

6.9 Turbo码

6.9.1 前言

6.9.2 Turbo码的编码结构

6.9.3 交织器的设计原则

6.9.4 Turbo码的译码

6.9.5 WCDMA系统中采用的Turbo码



6.9.1 前言

- 1993年，在国际通信会议(ICC)上法国学者C.Berrou等人在他们的论文“逼近Shannon限的纠错编码和译码—Turbo码”中首次提出了Turbo码。该文论述，在加性白高斯噪声的环境下，采用编码效率 $R=1/2$ 、交织长度为65536的Turbo码，经过18次迭代译码后，在 $E_b/N_0=0.7\text{dB}$ 时，其误码率到达 10^{-5} ，与香农极限只相差0.5dB，从而产生了在该领域的巨大兴趣。



➤ 但**Berrou**仅给出了**Turbo**码的基本组成和迭代译码的原理，而没有严格的理论解释和证明；随后**J. Hagenauer**阐明了迭代译码的原理，并推导了二进制分组码与卷积码的软输入软输出译码算法。由于在**Turbo**码中交织器的出现，性能分析非常困难，因此**S. Benedetto**提出了均匀交织（**UI**，**Uniform Interleaver**）的概念，并利用联合界技术给出了**Turbo**码的平均性能上界；**J. Seghers**系统地分析了**Turbo**码的距离特性；**Mackay**证明了**Turbo**码的校验矩阵与**LDPC**码的校验矩阵是等价的，从而可以将**Turbo**码看出一类特殊的**LDPC**码。

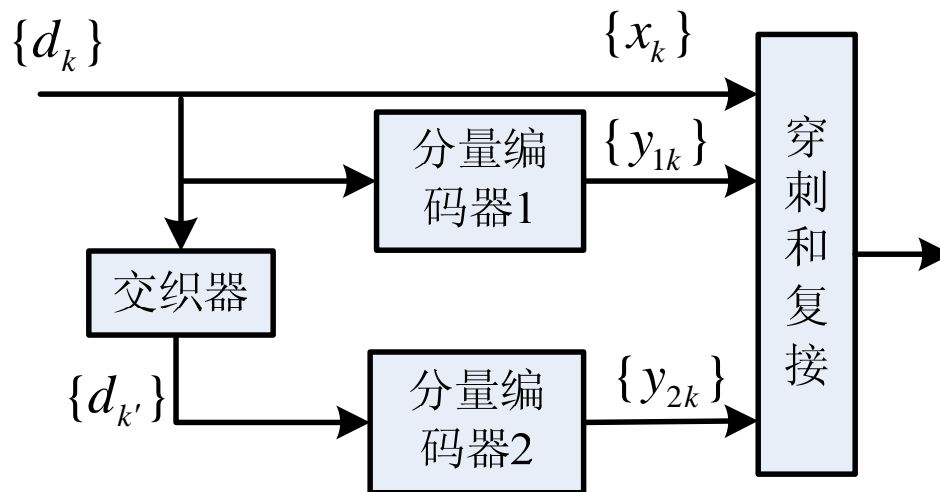
6.9.2 Turbo码的编码结构

- Turbo码的最大特点在于它通过在编译码器中交织器和解交织器的使用，有效地实现随机性编译码的思想，通过短码的有效结合实现长码，达到了接近Shannon理论极限的性能
- 通常，Turbo码编码器可分为并行级联卷积码(PCCC, Parallel Concatenated Convolutional Codes)结构、串行级联卷积码(SCCC, Serial Concatenated Convolutional Codes)结构和混合级联卷积码(HCCC, Hybrid Concatenated Convolutional Codes)结构。




PCCC编码器结构

- Berrou最初提出的Turbo码采用的是并行级联卷积码结构。下图给出了由两个分量编码器组成的Turbo码的编码框图。



- 分量码一般选择为递归系统卷积(RSC)码, 也可以为分组码、非递归卷积(NRC)码以及非系统码(NSC)码, 但分量码的最佳选择是递归系统卷积码。一般来说, 两个分量码采用相同的生成矩阵, 当然也可以不同。

- 
- 在编码过程中，两个分量码的输入信息序列是相同的，长度为 N 的信息序列 $\{d_k\}$ 一个支路直接作为系统输出 $\{x_k\}$ 送至复接器，另一个支路送入第一个分量编码器，得到校验序列 $\{y_{1k}\}$ ，第三个支路经过交织器 I 后信息序列为 $\{d_{k'}\}$ ，送入第二个分量编码器，得到校验序列 $\{y_{2k'}\}$ ，其中 $k'=I(k)$ 。 $I(.)$ 为交织映射函数， N 为交织长度，即信息序列长度。
 - 为提高码率和系统频谱利用率，可以将两个校验序列经过穿刺矩阵后，再与系统输出 $\{x_k\}$ 一起经过复接构成码字序列 $\{c_k\}$ 。

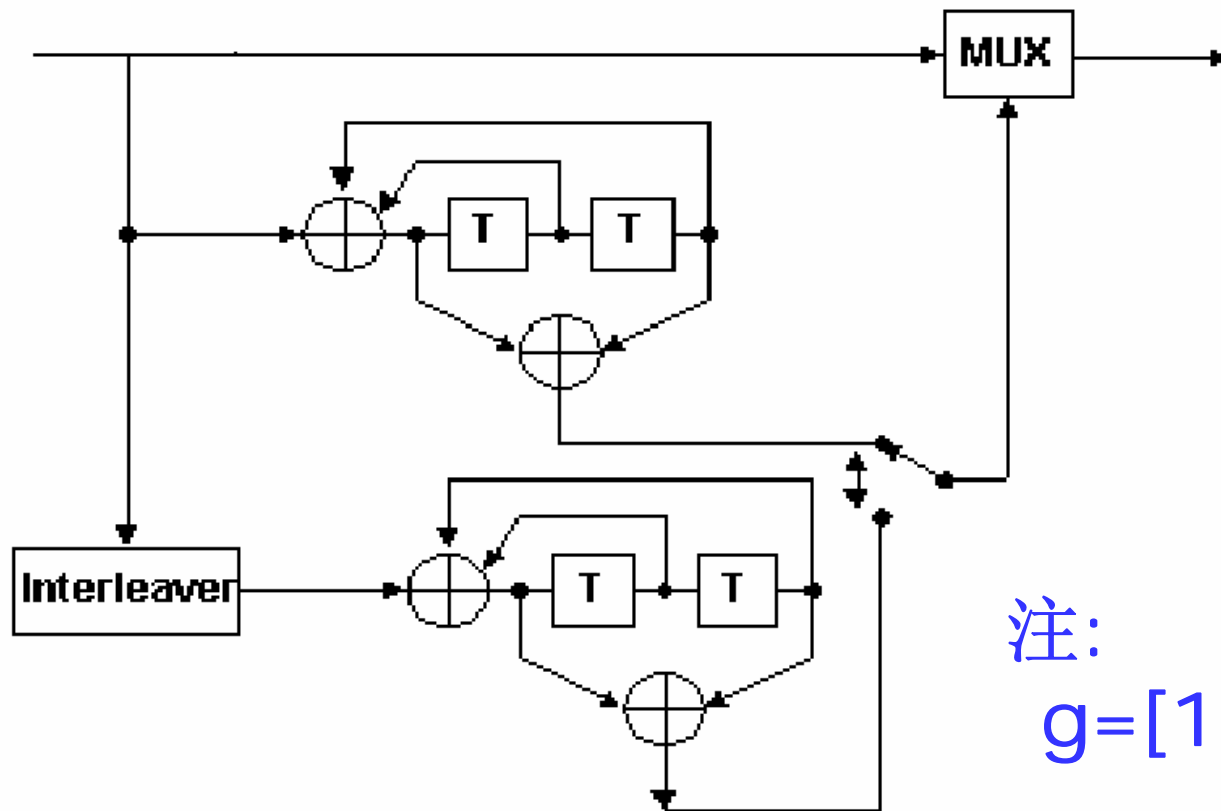
- 设输入序列在 k 时刻的比特为 d_k ，经约束长度为 N 的卷积码编码后，输出的校验位 Y_k 可表示为：

$$Y_{1k} = \sum_{i=0}^{N-1} g_{1i} d_{k-i} \quad Y_{2k} = \sum_{i=0}^{N-1} g_{2i} d'_{k-i}$$

其中， $G_1: \{g_{1i}\}$ ， $G_2: \{g_{2i}\}$ 分别是两个编码器的生成多项式。

- 若某一时刻输入比特 d_k ，则Turbo码编码器的输出码字为 $\{d_k, Y_{1k}, Y_{2k}\}$ ，即编码速率 R 为 $1/3$ 。为了获得更高的码率如 $R=1/2$ ，可用穿刺矩阵按一定的规则将校验位二选一。

例：Turbo编码器如下图所示



注：
 $g=[111,101]$

若输入序列 $d_k = [1 \ 0 \ 1 \ 0 \ 1]$

则第一个RSC编码器的输出

$$Y_{1K} = [1 \ 1 \ 0 \ 1 \ 1]$$

经交织器后的序列为

$$d_{k'} = [0 \ 1 \ 0 \ 1 \ 1]$$

则第二个RSC编码器的输出

$$Y_{2K} = [0 \ 1 \ 1 \ 0 \ 0]$$

编码器的输出是上述三个码序列的复合，为

$$X = [110 \ 011 \ 101 \ 010 \ 110]$$

序列码率 $R = 1/3$.

若 $R=1/2$,则需将校验位做穿刺处理,

$$Y_{1k}: [1 \ 1 \ 0 \ 1 \ 1]$$

$$Y_{2k}: [0 \ 1 \ 1 \ 0 \ 0]$$

穿刺矩阵为
$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$d_k = [1 \ 0 \ 1 \ 0 \ 1]$$

穿刺后的结果为

$$\Rightarrow [1 \ 1 \ 0 \ 0 \ 1]$$

则编码器的输出序列X为

$$[11 \ 01 \ 10 \ 00 \ 11]$$

Turbo码的码率

- 对于由两个分量码组成的**Turbo**码，其码率**R**与两个分量码的码率**R₁**和**R₂**之间满足：

$$R = \frac{R_1 R_2}{R_1 + R_2 - R_1 R_2}$$

显然，降低**R₁**和**R₂**的值可以使**R**减小；反之，**R**增大。

- 如果分量码的个数为**n**，不采用穿刺矩阵时，得到的码率为

$$R = \frac{1}{n+1}$$

- 可以综合上述两种方法得到任意码率的**Turbo**码。

SCCC的编码结构

- 在AWGN信道上对PCCC的性能仿真证明，当BER随SNR的增加下降到一定程度以后，就会出现下降缓慢甚至不再降低的情况，即出现了误码平台(error floor)。为此，S. Benedetto在1996年提出了串行级联卷积码(SCCC)的概念，如下图所示。

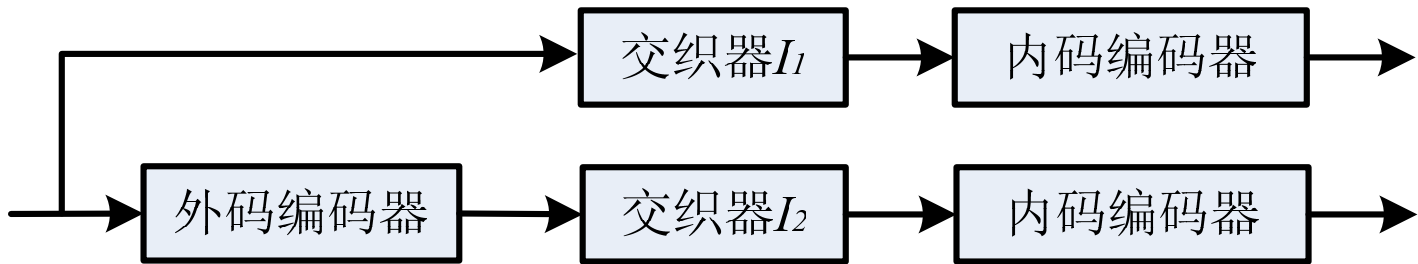


- 研究表明，为使SCCC达到较好的译码性能，至少其内码要采用递归系统卷积码，外码也应选择具有较好距离特性的卷积码。
- 若内外编码器的编码速率为 R_I 和 R_O ，则SCCC的编码速率为 $R=R_I \times R_O$ 。

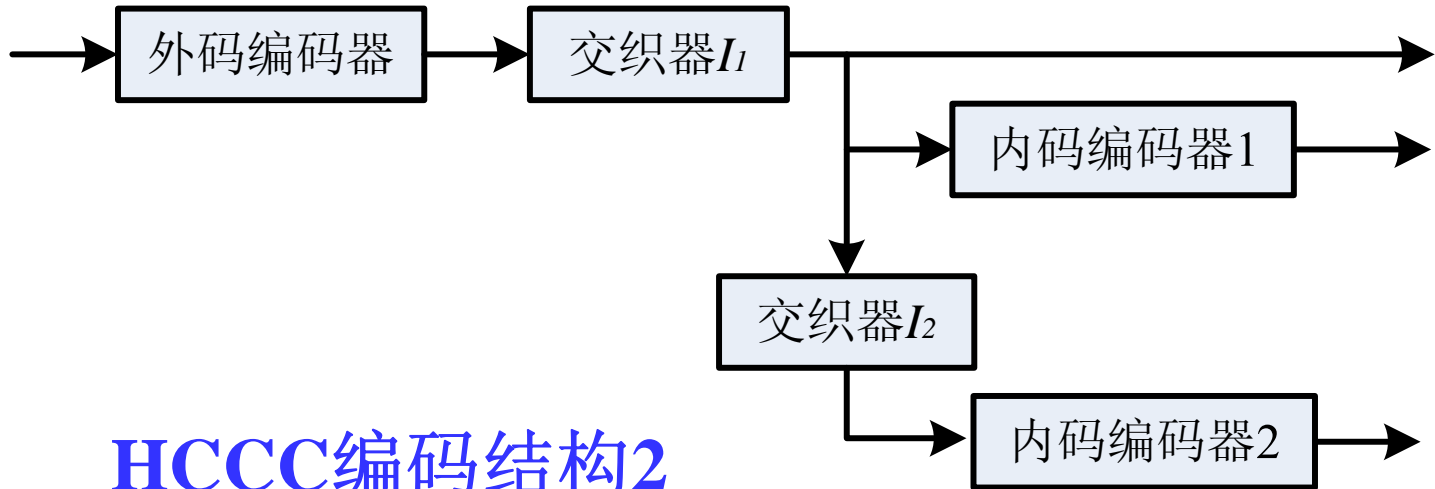
HCCC的编码结构

- 将上述两种方案结合起来，从而既能在低**SNR**条件下获得优异的译码性能，又能有效地消除**PCCC**所谓的误码平台，这种综合**PCCC**和**SCCC**的编码方案就称为混合级联卷积码(**HCCC**)。
- 综合串行和并行级联的方案很多，这里给出两种：一是采用卷积码和**SCCC**并行级联的编码方案；另一个是采用卷积码为外码，以**PCCC**为内码的混合级联编码结构。





HCCC编码结构1



HCCC编码结构2

6.9.3 交织器的设计原则

- **Turbo**码系统中交织器的作用是用于减少校验比特之间的相关性，进而在迭代译码过程中降低误比特率。
- 设计性能较好的交织器的特点和基本原则：
 - 通过增加交织器的长度，可以使译码性能得到提高，好的交织器可使总的码字的自由距离随交织器长度的增加而增加，即提供一定的交织器距离。
 - 交织器应该使输入序列尽可能地随机化，从而避免编码生成低重码字的信息序列在交织后编码仍旧生成低重码字，导致**Turbo**码的自由距离减小。

交织深度与**码重参数**是交织器设计时两个重要的参数指标，但它们之间还没有找到定量的关系式。目前，对交织器的设计一般都是采用计算机仿真的方法来搜索出较满意的交织器。

交织器的基本概念

➤ 交织实际上就是将数据序列中的元素的位置进行重置，从而得到交织序列的过程；其逆过程就是将交织后的序列元素恢复为原有顺序，也称为解交织。

➤ 例如，交织器 I 的输入为 $\mathbf{d} = (d_1, d_2, \dots, d_N)$

交织映射输出序列为 $\mathbf{d}' = (d'_1, d'_2, \dots, d'_N)$

序列 \mathbf{d}' 和 \mathbf{d} 仅仅是元素的位置顺序不同。如果把输入序列和交织输出序列看成一对含有 N 个元素的集合，则交织过程可以看成从集合 \mathbf{d} 到集合 \mathbf{d}' 的一个一一映射过程，即

$$I: d_i \rightarrow d'_j \quad i, j = 1, 2, \dots, N$$

交织器的设计

- 从信息论的角度看，在**turbo**码编码器中引入交织器的目的是实现随机性编码，但是在交织长度有限的实际情况下，实现完全随机编码是不可能的。交织长度越短，随机性越差，这时采用按照一定的确定规则设计的交织器可以得到比伪随机交织器更好的性能。而交织长度较大时，伪随机交织器或者满足一定距离属性要求的随机交织器可以获得比较好的性能。
- 因此，根据不同的设计思想，交织器大致可以分成两类：**规则交织器**和**随机交织器**。规则交织器通常按照一定的规则映射来实现交织，通常比较容易实现；基于随机性准则设计的交织器通常称为伪随机交织器。对于长度有限的输入信息序列而言，交织长度有限，实现完全随机是不可能的。当然也可以将规则交织和伪随机交织相结合作为交织器设计的方法。



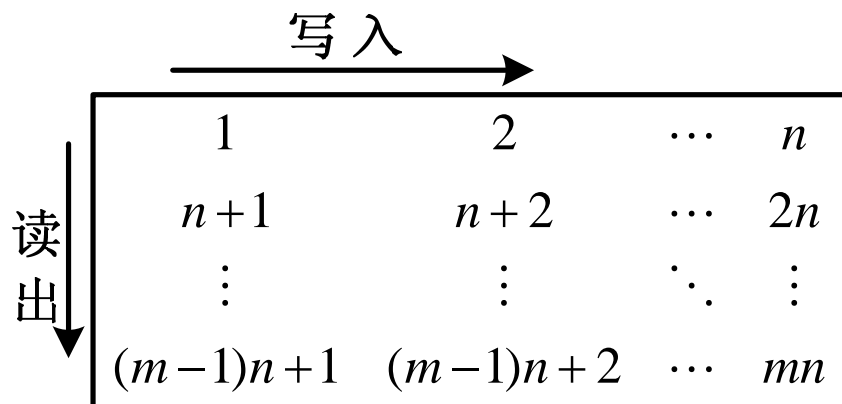
交织器性能优劣的衡量标准

- **交织前后比特之间的距离**。如果交织器能够通过交织在原始序列中距离较近的信息比特经过交织后有一定的距离，则可以在一定程度上提高**Turbo**码的性能。
- **在穿刺Turbo码中**，如果设计的交织器能够实现对系统比特的均匀保护，则有助于提高**Turbo**码的性能。例如，在采用伪随机交织器时，原始序列中某个位于奇数位置的比特经过交织后可能在交织序列的偶数位置出现，这样经过编码后它对应的两个校验比特也分别位于两个校验序列的奇数位置和偶数位置。根据前述穿刺方法，这两个校验比特要么都被删除，要么都被保留。在译码时，那些两个校验比特均被删除的信息比特出现错译的概率大大增加，降低了**Turbo**码的性能。因此所设计的交织器如果能够保证交织后信息比特位置的奇偶性不变，则有助于提高性能。
- 如果在**Turbo**码的编码器中引入交织器，那么在译码中就必须有与其相对应的解交织器，即需要两个设备来分别实现交织和解交织过程。如果所设计的**交织器满足对称特性**，则交织器和解交织器就是完全相同的，从而可以用同一个设备实现。

典型交织器设计

——块交织器

- 块交织器是最简单的一类交织器，其交织映射过程为：将数据序列按行写入 $m \times n$ 矩阵，然后按列的顺序读出，即完成交织；相应的解交织过程就是将交织后的数据按列的顺序写入，按行的顺序读出。



- 其交织映射函数可表示为：

$$I(i) = [(i-1) \bmod n] + \lfloor (i-1) / n \rfloor + 1, \quad i = 1, 2, \dots, N$$

N 为交织长度。

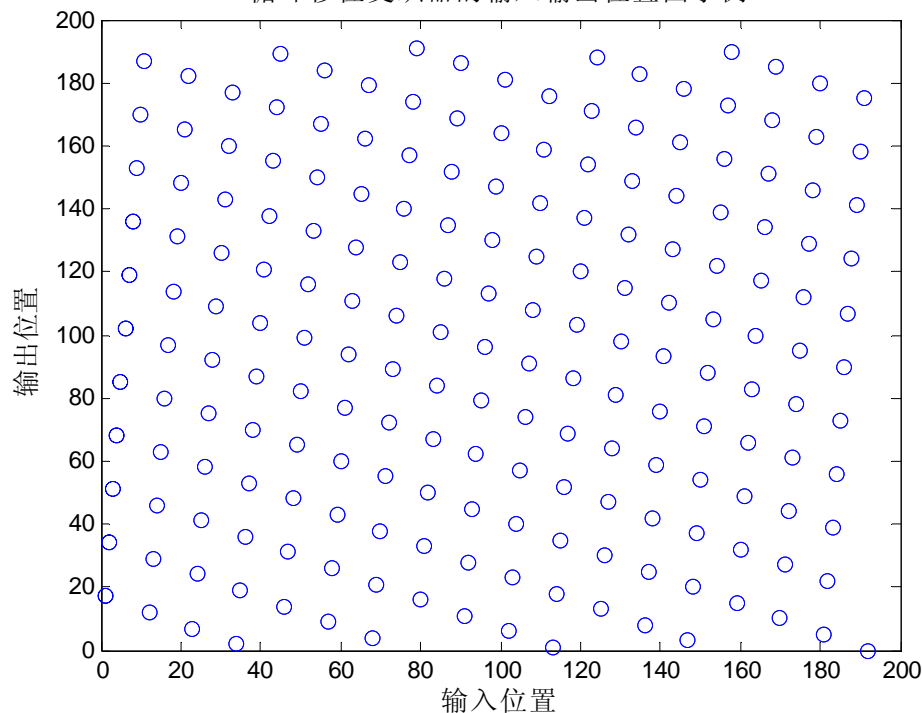
循环移位交织器

- 循环移位交织器的映射为： $I(i) = a * i \bmod N$
其中 a 是步长，为与交织长度 N 互素的正整数，且

$$a \leq \left\lfloor \sqrt{2N} - 1 \right\rfloor$$

- 步长 a 的值决定了原始序列中相邻的比特经过交织后在交织序列中的距离。例如右图中交织长度 $N=192$ 、 $a=17$ 的循环移位交织器的输入输出位置。

循环移位交织器的输入输出位置图示例

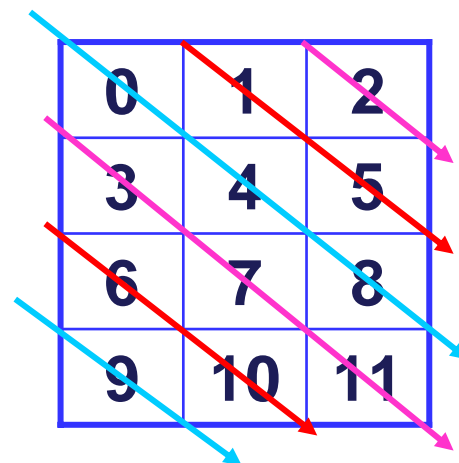


分组螺旋交织器

- 分组螺旋交织器首先将数据序列按行的顺序写入 $m \times n$ 矩阵，其中 m 与 n 互素。在交织时，从矩阵的左上角开始向右下方向读取数据，如下图所示。
- 在行的方向和列的方向分别对索引取模 m 和 n ，即若令 r_i 和 c_i 分别表示第 i 个比特的行索引和列索引，则分组螺旋交织器的数据读取顺序为：

$$r_{i+1} = r_i + 1 \bmod m$$

$$c_{i+1} = c_i + 1 \bmod n$$



伪随机交织器

伪随机交织器的交织过程可简单描述如下：

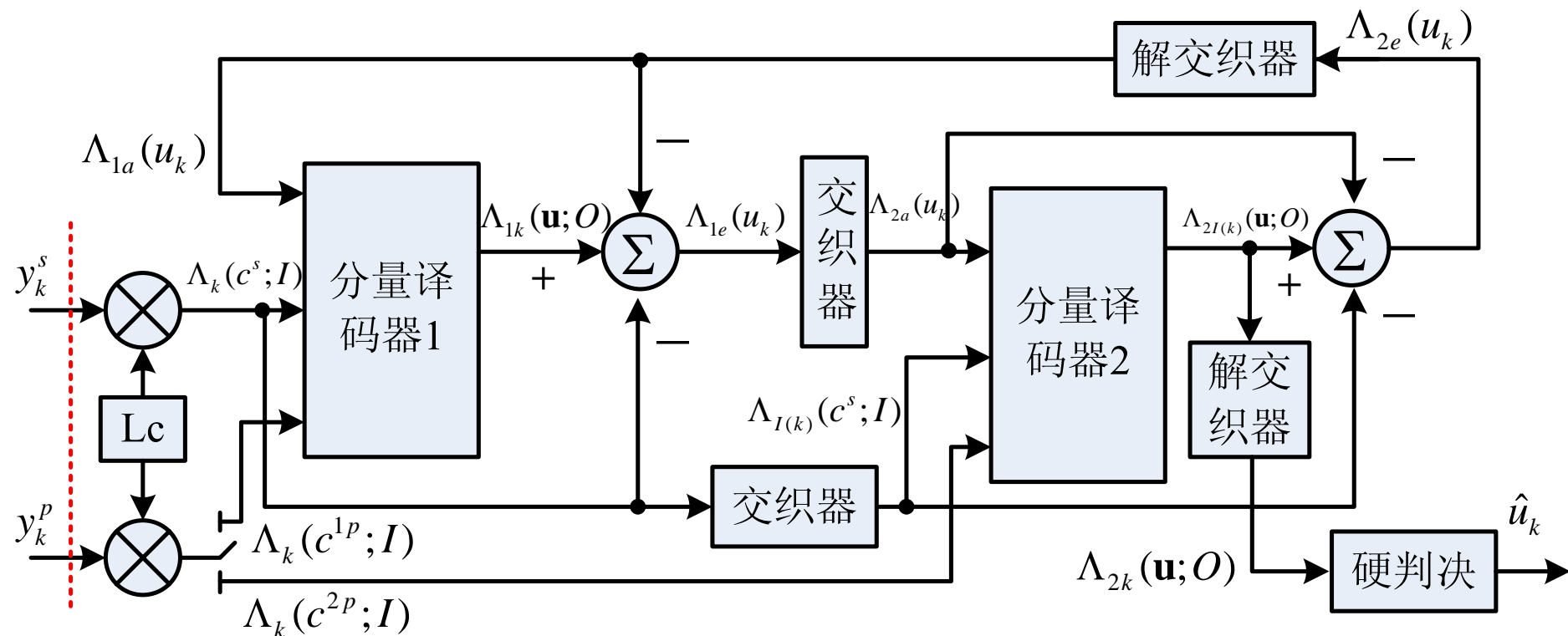
长为 n 的信息序列 $d_1 \quad d_2 \quad d_3 \quad d_4 \quad d_5 \quad d_6 \quad d_7 \quad d_8$
相应的 n 个随机数 $0.4 \quad 0.7 \quad 0.1 \quad 0.5 \quad 0.3 \quad 0.8 \quad 0.2 \quad 0.6$
按大小排列随机数 $0.1 \quad 0.2 \quad 0.3 \quad 0.4 \quad 0.5 \quad 0.6 \quad 0.7 \quad 0.8$
交织后的信息序列 $d_3 \quad d_7 \quad d_5 \quad d_1 \quad d_4 \quad d_8 \quad d_2 \quad d_6$

在以往的信道编码中使用交织器的目的主要是抗信道突发错误,而在**Turbo**码中,交织器除了抗信道突发错误外,主要是改变码的重量分布,控制编码序列的距离特性。

6.9.4 Turbo码的译码

- **Turbo**码获得优异性能的根本原因之一是采用了迭代译码，通过分量译码器之间软信息的交换来提高译码性能。对于并行级联码，如果分量译码器的输出为硬判决，则不可能实现分量译码器之间软信息的交换；同样，对于串行级联码，如果内码译码器的输出为硬判决结果，则外码译码器也无法采用软判决译码技术，从而限制了系统性能的进一步提高。从信息论的角度来看，任何硬判决都会损失部分信息，因此，如果分量译码器(内码译码器)能够提供一个反映其输出可靠性的软输出，则其他分量译码器(外码译码器)也可以来用软判决译码，从而系统的性能可以得到进一步提高。为此，人们又提出了软输出译码的概念和方法，即译码器的输入输出均为软信息。

PCCC的译码结构



➤ 假设编码输出信号为 $X_k = (x_k^s, x_k^p)$

➤ 接收信号为 $Y_k = (y_k^s, y_k^p)$

其中 $y_k^s = x_k^s + i_k$

$$y_k^p = x_k^p + q_k$$

i_k 和 q_k 是服从均值为0，方差为 $N_0/2$ 的独立同分布高斯随机变量。

➤ 经过s/p，可得到3个序列：

系统接收信息序列 $Y^s = (y_1^s, y_2^s, \dots, y_N^s)$

译码器1的接收校验序列 $Y^{1p} = (y_1^{1p}, y_2^{1p}, \dots, y_N^{1p})$

译码器2的接收校验序列 $Y^{2p} = (y_1^{2p}, y_2^{2p}, \dots, y_N^{2p})$

- 值得注意的是，若其中某些校验比特在编码过程中通过穿刺矩阵被删除了，则在接收校验序列的相应位置以“0”来补充。
- 上述三个接收序列 \mathbf{Y}^s 、 \mathbf{Y}^{1p} 和 \mathbf{Y}^{2p} 经过信道置信度 L_c 加权后作为系统信息序列 $\Lambda(c^s; I)$ 、校验信息 $\Lambda(c^{1p}; I)$ 和 $\Lambda(c^{2p}; I)$ 送入译码器。
- 对于**AWGN**信道， L_c 定义为：

$$L_c = 4\sqrt{E_s} / N_0$$

- 对于第 k 个被译比特，**PCCC**译码器中每个分量译码器都包括系统信息 $\Lambda_k(c^s; I)$ 、校验信息 $\Lambda_k(c^{ip}; I)$ 和先验信息 $\Lambda_{1a}(u_k)$ 。其中先验信息 $\Lambda_{1a}(u_k)$ 由另一个分量译码器生成的外部信息 $\Lambda_{3-i,e}(u_k)$ 经过解交织/交织后的对数似然比值。译码输出为对数似然比 $\Lambda_k(\mathbf{u}; O)$ ，其中 $i=1, 2$ 。

- 在迭代过程中，分量译码器1的输出 $\Lambda_{1k}(\mathbf{u}; O)$ 可表示为系统信息 $\Lambda_k(c^s; I)$ 、先验信息 $\Lambda_{1a}(u_k)$ 和外部信息 $\Lambda_{1e}(u_k)$ 之和的形式
- $$\Lambda_{1k}(\mathbf{u}; O) = \Lambda_k(c^s; I) + \Lambda_{1a}(u_k) + \Lambda_{1e}(u_k)$$

其中 $\Lambda_{1a}(u_{I(k)}) = \Lambda_{2e}(u_k)$

$I(k)$ 为交织映射函数。

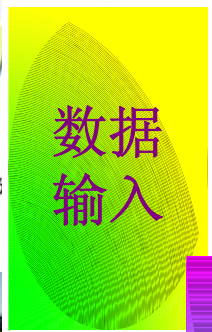
- 在第一次迭代时, $\Lambda_{2e}(u_k) = 0$, 从而 $\Lambda_{1a}(u_k) = 0$ 。
- 对于分量译码器**2**, 其外部信息 $\Lambda_{2e}(u_k)$ 为输出对数似然比 $\Lambda_{2k}(\mathbf{u}; O)$ 减去系统信息 $\Lambda_{I(k)}(c^s; I)$ (经过交织映射) 和先验信息 $\Lambda_{2a}(u_k)$ 的结果, 即

$$\Lambda_{2e}(u_k) = \Lambda_{2I(k)}(\mathbf{u}; O) - \Lambda_{I(k)}(c^s; I) - \Lambda_{2a}(u_k)$$

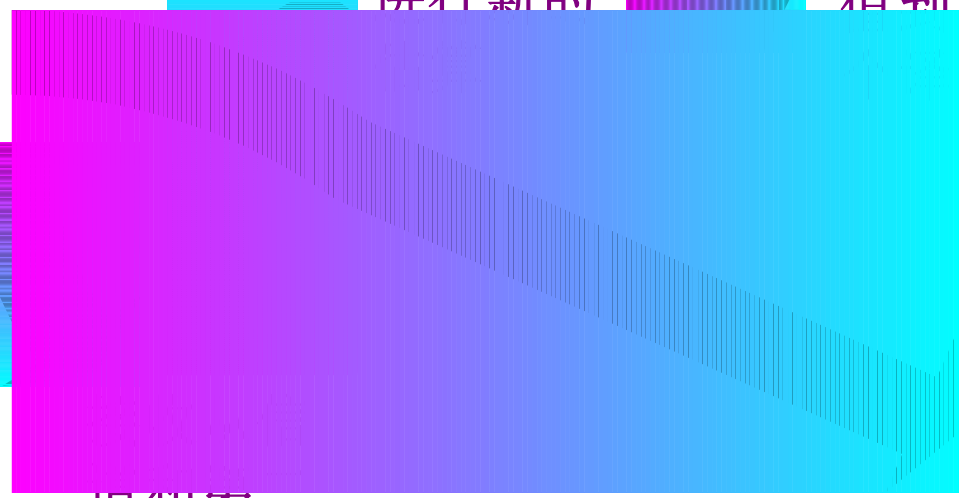
其中 $\Lambda_{2a}(u_k) = \Lambda_{1e}(u_{I(k)})$

外部信息 $\Lambda_{2e}(u_k)$ 解交织后反馈为分量译码器**1**的先验输入, 完成一轮迭代译码。

- 随着迭代次数的增加, 两个分量译码器得到的外部信息值对译码性能提高的作用会越来越小, 在一定迭代次数后, 译码性能不再提高。这时根据分量译码器**2**的输出 **LLR** 经过解交织后再进行硬判决即得到译码输出。

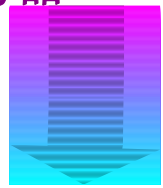


数据输入



根据信息进行新的

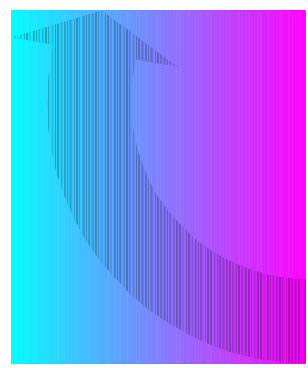
传递估算
值到第二
码器



接收从信
道和第一
个译码器
来的信息

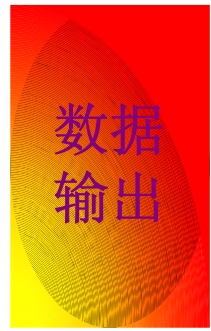
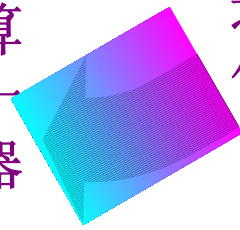


接收从第
一个译码器
来的信息



传递估算
值到第一
个译码器

根据信息
进行新的
估算



数据输出

Turbo码译码算法

Turbo码的译码算法主要有：最大后验概率(MAP)算法和软输出维特比译码算法(SOVA)。两者的共同点都是利用软输出来进行迭代译码。

MAP是最优的译码算法，但其缺点是具有较大的运算复杂度和需较大的存储空间；**SOVA**的译码性能虽不如**MAP**，但其运算复杂度较低，有利于硬件的实现。

MAP算法

在最初提出Turbo码时所采用的译码算法是修正的Bahl算法,也叫做最大后验概率(MAP)算法。它是Turbo码译码的最优算法。

MAP算法采用对数似然比函数(LLR),即后验概率比值的绝对值作为其软判决的输出。对于比特 u_k ,其后验概率表示为 $\Pr\{u_k = i / Y\}, i = 0, 1$, 软判决输出可表示为:

$$\Lambda(u_k) = \log \frac{P\{u_k = 1 / Y\}}{P\{u_k = 0 / Y\}}$$

其中, u_k 为信息序列, R 为观察序列。

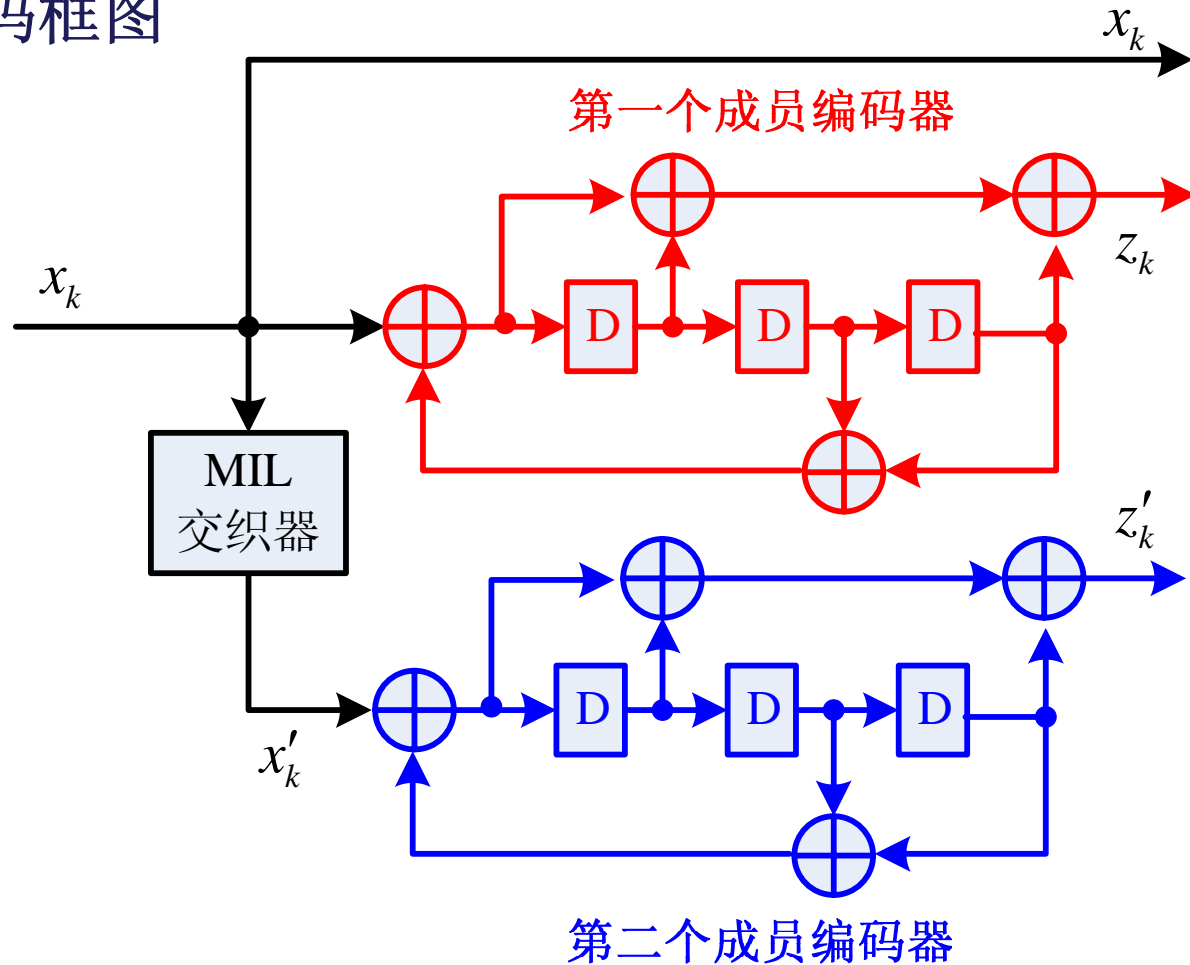
若 $\Lambda(u_k) > 1$ 判发 $u_k = 1$;反之判发 $u_k = 0$ 。

SOVA算法

对于卷积码，**Viterbi**算法是最优的最大似然译码方法，译码输出为卷积码的最优估计序列。但对于属于级联卷积码的**Turbo**码而言，传统的**Viterbi**算法存在两个缺陷：首先，一个分量译码器输出中存在的突发错误会影响另一个分量译码器的译码性能，从而使级联码的性能下降。其次，无论是软判决**Viterbi**算法还是硬判决**Viterbi**算法，其译码输出均为硬判决信息，若一个分量码采用**Viterbi**算法译码，则另一个分量译码器只能以硬判决结果作为输入，无法实现软判决译码，从而性能会有所下降。因此，如果**Viterbi**译码器能够提供软信息输出，则可以弥补上述两个缺陷，并且可以通过在分量译码器之间软信息的交换使级联码的性能大大提高。为此，需要在传统的**Viterbi**算法上进行修正，使之提供软信息输出，相应的算法就称为软输出**Viterbi**算法 (**SOVA**, **Soft Output Viterbi Algorithm**)。

6.9.5 WCDMA系统中的Turbo码

➤ 编码框图



成员编码器RSC

➤ 生成多项式

$$g_0(D)=1+D^2+D^3 \quad (13)_8$$

$$g_1(D)=1+D+D^3 \quad (15)_8$$

转移函数为：

$$G(D) = \begin{bmatrix} 1, \frac{g_1(D)}{g_0(D)} \end{bmatrix}$$

编码后的输出比特为：

$$x_1, z_1, z'_1, x_2, z_2, z'_2, \dots, x_K, z_K, z'_K$$

交织器

- 采用的是多级交织算法(MIL, Multi-stage InterLeaving method)的块交织器。协议规定，交织器的交织长度 K 在40~5114之间。具体的交织过程分三个步骤：
 - 比特流输入至内部矩阵；
 - 对内部矩阵进行行内重排和行间重排；
 - 对重排后的矩阵修剪输出；



1、比特流按行输入到交织矩阵

- 确定交织矩阵的行数**R**(从上到下依次为**0,1,2,..., R-1**);

$$R = \begin{cases} 5 & \text{if } 40 \leq K \leq 159 \\ 10 & \text{if } 160 \leq K \leq 200 \text{ or } 481 \leq K \leq 530 \\ 20 & \text{others} \end{cases}$$

- 确定交织矩阵的列数**C**(从左到右依次为**0,1,2,..., C-1**);

$$C = \begin{cases} p = 53 & \text{if } 481 \leq K \leq 530 \\ p - 1 & \text{if } K \leq R \times (p - 1) \\ p & \text{if } R \times (p - 1) < K \leq R \times p \\ p + 1 & \text{if } R \times p < K \leq R \times (p + 1) \end{cases}$$

其中**p**是满足 $K \leq R \times (p + 1)$ 的最小素数。

- 比特流从**0**行**0**列开始逐行写入到 **$R \times C$** 矩阵:

$$\begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_C \\ x_{C+1} & x_{C+2} & x_{C+3} & \cdots & x_{2C} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{C(R-1)+1} & x_{C(R-1)+2} & x_{C(R-1)+3} & \cdots & x_{RC} \end{bmatrix}$$

- 如果 $K < R \times C$, 则 $x_{K+1}, \dots, x_{R \times C}$ 可以任意填充**0**或者**1**, 因为它们在交织完成后还会被删除, 最终不会输出。

2、行内重排和行间重排

➤ 行内重排

(1) 根据p值从下表选取对应的原根v:

| p | v | p | v | p | v | p | v | p | v |
|----|---|----|---|-----|---|-----|----|-----|---|
| 7 | 3 | 47 | 5 | 101 | 2 | 157 | 5 | 223 | 3 |
| 11 | 2 | 53 | 2 | 103 | 5 | 163 | 2 | 227 | 2 |
| 13 | 2 | 59 | 2 | 107 | 2 | 167 | 2 | 229 | 6 |
| 17 | 3 | 61 | 2 | 109 | 6 | 173 | 2 | 233 | 3 |
| 19 | 2 | 67 | 2 | 113 | 3 | 179 | 2 | 239 | 7 |
| 23 | 5 | 71 | 7 | 127 | 3 | 181 | 2 | 241 | 7 |
| 29 | 2 | 73 | 5 | 131 | 2 | 191 | 19 | 251 | 6 |
| 31 | 3 | 79 | 3 | 137 | 3 | 193 | 5 | 257 | 3 |
| 37 | 2 | 83 | 2 | 139 | 2 | 197 | 2 | | |
| 41 | 6 | 89 | 3 | 149 | 2 | 199 | 3 | | |
| 43 | 3 | 97 | 5 | 151 | 6 | 211 | 2 | | |



(2) 建立一个行内重排基准序列 $s(i)$:

$$s(i)=[v \times s(i-1)] \bmod p, i=1,2,\dots,(p-2), s(0)=1$$

(3) 建立各行行内重排的步长 $\{q_j\}, j=1,2,\dots,R-1, q_0=1$

必须满足 $\text{g.c.d}\{q_j, p-1\}=1, q_j>6$ 且 $q_j>q_{j-1}$

注: g.c.d (greatest common divisor) 是取最大公约数;

(4) 计算 $\{r_j\}$

$$r_{T(j)}=q_j, j=0, 1, \dots, R-1$$

根据输入比特数目 K 的不同确定四种重排模式 $T(j): \text{Pat}_1, \text{Pat}_2, \text{Pat}_3, \text{Pat}_4$ 。 j 为重排后的行号, $T(j)$ 为重排后第 j 行在重排前的行号。



$$T(j) = \begin{cases} Pat_4 & 40 \leq K \leq 159 \\ Pat_3 & 160 \leq K \leq 200 \\ Pat_1 & 201 \leq K \leq 480 \\ Pat_3 & 481 \leq K \leq 530 \\ Pat_1 & 531 \leq K \leq 2280 \\ Pat_2 & 2281 \leq K \leq 2480 \\ Pat_1 & 2481 \leq K \leq 3160 \\ Pat_2 & 3161 \leq K \leq 3210 \\ Pat_1 & 3211 \leq K \leq 5114 \end{cases}$$

➤ 重排模式为:

Pat₁: {19,9,14,4,0,2,5,7,12,18,10,8,13,17,3,1,16,6,15,11}

Pat₂: {19,9,14,4,0,2,5,7,12,18,16,13,17,15,3,1,6,11,8,10}

Pat₃: {9,8,7,6,5,4,3,2,1,0}

Pat₄: {4,3,2,1,0}



(5) 进行行内重排

$U_j(i)$ 为第j行的第i列在重排前的列号。

若 $C=p$, 则

$$U_j(i) = s\left(\left\lfloor i \times r_j \right\rfloor \bmod (p-1)\right), \quad i = 0, 1, 2, \dots, (p-2), \quad U_j(p-1) = 0$$

若 $C=p+1$, 则

$$U_j(i) = s\left(\left\lfloor i \times r_j \right\rfloor \bmod (p-1)\right), \quad i = 0, 1, 2, \dots, (p-2), \quad U_j(p-1) = 0, \quad U_j(p) = 0$$

若 $K=C \times R$, 则还得交换 $U_{R-1}(p)$ 和 $U_{R-1}(0)$ 。

若 $C=p-1$, 则

$$U_j(i) = s\left(\left\lfloor i \times r_j \right\rfloor \bmod (p-1)\right) - 1, \quad i = 0, 1, 2, \dots, (p-2)$$

➤ 行间重排

将完成行内重排的矩阵基于行间重排模式 $T(j)$ 作行间重排,
 $T(j)$ 是重排后第j行在重排前的行号。

3、逐列带删减的比特输出

- 经过行内、行间重排后，矩阵转换为：

$$\begin{bmatrix} y'_1 & y'_{R+1} & y'_{2R+1} & \cdots & y'_{(C-1)R+1} \\ y'_2 & y'_{R+2} & y'_{2R+2} & \cdots & y'_{(C-1)R+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y'_R & y'_{2R} & y'_{3R} & \cdots & y'_{CR} \end{bmatrix}$$

- **Turbo**码交织器的输出是从重排后的 $R \times C$ 矩阵中从**0**行**0**列 y'_1 开始
 一列一列从上向下依次读到 **$R-1$** 行 **$C-1$** 列的 y'_{CR} 。输出时删除了输入序列中不存在的 $x_k(k > K)$ 重排对应的 y'_k 。实际上**Turbo**码内部
 交织器最后输出的比特数目仍为 **K** ，删除的比特数目为 **$R \times C - K$** 。

例如：假设信息长度 $K=62$

- 确定行数： $R=5$
- 确定列数： $C=13$, $p=13$
- 写入 5×13 矩阵：

| | | | | | | | | | | | | |
|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
| x_1 | x_2 | x_3 | x_4 | x_5 | x_6 | x_7 | x_8 | x_9 | x_{10} | x_{11} | x_{12} | x_{13} |
| x_{14} | x_{15} | x_{16} | x_{17} | x_{18} | x_{19} | x_{20} | x_{21} | x_{22} | x_{23} | x_{24} | x_{25} | x_{26} |
| x_{27} | x_{28} | x_{29} | x_{30} | x_{31} | x_{32} | x_{33} | x_{34} | x_{35} | x_{36} | x_{37} | x_{38} | x_{39} |
| x_{40} | x_{41} | x_{42} | x_{43} | x_{44} | x_{45} | x_{46} | x_{47} | x_{48} | x_{49} | x_{50} | x_{51} | x_{52} |
| x_{53} | x_{54} | x_{55} | x_{56} | x_{57} | x_{58} | x_{59} | x_{60} | x_{61} | x_{62} | 0 | 0 | 0 |

行内重排

(1) 由于 $p=13$, 查表得 $v=2$

(2) $s(0)=1$, 由 $s(i)=[v \times s(i-1)] \bmod p$, $i=1,2,\dots,(p-2)$, 可得到

$s(1)=2, s(2)=4, s(3)=8, s(4)=3, s(5)=6, s(6)=12$

$s(7)=11, s(8)=9, s(9)=5, s(10)=10, s(11)=7$

(3) $q_0=1$, so $q_1=7, q_2=11, q_3=13, q_4=17$

(4) $r_4=q_0=1, r_3=q_1=7, r_2=q_2=11, r_1=q_3=13, r_0=q_4=17$

(5) 由于 $C=p$, $U_j(i) = s\left(\left\lfloor i \times r_j \right\rfloor \bmod (p-1)\right)$, $i=0,1,2,\dots,(p-2)$, $U_j(p-1)=0$

$U_j(i)$ 为第 j 行的第 i 列在重排前的列号

对于第 0 行: 当 i 从 0 到 $p-2=11$ 变化时, 得到

$U_0(0)=s(0)=1, U_0(1)=s(5)=6, U_0(2)=s(10)=10, U_0(3)=s(3)=8$

$U_0(4)=s(8)=9, U_0(5)=s(1)=2, U_0(6)=s(6)=12, U_0(7)=s(11)=7$

$U_0(8)=s(4)=3, U_0(9)=s(9)=5, U_0(10)=s(2)=4, U_0(11)=s(7)=11, U_0(12)=0$

➤ 这样对于第**0**行的数据，行内重排的结果为：

1 6 10 8 9 2 12 7 3 5 4 11 0

即对应着 x_2 x_7 x_{11} x_9 x_{10} x_3 x_{13} x_8 x_4 x_6 x_5 x_{12} x_1

➤ 同样的道理，对于第**1**行，当*i*从**0**到**p-2=11**变化时，得到

$U_0(0)=s(0)=1$, $U_0(1)=s(1)=2$, $U_0(2)=s(2)=4$, $U_0(3)=s(3)=8$

$U_0(4)=s(4)=3$, $U_0(5)=s(5)=6$, $U_0(6)=s(6)=12$, $U_0(7)=s(7)=11$

$U_0(8)=s(8)=9$, $U_0(9)=s(9)=5$, $U_0(10)=s(10)=10$, $U_0(11)=s(11)=7$,
 $U_0(12)=0$

➤ 这样对于第**1**行的数据，行内重排的结果为：

1 2 4 8 3 6 12 11 9 5 10 7 0

即对应着 x_{15} x_{16} x_{18} x_{22} x_{17} x_{20} x_{26} x_{25} x_{23} x_{19} x_{24} x_{21} x_{14}

- 同样的道理，对于第2行，当i从0到p-2=11变化时，得到
- $U_0(0)=s(0)=1, U_0(1)=s(11)=7, U_0(2)=s(10)=10, U_0(3)=s(9)=5$
- $U_0(4)=s(8)=9, U_0(5)=s(7)=11, U_0(6)=s(6)=12, U_0(7)=s(5)=6$
- $U_0(8)=s(4)=3, U_0(9)=s(3)=8, U_0(10)=s(2)=4, U_0(11)=s(1)=2, U_0(12)=0$
- 这样对于第2行的数据，行内重排的结果为：

1 7 10 5 9 11 12 6 3 8 4 2 0

即对应着 x_{28} x_{34} x_{37} x_{32} x_{36} x_{38} x_{39} x_{33} x_{30} x_{35} x_{31} x_{29} x_{27}

➤ 同样的道理，对于第3行，当i从0到p-2=11变化时，得到

$$U_0(0)=s(0)=1, U_0(1)=s(7)=11, U_0(2)=s(2)=4, U_0(3)=s(9)=5$$

$$U_0(4)=s(4)=3, U_0(5)=s(11)=7, U_0(6)=s(6)=12, U_0(7)=s(1)=2$$

$$U_0(8)=s(8)=9, U_0(9)=s(3)=8, U_0(10)=s(10)=10, U_0(11)=s(5)=6,$$

$$U_0(12)=0$$

➤ 这样对于第2行的数据，行内重排的结果为：

| | | | | | | | | | | | | | |
|------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
| | 1 | 11 | 4 | 5 | 3 | 7 | 12 | 2 | 9 | 8 | 10 | 6 | 0 |
| 即对应着 | x_{41} | x_{51} | x_{44} | x_{45} | x_{43} | x_{47} | x_{52} | x_{42} | x_{49} | x_{48} | x_{50} | x_{46} | x_{40} |

➤ 同样的道理，对于第4行，当i从0到p-2=11变化时，得到
 $U_0(0)=s(0)=1$, $U_0(1)=s(1)=2$, $U_0(2)=s(2)=4$, $U_0(3)=s(3)=8$
 $U_0(4)=s(4)=3$, $U_0(5)=s(5)=6$, $U_0(6)=s(6)=12$, $U_0(7)=s(7)=11$
 $U_0(8)=s(8)=9$, $U_0(9)=s(9)=5$, $U_0(10)=s(10)=10$, $U_0(11)=s(11)=7$,
 $U_0(12)=0$

➤ 这样对于第1行的数据，行内重排的结果为：

| | | | | | | | | | | | | | |
|------|----------|----------|----------|----------|----------|----------|----|----|----------|----------|----|----------|----------|
| | 1 | 2 | 4 | 8 | 3 | 6 | 12 | 11 | 9 | 5 | 10 | 7 | 0 |
| 即对应着 | x_{54} | x_{55} | x_{57} | x_{61} | x_{56} | x_{59} | 0 | 0 | x_{62} | x_{58} | 0 | x_{60} | x_{53} |

➤ 经过行内重排后，矩阵变为：

$$\begin{bmatrix} x_2 & x_7 & x_{11} & x_9 & x_{10} & x_3 & x_{13} & x_8 & x_4 & x_6 & x_5 & x_{12} & x_{11} \\ x_{15} & x_{16} & x_{18} & x_{22} & x_{17} & x_{20} & x_{26} & x_{25} & x_{23} & x_{19} & x_{24} & x_{21} & x_{14} \\ x_{28} & x_{34} & x_{37} & x_{32} & x_{36} & x_{38} & x_{39} & x_{33} & x_{30} & x_{35} & x_{31} & x_{29} & x_{27} \\ x_{41} & x_{51} & x_{44} & x_{45} & x_{43} & x_{47} & x_{52} & x_{42} & x_{49} & x_{48} & x_{50} & x_{46} & x_{40} \\ x_{54} & x_{55} & x_{57} & x_{61} & x_{56} & x_{59} & 0 & 0 & x_{62} & x_{58} & 0 & x_{60} & x_{53} \end{bmatrix}$$

➤ 经过行间重排后，矩阵变为：

$$\begin{bmatrix} x_{54} & x_{55} & x_{57} & x_{61} & x_{56} & x_{59} & 0 & 0 & x_{62} & x_{58} & 0 & x_{60} & x_{53} \\ x_{41} & x_{51} & x_{44} & x_{45} & x_{43} & x_{47} & x_{52} & x_{42} & x_{49} & x_{48} & x_{50} & x_{46} & x_{40} \\ x_{28} & x_{34} & x_{37} & x_{32} & x_{36} & x_{38} & x_{39} & x_{33} & x_{30} & x_{35} & x_{31} & x_{29} & x_{27} \\ x_{15} & x_{16} & x_{18} & x_{22} & x_{17} & x_{20} & x_{26} & x_{25} & x_{23} & x_{19} & x_{24} & x_{21} & x_{14} \\ x_2 & x_7 & x_{11} & x_9 & x_{10} & x_3 & x_{13} & x_8 & x_4 & x_6 & x_5 & x_{12} & x_{11} \end{bmatrix}$$

$$\begin{bmatrix} x_{54} & x_{55} & x_{57} & x_{61} & x_{56} & x_{59} & 0 & 0 & x_{62} & x_{58} & 0 & x_{60} & x_{53} \\ x_{41} & x_{51} & x_{44} & x_{45} & x_{43} & x_{47} & x_{52} & x_{42} & x_{49} & x_{48} & x_{50} & x_{46} & x_{40} \\ x_{28} & x_{34} & x_{37} & x_{32} & x_{36} & x_{38} & x_{39} & x_{33} & x_{30} & x_{35} & x_{31} & x_{29} & x_{27} \\ x_{15} & x_{16} & x_{18} & x_{22} & x_{17} & x_{20} & x_{26} & x_{25} & x_{23} & x_{19} & x_{24} & x_{21} & x_{14} \\ x_2 & x_7 & x_{11} & x_9 & x_{10} & x_3 & x_{13} & x_8 & x_4 & x_6 & x_5 & x_{12} & x_{11} \end{bmatrix}$$

➤ 输出序列为:

$x_{54}, x_{41}, x_{28}, x_{15}, x_2, x_{55}, x_{51}, x_{34}, x_{16}, x_7, x_{57}, x_{44}, x_{37},$
 $x_{18}, x_{11}, x_{61}, x_{45}, x_{32}, x_{22}, x_9, x_{56}, x_{43}, x_{36}, x_{17}, x_{10}, x_{59},$
 $x_{47}, x_{38}, x_{20}, x_3, x_{52}, x_{39}, x_{26}, x_{13}, x_{42}, x_{33}, x_{25}, x_8, x_{62},$
 $x_{49}, x_{30}, x_{23}, x_4, x_{58}, x_{48}, x_{35}, x_{19}, x_6, x_{50}, x_{31}, x_{24}, x_5,$
 $x_{60}, x_{46}, x_{29}, x_{21}, x_{12}, x_{53}, x_{40}, x_{27}, x_{14}, x_{11}$