差错控制编码和线性分组码（2）

循环码译码器

循环码译码过程

（1）由接收的码多项式 $B(D)$ 计算校正子多项式 $S(D)$

（2）由 $S(D)$ 确定错误图样 $E(D)$

（3）$E(D)+B(D)$—》纠正错误

校正子起到关键作用，问题如何计算？

对于循环码：

$$S(D) \equiv B(D) \bmod g(D)$$
$$\equiv E(D) \bmod g(D)$$

差错控制编码和线性分组码（2）

一个有关校正子的重要结果 –

循环码组移位 i 次后的校正子等于原码组校正子在除法电路中移位 i 次后所得的结果

$$B(D) \to S(D) \qquad S(D) \equiv B(D) \bmod g(D)$$

$$D^i B(D) \to S_i(D)$$

$$S_i(D) \equiv D^i B(D) \bmod g(D)$$

对译码电路的简化作用：

可纠正错误图样 $E(D)$ i 次循环移位—》$D^i E(D)$

—》可纠正错误图样

差错控制编码和线性分组码（2）

将多个错误图样 $E(D), DE(D), \cdots D^i E(D)$ 归为一类，用一个错误图样代表—》简化识别图样数—》简化译码电路

纠正错误数	识别错误图样数
1	1 -- C_{n-1}^{0}
2	C_{n-1}^{1}
⋮	⋮
t	C_{n-1}^{t-1}

差错控制编码和线性分组码（2）

梅吉特译码器 构造

（1）校正子计算电路

$$S(D) \equiv B(D) \bmod g(D)$$

（2）错误图样识别

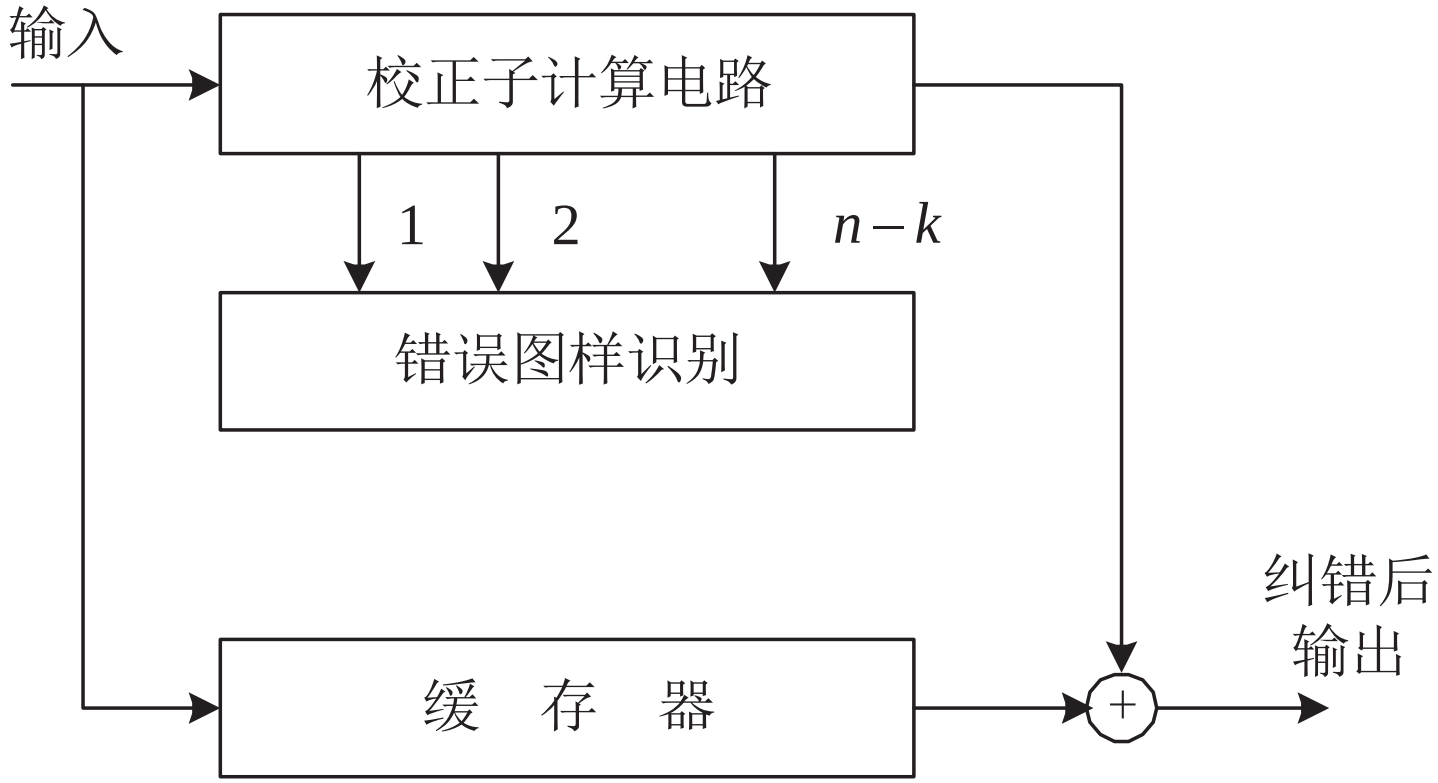
错误图样预先存储，由 $S(D)$ 输出的 $n-k$ 个端子进行查表操作—》确定错误图样 $E(D)$

（3）错误纠正电路

输入循环码组 $B(D)$ 先存在缓存，待确定 $E(D)$

$$A(D) = B(D) + E(D)$$

差错控制编码和线性分组码（2）



梅吉特译码器框图

差错控制编码和线性分组码（2）

梅吉特译码器应用举例：

梅吉特译码器适用于纠正 $t \le 2$ 个独立随机错误

假定 $(7,4)$ 循环汉明码生成多项式 $g(D) = D^3 + D + 1$，

设计纠正一位错误的译码器

分析：

（1）只要识别一个错误图样的类代表 （1000000），

$$E(D) = D^6, \quad S(D) \equiv E(D) \bmod g(D) = D^2 + 1$$

$S(D)$《——》101

（2）电路设计基于 $g(D)$ 除法电路的校正子计算电路

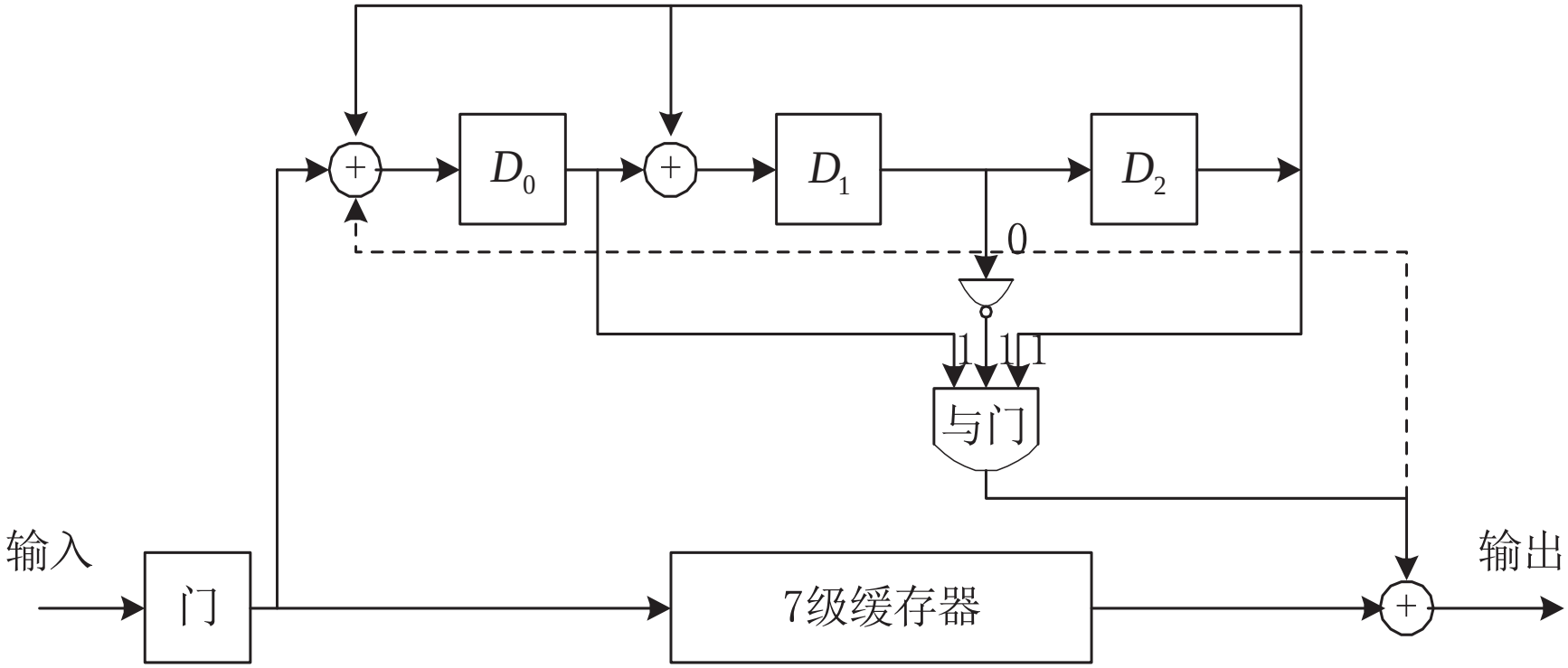
差错控制编码和线性分组码（2）

（3）电路设计有 7 级缓存器，用于存放输入循环码组

（4）码组同时进入缓存器和校正子计算电路，经过 7 个节拍，校正子计算电路输出 101—》译码—》纠错信号“1”—》和循环码 D^7 位模 2 加纠正错误，纠错信号反馈至校正子计算电路，使其不对后续码组发生影响

（5）如错误发生在第 7-i 位，第 7 节拍后，D^7 处于输出位，纠错信号为 0，再经 i 个节拍，错误位 D^{7-i} 处于输出位，此时 S 电路输出 101，译码输出纠错信号“1”，正好用于第 7-i 位纠错

差错控制编码和线性分组码（2）



（7，4）循环汉明码梅吉特译码器

差错控制编码和线性分组码（2）

四、BCH 码

BCH 码的特点：

（1）属于循环码的一个重要子类，具有纠正多个随机错误的能力，应用最普遍

（2）具有严密的代数结构，最小码距 $d_{\min}$ 和生成多项式密切相关，可根据纠错能力直接构造 BCH 码

有关本原循环码

（1）码长为 2^m-1，m 为正整数

（2）生成多项式由若干 m 阶或 m 的因子为最高阶的多项式相乘构成

差错控制编码和线性分组码（2）

如何判断本原循环码的存在？

对于 $(2^m-1,k)$ 循环码，判断其 2^m-1-k 阶的生成多项式

是否由多项式 $D^{2^m-1}+1$ 的因式构成

相关代数理论的结果：

每个 m 阶既约多项式一定能除尽　多项式 $D^{2^m-1}+1$

既约多项式 –

举例 1：对于 $m=5$ 的 $D^{31}+1$

有 6 个 $m=5$ 阶既约多项式，另有因式 $D+1$

见 P。352

差错控制编码和线性分组码（2）

有关多项式 $D^{2^m-1}+1$ 的既约多项式分解，可参见相应表格

说明：

（1）表格中的 m 阶和 $D^{2^m-1}+1$ 中的 m 相对应

（2）多项式系数用 8 进制表示

如 $m=4$ 时，$(23)_8=(10011)_2$—》D^4+D+1

（3）系数前的数字为多项式序号，$m_i(D)$ -- 最小多项式

（4）系数后的字母表示本原和非本原多项式

（5）表格中列出多项式的反多项式也是既约多项式，并能除尽 $D^{2^m-1}+1$

差错控制编码和线性分组码（2）

反多项式：对 $m=4,\ f(D)=D^4+D+1$

$$f^*(D)=D^{4-0}+D^{4-1}+D^{4-4}=D^4+D^3+1$$

举例 2：

对 $D^{15}+1$ 因式分解

$2^m-1=15\rightarrow m=4$

查表：$m_1(D)=23\rightarrow D^4+D+1$

$m_3(D)$、$m_5(D)$ ；$m_0(D)=D+1$

$m_7(D)=m_1^*(D)=D^4+D^3+1$

差错控制编码和线性分组码（2）

BCH 码的生成多项式

循环码编码的关键为生成多项式，BCH 码生成多项式形式：

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

其中，LCM 表示取最小公倍数

$d_{\min} \geq 2t+1$，纠正 t 个错误

本原 BCH 码 – 码长为 $2^m - 1$

非本原 BCH 码 – 码长为 $2^m - 1$ 因子

BCH 码的监督码元最多可有 mt 位

对于纠正 t 个随机错误的 BCH 本原码，有：

$$g(D) = m_1(D) \bullet m_3(D) \cdots m_{2t-1}(D)$$

差错控制编码和线性分组码（2）

举例：设计能纠正 3 个随机错误，码长为 15 的 BCH 码

$n=15 \rightarrow m=4$，为本原 BCH 码

有生成多项式 $g(D)=m_1(D)\bullet m_3(D)\bullet m_5(D)$

$$=D^{10}+D^8+D^5+D^4+D^2+D+1$$

为一个（15，5）循环码

有关扩展性问题：

（1）BCH 码长为奇数

（2）为得到偶数码长，并增强检错能力，可在 $g(D)$ 中增加一因式 $(D+1)$，相当于增加一校验位

差错控制编码和线性分组码（2）

（3）扩展后得到 $(n+1,k+1)$ 扩展 BCH 码，不具有循环码特性

BCH 译码

BCH 译码问题涉及更多代数学理论，本课程略

五、能纠正和检测突发错误的分组码

有关概念

突发错误 – 发生连续的一个错误序列

突发长度 – 错误序列长度

码的分类：交织码和专门设计码

差错控制编码和线性分组码（2）

交织码（ interleave code ）

交织码的引入：水平垂直监督码—》检测突发错误

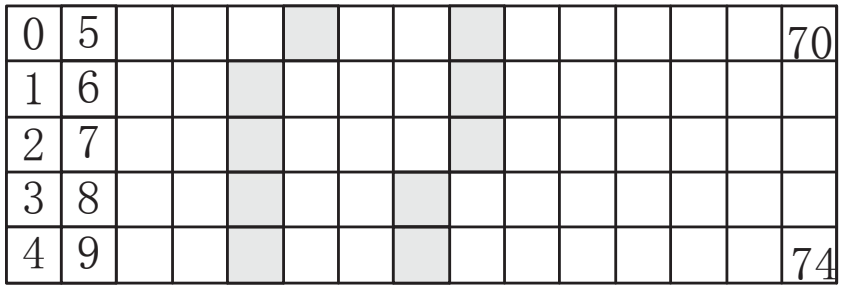
—》进一步扩展—》交织码

将能纠正 t 个随机错误的码作为行码，i 个行码构成一个方阵，该码长为 $n \bullet i$ 的码为交织码，可纠正 t 个突发长度为 i 的突发错误

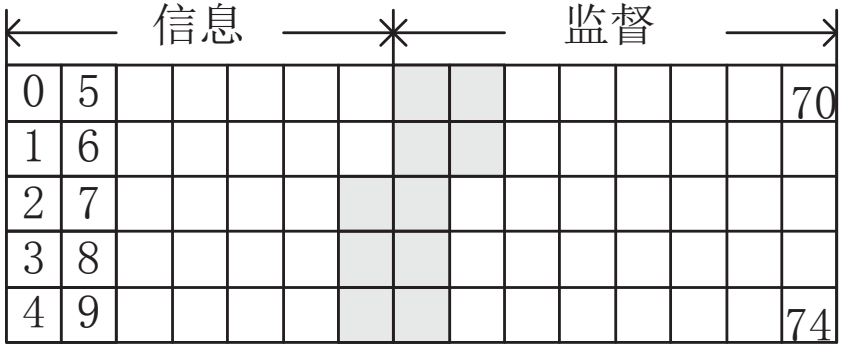
另一种形式

将能纠正 b 个突发错误的码作为行，i 个码组构成一个方阵，该交织码可纠正长度为 $b \bullet i$ 的突发错误

差错控制编码和线性分组码（2）



$t = 2,\ i = 5$



$b = 2,\ i = 5$

交织码图示

差错控制编码和线性分组码（2）

采用循环码构造交织码

利用原循环码的生成多项式直接构造交织码的生成多项式

假定交织码每行为具有生成多项式 $g(D)$ 的 (n,k) 循环码，

则交织度为 i 的交织码 (ni,ki)，其生成多项式为：

$$g_i(D) = g(D^i)$$

说明：（1）$g_i(D)$ 相当于在 $g(D)$ 的各项间插入（i-1）个 0

（2）$g_i(D)$ 能除尽 $D^{ni}+1$ ，(ni,ki) 码是循环码

差错控制编码和线性分组码（2）

举例：用生成多项式为 $g(D)=D^3+D^2+1$ 的（7，4）汉明码，构成交织度为 3 的（21，12），求交织码的生成多项式和监督矩阵

分析：利用原循环码和交织码间的对应关系

$$g_3(D)=g(D^3)=D^9+D^6+1$$

假定（7，4）码监督矩阵 H

H_3 为在 H 的每行、每列的元素间插入 $3-1=2$ 个“0”后构成的矩阵

参见 P。360

差错控制编码和线性分组码（2）

Fire 码

Fire 是循环码，用以纠正单个突发错误

以下构造仅给出结果

假定 $p(D)$ 是一个 m 阶的既约多项式，l 与 m 互素，则有

Fire 码的生成多项式：

$$g(D) = p(D)(D^l + 1)$$

Fire 码码长：$n = LCM(l, e)$，$e = 2^m - 1$

Fire 码监督位：$r = n - k = l + m$

差错控制编码和线性分组码（2）

CRC 码

CRC 码 – 循环冗余检验码，是一种循环码，具有很强的检错能力，编码和检错译码易于实现

表 11-16 给出了四种已成为标准的 CRC 码的生成多项式$g(D)$

编码格式：$M(D)+r(D)$

编码过程：$r(D)=M(D)D^{n-k} \bmod g(D)$

译码检错：假定 $B(D)$ 为接收码组，$\dfrac{B(D)}{g(D)}=q'(D)+\dfrac{r'(D)}{g(D)}$

检验 $r'(D)$ 是否为 0

卷积码概要

一、卷积码的引入

（1）分组码的编/译码，前后各组是无关的

编码时，码组的检验位只决定于本组信息位

译码时，从一个长为 n 码组中还原本组信息位

（2）分组码要增加纠错能力—》增加检验位—》使编/译码设备复杂

（3）如既要求 n, r 较小，又要求纠错能力较强，考虑使用卷积码

卷积码 – 码组中的监督码元不仅取决于本组的信息码元，也取决于前 m 组的信息码元

记为 (n, k, m)

卷积码概要

基本术语

在 (n,k,m) 码中，m 称为编码记忆

称 $K=m+1$ 为编码约束度，说明编码过程中互相有约束的码段个数

称 $N_c=K\bullet n$ 为编码约束长度，说明编码过程中相互约束的码元个数

对于译码过程：

根据当前输入码组，及以后 $L+m$ 段 $(L\geq 1)$ 所接收的码组，译出一个码组的信息码元，称 $L+m$ 译码约束度

卷积码概要

通信链路的卷积编码/译码和调制/解调结构

—输入信息源 $\mathbf{m}=m_1,m_2,\cdots,m_i,\cdots$，$m_i$为数字比特，且独立/等概；

—卷积码编码器将输入 $\mathbf{m}$ 转换成唯一的码字序列$\mathbf{U}=G(\mathbf{m})$，

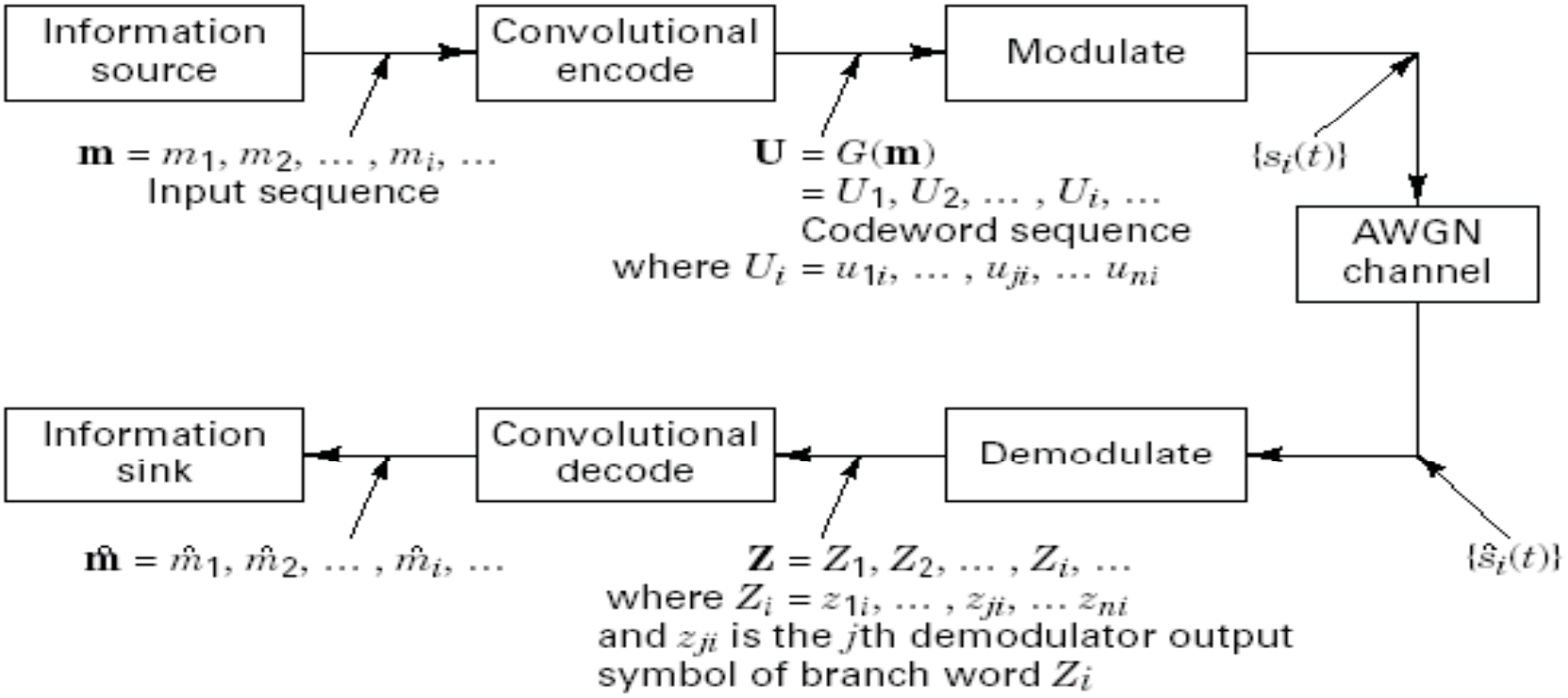
序列$\mathbf{U}=U_1,U_2,\cdots U_i,\cdots$，其中$U_i$为分支码字；

—码字序列$\mathbf{U}$对波形$s(t)$进行调制—》干扰—》$\hat{s}(t)$

—对$\hat{s}(t)$解调得解调序列$\mathbf{Z}=Z_1,Z_2,\cdots,Z_i,\cdots$

—译码器根据接收序列 $\mathbf{Z}$ 及编码过程先验知识，生成对原信息序列的估计$\hat{\mathbf{m}}=\hat{m}_1,\hat{m}_2,\cdots,\hat{m}_i,\cdots$

卷积码概要



通信链路的编码/译码和调制/解调分析

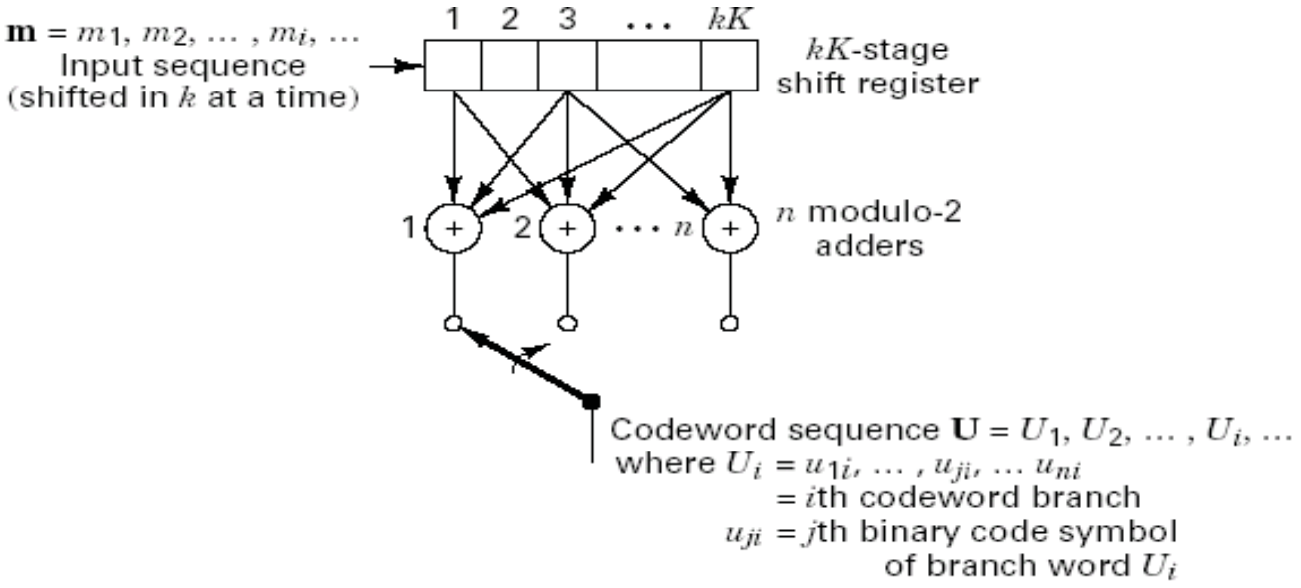
卷积码概要

典型卷积码编码器结构

—包含K个k位移位寄存器和 n个模 2 加法器；

—每个时间单元，输入k个信息比特，寄存器中其他比特向右移 k 位；

—顺序采样n个加法器输出得到 n元编码比特，效率k/n



卷积码概要

二、卷积码的基本概念

考虑一个（2，1，2）卷积码编码器，包括二级移位寄存器和 2 个模 2 加法器，输出码组包含 2 个输出码元$C_i^{(1)}, C_i^{(2)}$

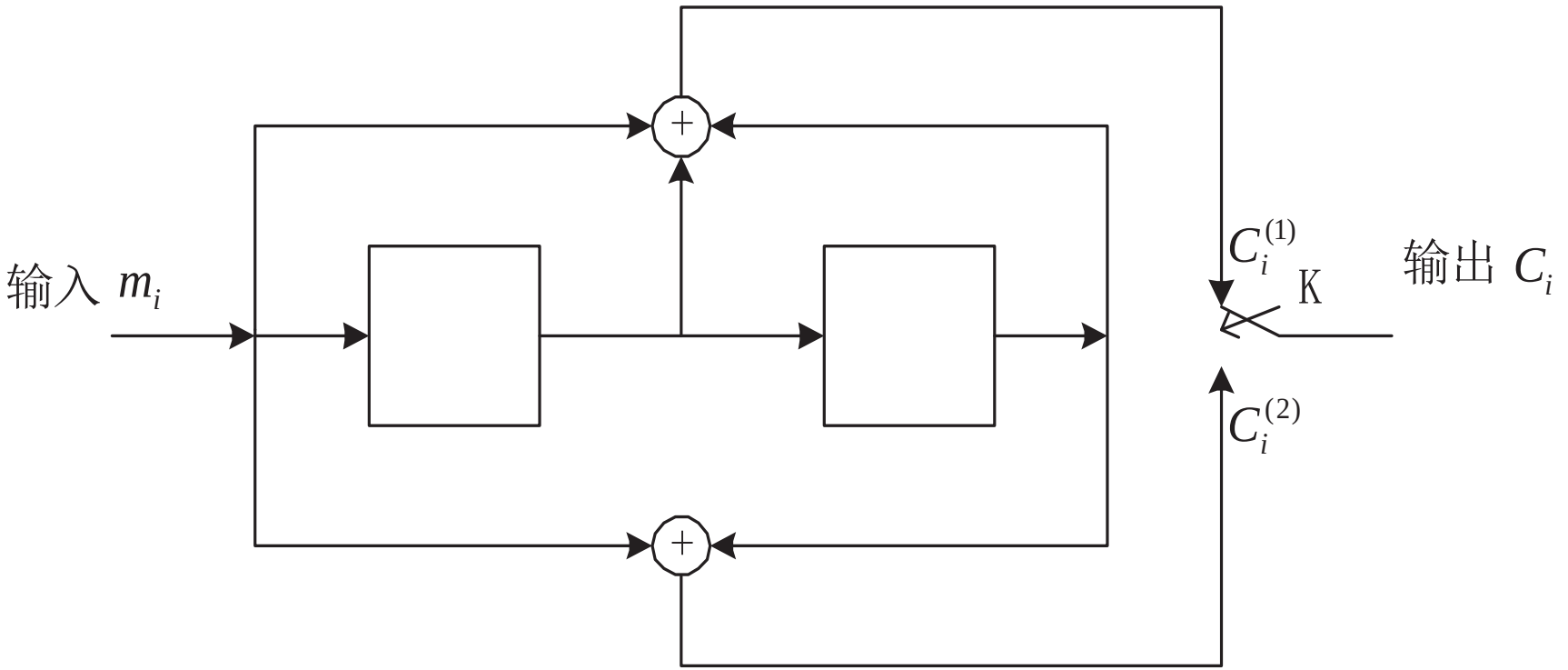
假定寄存器初始状态为 0，输出码组和输入信息码元关系：

$$C_i^{(1)} = m_{i-2} + m_{i-1} + m_i$$

$$C_i^{(2)} = m_{i-2} + m_i$$

如输入信息序列为 $m = (m_0, m_1, \cdots) = (1,1,1,\cdots)$ 时，有输出序列：$C = (11,01,10,\cdots) = (C_0, C_1, C_2, \cdots)$

卷积码概要



（2，1，2）卷积码编码器

卷积码概要

其中 $C_0, C_1, C_2, \cdots$ 称为卷积码的子码，形式可以有系统码和非系统码

以生成矩阵描述卷积码编码过程

$$C_0^{(1)} = m_0, \quad C_0^{(2)} = m_0$$

$$C_1^{(1)} = m_0 + m_1, \quad C_1^{(2)} = m_1$$

$$C_2^{(1)} = m_0 + m_1 + m_2, \quad C_2^{(2)} = m_0 + m_2$$

$$C_i^{(1)} = m_{i-2} + m_{i-1} + m_i, \quad C_i^{(2)} = m_{i-2} + m_i$$

卷积码概要

表示成矩阵形式：

$$\begin{bmatrix} C_0^{(1)} \\ C_0^{(2)} \\ C_1^{(1)} \\ C_1^{(2)} \\ C_2^{(1)} \\ C_2^{(2)} \\ \cdots \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \\ 1 & 0 & 0 & \\ 1 & 1 & 0 & \\ 0 & 1 & 0 & \\ 1 & 1 & 1 & \\ 1 & 0 & 1 & \\ \cdots & \cdots & \cdots & \cdots \end{bmatrix} \begin{bmatrix} m_0 \\ m_1 \\ m_2 \\ m_3 \\ m_4 \\ m_5 \\ \cdots \end{bmatrix}$$

上式可改写为：

卷积码概要

$$\begin{bmatrix} C_0^{(1)} \\ C_0^{(2)} \\ C_1^{(1)} \\ C_1^{(2)} \\ C_2^{(1)} \\ C_2^{(2)} \\ \cdots \end{bmatrix}^{\tau} = \begin{bmatrix} m_0 & m_1 & m_2 & m_3 & m_4 & m_5 & \cdots \end{bmatrix} \begin{bmatrix} 11 & 10 & 11 & 00 & \cdots \\ & 11 & 10 & 11 & \cdots \\ & & 11 & 10 & \cdots \\ & & & 11 & \cdots \\ & & & \cdots & \\ & & & \cdots & \\ & & & \cdots & \end{bmatrix}$$

或 $C_\infty = MG_\infty$

卷积码长度和信息序列成正比

C_∞：编码输出序列；G_∞：生成矩阵；M：信息序列

卷积码概要

上式中生成矩阵 G_∞ 为半无穷矩阵，决定于矩阵第 1 行，如卷积码子码为 n 位，第 i 行将第 1 行向右移动 $(i-1)n$ 位

称第 1 行为基本生成矩阵 g_∞

$$g_\infty = (11,10,11,00\cdots)$$

基本生成矩阵是卷积码的关键，其码元数为编码约束长度

子码的生成多项式为：

$$g_1(x) = 1 + x + x^2$$

$$g_2(x) = 1 + x^2$$

卷积码概要

假定输入信息序列为：

$$m(x) = m_0 + m_1 x + m_2 x^2 + \cdots + m_i x^i + \cdots$$

卷积码输出为：

$$C_1(x) = m(x) g_1(x)$$
$$C_2(x) = m(x) g_2(x)$$

考察生成矩阵 G_∞，从第 7 列开始，以后每二列均与前二列相同，故对于卷积码只需研究前六列组成的截短矩阵 G'

$$G' = \begin{bmatrix} 11 & 10 & 11 \\ & 11 & 10 \\ & & 11 \end{bmatrix}$$

卷积码概要

三、树图、状态图和距离

卷积码生成矩阵 – 半无限；信息序列 – 半无限

—》输出码序列 – 半无限

对于卷积码 (n,k,m)，在 L 组有限序列内，卷积码有 2^{kL} 个不同码序列，码序列数按 L 以指数增长

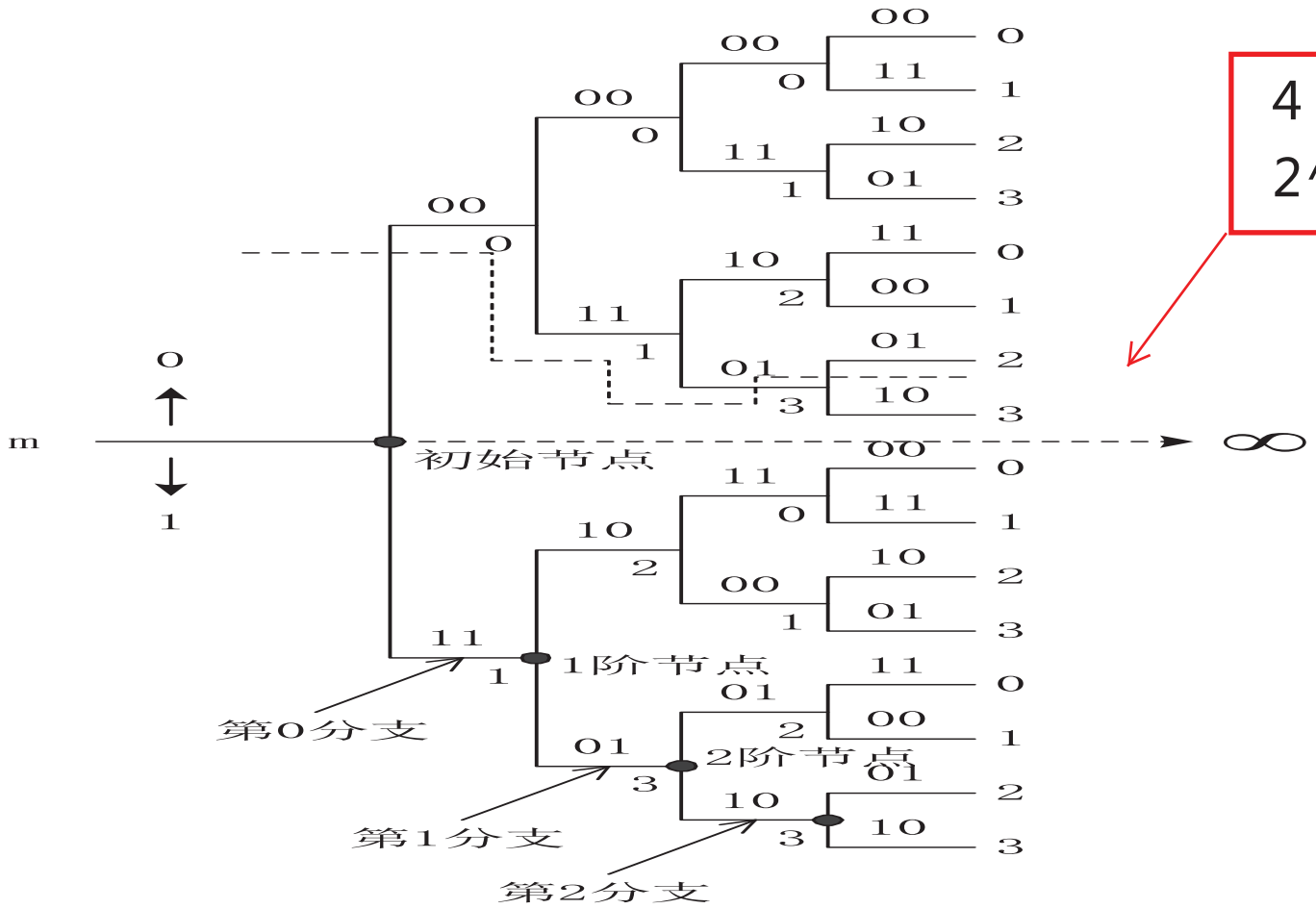
卷积码属于树码，码树图可表示卷积码的编码过程

对于上述（2，1，2）卷积码，

$$g_1(x)=1+x+x^2;\quad g_2(x)=1+x^2$$

直接有：$g=(11\ 10\ 11)$—》$G^{'}$—》*码树图*

卷积码概要



（2，1，2）码树图

4个节拍，输出2^4=16个码序列

卷积码概要

编码状态图

以（2，1，2）卷积码为例，2 个寄存器的不同存数对应 4 个不同状态

存数 00—》状态S_0；存数 01—》状态 S_1

存数 10—》状态S_2；存数 11—》状态S_3

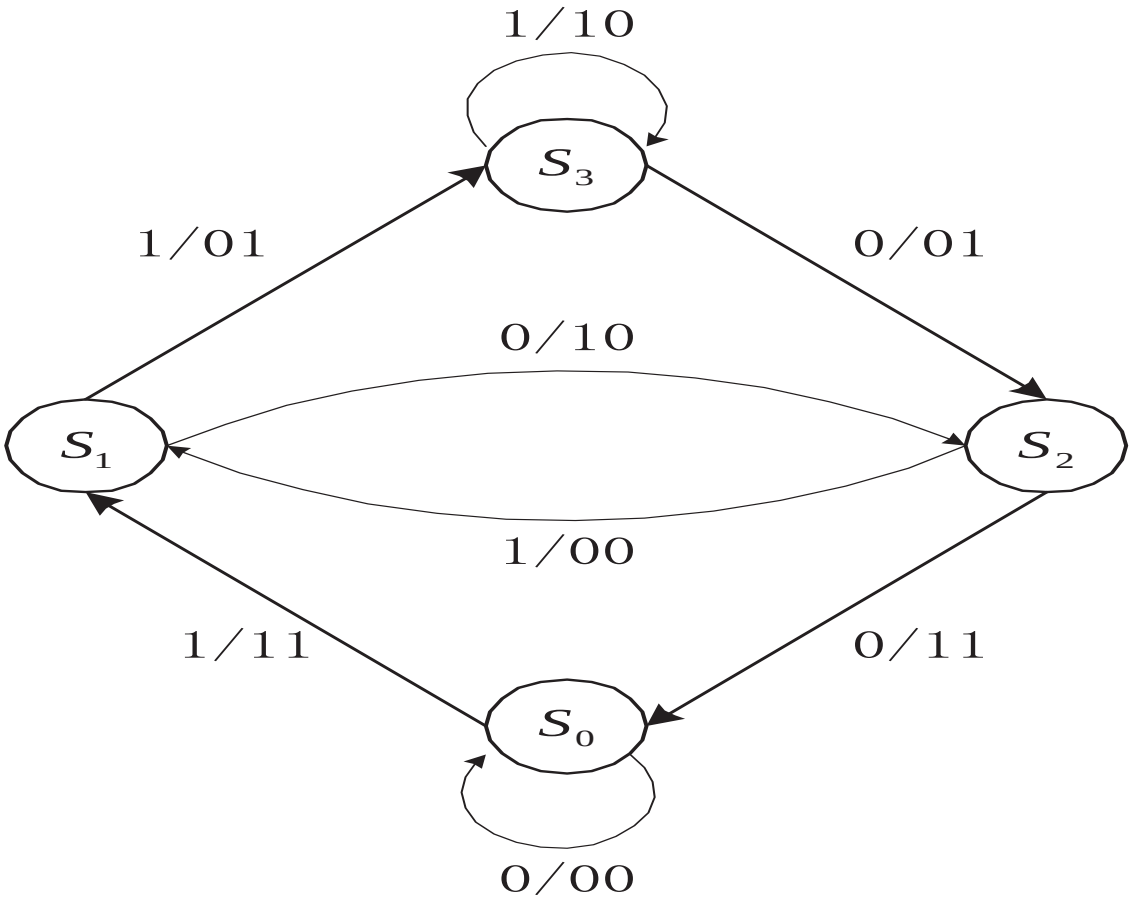
k/n 表示输入 k 位信息元后使状态发生变化，输出 n 位码元

如从 S_0 开始，输入 1，输出 11，寄存器存数—》01—》S_1

对 S_1，输入 0，输出 10，寄存器存数—》10—》S_2

对 S_2，输入 1，输出 00，寄存器存数—》01—》S_1

卷积码概要



（2，1，2）码的状态图

卷积码概要

利用状态图，已知输入信息序列 m，可以容易得出状态转移过程和输出码序列

利用码树图描述编码过程，树的每一个分支表示一单独输入信息元，“0”和“1”分别对应上/下分支，编码输出对应树的一个特殊路径上的数码组成的序列
输入 0110—》00110101 见图中虚线

表面上，随着输入序列增加，路径按指数增加
实际上路径“可控”码树中标明“0”~“3”的节点对应
状态 S_0 ~ S_3，同一“状态”任意两个发散的树分支是相同的

卷积码概要

总结：

（1）码树的以上特性称为重复特性

（2）研究卷积码时，只需研究 N 个分支长的有限码树，即截短码树，对（2，1，2）码，由三阶节点和三段分支组成，有 $2^{k(m+1)}=2^3=8$ 条路径《—》8 个码序列

（3）如图中第 2 分支包含的两组四个节点可以合并，这种码树中相同状态合并的可能性构成了 Viterbi 译码算法的原理

卷积码概要

四、卷积码的最佳译码 -- Viterbi 算法

Viterbi 算法的设想

（1）卷积码编码过程视为输入信息元通过码树的某一条路径的过程

（2）译码器根据接收到的序列、信道统计特性及编码规则，力图寻找原编码时通过码树的路径，找到该路径相当于完成了译码并纠正传输错误

（3）概率译码中，通过计算各路径所对应序列和接收序列的偏差，作出最大相似然估计来寻找路径

（4）Viterbi 译码利用码树重复特性大大减少计算量

卷积码概要

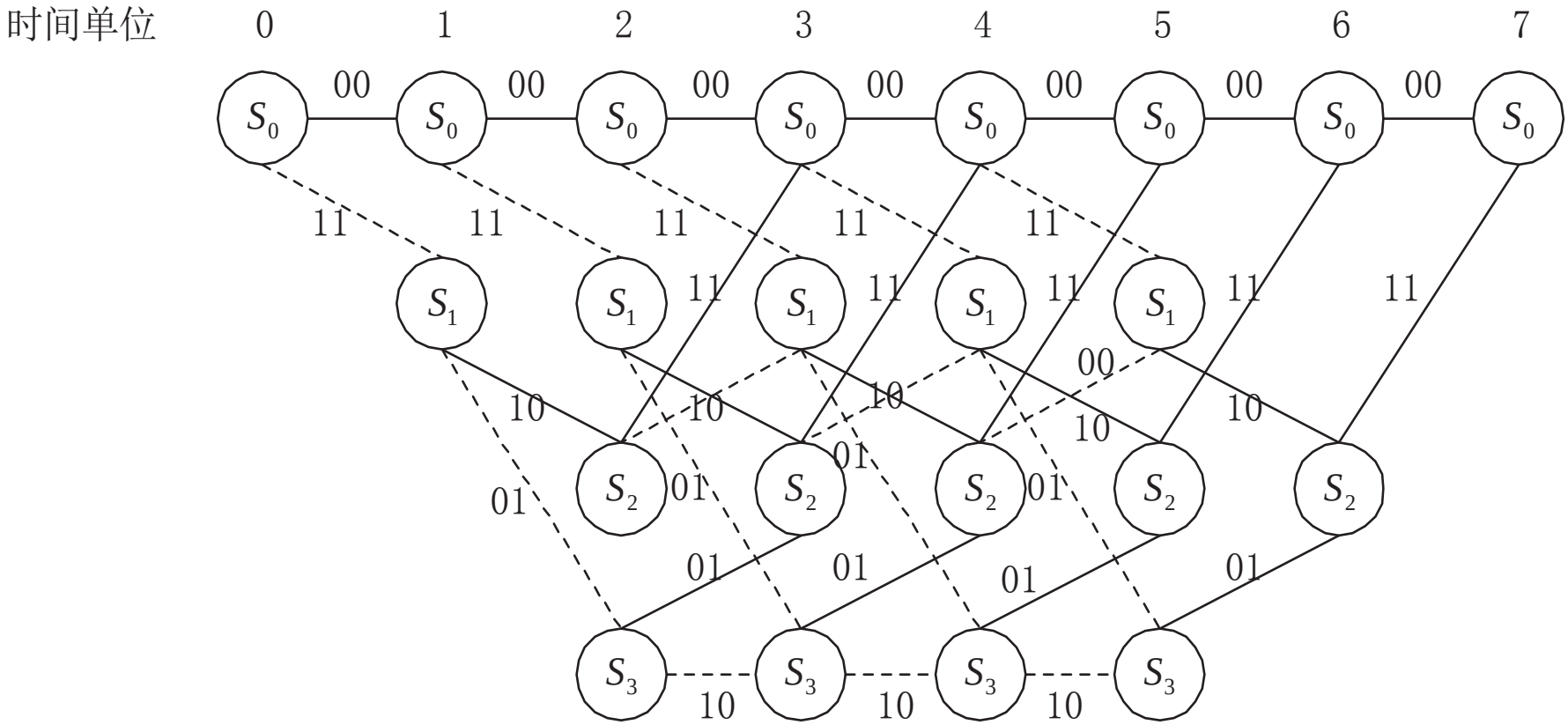
网格图

网格图将码树中所有相同状态的节点合并成一个节点，用网格图可以表现码的状态转移时间关系图

对于 $L=5$ 时（2，1，2）码的网格图，L 表示输入信息组数目，节点数 $L+m+1=8$ 《—》时间单元 0~7

图中每一个状态有两个输出分支，实线 – 输入编码器信息码组为“0”，虚线为“1”，分支上的 2 位数字代表编码器输出

卷积码概要



（2，1，2）码 L=5 时的网格图

卷积码概要

最大似然译码

假定所有输入信息序列等概，构造似然函数$P(\mathbf{Z}|\mathbf{U}^{(m)})$，可以得到具有最小差错概率的译码器：

$$P(\mathbf{Z}|\mathbf{U}^{(m')}) = \max_{all\ \mathbf{U}^{(m)}} P(\mathbf{Z}|\mathbf{U}^{(m)})$$

其中$\mathbf{Z}$为接收序列，$\mathbf{U}^{(m)}$是可能的发送序列

如噪声是零均值高斯白噪声，且信道无记忆，对 $1/n$卷积码：

$$P(\mathbf{Z}|\mathbf{U}^{(m)}) = \prod_{i=1}^{\infty} P(Z_i|U_i^{(m)}) = \prod_{i=1}^{\infty}\prod_{j=1}^{n} P(z_{ji}|u_{ji}^{(m)})$$

信道无记忆

卷积码概要

最大似然译码相当于在网格图中选择一条路径(一个码字序列)，使得下式最大:

$$\prod_{i=1}^{\infty}\prod_{j=1}^{n} P(z_{ji} \mid u_{ji}^{(m)})$$

取对数得对数最大似然函数:

$$\gamma_U(m) = \log P(\mathbf{Z} \mid \mathbf{U}^{(m)}) = \sum_{i=1}^{\infty} \log P(Z_i \mid U_i^{(m)}) = \sum_{i=1}^{\infty}\sum_{j=1}^{n} \log P(z_{ji} \mid u_{ji}^{(m)})$$

分析:—译码问题等价于在网格图中寻一条路径，使得$\gamma_U(m)$最大

—对于有 L个分支字的码字序列有 2^L种可能;

—采用网格图译码路径从幸存路径中选取，降低了复杂度

卷积码概要

信道模型：硬判决和软判决的比较

编码/调制过程

—卷积码字序列 $\mathbf{U}^{(m)}$ 由分支字组成 $k/n, K$；

—每个分支字由 n 个码元组成，码字序列构成比特流；

—码字序列序列经调制器将码元转换为波形 $s_i(t)$；

—假定信道干扰为高斯噪声，接收信号 $\hat{s}_i(t) = s_i(t) + n_i(t)$；

—$\hat{s}_i(t)$ 先由解调器解调，再经译码器处理

卷积码概要

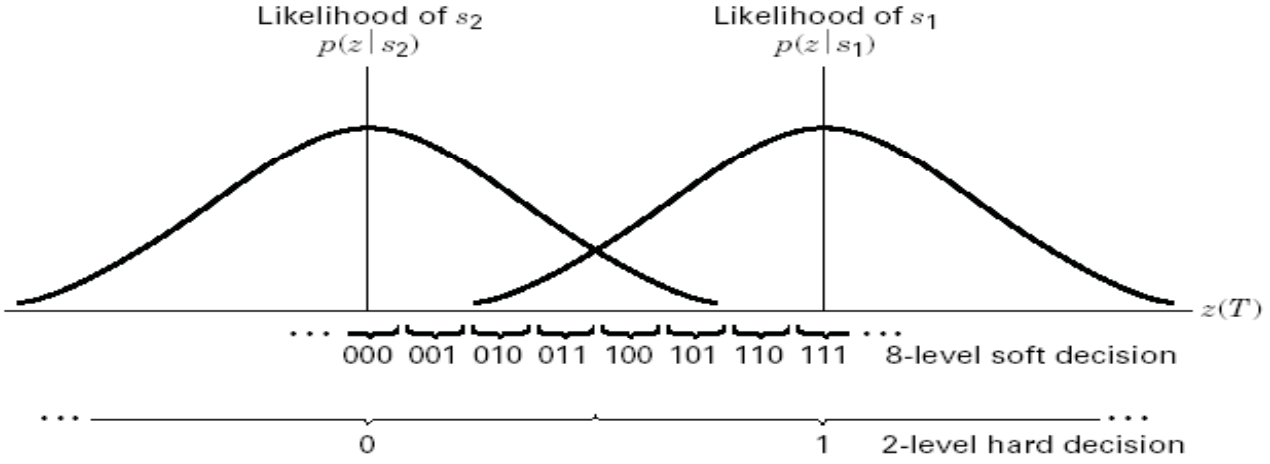
假定为二进调制，在每个码元间隔内传递波形 $s_1(t)$和 $s_2(t)$

接收信号$z(t)=s_i(t)+n(t)$ —》解调取决于似然函数$p(z\mid s_i)$

解调器输出构造：

—硬判决实现，判定 $z(t)$为 0 或 1，量化级为 2

—软判决实现，解调器将两个以上$z(t)$量化值馈送给译码器



卷积码概要

分析：

—解调器发送二进制硬判决给译码器，作硬判决译码；

—解调器软判决时，相当于发送带置信度的码元判决给译码器，作软判决译码；

—软判决译码时，解调器给译码器提供更多中间信息，以更好恢复信息序列；

—所有译码器作出的信息最终判决是硬判决；

—对于高斯信道，获取同样的比特差错性能，软判决需要的E_b/n_0比硬判决低，如 8 级软判决时低 $2dB$；

—软判决代价是译码器存储要求增大

卷积码概要

二进制对称信道(BSC)

对 BSC 信道，有： $p(0|1)=p(1|0)=p$

$$p(0|0)=p(1|1)=1-p$$

BSC 信道只允许硬判决，解调器输出 $\mathbf{Z}=\{Z_i\}$送译码器

假定码字序列 $\mathbf{U}^{(m)}$经差错概率为 p的 BSC 信道传输，$\mathbf{Z}$是相应接收译码序列，若码字长度为 L，且有 d_m比特不同

于是： $P(\mathbf{Z}|\mathbf{U}^{(m)})=p^{d_m}(1-p)^{L-d_m}$

卷积码概要

上式对数似然函数为：

$$\log P(\mathbf{Z}\,|\,\mathbf{U}^{(m)}) = -d_m \log\left(\frac{1-p}{p}\right) + L\log(1-p)$$

假定 $p < 0.5$，上式改写为：

$$\log P(\mathbf{Z}\,|\,\mathbf{U}^{(m)}) = -Ad_m - B，\ A,B\text{为正数}$$

结论：对 BSC 信道，译码应选择码字$\mathbf{U}^{(m)}$，它与 $\mathbf{Z}$之间的汉明距离d_m最小

卷积码概要

高斯信道

对于高斯信道，解调器作软判决，输出码元 z_{ji}取自连续字符号值，可以证明，最大化 $P(\mathbf{Z}|\mathbf{U}^{(m)})$相当于最大化

$$\sum_{i=1}^{\infty}\sum_{j=1}^{n} z_{ji}u_{ji}^{(m)}$$

分析：

—上式相当于最大化码字序列 $\mathbf{U}^{(m)}$和模拟接收序列 $\mathbf{Z}$的内积；

—选择与 $\mathbf{Z}$具有最小欧氏距离的码字 $\mathbf{U}^{(m)}$；

卷积码概要

Viterbi 算法

（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，计算与送入译码器的序列 R 的汉明距离，对每一状态，挑选并存储一条与 R 有最小汉明距离的路径，称幸存路径

（2）$j=j+1$，计算此时进入每一状态的所有分支和同这些分支相连的前一时刻留下的幸存路径与 R 的汉明距离，仍挑选与 R 有最小汉明距离的幸存路径

（3） 若 $j<L+m$，重复（2），当 $j>L$ 后，状态数减少，最后只剩下一条幸存路径，即译码输出

卷积码概要

举例：输入到（2，1，2）编码器的信息序列为 M=（1011100），

编码器输出 c=（11100001100111），经信道送入译码器的

序列 R=（10100001110111），有 2 个错误，利用 Viterbi

算法求估值信息序列 $\hat{m}$ 和码序列 $\hat{c}$

解： Viterbi 译码器接收 R 序列过程见图

画出各时刻进入每一状态的幸存路径及最小汉明距离值

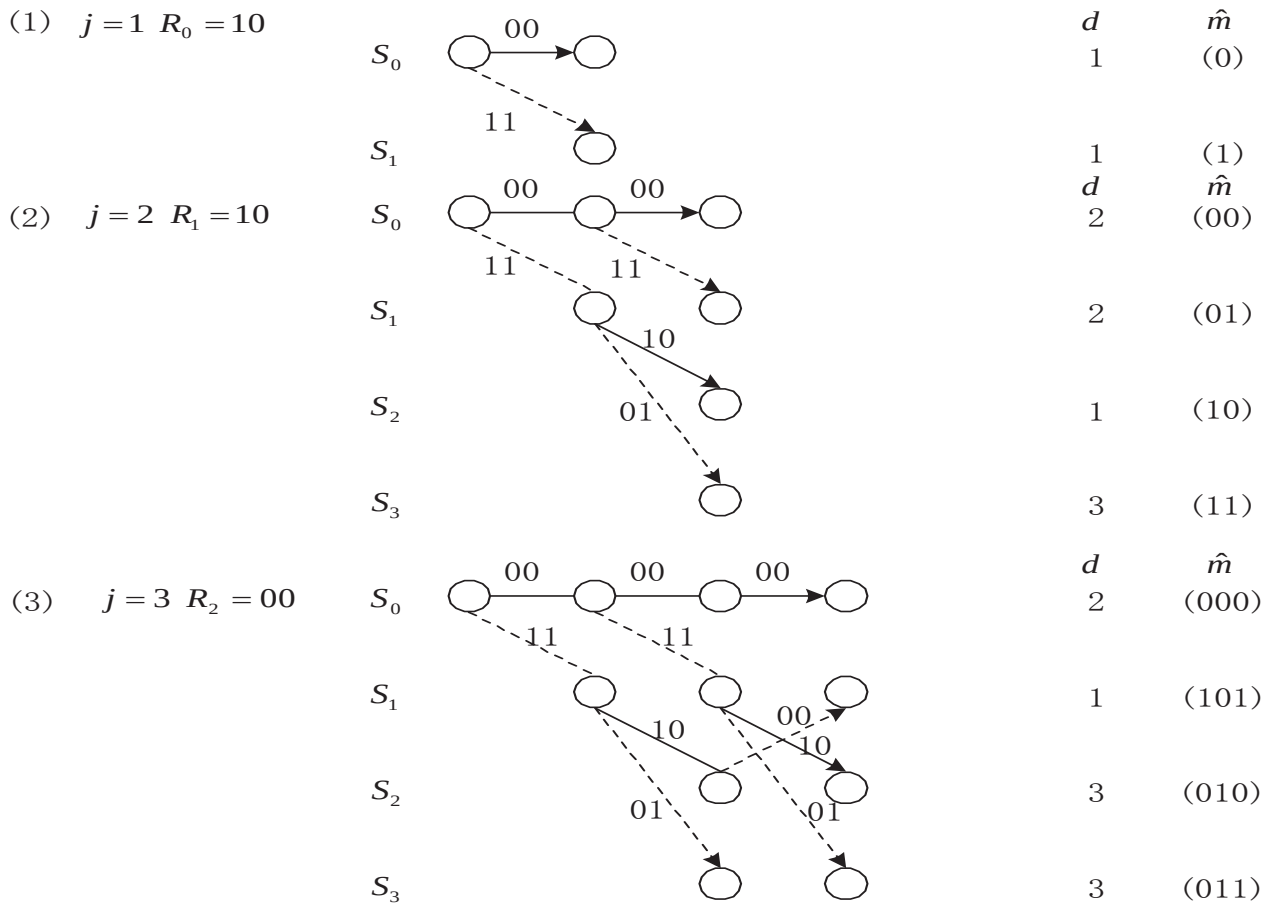
及对应 $\hat{m}$

当 $j=7$ 后，4 条幸存路径只剩一条，即输出码序列

$\hat{c}=(11100001100111)$，信息码估值$\hat{M}=(1011100)$

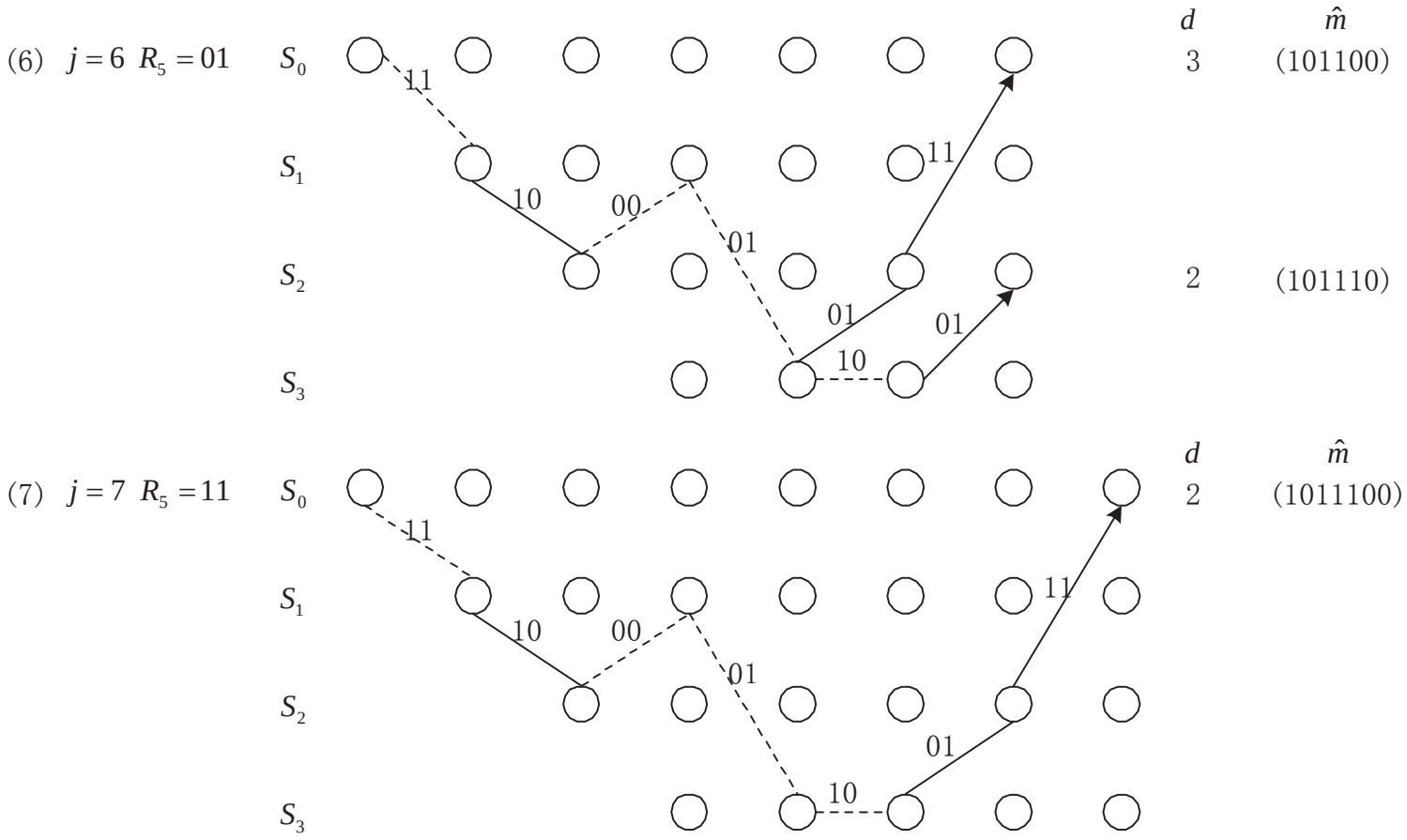
R 中 2 个错误得到纠正

卷积码概要



Viterbi算法译码过程（1）

卷积码概要



Viterbi算法译码过程（2）