16.Turbo码

引言

- Turbo码是在1993年4月瑞士日内瓦国际通信会议(ICC'93-IEEE International Conference on Communications）上由两位法国工程师Claude Berrou和Alain Glavieux首先提出的
- 在译码器复杂度相当的情况下，Turbo的性能远远优于传统的卷积码

引言

- 根据Shannon信道编码定理，为达到10^{-6}的误码率，需要E_b/N_0 = -1.6dB.

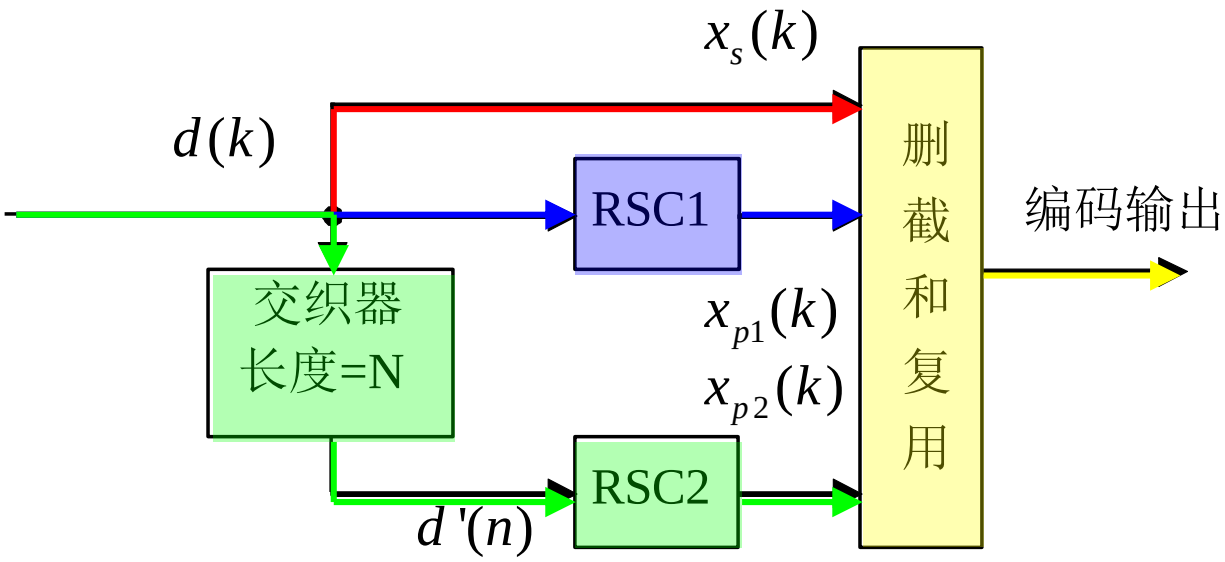
信道编码	与Shannon极限差距
不采用信道编码	12dB
卷积码+序列译码	3dB
级联码（RS码+卷积码）	2.3dB
Turbo码	0.5dB
LDPC	0.04dB

目录

- 编码器结构
- 解码原理
- 软输出译码算法
- 性能分析

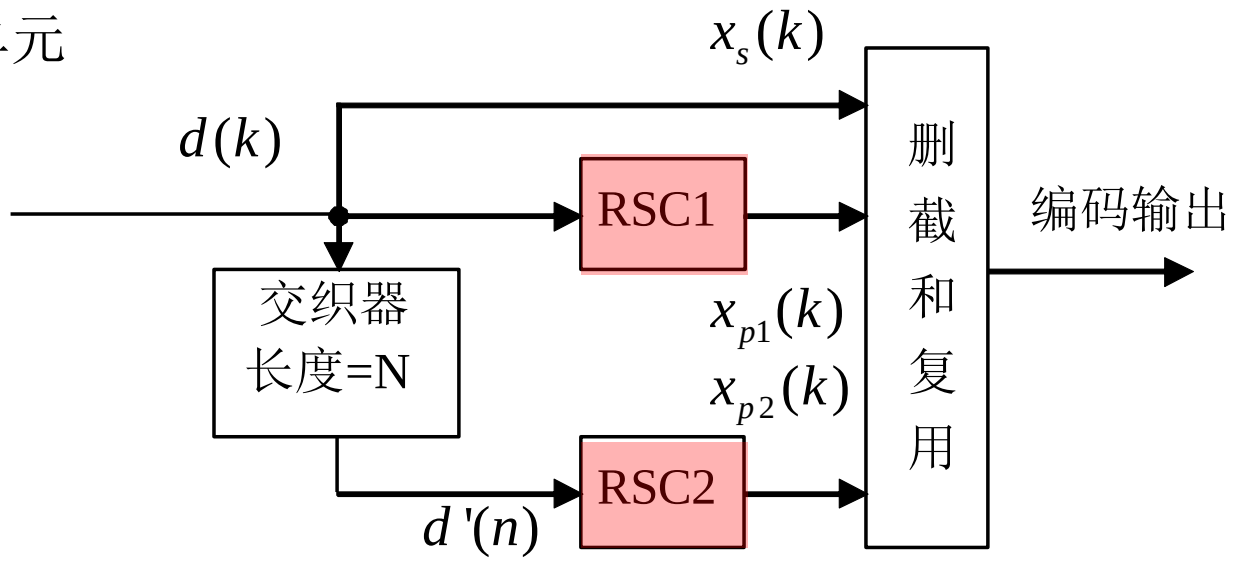
编码器结构

- Turbo Code采用的是并行级联系统递归卷积码编码器结构：



编码器结构

- 组成部分：
  - 分量卷积编码器
  - 交织器
  - 删截和复用单元



编码器结构—分量编码器

- 卷积编码方法被广泛用于数字通信系统中
- 分类：
  - 系统卷积编码和非系统卷积编码
  - 递归卷积编码和非递归卷积编码
- 实际常用：
  - 非系统卷积码

    NSC——Non Systematic Convolutional Codes
  - 系统递归卷积码

    RSC——Recursive Systematic Convolutional Codes
  - Turbo码中一般采用RSC编码器，有利于通过交织改变码的重量分布

编码器结构—分量编码器

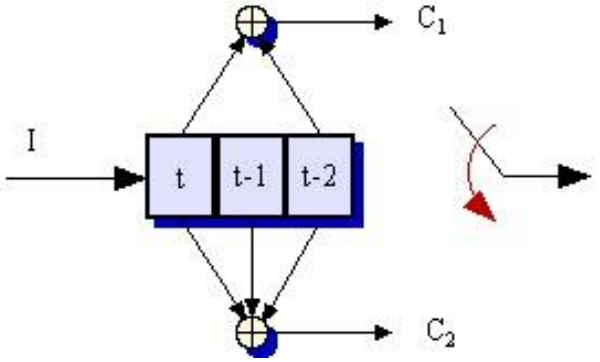
- NSC编码器

  对于 (K,1/2)非系统卷积码（*NSC*）编码器，*k*时刻输入到编码器的信息比特$d(k)$与相应的二进制编码输出之间的关系为：

$$x_{p1}(k)=\sum_{i=0}^{v} g_{1i}d(k-i) \quad \text{mod.2} \quad g_{1i}=0,1$$

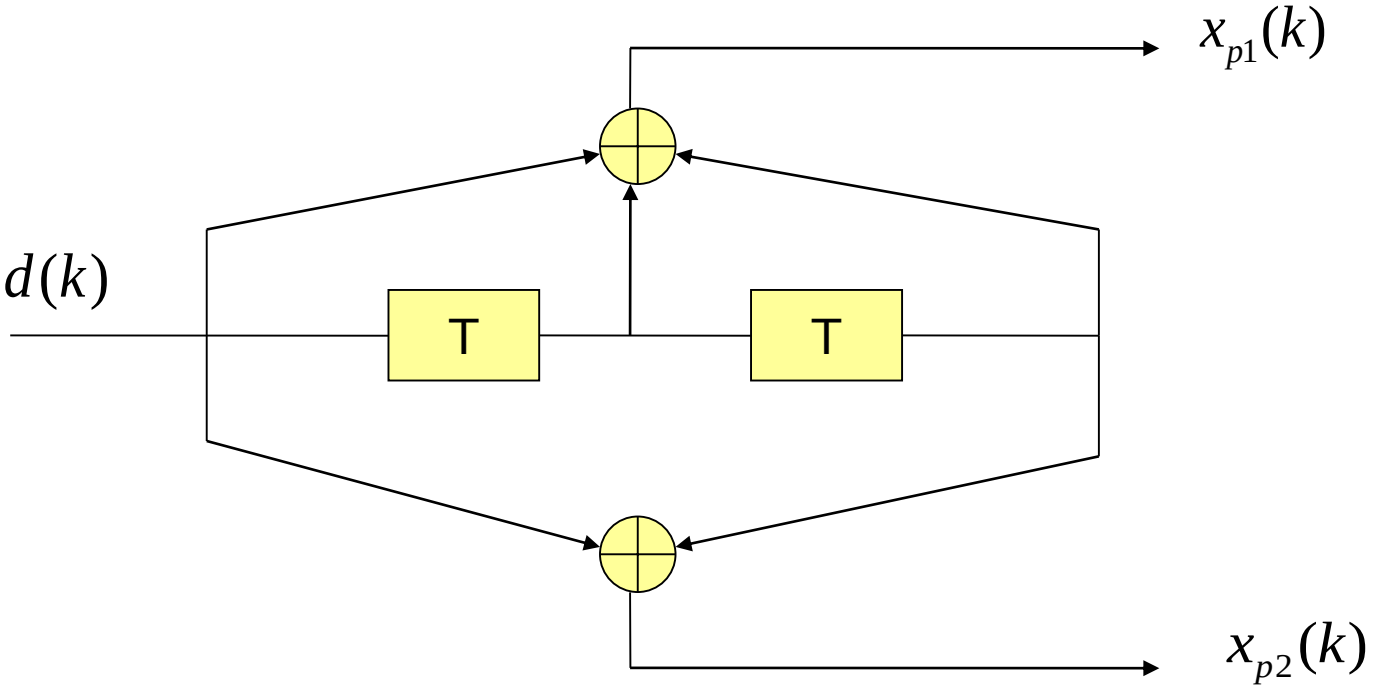
$$x_{p2}(k)=\sum_{i=0}^{v} g_{2i}d(k-i) \quad \text{mod.2} \quad g_{2i}=0,1$$

编码器结构—分量编码器



- NSC编码器

  若相应的NSC编码器结构为

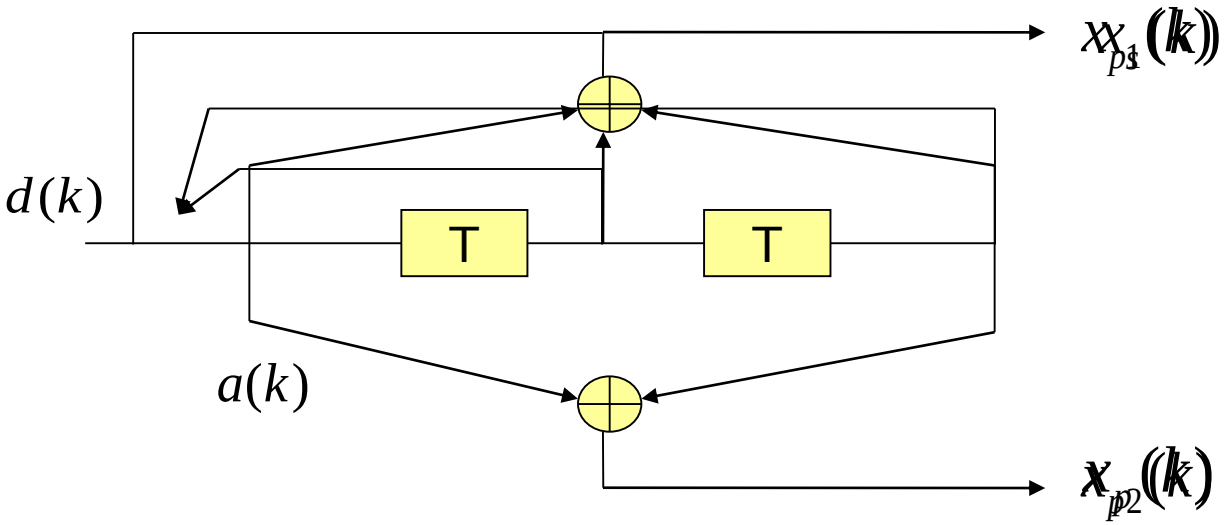


编码器结构—分量编码器

- RSC编码器

  只需做如下改动，就可以将NSC编码器变成RSC编码器：

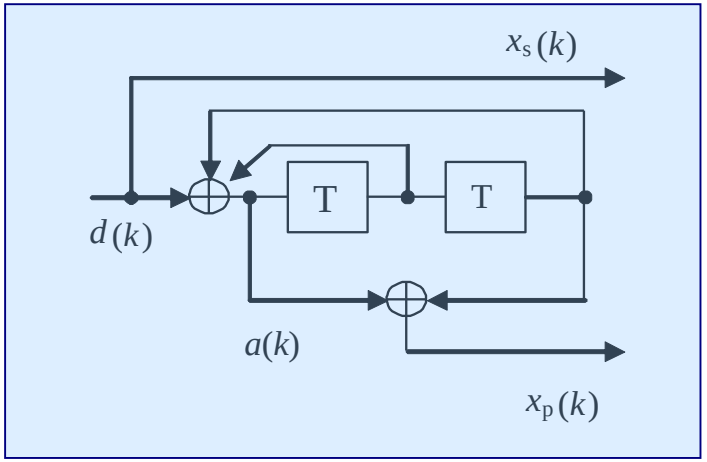
  - 令其中一路输出$x_s(k)$直接等于输入比特$d(k)$
  - 添加一个反馈环路



编码器结构—分量编码器

- RSC编码器

此时，移位寄存器的输入不再是$d(k)$，而变成$a(k)$，二者之间的关系：



$$a(k)=d(k)+\sum_{i=1}^{v}g_{1i}a(k-i)\ \ \mathrm{mod.2}$$

$$d(k)=\sum_{i=0}^{v}g_{1i}a(k-i)\ \ \mathrm{mod.2}$$

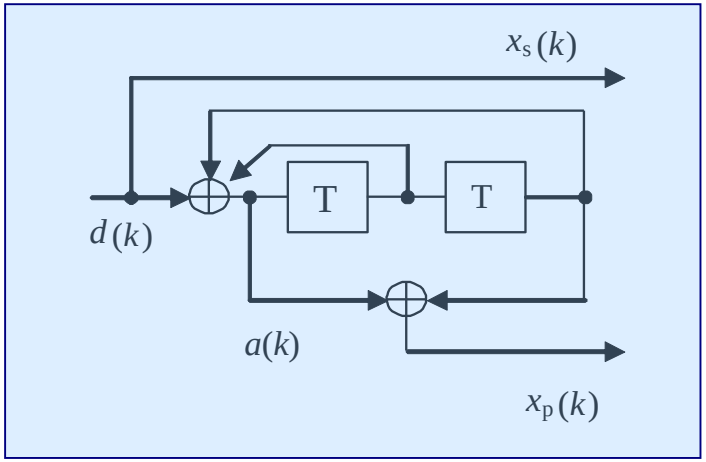
编码器结构—分量编码器

- RSC编码器
  - RSC编码器的输出为

$$\left[x_s(k), x_p(k)\right]$$

$$x_s(k) = d(k)$$

$$x_p(k) = \sum_{i=0}^{v} g_{2i} a(k-i) \qquad g_{2i} = 0,1$$



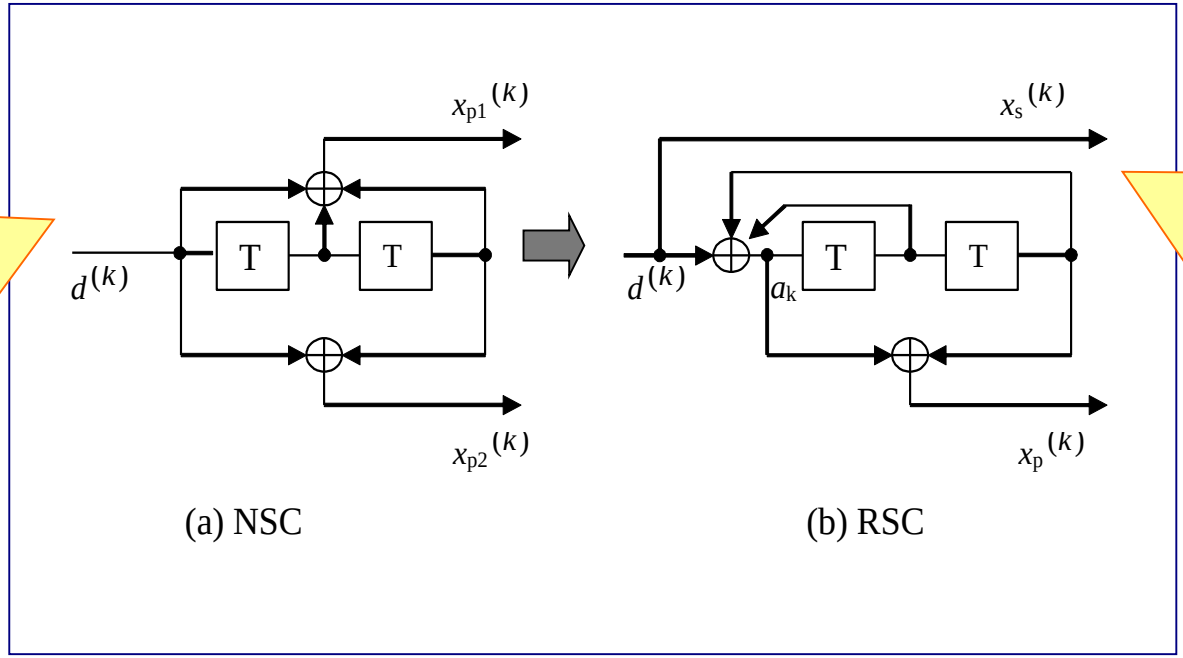
编码器结构—分量编码器

- NSC到RSC的转换

  例如前面讲过的例子

*NSC*码
生成矩阵

$$G=\begin{bmatrix}1+D+D^2,\\1+D^2\end{bmatrix}$$



*RSC*码
生成矩阵

$$G=\begin{bmatrix}1,\\\dfrac{1+D^2}{1+D+D^2}\end{bmatrix}$$

编码器结构—分量编码器

- NSC到RSC的转换
  - 根据上述NSC到RSC的转换过程，可以直接由任意（K，$1/n$）NSC码的生成矩阵来获得相应RSC的生成矩阵
  - 如果码率为$1/n$的NSC的生成矩阵为

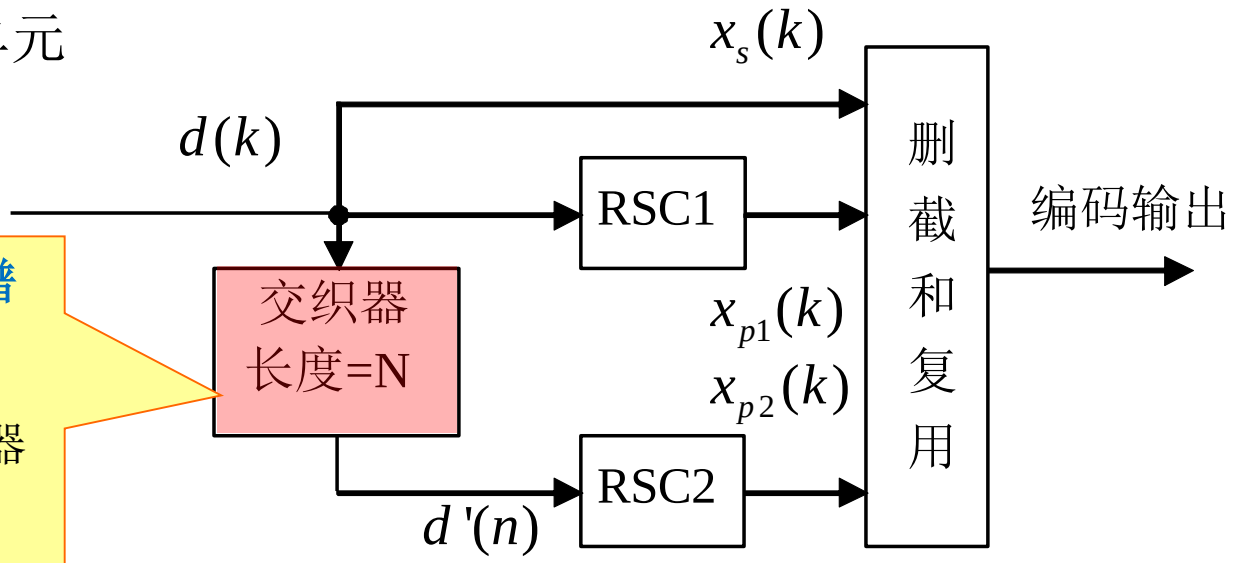
$$G(D)=\begin{pmatrix} g_0(D) & g_1(D)\cdots g_{n-1}(D) \end{pmatrix}$$

则相应RSC的生成矩阵为

$$G(D)=\left(1,\frac{g_1(D)}{g_0(D)},\cdots,\frac{g_{n-1}(D)}{g_0(D)}\right)$$

编码器结构

- 组成部分：
  - 分量卷积编码器
  - 交织器
  - 删截和复用单元



交织器对Turbo码重量谱起着关键作用

它是连接两个子编译码器之间的桥梁

它能使整个码接近随机编码的特性

编码器结构—交织器

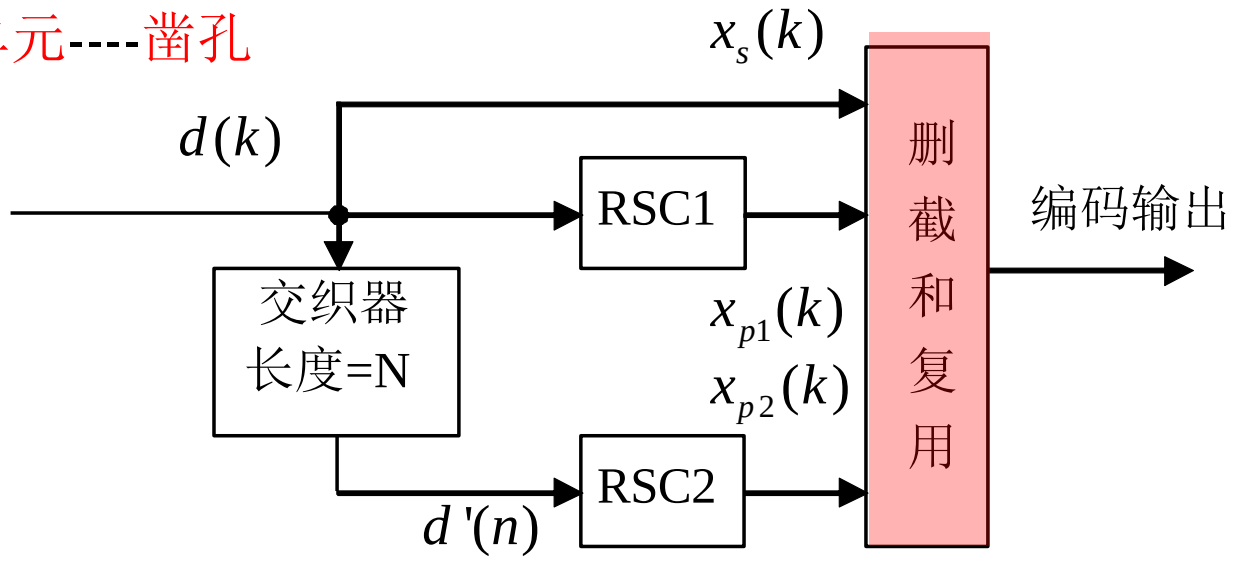
- 交织器长度对Turbo码性能的影响
  - 一般情况下，码中具有最小汉明重量的码字数量越多，在传送过程中发生误码的可能就越大
  - 因此可以用最小汉明重量码字的数量衡量其性能
  - 定义最小汉明重量码字的数量为D_m，则对于Turbo码，D_m近似地随交织长度N的增加按照$1/N$递减，从而极大地降低了误比特率

编码器结构—交织器

- 交织器长度对Turbo码性能的影响
  - 例如，当交织长度为1000时，可得到4dB以上的增益，这种增益被称为“交织增益”
  - 一方面，选择交织长度较长的交织器，可以在一定程度上提高Turbo码的性能
  - 另一方面，较长的交织器长度也带来了较大的编解码时延!

编码器结构

- 组成部分：
  - 分量卷积编码器
  - 交织器
  - 删截和复用单元----凿孔



编码器结构—凿孔

- NSC码的凿孔
  - 若要提高编码速率，通常需要考虑凿孔
  - 凿孔即按照某种方式，将输出数据中的一些比特删去
  - 对于每种特定NSC的卷积码，都有相应的**最佳凿孔图案**，即相应的凿孔矩阵

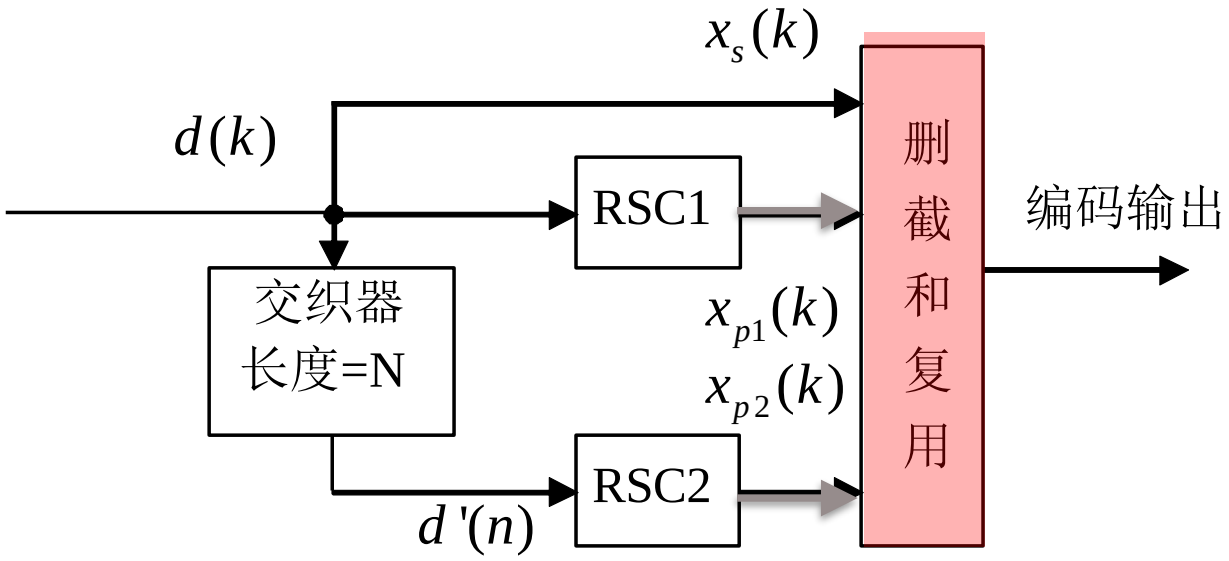
➢ 例如对于 $G_1 = 7, G_2 = 5$ ，码率为1/2的卷积码，要产生码率为2/3的卷积码，对应的最佳凿孔矩阵为：

$$P = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

表示每2个信息比特送入编码器，生成4个编码符号，发送时只发送前3个比特，而将第四个比特去掉

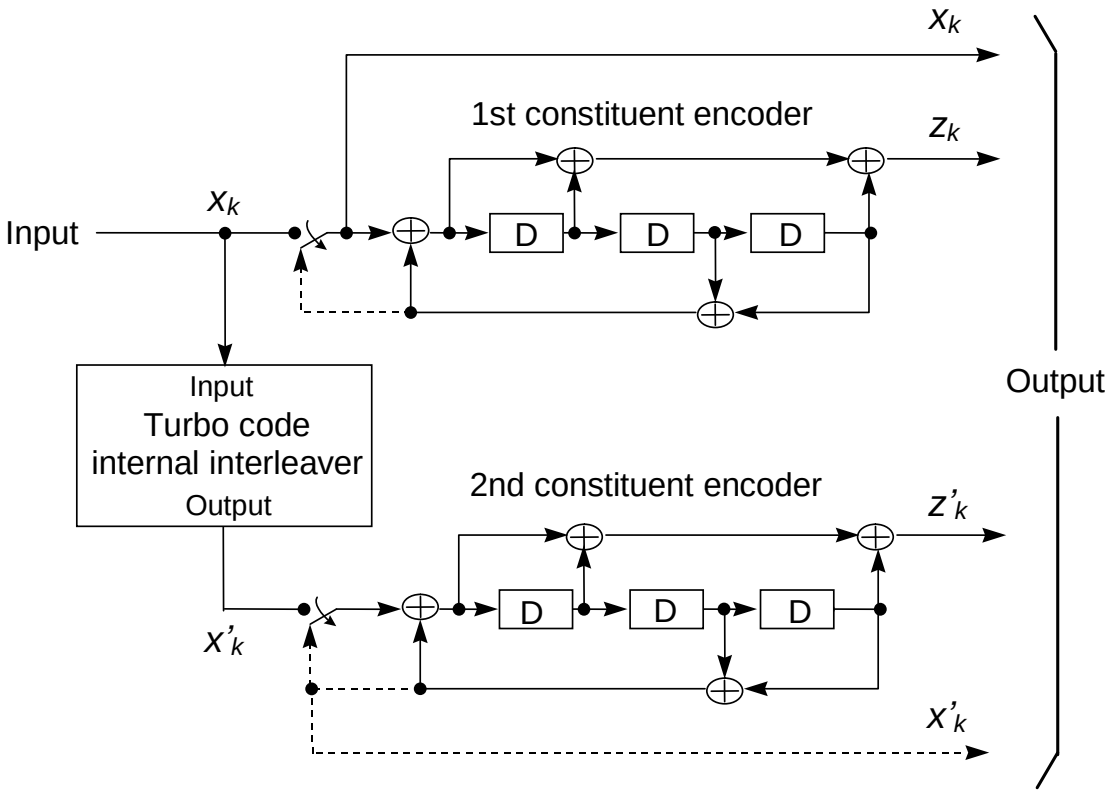
编码器结构—凿孔

- RSC码的凿孔
  - 为了使RSC凿孔后仍然保持系统码的特性，凿孔应发生在RSC的$x_p(k)$所在的输出端口上



编码器结构

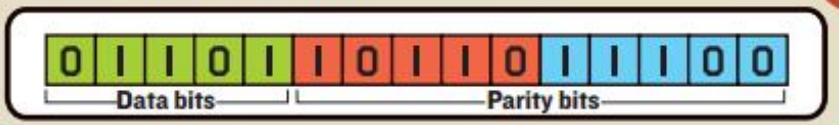
例：3GPP方案

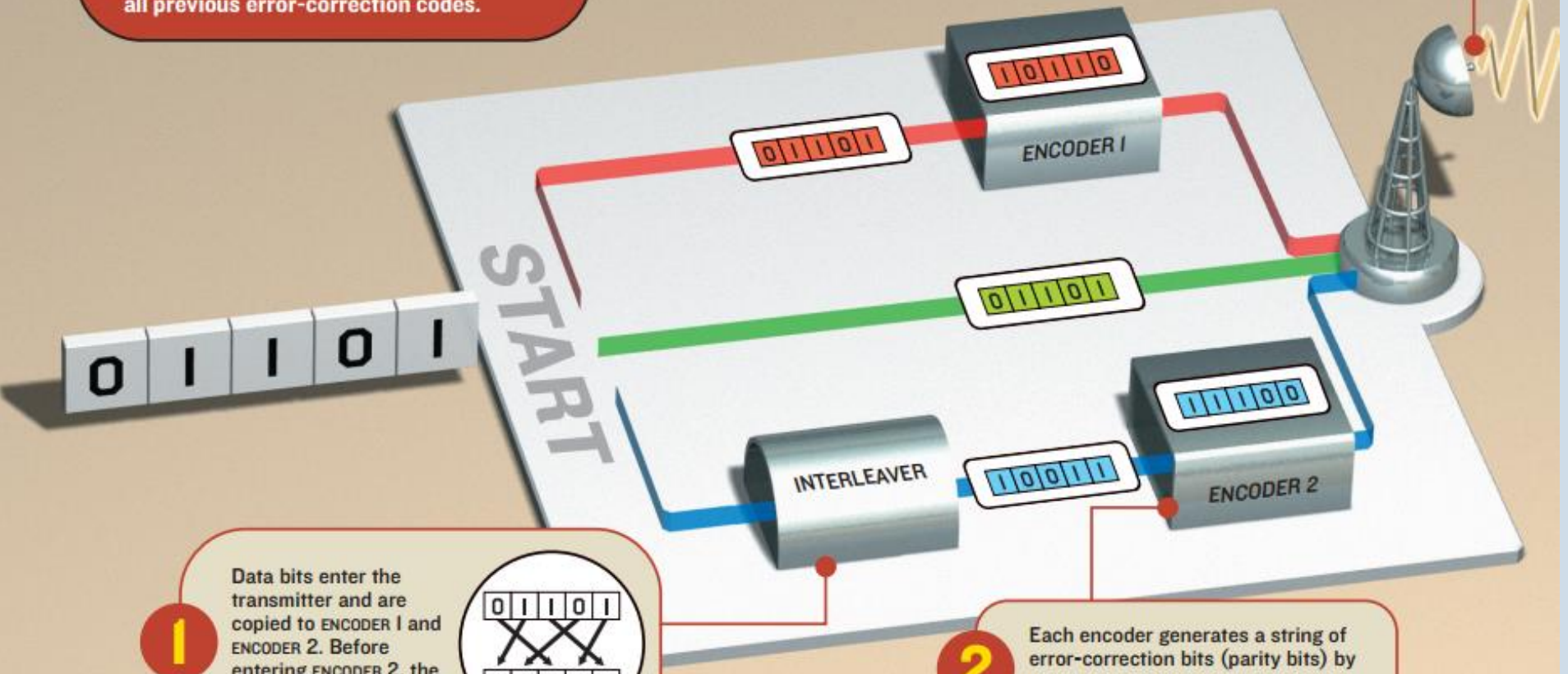


HOW TURBO CODES WORK

Turbo codes use two encoders at the transmitter and two decoders at the receiver. With this divide-and-conquer approach, turbo codes outperform all previous error-correction codes.

3 The original data bits plus the two strings of parity bits are combined into a single block and then sent over the channel, where noise can cause errors in the transmission.





1 Data bits enter the transmitter and are copied to ENCODER 1 and ENCODER 2. Before entering ENCODER 2, the data bits are scrambled by the INTERLEAVER.

2 Each encoder generates a string of error-correction bits (parity bits) by perfoming a series of calculations on the data bits it receives.

目录

- 编码器结构
- 解码原理
- 软输出译码算法
- 性能分析

解码原理

- 概率对数比
- 信道软输出
- 迭代译码器结构

解码原理—概率对数比

- 假设二元随机变量X的取值为 {+1,-1}，则先验概率对数比 $L_X(x)$ 定义为：

$$L_X(x) = \log \frac{P_X(x=+1)}{P_X(x=-1)}$$

其中 $P_X(x)$ 表示随机变量 X 取值为 x 的概率

除非特别说明，对数运算取自然对数

解码原理—概率对数比

- $L_X(x)$作为二元随机变量取值的“软”输出值
  - $L_X(x)$的**符号**作为二元随机变量X的输出
  - $|L_X(x)|$的**取值**作为二元随机变量X输出的可靠度

解码原理—概率对数比

- 由先验概率对数比$L_X(x)$可以得到$P_X(x)$：

$$P(x=+1)=\frac{e^{L(x)}}{1+e^{L(x)}}$$

$$P(x=-1)=\frac{1}{1+e^{L(x)}}$$

解码原理—概率对数比

- 如果二进制随机变量X以另一个随机变量Y为条件，定义条件概率对数比$L_{X|Y}(x|y)$为：

$$L_{X|Y}(x|y)=\log\frac{P_{X/Y}(x=+1|y)}{P_{X/Y}(x=-1|y)}$$

$$=\log\frac{P_X(x=+1)}{P_X(x=-1)}+\log\frac{P_{Y|X}(y|x=+1)}{P_{Y|X}(y|x=-1)}$$

$$=L_X(x)+L_{Y|X}(y|x)$$

解码原理

- 概率对数比
- 信道软输出
- 迭代译码器结构

解码原理—信道软输出

- 考虑*AWGN*信道，概率密度函数为：

$$p(y/x)=\frac{1}{\sqrt{2\pi}\sigma}\exp\left\{-\frac{1}{2\sigma^2}(y-x)^2\right\}$$

$$=\frac{1}{\sqrt{2\pi}\sigma}\exp\left\{-\frac{E_s}{N_0}(y-x)^2\right\}$$

这里 x 是信道传输的符号值

σ^2是对信号能量归一化条件下噪声的方差，即

$$1/2\sigma^2=E_s/N_0$$

解码原理—信道软输出

- 此时 $AWGN$ 信道情况下的条件概率对数比为

$$L_{X/Y}(x \mid y) = \log \frac{P(y \mid x=+1)}{P(y \mid x=-1)} + \log \frac{P(x=+1)}{P(x=-1)}$$

$$= \log \frac{\exp\left[-\frac{E_s}{N_0}(y-A)^2\right]}{\exp\left[-\frac{E_s}{N_0}(y+A)^2\right]} + \log \frac{P(x=+1)}{P(x=-1)} = L_c \cdot y + L(x)$$

其中 $L_c = 4A \cdot E_s / N_0$ ，表示信道传输的**可靠度**

A表示信号的幅度值，对于恒定信道，设A=1

解码原理—信道软输出

- 对于一个离散BSC，L_c是交叉概率的对数比：

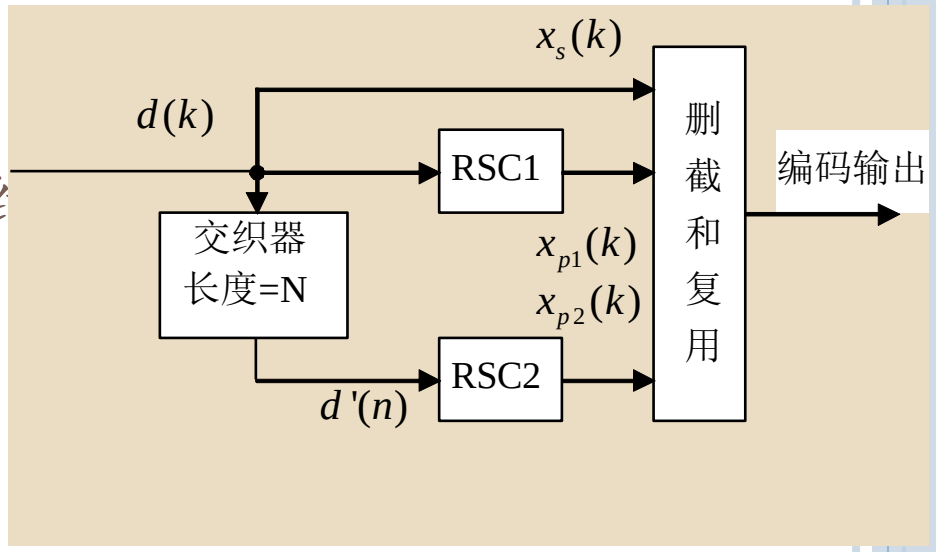
$$L_c = \log \frac{1-p}{p}$$

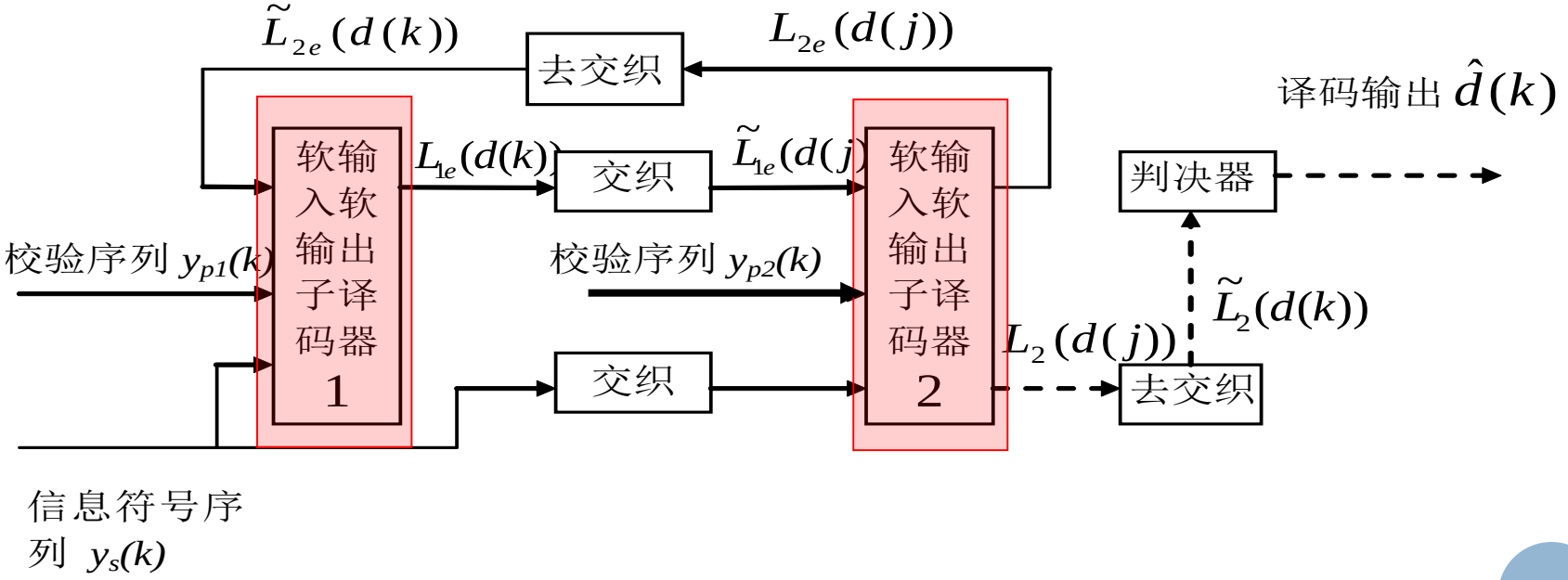
解码原理

- 概率对数比
- 信道软输出
- 迭代译码器结构

解码原理—迭代译码器结

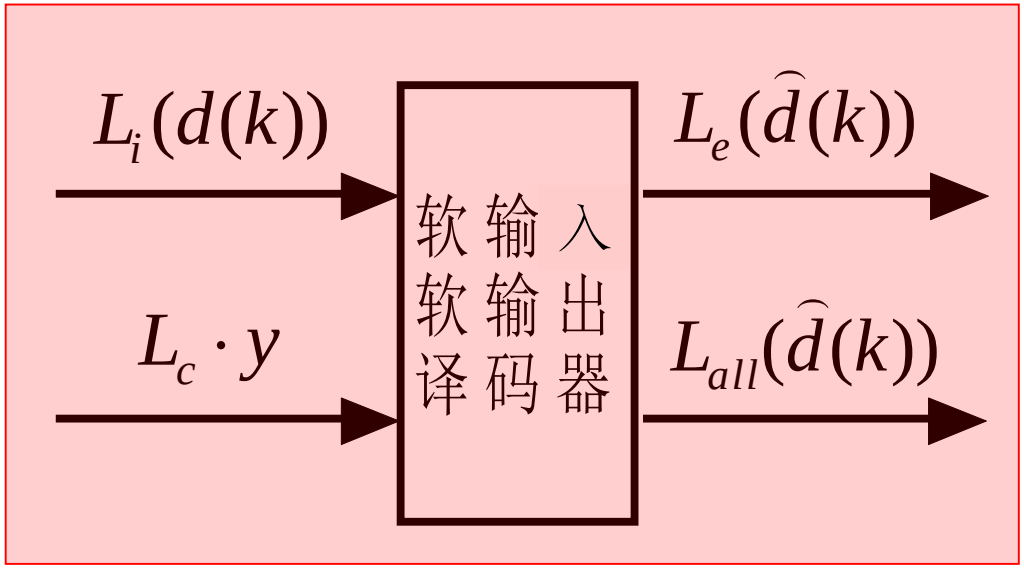
- 译码器结构



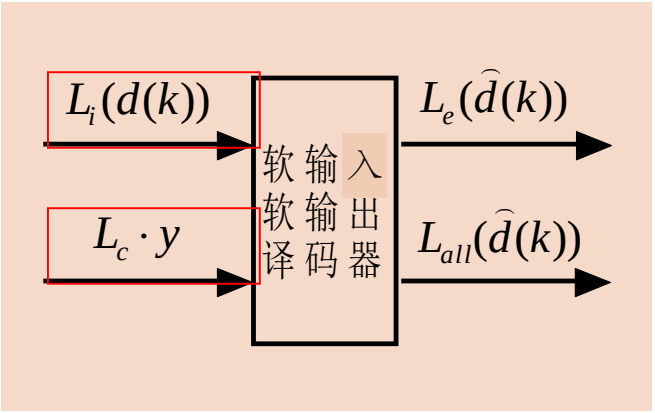


解码原理—迭代译码器结构

- 核心是一个软输入/软输出的译码器，它可获得每个译码输出(信息)比特$d(k)$的后验概率的对数比

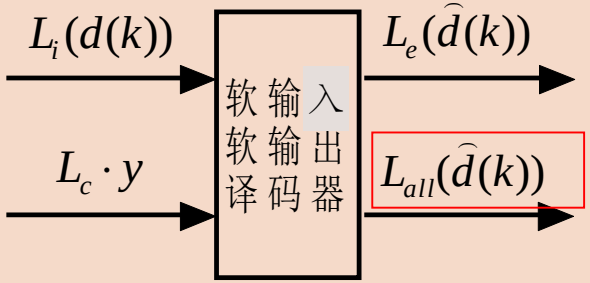


解码原理—迭代译码器结构



- SISO译码器输入
  - $L_i(d(k))$ 是信息比特$d(k)$的先验信息（先验概率的对数比）
  - y是编码比特经信道传输受噪声干扰后的接收信号
  - 由于采用的是系统编码，y可以分为三部分：
    - $y_s(k)$ 是信息比特对应的信号
    - $y_{p1}(k)$ 是第一个子RSC编码器产生校验比特对应的信号
    - $y_{p2}(k)$ 是第二个子RSC编码器产生校验比特对应的信号

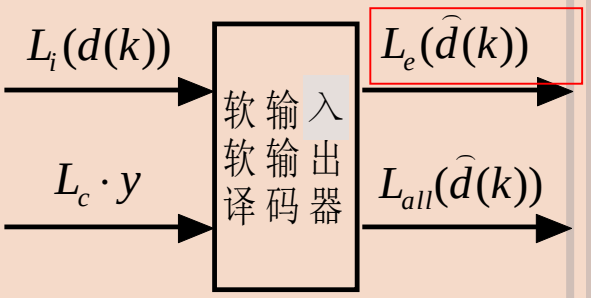
解码原理—迭代译码器结构



- SISO译码器输出
  - $L_{all}(\hat{d}(k))$ 是信息比特 $d(k)$ 的后验概率的对数比
  - 它是通过软输出译码算法得到的，定义为：

$$L_{all}(\hat{d}(k)) = L(d(k)|\text{观察区间}) = \log \frac{P(d(k)=+1|\text{观察区间})}{P(d(k)=-1|\text{观察区间})}$$

解码原理—迭代译码器结构



- SISO译码器输出
  - $L_e(\hat{d}(k))$ 是信息比特$d(k)$ 的外信息(extrinsic information)
  - 它是从输入的 $L_i(d(k))$ 和 $L_c \cdot y$ 中经过软输出译码获得的关于信息比特 $d(k)$ 的软值
  - $L_e(\hat{d}(k))$将作为下一级迭代译码中信息比特$d(k)$ 的先验信息

解码原理—迭代译码器结构

- 系统码的软输出
  - 对系统码，信息比特 $d(k)$ 的软输出可表示成三项的和：

$$L_{all}(\hat{d}(k)) = L_s\left(y_s(k)\right) + L_i(d(k)) + L_e(\hat{d}(k))$$

信道产生的软值，对高斯信道，它等于 $L_c \cdot y$

先验信息

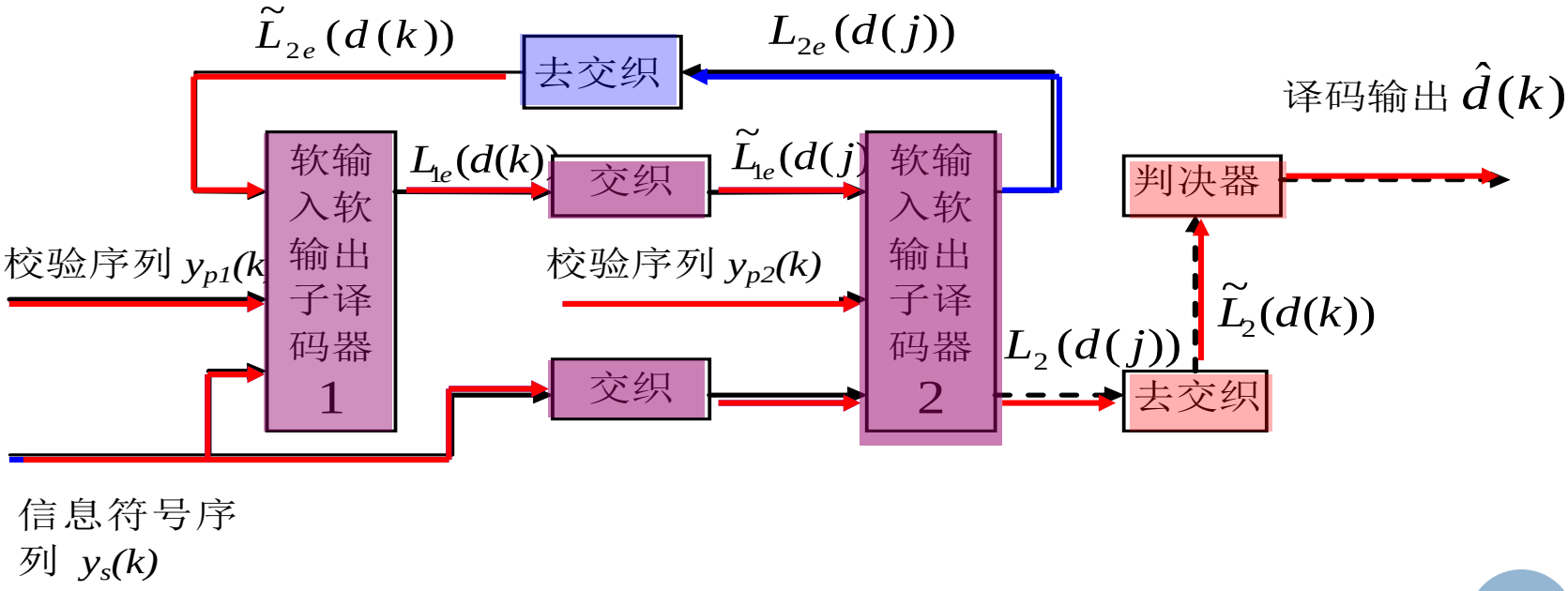
外信息

解码原理—迭代译码器结构

- 迭代译码过程
  - 迭代译码通常都要反反复复经过若干次译码
  - 如果编码结构中有m个子编码器，每次译码又可以分为m级，第i级译码针对第i个子编码器(i=1、2、…、m)
  - 这里假设有q次迭代译码，每次迭代译码又分为2级，则迭代译码器结构及工作过程如下：

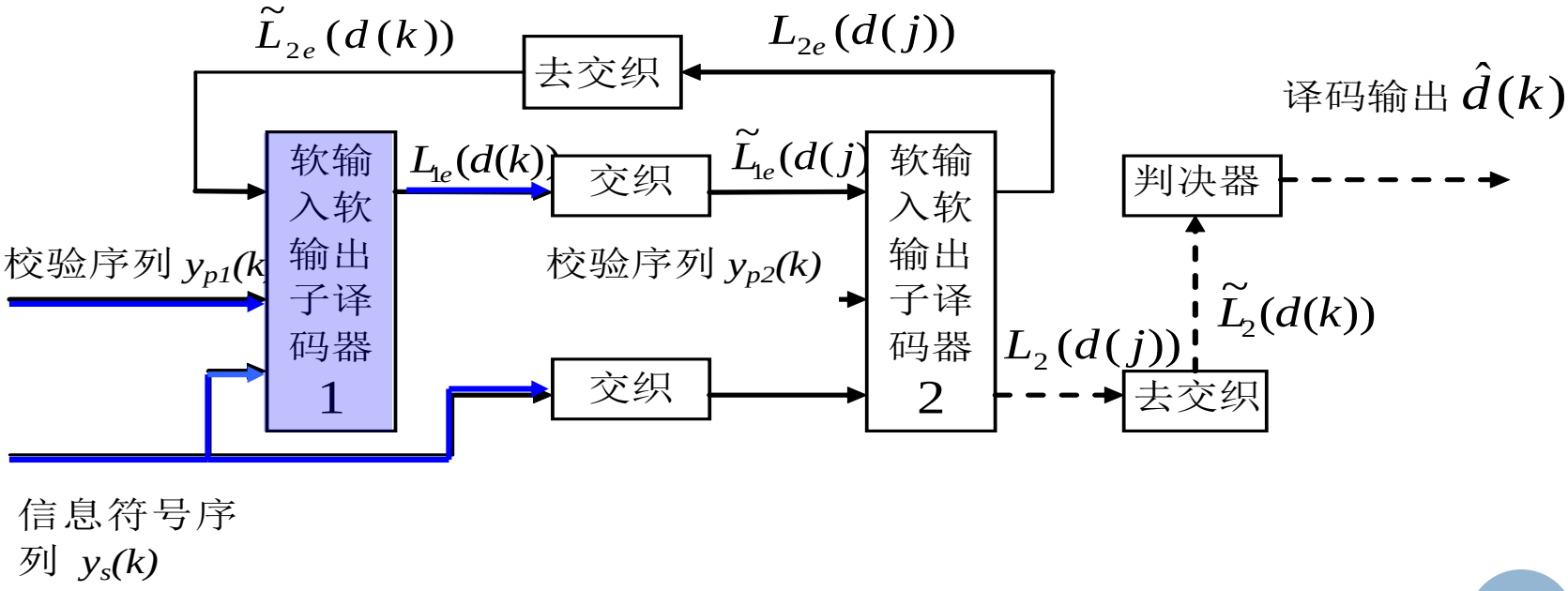
解码原理—迭代译码器结构

- 迭代译码过程



解码原理—迭代译码器结构

- 迭代译码过程—第一次迭代的第一级译码



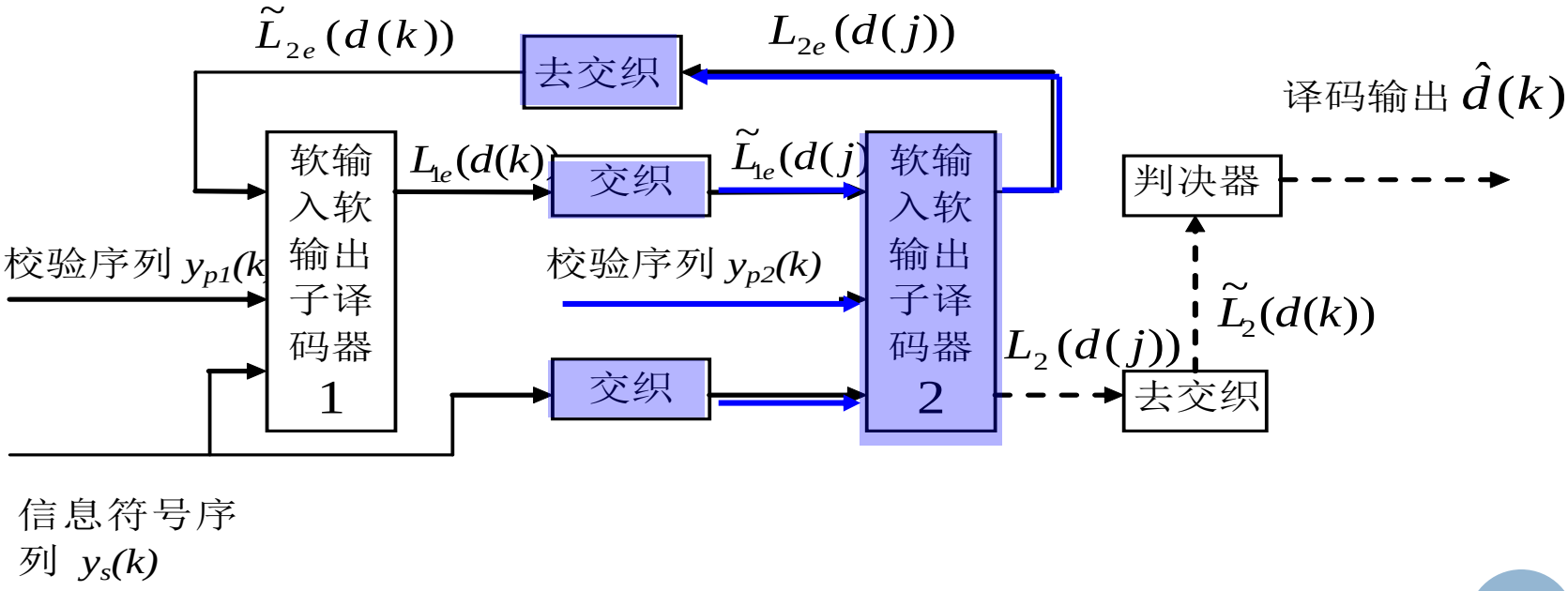
解码原理—迭代译码器结构

- 迭代译码过程—第一次迭代的第一级译码
  - 通常假定信息发送是等概的，因此在第一次迭代的第一级译码时，没有先验信息可用，即$L_i^{(1)}(d(k))=0$
  - 第一级译码器通过 $y_{s1}(k)$ 和 $y_{p1}(k)$ 两个输入获得信息比特的软输出 $L_{1,all}^{(1)}(\hat{d}(k))$
  - 关于信息比特$d(k)$ 的外信息为：

$$L_{1,e}^{(1)}(\hat{d}(k))=L_{1,all}^{(1)}(\hat{d}(k))-L_s\left(y_{s1}(k)\right)$$

解码原理—迭代译码器结构

- 迭代译码过程—第一次迭代的第二级译码



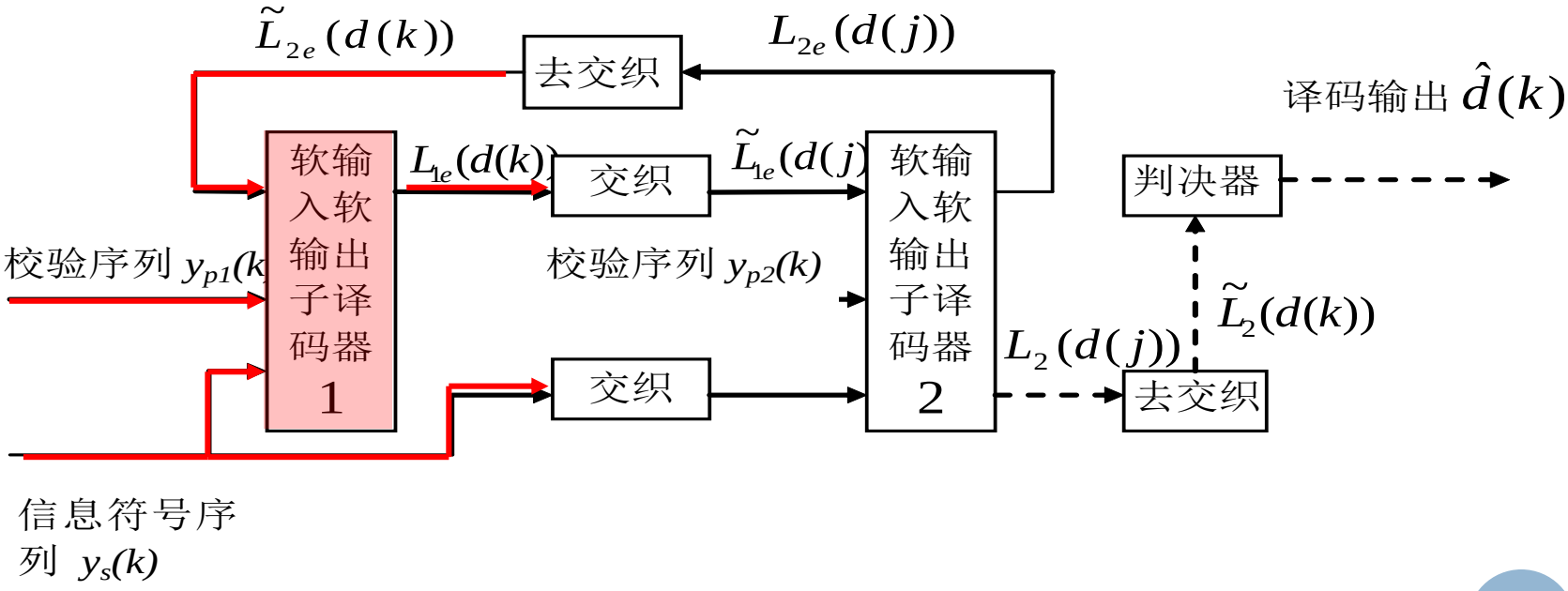
解码原理—迭代译码器结构

- 迭代译码过程—第一次迭代的第二级译码
  - 在第二级译码时，将第一级输出的外信息作为信息输入的先验软值，译码器通过三个输入 $y_{s2}(k)$，$y_{p2}(k)$，$L_{1,e}^{(1)}(\hat{d}(k))$ 获得信息比特 $d(k)$ 的软输出 $L_{2,all}^{(1)}(\hat{d}(k))$
  - 同样可获得信息比特 $d(k)$ 的外信息：

$$L_{2,e}^{(1)}(\hat{d}(k)) = L_{2,all}^{(1)}(\hat{d}(k)) - L_s(y_{s2}(k)) - L_{1,e}^{(1)}(\hat{d}(k))$$

解码原理—迭代译码器结构

- 迭代译码过程—第p次迭代的第一级译码



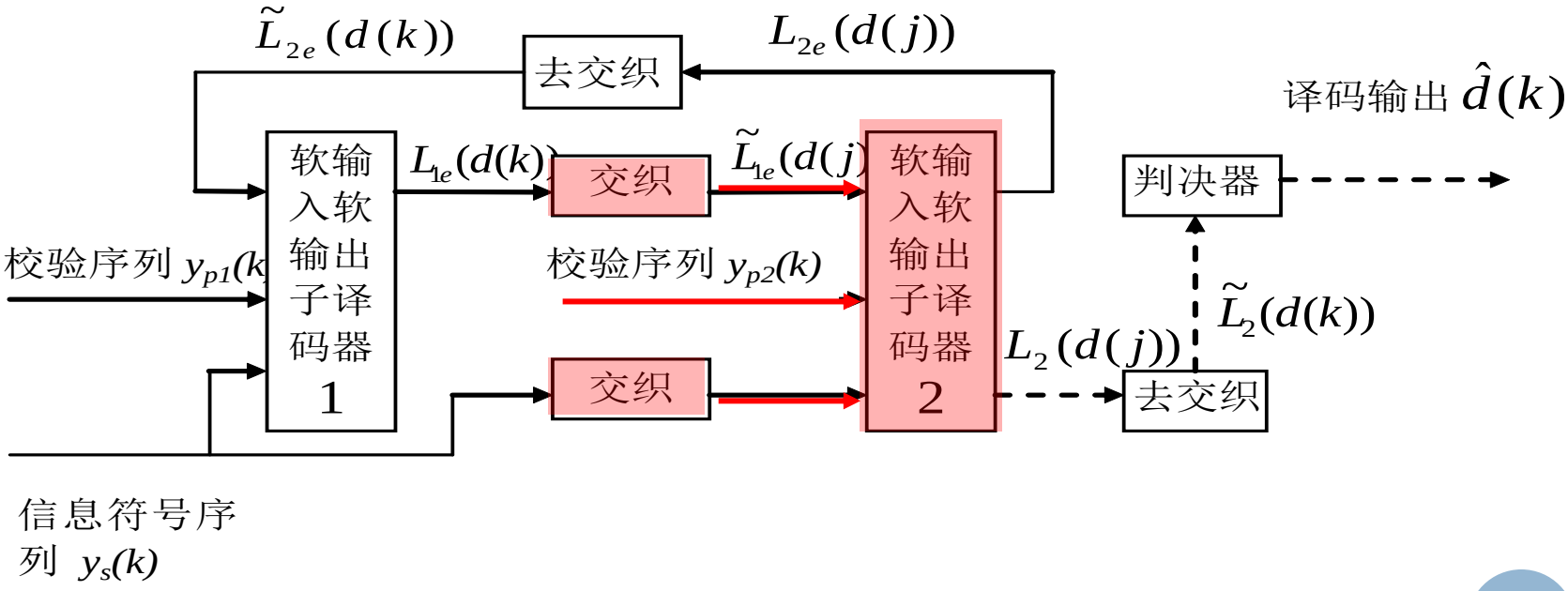
解码原理—迭代译码器结构

- 迭代译码过程—第p次迭代的第一级译码
  - 在第p次迭代的第一级译码时，将p-1次迭代的第二级输出的外信息作为信息输入的先验软值，译码器通过三个输入 $y_{s1}(k)$, $y_{p1}(k)$, $L_{2,e}^{(p-1)}(\hat{d}(k))$ 获得信息比特$d(k)$ 的软输出 $L_{1,all}^{(p)}(\hat{d}(k))$
  - 关于信息比特 $d(k)$ 的外信息为：

$$L_{1,e}^{(p)}(\hat{d}(k)) = L_{1,all}^{(p)}(\hat{d}(k)) - L_s\left(y_{s1}(k)\right) - L_{2,e}^{(p-1)}(\hat{d}(k))$$

解码原理—迭代译码器结构

- 迭代译码过程—第p次迭代的第二级译码



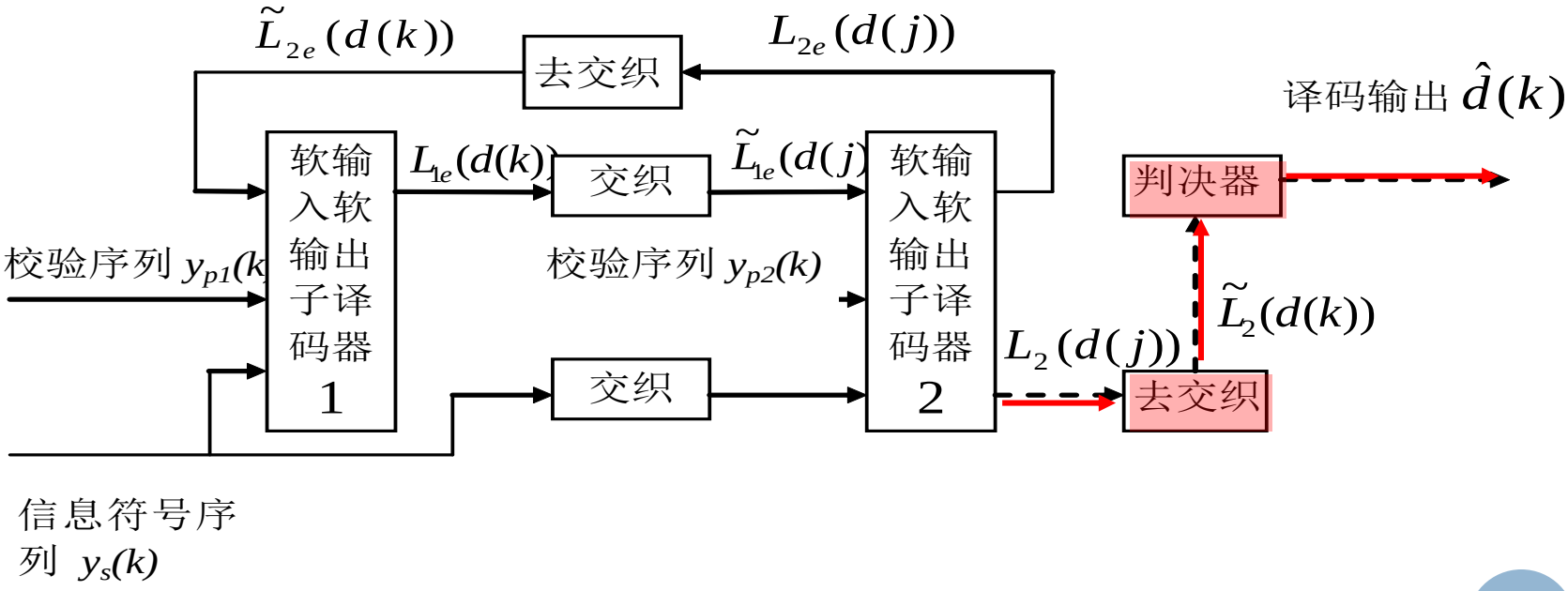
解码原理—迭代译码器结构

- 迭代译码过程—第p次迭代的第二级译码
  - 在第二级译码时，将第一级输出的外信息作为信息输入的先验软值，译码器通过三个输入 $y_{s2}(k)$，$y_{p2}(k)$，$L_{1,e}^{(p)}(\hat{d}(k))$ 获得信息比特 $d(k)$ 的软输出 $L_{2,all}^{(p)}(\hat{d}(k))$
  - 同样可获得信息比特$d(k)$ 的外信息：

$$L_{2,e}^{(p)}(\hat{d}(k)) = L_{2,all}^{(p)}(\hat{d}(k)) - L_s(y_{s2}(k)) - L_{1,e}^{(p)}(\hat{d}(k))$$

解码原理—迭代译码器结构

- 迭代译码过程



解码原理—迭代译码器结构

- 迭代译码过程
  - 在首次迭代中，第一级译码和第二级译码输出的外信息是相互统计独立的，所以从第一次迭代到第二次迭代所获得的增益较大
  - 但由于译码时采用相同的信息，在后面迭代中输出的外信息相关性越来越大，因而通过迭代改善的性能越来越小
  - 最后一次迭代后，将软输出 $L_{all}(\hat{d}(k))$ 用来进行硬判决：

$$\hat{d}(k)=\begin{cases}+1 & \text{如果} L_{all}(\hat{d}(k))\geq 0\\ -1 & \text{如果} L_{all}(\hat{d}(k))<0\end{cases}$$

TRANSMITTER

The Comms. Research Group deep in thought

CHANNEL

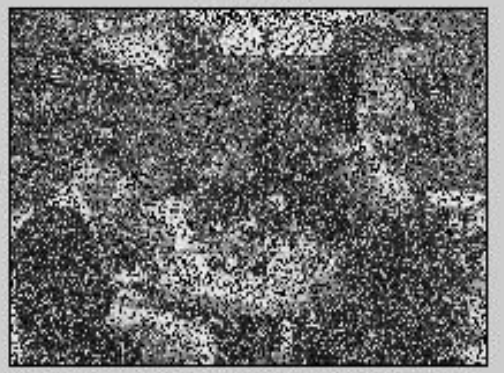


Image Corrupted by Channel Noise (1.0 dB)

RECEIVER

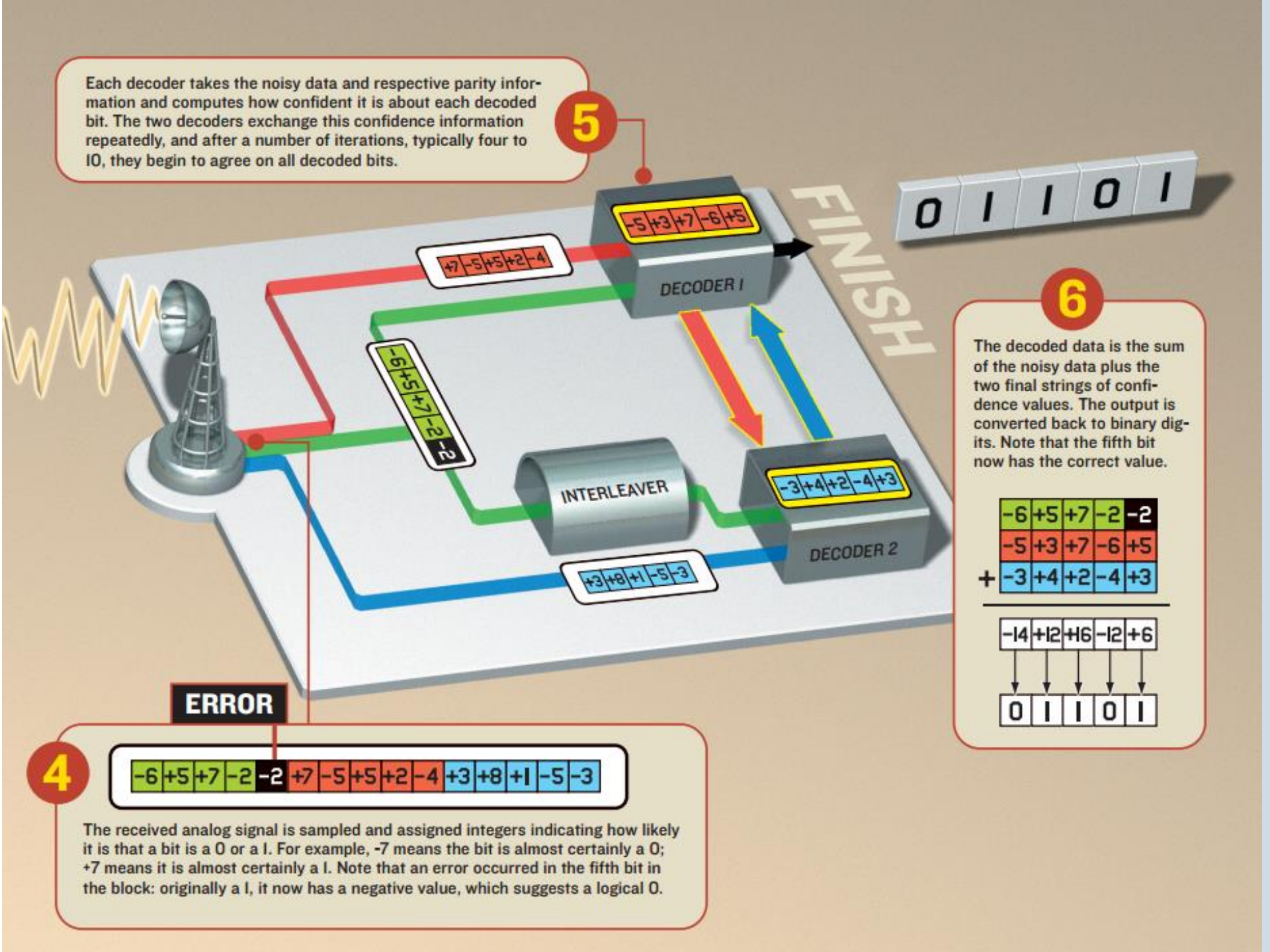
After 1 Turbo Decoding Step

After 2 Iterations

After 10 Iterations

Turbo Decoding of a Bit-Map Image

Turbo Decoding is a powerful error correcting tool, that can be used to protect digital signals transmitted over radio channels. The term 'Turbo' refers to the iterative (repeated) processing of the received signal. Turbo codes tend to perform better than any other code at low signal-to-noise ratios (noisy signals). Here we can see a simulation of a digitised image being transmitted over a noisy radio channel.


Each decoder takes the noisy data and respective parity information and computes how confident it is about each decoded bit. The two decoders exchange this confidence information repeatedly, and after a number of iterations, typically four to 10, they begin to agree on all decoded bits.
5
-5 +3 +7 -6 +5
+7 -5 +5 +2 -4
DECODER 1
FINISH
0 1 1 0 1
6
The decoded data is the sum of the noisy data plus the two final strings of confidence values. The output is converted back to binary digits. Note that the fifth bit now has the correct value.
-6 +5 +7 -2 -2
-5 +3 +7 -6 +5
+ -3 +4 +2 -4 +3
-14 +12 +16 -12 +6
0 1 1 0 1
-6 +5 +7 -2 -2
INTERLEAVER
-3 +4 +2 -4 +3
DECODER 2
+3 +8 +1 -5 -3
ERROR
4
-6 +5 +7 -2 -2 +7 -5 +5 +2 -4 +3 +8 +1 -5 -3
The received analog signal is sampled and assigned integers indicating how likely it is that a bit is a 0 or a 1. For example, -7 means the bit is almost certainly a 0; +7 means it is almost certainly a 1. Note that an error occurred in the fifth bit in the block: originally a 1, it now has a negative value, which suggests a logical 0.

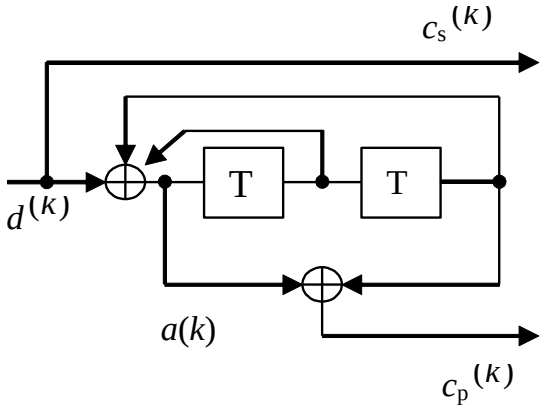
目录

- 编码器结构
- 解码原理
- 软输出译码算法
- 性能分析

软输出译码算法

- 软输出译码算法目前可以被划分为两大类：
  - 软输出维特比译码算法（SOVA）
  - 逐符号的最大后验概率（MAP）算法以及其简化算法，例如：
    - Log-MAP算法
    - Max-Log-MAP算法
  - MAP算法复杂度较高，但在低信噪比条件下性能优越

软输出译码算法—MAP



- MAP算法概述
  - 设RSC码的编码约束长度为K,
  - 编码器的状态是S_k,
  - 与第k个输入$a(k)$ 对应的编码器更新状态为K-1维向量:

$$S_k = \left(a(k), a(k-1), \cdots a(k-K+2)\right)$$

  - 这里假定:
    - 信息比特序列$\{d(k)\}$ 是由N个统计独立的比特 $d(k)$ 组成;
    - $d(k)$ 取0和取1的概率相等;
    - 编码器的初始状态S_0和终止状态S_N都为0，即:

$$S_0 = S_N = (0,0,\cdots 0) = \vec{0}$$

软输出译码算法－MAP

- MAP算法概述

设：

- 编码器输出码字序列

$$C_1^N = \{C(1)\cdots C(k)\cdots C(N)\},\quad C(i)\in\{0,1\}$$

- 经过调制器后的信道输入序列

$$X_1^N = \{X(1)\cdots X(k)\cdots X(N)\},\quad X(i)\in\{+1,-1\}$$

- 经过（时间离散）无记忆AWGN信道后的输出序列

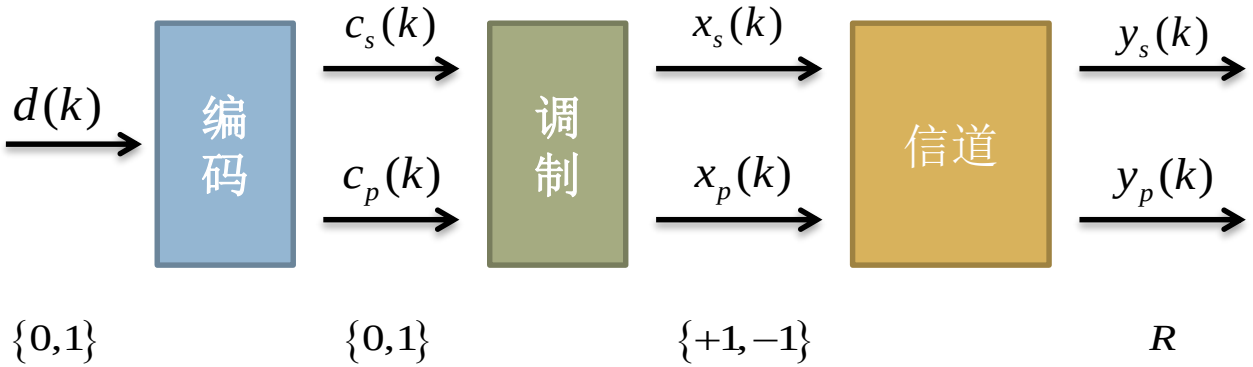
$$Y_1^N = \{Y(1)\cdots Y(k)\cdots Y(N)\},\quad Y(i)\in R$$

这里：$C(k)=\left(c_s(k),c_p(k)\right)$　　$X(k)=\left(x_s(k),x_p(k)\right)$

$$Y(k)=\left(y_s(k),y_p(k)\right)$$

软输出译码算法－MAP

- MAP算法概述



软输出译码算法－MAP

- MAP算法概述

对（时间离散）无记忆AWGN信道，在k时刻满足关系式：

0码映射为+1
1码映射为-1

$$x_s(k)=(1-2c_s(k)) \qquad y_s(k)=x_s(k)+n_s(k)$$

$$x_p(k)=(1-2c_p(k)) \qquad y_p(k)=x_p(k)+n_p(k)$$

这里 $n_s(k)$ 和 $n_p(k)$是均值都为0、方差都为σ^2 的两个独立的信道高斯噪声

软输出译码算法—MAP

- MAP算法概述

MAP准则的目标是获得信息比特 $d(k)$ 的后验概率对数比（译码器的软输出）：

Y_1^N表示从时刻1到时刻N的整个接收符号序列

$$L(\hat{d}(k)) = \log \frac{P(d(k)=0 \mid Y_1^N)}{P(d(k)=1 \mid Y_1^N)} = \log \frac{P(d(k)=0, Y_1^N)}{P(d(k)=1, Y_1^N)}$$

$$= \log \frac{\sum_{m',m} p(d(k)=0, S_{k-1}=m', S_k=m, Y_1^N)}{\sum_{m',m} p(d(k)=1, S_{k-1}=m', S_k=m, Y_1^N)}$$

软输出译码算法－MAP

- 逐个信息比特的后验概率对数比的计算

$$L(\hat{d}(k)) = \log \frac{\sum_{m',m} p(d(k)=0, S_{k-1}=m', S_k=m, Y_1^N)}{\sum_{m',m} p(d(k)=1, S_{k-1}=m', S_k=m, Y_1^N)}$$

状态转换标记对(m',m) 确定了输入信息比特$d(k)$，编码输出比特 $C(k)$, 信道输入比特 $X(k)$.

对数比计算公式的分子或分母上联合概率$p(d(k)=i, S_k=m', S_k=m, Y_1^N)$之和，分别表示了信息比特$d(k)=1$ 或 $d(k)=0$ 从m’状态到m状态的所有转化可能.

$$L(\hat{d}(k)) = \log \frac{\sum_{m',m} p(d(k)=0, S_{k-1}=m', S_k=m, Y_1^N)}{\sum_{m',m} p(d(k)=1, S_{k-1}=m', S_k=m, Y_1^N)}$$

MAP

- 逐个信息比特的后验概率对数比的计算

$$p(d(k)=i, S_{k-1}=m', S_k=m, Y_1^N)$$

$$= p\left(d(k)=i, S_{k-1}=m', S_k=m, Y_1^{k-1}, Y(k), Y_{k+1}^N\right)$$

$$= p\left(S_{k-1}=m', Y_1^{k-1}\right) \cdot p\left(d(k)=i / S_{k-1}=m', Y_1^{k-1}\right)$$

$$\cdot p\left(S_k=m / d(k)=i, S_{k-1}=m', Y_1^{k-1}\right)$$

$$\cdot p\left(Y(k) / d(k)=i, S_{k-1}=m', S_k=m, Y_1^{k-1}\right)$$

$$\cdot p\left(Y_{k+1}^N / d(k)=i, S_{k-1}=m', S_k=m, Y_1^{k-1}, Y(k)\right)$$

$$= \underbrace{p\left(S_{k-1}=m', Y_1^{k-1}\right)}_{\cdot} \cdot$$

$$\underbrace{\cdot p\left(d(k)=i\right) \cdot p\left(S_k=m / d(k)=i, S_{k-1}=m'\right) \cdot p\left(Y(k) / d(k)=i, S_{k-1}=m', S_k=m\right)}_{\cdot}$$

$$\underbrace{\cdot p\left(Y_{k+1}^N / S_k=m\right)}_{\cdot}$$

Y_1^{k-1} 表示从初始状态到时刻k-1的接收符号序列

$Y(k)$ 表示k时刻的接收符号

Y_{k+1}^N 表示从时刻k+1到结束状态的接收符号序列

对于一个无记忆传输信道，联合概率

$p(d(k)=i, S_{k-1}=m', S_k=m, Y_1^N)$

可以表示成三组独立概率的乘积形式

$$L(\hat{d}(k)) = \log \frac{\sum_{m',m} p(d(k)=0, S_{k-1}=m', S_k=m, Y_1^N)}{\sum_{m',m} p(d(k)=1, S_{k-1}=m', S_k=m, Y_1^N)}$$

MAP

- 逐个信息比特的后验概率对数比的计算
  - 定义

过去 $\alpha_{k-1}(m') = p(S_{k-1}=m', Y_1^{k-1})$

未来 $\beta_k(m) = p(Y_{k+1}^N \mid S_k = m)$

现在 $\gamma_{i,k}(m',m) = p\left(d(k)=i\right) \cdot p\left(S_k = m / d(k)=i, S_{k-1}=m'\right)$

$\cdot p(Y(k) \mid d(k)=i, S_{k-1}=m', S_k=m)$

则有

$$p(d(k)=i, S_k=m', S_k=m, Y_1^N)$$

$$= \alpha_{k-1}(m') \cdot \gamma_{i,k}(m',m) \cdot \beta_k(m)$$

MAP

- 逐个信息比特的后验概率对数比的计算
  - 因此后验概率对数比可以表示为

$$L(\hat{d}(k)) = \log \frac{\sum_{m',m} \alpha_{k-1}(m')\gamma_{0,k}(m',m)\beta_k(m)}{\sum_{m',m} \alpha_{k-1}(m')\gamma_{1,k}(m',m)\beta_k(m)}$$

MAP

- 后验概率对数比的递归计算
  - 为了完成MAP准则，信息帧必须是有限长度
  - 假设初始状态和所有序列路径的结束状态最后都归结到零状态
  - 所以前向递归和反向递归可以初始化为:

$$\alpha_{start}(0)=1 \qquad \beta_{end}(0)=1$$

$$L(\hat{d}(k)) = \log \frac{\sum_{m',m} \alpha_{k-1}(m')\gamma_{0,k}(m',m)\beta_k(m)}{\sum_{m',m} \alpha_{k-1}(m')\gamma_{1,k}(m',m)\beta_k(m)}$$

MAP

- 后验概率对数比的递归计算
  - MAP准则中的前向递归：

$$\begin{aligned}
&\alpha_k(m) \\
&= p(S_k = m, Y_1^k) \\
&= \sum_{m',\, i} p(d(k) = i, S_{k-1} = m', S_k = m, Y_1^{k-1}, Y(k)) \\
&= \sum_{m',i} p(Y_1^{k-1}, S_{k-1} = m')\, p(d(k) = i, Y(k), S_k = m \mid Y_1^{k-1}, S_{k-1} = m') \\
&= \sum_{m',i} \alpha_{k-1}(m') \cdot \gamma_{i,k}(m',m)
\end{aligned}$$

$$L(\hat{d}(k)) = \log \frac{\sum_{m',m} \alpha_{k-1}(m')\gamma_{0,k}(m',m)\beta_k(m)}{\sum_{m',m} \alpha_{k-1}(m')\gamma_{1,k}(m',m)\beta_k(m)}$$

MAP

- 后验概率对数比的递归计算

  - MAP准则中反向递归：

$$\beta_{k-1}(m')$$

$$= p\left(Y_k^N \mid S_{k-1} = m'\right)$$

$$= \sum_{m,i} p\left(d(k)=i, S_k = m, Y_k^N \mid S_{k-1} = m'\right)$$

$$= \sum_{m,i} p\left(d(k)=i, Y(k), S_k = m \mid S_{k-1} = m'\right) p\left(Y_{k+1}^N \mid S_k = m\right)$$

$$= \sum_{m,i} \gamma_{i,k}(m',m) \cdot \beta_k(m)$$

$Y(k), Y_{k+1}^N$

$$L(\hat{d}(k)) = \log \frac{\sum_{m',m} \alpha_{k-1}(m')\gamma_{0,k}(m',m)\beta_k(m)}{\sum_{m',m} \alpha_{k-1}(m')\gamma_{1,k}(m',m)\beta_k(m)}$$

MAP

- 后验概率对数比的递归计算
  - 在状态m’和m间的支路转移概率为：

$$\begin{aligned}&\gamma_{i,k}(m',m)\\ &= p(d(k)=i)\cdot p(S_k=m \mid d(k)=i, S_{k-1}=m')\\ &\cdot p(Y(k) \mid d(k)=i, S_{k-1}=m', S_k=m)\end{aligned}$$

MAP

$$\gamma_{i,k}(m',m) = p(d(k)=i)\cdot p(S_k=m \mid d(k)=i, S_{k-1}=m') \cdot p(Y(k) \mid d(k)=i, S_{k-1}=m', S_k=m)$$

- 后验概率对数比的递归计算
  - 前面讲过先验概率对数比 $L(d(k))$ 和概率 $P(d(k))$ 之间的关系

$$p(d(k)=0)=\frac{e^{L(d(k))}}{1+e^{L(d(k))}} \qquad p(d(k)=1)=\frac{1}{1+e^{L(d(k))}}$$

将其合并后得到上式中的第一项

$$p(d(k)=i)=\frac{e^{(1-i)\cdot L(d(k))}}{1+e^{L(d(k))}}$$

MAP

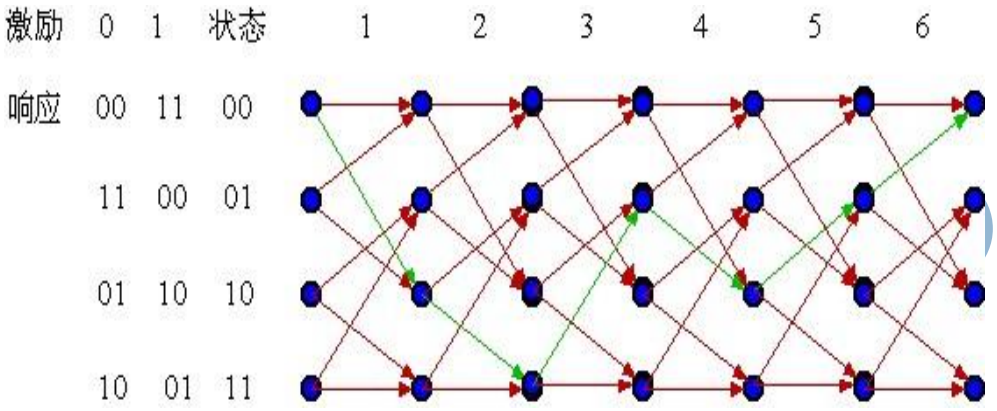
$$\gamma_{i,k}(m',m) = p(d(k)=i)\, p(S_k=m \mid d(k)=i, S_{k-1}=m') \cdot p(Y(k) \mid d(k)=i, S_{k-1}=m', S_k=m)$$

- 后验概率对数比的递归计算
  - 上式中的第二项，即概率

    $p(S_k=m \mid d(k)=i, S_{k-1}=m')$　非1即0
  - 如果在状态(m',m) 之间有 $d(k)=i$ 的路径，

    $$p(S_k=m \mid d(k)=i, S_{k-1}=m')=1$$
  - 如果在状态(m',m) 之间没有 $d(k)=i$ 的路径，

    $$p(S_k=m \mid d(k)=i, S_{k-1}=m')=0$$



MAP

$$\gamma_{i,k}(m',m)$$
$$= p(d(k)=i)\cdot p(S_k=m \mid d(k)=i, S_{k-1}=m')$$
$$\cdot p(Y(k) \mid d(k)=i, S_{k-1}=m', S_k=m)$$

- 后验概率对数比的递归计算
  - 上式中的第三项

$$p(Y(k) \mid d(k)=i, S_{k-1}=m', S_k=m)$$
$$= p(y_s(k) \mid x_s(k))\, p(y_p(k) \mid x_p(k))$$
$$= \exp\left(-\frac{(y_s(k)-x_s(k))^2}{2\sigma^2}\right)\exp\left(-\frac{(y_p(k)-x_p(k))^2}{2\sigma^2}\right)$$
$$= \exp\left(-\frac{y_s^2(k)+x_s^2(k)+y_p^2(k)+x_p^2(k)}{2\sigma^2}\right)\exp\left(\frac{x_s(k)y_s(k)+x_p(k)y_p(k)}{\sigma^2}\right)$$
$$= A(k)\cdot\exp\left[\frac{1}{2}L_c x_s(k)\cdot y_s(k)\right]\cdot\exp\left[\frac{1}{2}L_c x_p(k)\cdot y_p(k)\right]$$

其中从k-1到k的传输中各状态的$A(k)$值相同，在计算对数比时可以消掉！

$$L(\hat{d}(k)) = \log \frac{\sum_{m',m} \alpha_{k-1}(m')\gamma_{0,k}(m',m)\beta_k(m)}{\sum_{m',m} \alpha_{k-1}(m')\gamma_{1,k}(m',m)\beta_k(m)}$$

MAP

- 后验概率对数比的递归计算
  - 如果在状态 (m',m) 之间没有 $d(k)=i$ 的路径，

    $$\gamma_{i,k}(m',m)=0$$

  - 如果在状态 (m',m) 之间有 $d(k)=i$ 的路径，$\gamma_{i,k}(m',m)$

  可以简化为：

$$\begin{aligned}
&\gamma_{i,k}(m',m) \\
&= p(d(k)=i)\cdot p(S_k=m \mid d(k)=i, S_{k-1}=m') \\
&\cdot p(Y(k) \mid d(k)=i, S_{k-1}=m', S_k=m)
\end{aligned}$$

$$= \frac{\exp\left[(1-i)L(d(k))\right]}{\left[1+\exp(L(d(k))\right]} \cdot \exp\left\{\frac{1}{2}\left[L_c x_s(k)\cdot y_s(k)\right] + \frac{1}{2}\left[L_c x_p(k)\cdot y_p(k)\right]\right\}$$

$$L(\hat{d}(k)) = \log \frac{\sum_{m',m} \alpha_{k-1}(m')\gamma_{0,k}(m',m)\beta_k(m)}{\sum_{m',m} \alpha_{k-1}(m')\gamma_{1,k}(m',m)\beta_k(m)}$$

MAP

- 后验概率对数比的递归计算
  - 定义 $\gamma_{i,k}^{(e)}(m',m) = \exp\left[\frac{1}{2}L_c x_p(k)\cdot y_p(k)\right],$

    则有

$$\gamma_{i,k}(m',m) = \frac{1}{\left[1+\exp(L(d(k))\right]}\gamma_{i,k}^{(e)}(m',m)\exp\left[(1-i)\cdot L(d(k)) + \frac{1}{2}\left[L_c x_s(k)\cdot y_s(k)\right]\right]$$

$$L(\hat{d}(k)) = \log \frac{\sum_{m',m} \alpha_{k-1}(m')\gamma_{0,k}(m',m)\beta_k(m)}{\sum_{m',m} \alpha_{k-1}(m')\gamma_{1,k}(m',m)\beta_k(m)}$$

MAP

- 后验概率对数比的递归计算

注意到由于这里进行的是系统码编码，

因此　$x_s(k) = 1 - 2d(k) = 1 - 2i$

$\gamma_{i,k}(m',m)$ 可化为

$$\gamma_{i,k}(m',m) = \frac{1}{[1+\exp(L(d(k))]} \gamma_{i,k}^{(e)}(m',m) \exp\left[(1-i)L(d(k)) + \frac{1-2i}{2}\left[L_c \cdot y_s(k)\right]\right]$$

MAP

- 后验概率对数比的递归计算
  - 最后可以得到表达式

$$L(\hat{d}(k))$$

$$= \log \frac{\sum_{m',m} \alpha_{k-1}(m')\gamma_{0,k}(m',m)\beta_k(m)}{\sum_{m',m} \alpha_{k-1}(m')\gamma_{1,k}(m',m)\beta_k(m)}$$

$$= \left[(1-i)L(d(k)) + \frac{1-2i}{2}\left[L_c y_s(k)\right]\right]_{i=0} - \left[(1-i)L(d(k)) + \frac{1-2i}{2}\left[L_c y_s(k)\right]\right]_{i=1}$$

$$+ \log \frac{\sum_{m',m} \gamma_{0,k}^{(e)}(m',m)\alpha_{k-1}(m')\beta_k(m)}{\sum_{m',m} \gamma_{1,k}^{(e)}(m',m)\alpha_{k-1}(m')\beta_k(m)}$$

信道值　　先验信息

$$= L_c y_s(k) + L(d(k)) + \log \frac{\sum_{m',m} \gamma_{0,k}^{(e)}(m',m)\alpha_{k-1}(m')\beta_k(m)}{\sum_{m',m} \gamma_{1,k}^{(e)}(m',m)\alpha_{k-1}(m')\beta_k(m)}$$

外信息

软输出译码算法－MAP

- MAP算法步骤
  - 步骤1：对概率 $\alpha_0(m)$ 和 $\beta_N(m)$ 进行初始化
    - $\alpha_0(0)=1$，　$\alpha_0(m)=0, \forall m \neq 0$
    - 对于 $\beta_N(m)$，
      - 如果两个RSC子编码器编码结束时状态全为零，

        $$\beta_N(0)=1, \beta_N(m)=0, \forall m \neq 0$$

      - 如果两个RSC子编码器编码结束时状态并未归零，

        $$\beta_N(m)=1/M, \forall m$$

      - 其中M为编码器总状态数
      - 此时在计算BER时将不考虑每个分组的最后几个比特

软输出译码算法－MAP

- MAP算法步骤

  - 步骤2

    对每个接收到的$Y(k)$，计算存储概率$\alpha_k(m)$和$\gamma_{i,k}(m',m)$

软输出译码算法－MAP

- MAP算法步骤
  - 步骤3

当序列 Y_1^N 完全接收到后，计算概率 $\beta_k(m)$,同时计算出每个译码输出比特$d(k)$所对应的后验概率对数比

$$L(\hat{d}(k)) = \log \frac{\sum_{m',m} \alpha_{k-1}(m')\gamma_{0,k}(m',m)\beta_k(m)}{\sum_{m',m} \alpha_{k-1}(m')\gamma_{1,k}(m',m)\beta_k(m)}$$

$$L_{1,e}^{(p)}(\hat{d}(k)) = L_{1,all}^{(p)}(\hat{d}(k)) - L_s\left(y_{s1}(k)\right) - L_{2,e}^{(p-1)}(\hat{d}(k))$$

$$L_{2,e}^{(p)}(\hat{d}(k)) = L_{2,all}^{(p)}(\hat{d}(k)) - L_s(y_{s2}(k)) - L_{1,e}^{(p)}(\hat{d}(k))$$

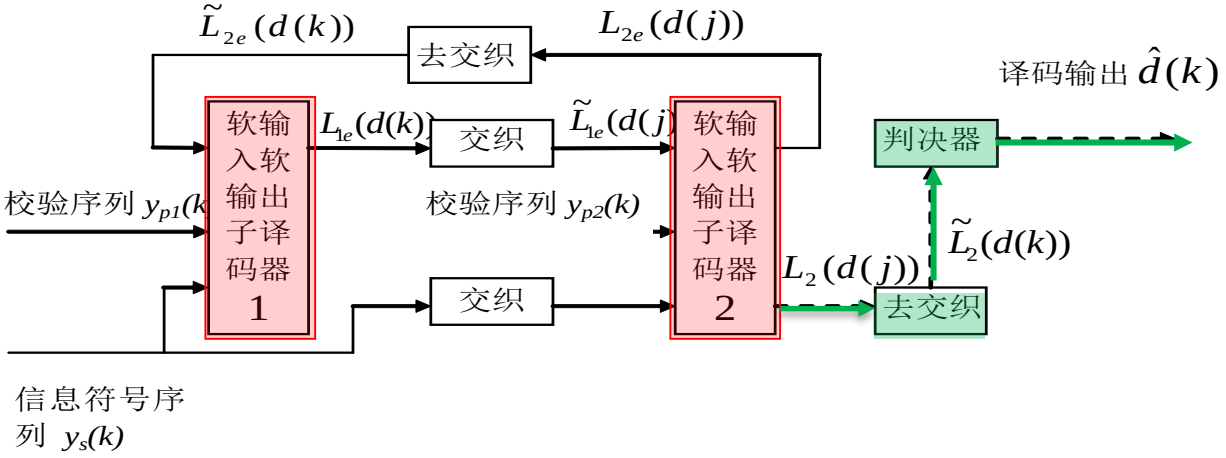
软输出译码算法—MAP

- MAP算法概述

译码器可根据这一软输出与门限0比较即可得到 $d(k)$ 的硬判结果 $\hat{d}(k)$：

$$L(\hat{d}(k)) \geq 0 \rightarrow \hat{d}(k) = 0$$

$$L(\hat{d}(k)) < 0 \rightarrow \hat{d}(k) = 1$$



目录

- 编码器结构
- 解码原理
- 软输出译码算法
- 性能分析

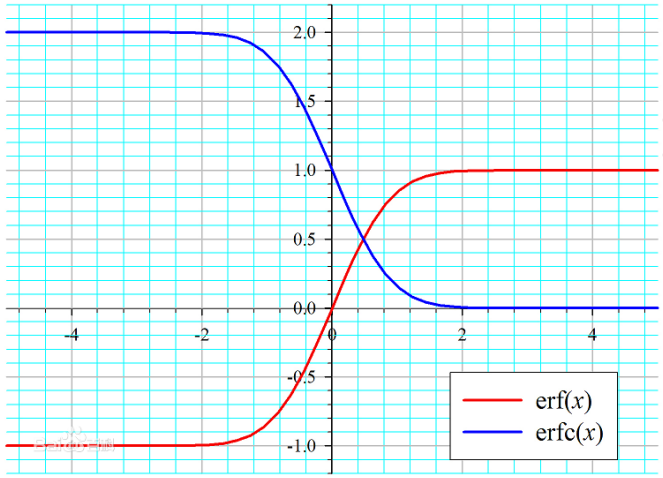
$$\operatorname{erfc}(x) = 1 - \operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_x^{\infty} e^{-\eta^2} d\eta$$

性能分析

- Turbo码的误比特率可近似表示为：

$$P_b \approx \frac{\omega_{d_{\min}} A_{d_{\min}}}{N} erfc\left(\sqrt{\frac{2 d_{\min} R E_b}{N_0}}\right)$$

其中 $\omega_{d_{\min}} = \frac{W_{d_{\min}}}{A_{d_{\min}}}$



$A_{d_{\min}}$ 表示重量为d_{min}的码字数目

$W_{d_{\min}}$ 表示所有重量为d_{min}的码字的信息位重量之和

N表示总信息位数，即交织长度

R表示编码速率

性能分析

$$P_b \approx \frac{\omega_{d_{\min}} A_{d_{\min}}}{N} erfc\left(\sqrt{\frac{2d_{\min} RE_b}{N_0}}\right)$$

- 在低信噪比区间，上式中的系数起主要作用：
  - N越大，性能越好
  - 误比特率反比于交织长度的大小N，即有所谓的交织增益
- Turbo码的设计主要是最大化d_{min}，它决定了中、高信噪比条件下的性能
- 同时要最小化 $\frac{\omega_{d_{\min}} A_{d_{\min}}}{N}$ ，这些参数与分量编码器的选取以及交织器的设计密切相关：
  - 分量编码器采用RSC的主要优点是RSC为无限冲击响应，对于小重量的输入序列，RSC可以输出大重量的码字序列

性能分析

- 在中、高信噪比时，Turbo码的性能受到限制，误比特率曲线下降缓慢，出现所谓的“错误地板”（error floor）效应：

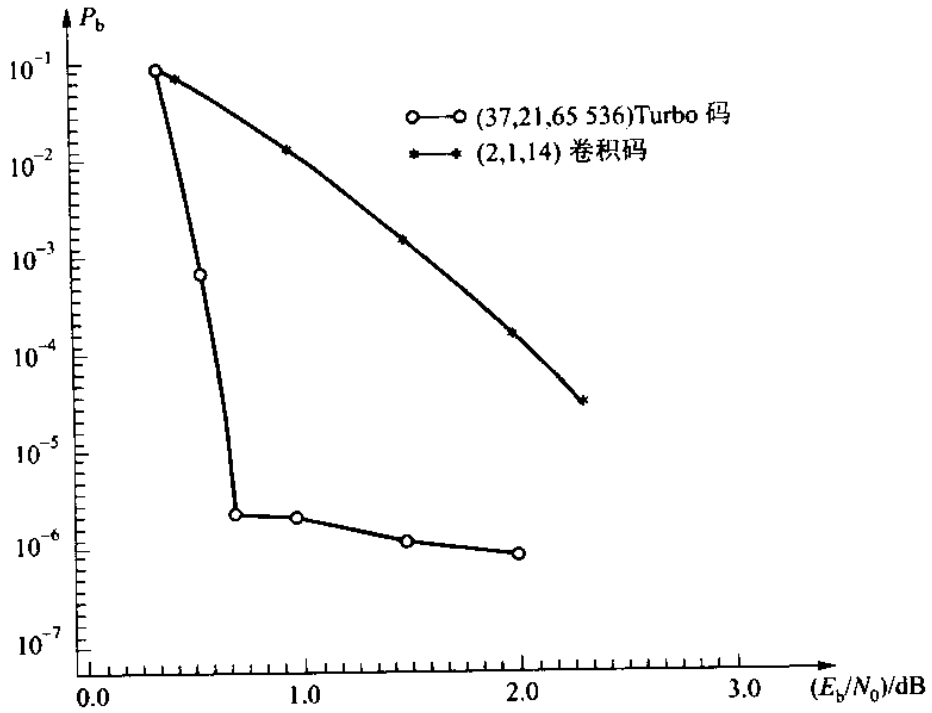


图 8.3.1 (37,21,65 536)Turbo 码与(2,1,14)卷积码性能

Source: Richardson, T.J.; Shokrollahi, M.A.; Urbanke, R.L.; Design of capacity-approaching irregular low-density parity-check codes. Information Theory, IEEE Transactions on, Volume: 47 , Issue: 2， 2001

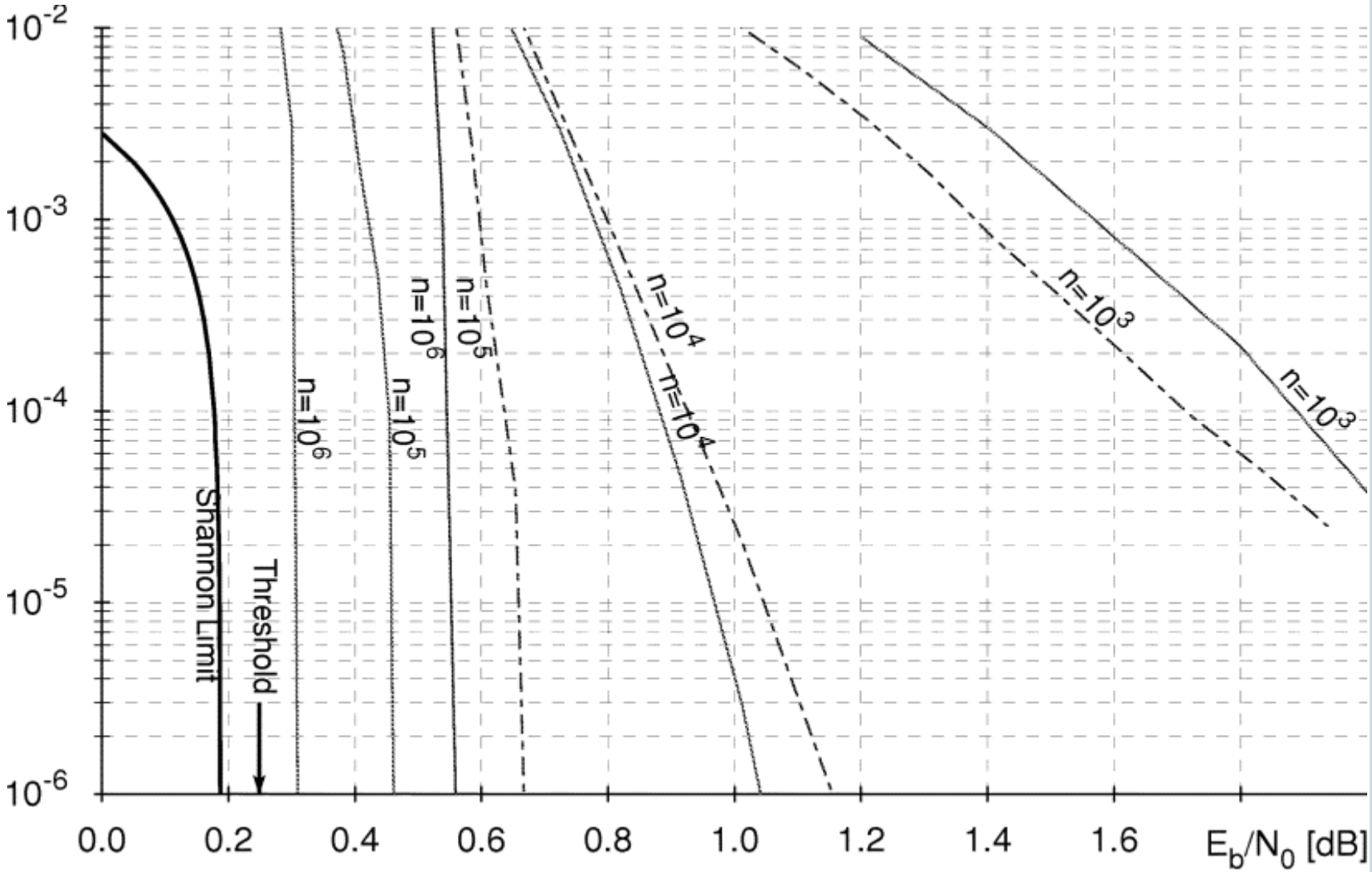
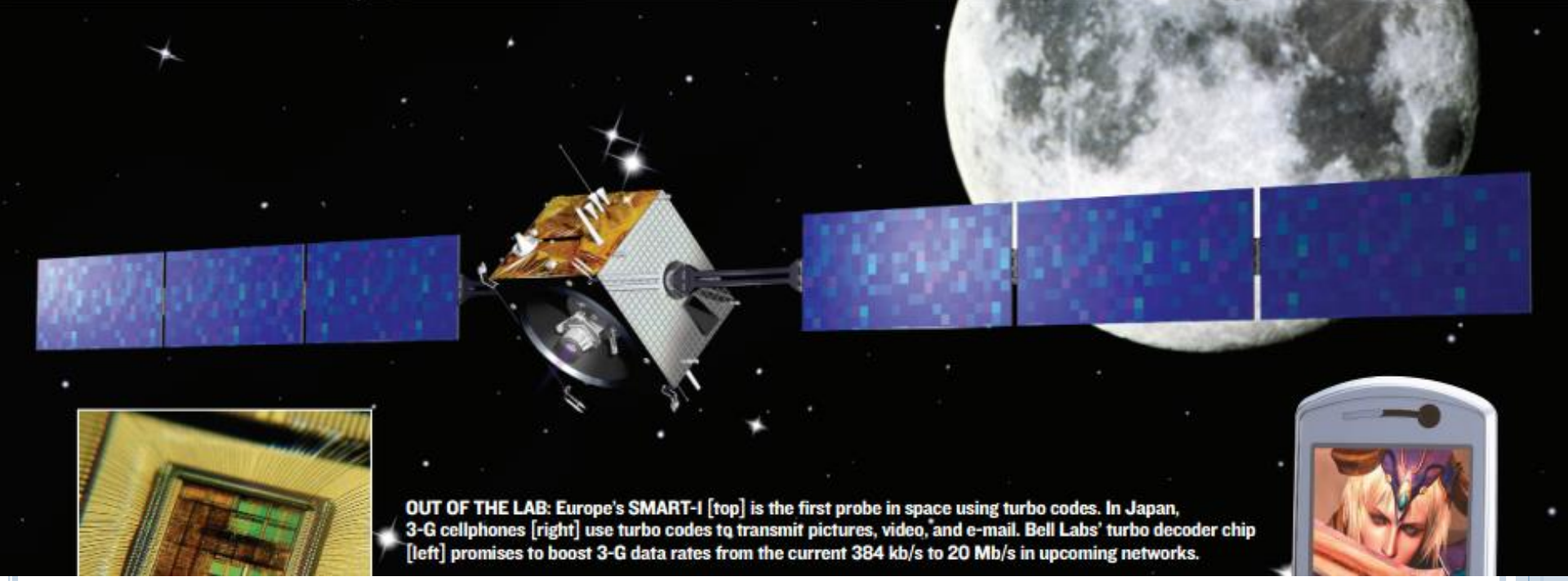


Fig. 3. Comparison between bit-error rates achieved by turbo codes (dashed curves) and LDPC codes (solid curves) . All codes are of rate one-half. Observe that longer LDPC codes outperform turbo codes and that the gap becomes the more significant the larger n is chosen. For short lengths, it appears that the structure of turbo codes gives them an edge over LDPC codes despite having a lower threshold.



OUT OF THE LAB: Europe's SMART-I [top] is the first probe in space using turbo codes. In Japan, 3-G cellphones [right] use turbo codes to transmit pictures, video, and e-mail. Bell Labs' turbo decoder chip [left] promises to boost 3-G data rates from the current 384 kb/s to 20 Mb/s in upcoming networks.

THANKS