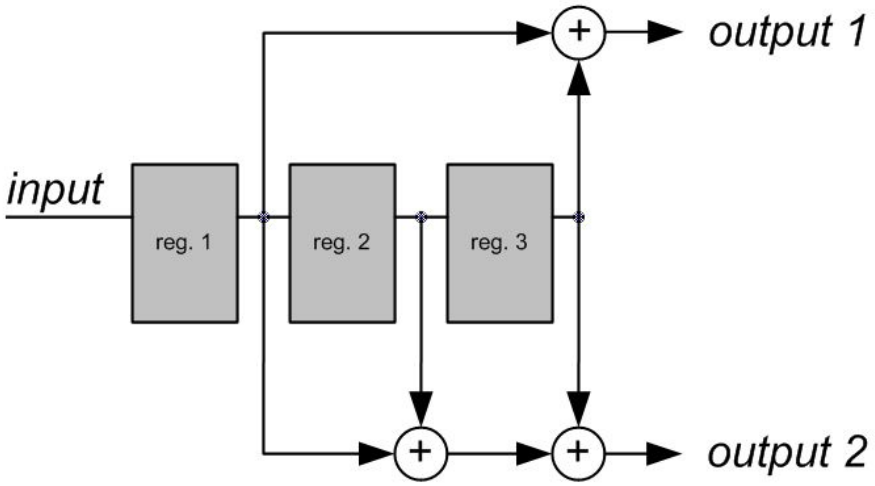
下面是一个 (n, k, m) 卷积码的编码器。有 k 个信息元并行输入编码器，经过编码得到 n 个码元并行输出。编码器的两边是为适应信道传输的串/并、并/串变换器。卷积码的编码效率 R 定义为信息位数与码长之比，即 R = k/n



3.2 卷积码的编码

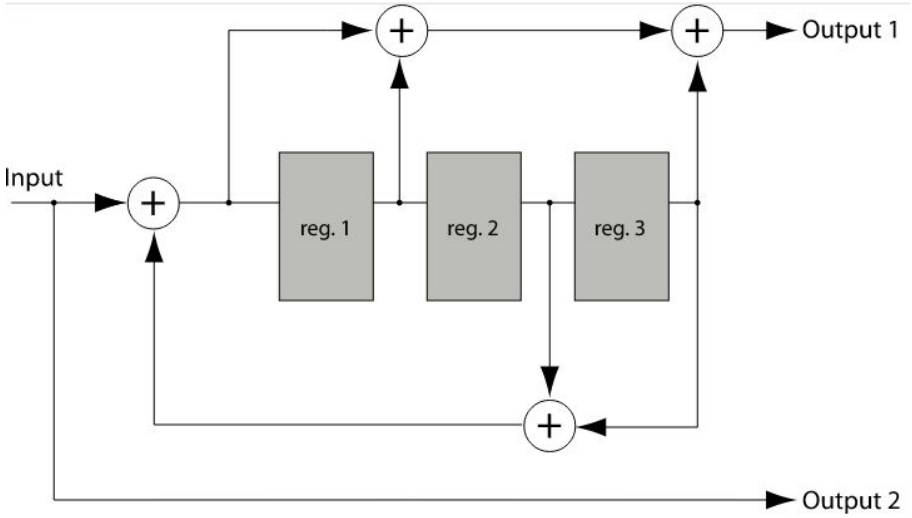
原始信号依序由输入端(input)进入编码器的暂存器(register)，每一个暂存器会储存一个输入字符，而它们的起始值都是 0。依图一而言，编码器内有 3 个'模 2 加法器'(modulo-2 adder，可等价于一个异或门(Boolean XOR gate)，对储存的 3 位元原始信号，做各自的加法运算。接着，暂存器(register)内的码元会移往下一格，(reg1 moves to reg2, reg2 moves to reg3)；然后继续将信号传至输出端(output)，如此便可以得到输出为：

$$output1 = reg.1 + reg.3$$
$$output2 = reg.1 + reg.2 + reg.3$$



每次卷积码输出信号都与过去的输入信号有关系，因而保有记忆效应(memory property)。

上图是一个非递回编码(non-recursive code)的类型，而下面则是一个递回编码(recursive code)再处理的类型，其即将被进行编码的输入信号同时也是输出信号；此外，递回编码几乎都是系统性的(systematic)，反之非递回编码则是非系统性的(non-systematic)。



卷积码之所以得其名是因为其处理方法是将输入端(input)信号以及编码器(encoder)中的脉冲响应(impulse response)进行卷积(convolution)。

$$y_i^j = \sum_{k=0}^{\infty} h_k^j x_{i-k}$$

此处x是输入讯号，y^j是输出信号 j，而h^j则是输出信号 j 的脉冲响应。卷积码的编码器(encoder)是一个离散的(discrete)线性非时变系统(linear time-invariant system)，因此每个输出端子的讯息都可以视为该编码器的转移函数(transfer function)；此外，脉冲响应可以透过 Z 转换(Z-transform)与转移函数建立关联性。

上图是一个非递回(non-recursive)编码器，它的系统函数如下：

$$H_1(z)=\frac{1+z^{-1}+z^{-3}}{1-z^{-1}-z^{-3}}$$

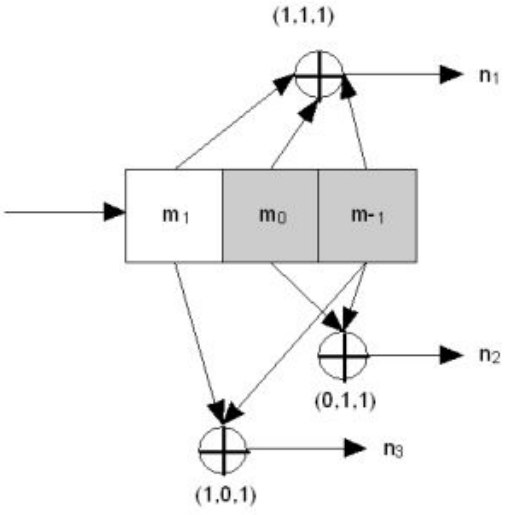
$$H_2(z)=1$$

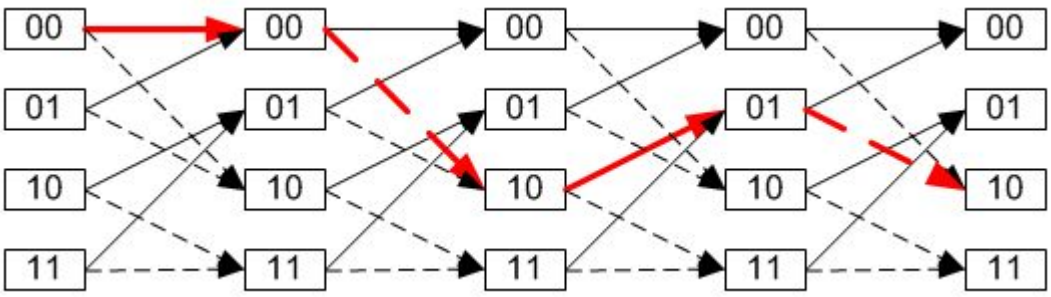
3.3 树状图

树状图(Trellis diagram)又称篱笆图，树图表。卷积码的编码器(encoder)可以表示成有限状态机(finite-state machine, FSM)，拥有n组输出的编码器在 FSM 上会有2^n个状态(states)。

以下图的非递回编码器来说，假设m_0现在的码元为‘1’、'0'被存放在m_{-1}(m_1不用列入考量因为它存放的是现在这个时刻的值)，那我们便定义现在位在"10"这个状态。而在下一个时刻，新的输入端讯号进入编码器时可能产生'1'或者'0'，因此下一时刻编码器可抵达的状态是"01"或者"11"；整个树状图如图所示，显而易见的，并不是所有的状态间都可以进行相连，比如"10"就不会连到"00"或者"10"这两个状态。

这个树状图是卷积码在解码(decoding)时的基础，唯有能够从头连到尾的输出端信号(output sequences)，才有可能是解码出来的结果，否则便会产生错误。





现存有许多解码卷积码的方法。对于较小的输出端组数，维特比算法(Viterbi algorithm)是一种普遍被使用来解码的算法，其以最大似然估计(maximum likelihood)来寻找最有可能产生观测事件序列的路径。

3.4 维特比算法

维特比算法是一种动态规划算法用于寻找最有可能产生观测事件序列的-维特比路径-隐含状态序列,特别是在马尔可夫信息源上下文和隐马尔可夫模型中。术语“维特比路径”和“维特比算法”也被用于寻找观察结果最有可能解释相关的动态规划算法。例如在统计句法分析中动态规划算法可以被用于发现最可能的上下文无关的派生(解析)的字符串，有时被称为“维比特分析”。维特比算法由安德鲁·维特比(Andrew Viterbi)于 1967 年提出，用于在数字通信链路中解卷积以消除噪音。 此算法被广泛应用于 CDMA 和 GSM 数字蜂窝网络、拨号调制解调器、卫星、深空通信和 802.11 无线网络中解卷积码。现今也被常常用于语音识别、关键字识别、计算语言学和生物信息学中。

维特比译码算法是卷积码中的一种最大似然译码算法, 在码的约束度较小时, 它的译码算法效率很高, 速度很快, 译码器也较简单。

3.4.1 维特比译码算法

维特比译码算法的步骤简述如下(对于(n, k,v) 卷积码, 信息序列长为 L) :

1) 从某一时间单位 j = m 开始, 对进入每一状态的所有长为 j 段分支的部分路径, 计算部分路径度量。对每一状态, 挑选并存储一条有最大度量的部分路径及其 部分度量值, 称此部 分路径为 保留(幸存) 路径。

2) j 增加 1 , 把此时刻进入每一状态的所有分支度量和同这些分支相连的前一时刻的保留路径的度量相加, 得到了此时刻进入每一状态的保留路径, 加以存储并删去其它所有路径, 因此保留路径延长了一个分支。

3) 若 j < L + m, 则重复以上各步, 否则停止,译码器得到了有**最大路径度量的路径。**

3.4.2 SOVA 算法

在应用中维特比算法只提供硬判决算法，不能带来额外的增益，Joachim Hagenauer 于 1989 年提出了 SOVA(Soft Output Viterbi Algorithm)算法，该算法除了提供一个硬判决外，还提供该硬判决的可靠度值。应用 SOVA 算法系统性能可以得到很大的改善 特别在并行连接码(Turbo)应用中，和应用维特比算法相比，大概可以获得 3dB 左右的增益。

SOVA 算法和传统的维特比算法相比，主要是多了一个可靠度的运算和刷新过程，从而增加了算法的复杂度。在硬输出维特比算法中，解码器从进入每一个节点的两条路径中选择一条。 软输出维特比算法除提供译码信息外 还输出译码信息的对数似然比 $\hat{L}_j$ ，代表译码的可信度量，可信度公式为：

$$\hat{L}_j < -f(\hat{L}_j, \Delta/\alpha) = \min(\hat{L}_j, \Delta/\alpha)$$

其中 Δ 是进入同一个节点的两条路径度量的差值，也可称为加权值。

$$\alpha = 4d_{free}\frac{E_S}{N_0}$$

其中 d_{free} 是码间的自由距离。

SOVA 算法的具体步骤：

第一步：前向计算分支度量和路径度量计算加权值。

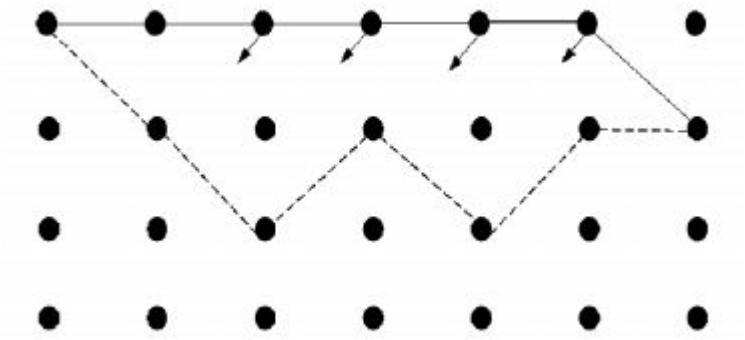
这是区别于硬输出维特比解码的一步。假设 $P_s(k,m)$ 是幸存分支的路径度量。k 为时间，m 为状态。 $P_c(k,m)$ 是竞争分支的路径度量，则加权值 $R=|P_s(k,m)-P_c(k,m)|$ 。

计算状态转换表。

第二步：反向回退获得信息译码及其相对应的加权值。

第三步：刷新加权值，获得可靠度加权值。

这是最关键的一步。如下图所示，**假设图中所示实线是已知的存活路径，从时间 k 出发，存在一条竞争路径，如果竞争路径上的判决和幸存路径上的不一样，则比较竞争路径和幸存路径的加权值，如果幸存路径的加权值大，则刷新幸存路径的加权值。幸存路径上的每一点都有一条竞争路径，所以每一个幸存路径上的节点的加权值都要进行多次刷新。**刷新的最多次数取决于幸存路径和存活路径何时重合。这一步可以和第二步并行进行。



3.4.3 BCJR 算法

BCJR 算法由贾里尼克（Fred Jelinek)和波尔，库克(Cocke)以及拉维夫(Raviv)设计，这是今天数字通信的最广泛的两个算法之一（另一个是 Viterbi 算法）。

BCJR 算法是一种**定义在网格图上的用来最大化纠错编码的后验概率的算法，主要用于卷积编码（详细参考《Turbo 码译码中的 BCJR 算法》)**。这种算法以它的发明者的名字命名，分别是 Bahl, Cocke,Jelinek 和 Raviv。这个算法对于现在的迭代的纠错编码来说是非常重要的，其中包括 Turbo 码和低密度部分检错编码。

算法步骤（基于格型结构)：

- 计算前向概率（Forward probabilities α）
- 计算后向概率 (Backward probabilities β)
- 基于其他信息（例如高斯白噪声的方差，二进制对称信道的位交叉概率）计算出平滑概率(smoothed probabilities)。

码器的主要目标是根据接收到的码序列及先验信息计算信息位 u_n 分别为 1 和 0 的后验概率，即计算 $P_r(u_n=1|R_1^N))$和 $P_r(u_n=0|R_1^N)$ 。这一概率可以通过对网格图中的状态转

移概率求和而得到。**为了计算** $P_r(u_n=1|R_1^N)$ 和 $P_r(u_n=0|R_1^N)$,可以将网格图中对应于 $u_n=1$ 和 $u_n=0$ 的转移概率相加。n 时刻编码器状态记为 $S_n \in \{0,1,...,2^v-1\}$，其中 v 为编码器的级数，则：

$$P_r(u_n=i|R_1^N)=\sum_{s'=0}^{2^v-1}\sum_{s=0}^{2^v-1}P_r(u_n=i,S_{n-1}=s',S_n=s|R_1^N) \qquad u_n=1,0$$

为了计算上述状态转移概率，定义联合概率密度函数为：

$$e_n^i(s',s)=p(u_n=i,S_{n-1}=s',S_n=s,R_1^N)$$

从而有：

$$P_r(u_n=i|R_1^N)=\sum_{s'=0}^{2^y-1}\sum_{s=0}^{2^y-1}e_n^i(s',s)/\,p(R_1^N)$$

再根据递归的方法，得到译码器的外信息输出。

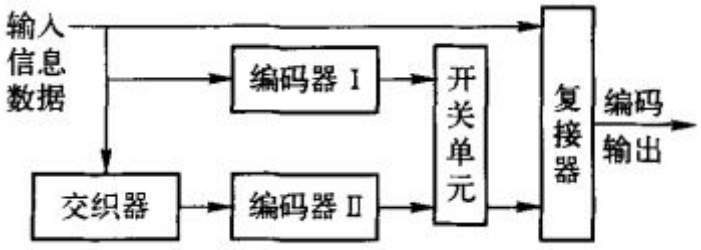
与 SOVA 算法相比，BCJR 算法在硬件上实现较为复杂，但准确度更高。

3.5 Turbo 码

1993 年，C．Berrou 等人提出一种新的信道编码方案——Turbo 码，Turbo 译码算法的特点是，利用两个子译码器之间信息的往复迭代递归调用，加强后验概率对数似然比，从而提高判决可靠性。Turbo 码很好地应用了 Shannon 信道编码定理中的随机性编译码条件，从而获得几乎接近 Shannon 理论极限的译码性能。

3.5.1 Turbo 编码器

Turbo 码由 2 个循环系统卷积码(recurrence system code，RSC)并行级联而成：译码采用迭代的串行译码；交织器是 Turbo 码所特有的，**它可以使得信息序列随机化，增加各码字间的重量**，从而提高码的保护能力。



典型的 Turbo 码编码器由 2 个相同的 RSC 编码器和交织器等构成，其原理如图。编码器 I、II 又称为成员码，都为非循环非系统卷积码编码器，共同构成了 RSC 编码器。编码器 I、II 只输出编码的校验序列，通过开关单元(有必要，进行增信删余)与输入信息序列一起输出得到 Turbo 码编码输出序列。例如，对于生成矩阵为 g=[g1，g2]的(2，1，2)卷积码通过编码后，如果进行增信删余，得到码率为 1 / 2 的编码输出序列；如果不进行增信删余，则得到码率为 1 / 3 的序列。

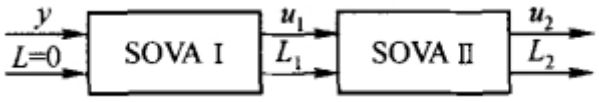
Turbo 码之所以利用 RSC 编码器，是因为循环编码器可以改善码的比特误码率性能。

对于卷积码编码器，通过在输入信息序列后加入 m(编码约束长度)个“0”，就可以使编码状态回到全“0”状态，使网格终止；但对于 RSC 编码器，由于有反馈，RSC 编码器的状态

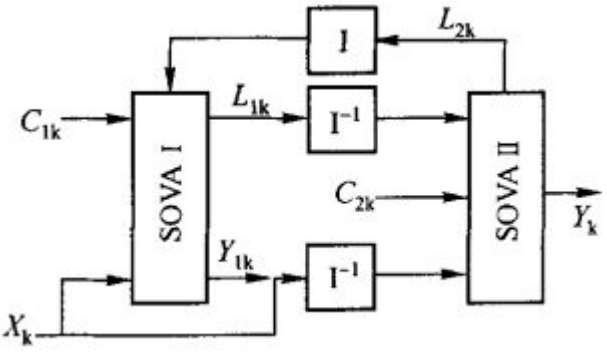
很难确定，加“0”的策略不能达到网格终止的目的；而且，在编码器 I 回到了全“0”状态，由于加入了交织器，编码器 II 却不一定回到全“0”状态，所以需要寻求别的办法来终止网格。最简单的方法就是在输入端设置开关电路来实现网格的终止。方法之二为：初始状态为全“0”，在 RSC 编码器 I 的信息序列后加入 m 个“0”比特，使其状态回到全“0”，从而使网格终止；而编码器 II 不终止，处于开放状态，可以在任何状态停止编码.

3.5.2 Turbo 码译码原理

卷积码的 Viterbi 译码算法可以得到最大似然输出序列，是一种最优的算法。但对于级联的卷积码，传统的 Viterbi 译码器有 2 大缺陷：内部的 Viterbi 译码器产生的突发错误会降低外部的 Viterbi 译码器的性能；内部的 Viterbi 译码器的硬判决输出，使得外部的 Viterbi 译码器不能利用软判决带来的好处，但引入可靠性参数就可以提高级联译码系统的性能。如果 Viterbi 译码器能产生可靠性参数(软输出)，再将该软判决输出值输入到随后的 Viterbi 译码器中作为先验概率来提高译码性能。这种修正的 Viterbi 译码算法构成的译码器称为软输出的 Viterbi 算法(Soft Output Viterbi Algorithm，SOVA)译码器。下图为级联的 SOVA 译码器框图：



其中：L 为相关的可靠性值，与信道参数有关，可以定义为信噪比的函数，$L = 4E_b / N_0$，y 为信道接收值；u 为硬判决输出。**Turbo 码译码算法有多种：最大似然译码 MAP、对数的最大似然译码(Log-MAP)、基于修正的 Viterbi 译码算法等。**从性能好坏和实现难易程度来看，**MAP** 译码精度高，但延迟大，存储量大，不易实现；而软判决算法，如 **SOVA** 算法，通过引入可靠性参数(软输出)可以改善译码性能，同时译码延迟和计算复杂度低，适合于硬件实现，所以比较常用。



上面表示 Turbo 码迭代的 SOVA 译码原理图，图中：I 为交织器；I^{-1} 为解交织器；C_{1k}，C_{2k} 为校验序列；X_k 为信息序列；L_{1k}，L_{1k} 为可靠性值，即 SOVA 译码器的软输出值；Y_{1k} 为经 SOVA I 译码的估值；Y_k 为译码得到的估值。

3.5.3 Turbo 码的实际应用

针对不同环境和不同译码要求，可以不同程度地将通信系统中的检测、调制等技术与 Turbo 码结合起来，从而达到更好的通信质量，如针对衰落形式中 DPSK(Differential PhaSe2shift Keying)信号的特点，提出的一种基于判决反馈的 Turbo DPSK 解调／解码算法，

它是利用判决反馈进行衰落信道的估计从而计算信息比特的似然比，相对传统的卷积码多用户与单用户检测系统，它有效地改进了快速瑞利衰落信道中数字通信系统的性能；而针对新近提出的空时码，Turbo 码调制(Turb02Code Modulation)方案在衰落信道中优于空时码的性能；针对无线通信系统中信道的特点，Ngatched 等比较了 Turbo—TCM 方案与编码分集技术(ACD)级联及 Turbo—TCM 方案与正交发射分集技术(OTD)级联在衰落信道中的性能，提出了既可以保持良好译码性能又具有较低译码复杂度的次优译码算法.在与调制编码技术相结合方面的改进还体现在：将栅格调制编码和 Turb0 码结合可以得到大的编码增益和高的带宽效率，如将 Turbo 码与 OFDM 调制、差分检测技术相结合，具有较高的频率利用率，可有效地抑制短波信道中多径时延、频率选择性衰落、人为干扰与噪声带来的不利影响；在级联树码的基础上提出的 CT—TCM(Concatenated Two2state TCM)方案，相对于其他的 TTCM 方案不仅得到了优异的性能，而且还有较低的译码复杂度．在 Turbo 码与 PSK 和删调制技术相结合方面也有不少研究者进行了研究．Turbo 码与检测技术相结合的研究主要有：在 CDMA 系统中将多用户检测与 Turbo 码及 SCCC 码相结合，通过更新、交换外部信息产生新的软信息使其达到单用户的性能;在 DS-CDMA 系统中利用 Turbo 码译码输出信息反馈给多用户接收机的接收端作先验信息，可以增强多用户检测的性能．在其他方面对 Turbo 码的研究也取得了一定的成果，例如，通过结合最小均方误差(MMSE)接收机，从联合界方面得出关于 Turbo 码在 DSCDMA 系统中性能的优越性；针对信源与信道编码的特点，提出了结合 SPIHT(Set Partitioning in Hierarchical Trees)算法和改进的 Turbo Codes 的信源与信道联合编码方案．此外，还有关于短帧 Turbo 码在 Rician 信道上的性能及设计研究。