第9章　差错控制编码

- 9.1　引言
- 9.2　纠错编码的基本原理
- 9.3　常用的简单编码
- 9.4　线性分组码
- 9.5　循环码
- 9.6　卷积码
- 9.7　网格编码调制

本章内容目的要求

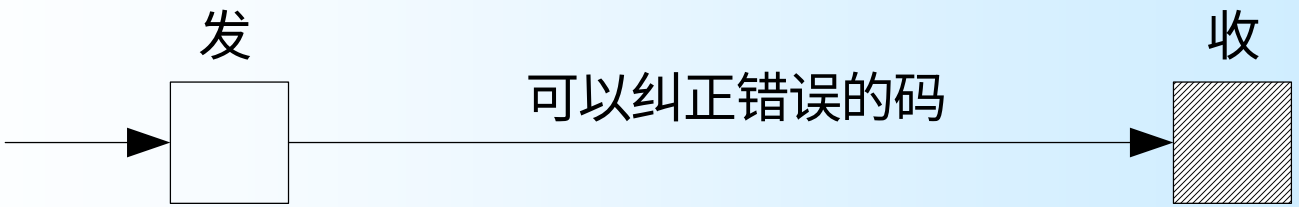
- **教学要求:**了解差错控制编码的基本方法和基本原理，掌握线性分组码的一般构造原理及汉明码、循环码、卷积码的概念。理解m序列的产生原理、性质及数字加密的概念。
- **内容提要:**差错控制的基本方式及信道编码的概念；检错码；线性分组码；卷积码；m序列；数字加密基本方法介绍。
- **重点**：汉明码的生成矩阵、监督矩阵的计算；循环码的生成矩阵、监督矩阵的计算。
- **难点**：卷积码的原理。

9.1 引言

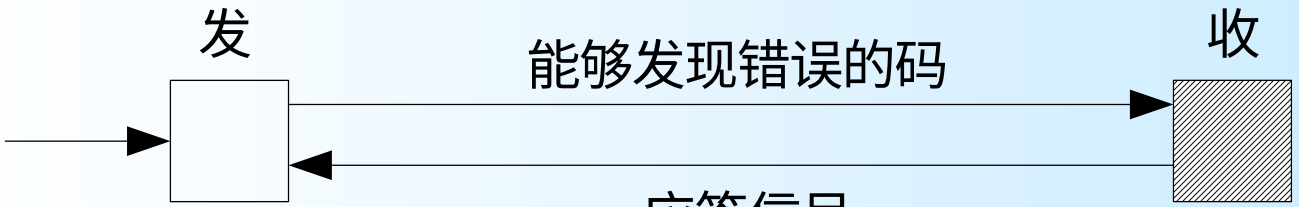
9.1.1 信道编码

- 信元编码：为了提高数字信号传输的有效性而采取的编码。
- 信道编码：为了提高数字通信的可靠性而采取的编码。
- 信道编码方法：在信息序列上附加上一些监督码元，发现和纠正错误。

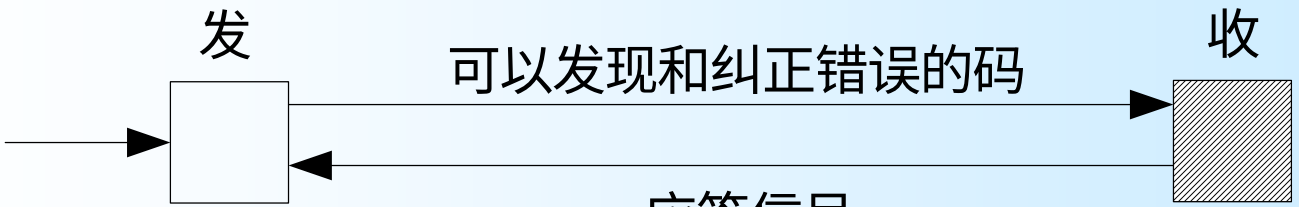
9.1.2 差错控制方式



(a) 前向纠错(FEC)



(b) 检错重发(ARQ)



(c) 混合纠错检错(HEC)

1、检错重发法 ：

- 检错重发（ARQ）的优点主要表现在：
  - （1）只需要少量的冗余码，就可以得到极低的输出误码率；
  - （2）有一定的自适应能力；
- 某些不足主要表现在：
  - （1）需要反向信道，故不能用于单向传输系统，并且实现重发控制比较复杂；
  - （2）通信效率低，不适合严格实时传输系统。

2、前向纠错法

- 发送端经信道编码后可以发出具有纠错能力的码字；
- 接收端译码后不仅可以发现错误码，而且可以判断错误码的位置并予以自动纠正。

3、反馈校验法

- 接收端将收到的新码原封不动的转发回发送端，与源码比较。如果发现错误则发送端再进行重传。

9.1.3 纠错编码的分类

（1）按照信道编码的不同功能，可以将它分为检错码和纠错码。

（2）按照信息码元和监督码元之间的检验关系，可以将它分为线性和非线性码。

（3）按照信息码元和监督码元之间的约束方式不同，可以将它分为分组码和卷积码。

（4）按照信息码元在编码后是否保持原来的形式，可以将它分为系统码和非系统码。

（5）按照纠正错误的类型不同，可以将它分为纠正随机错误码和纠正突发错误码。

随着数字通信系统的发展，可以将信道编码器和调制器统一起来综合设计，这就是所谓的网格编码调制。

第9章　差错控制编码

- 9.1　引言
- 9.2　纠错编码的基本原理
- 9.3　常用的简单编码
- 9.4　线性分组码
- 9.5　循环码
- 9.6　卷积码
- 9.7　网格编码调制

9.2 纠错编码的基本原理

分组码举例

- 设：有一种由3个二进制码元构成的编码，它共有2^3 = 8种不同的可能码组：

  000 – 晴　001 – 云　010 – 阴　011 – 雨

  100 – 雪　101 – 霜　110 – 雾　111 – 雹

  这时，若一个码组中发生错码，则将收到错误信息。

  - 若在此8种码组中仅允许使用4种来传送天气，例如：令

    000 – 晴　011 – 云　101 – 阴　110 – 雨

    为许用码组，其他4种不允许使用，称为禁用码组。

    这时，接收端有可能发现（检测到）码组中的一个错码。

    – 这种编码只能检测错码，不能纠正错码。

  - 若规定只许用两个码组：例如

    000 – 晴　111 – 雨

    就能检测两个以下错码，或纠正一个错码。

1、分组码

表示为（n，k），n表示码组的长度；k信息的长度；$r = n-k$表示监督位长度。

几个概念：

码长：码字中码元的数目；

码重：码字中非0数字的数目；

码距：两个等长码字之间对应位不同的数目，有时也称作这两个码字的汉明距离。

最小码距：在码字集合中全体码字之间距离的最小数值。

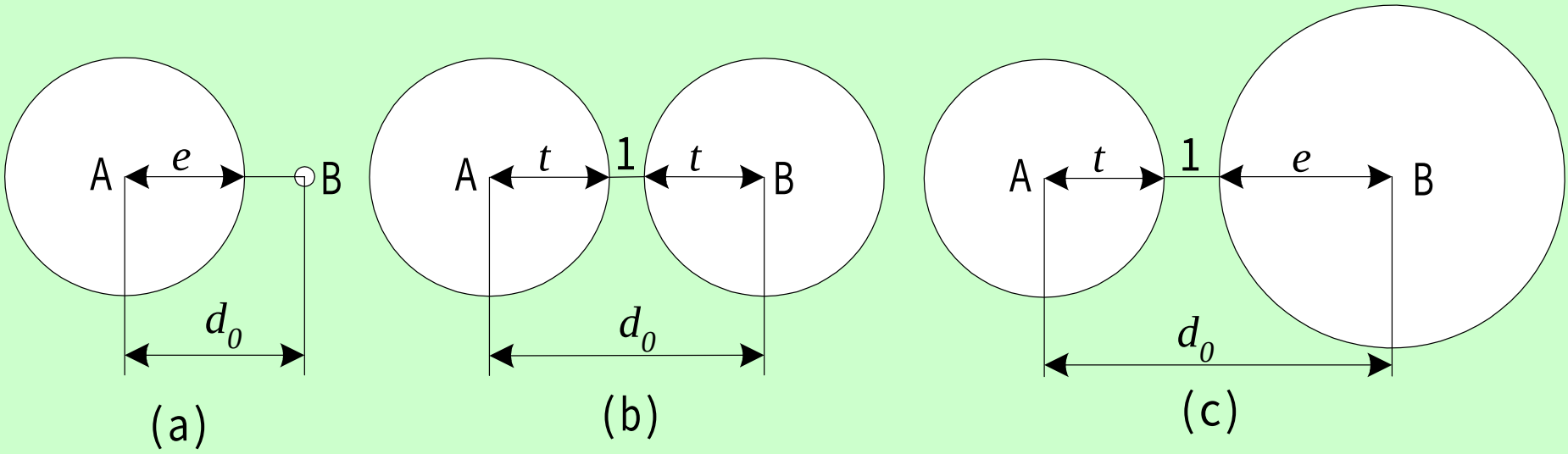
纠错码的抗干扰能力完全取决于许用码字之间的距离，码的最小距离越大，说明码字间的最小差别越大，抗干扰能力就越强。分组码的最小汉明距离为$d0$

2、检错和纠错能力

（1）当码字用于检测错误时，如果要检测e个错误，则 $d_0 \geq e+1$；

（2）当码字用于纠正错误时，如果要纠正t个错误，则 $d_0 \geq 2t+1$；

（3）若码字用于纠t个错误，同时检e个错误时（$e > t$），则 $d_0 \geq t+e+1$。



编码效率R_c可以用下式表示：

$$R_c = k/n = (n-r)/n = 1 - r/n$$

- 9.1 引言
- 9.2 纠错编码的基本原理
- 9.3 常用的简单编码
- 9.4 线性分组码
- 9.5 循环码
- 9.6 卷积码
- 9.7 网格编码调制

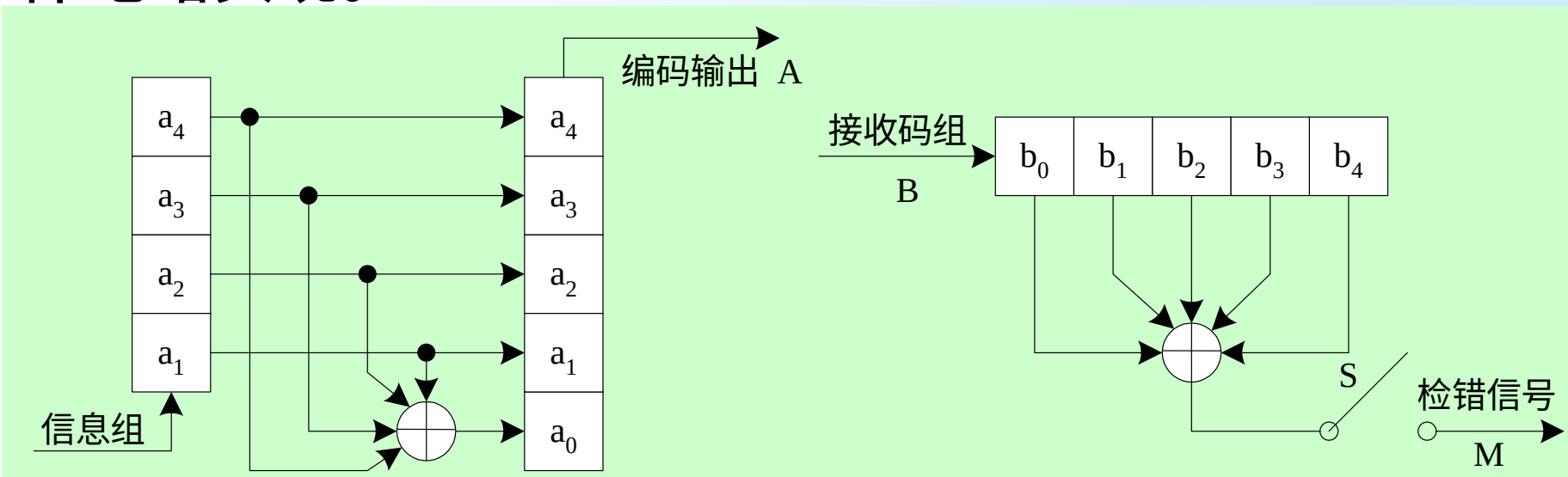
9.3 常用的简单编码

9.3.1 奇偶监督码

可以表示成为（n，n-1）。如果是奇监督码，在附加上一个监督元以后，码长为n的码字中“1”的个数为奇数个；如果是偶监督码，在附加上一个监督元以后，码长为n的码字中“1”的个数为偶数个。

$$a_{n-1}+a_{n-2}+\cdots+a_1+a_0=0$$

奇偶监督码的编码可以用软件实现，也可用硬件电路实现。



如果码组B无错，B＝A，则M＝0；如果码组B有单个（或奇数个）错误，则M＝1。

编码效率：　R=(n-1)/n

9.3.2 二维奇偶监督码

- 二维奇偶监督码又称方阵，有时还被称为矩阵码。

1 1 0 0 1 0 1 0 0 0 0 1 0 0 0 0 1 1 0 1 0 1 1 1 1 0 0 0 0 1 1 0 0 1 1 1 0 0 0 0 1 0 1 0 1 0 1 0 1 0	0 0 1 0 1
1 1 0 0 0 1 1 1 1 0	0

- 二维奇偶监督码适于检测突发错码。二维奇偶监督码不仅可用来检错，还可用来纠正一些错码。

9.3.3 恒比码

- 恒比码又称等重码，该码的码字中1和0的位数保 持恒定的比例。具体情况见表9-2。

目前我国电传通信中普遍采用3：2码即5中取3码，国际上通用的ARQ电报通信系统中，采用3：4码即7中取3码。

- 9.1 引言
- 9.2 纠错编码的基本原理
- 9.3 常用的简单编码
- 9.4 线性分组码
- 9.5 循环码
- 9.6 卷积码
- 9.7 网格编码调制

9.4 线性分组码

9.4.1 基本概念

分组码是一组固定长度的码组，可表示为（n，k），通常它用于前向纠错。在编码时，k个信息位被编为n位码组长度，而n-k个监督位的作用就是实现检错与纠错。

这样，一个k比特信息的线性分组码可以映射到一个长度为n码组上。

- 线性分组码的主要性质如下：

（1）任意两许用码之和仍为一许用码，也就是说，线性分组码具有封闭性；

（2）码组间的最小码距等于非零码的最小码重。

- 对偶校验时的监督关系。在接收端解码时，实际上就是在计算：

$$S = b_{n-1}+b_{n-2}+\ldots+b_1+b_0$$

若$S=0$，则无错；若$S=1$就认为有错。

- 以（7，4）码为例进行分析，可以设码字$A=[a_6,a_5,a_4,a_3,a_2,a_1,a_0]$，其中$[a_6,a_5,a_4,a_3]$为信息位，$[a_2,a_1,a_0]$为监督位，进而得到下面的方程组形式：

$$\begin{cases} a_6+a_5+a_4+a_2=0 \\ a_6+a_5+a_3+a_1=0 \\ a_6+a_4+a_3+a_0=0 \end{cases} \qquad \begin{cases} a_6+a_5+a_4=a_2 \\ a_6+a_5+a_3=a_1 \\ a_6+a_4+a_3=a_0 \end{cases}$$

不难看出，上述（7，4）码的最小码距$d_{min}=3$。

9.4.2 监督矩阵H和生成矩阵G

将（7，4）码的三个监督方程式可以重新改写为如下形式：

$$\begin{cases} 1\cdot a_6+1\cdot a_5+1\cdot a_4+0\cdot a_3+1\cdot a_2+0\cdot a_1+0\cdot a_0=0 \\ 1\cdot a_6+1\cdot a_5+0\cdot a_4+1\cdot a_3+0\cdot a_2+1\cdot a_1+0\cdot a_0=0 \\ 1\cdot a_6+0\cdot a_5+1\cdot a_4+1\cdot a_3+0\cdot a_2+0\cdot a_1+1\cdot a_0=0 \end{cases}$$

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 \end{bmatrix}^T = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

上式可以记作：$\boldsymbol{HA^T=0^T}$或$\boldsymbol{AH^T=0}$，其中

$$\boldsymbol{0}=\begin{bmatrix} 0 & 0 & 0 \end{bmatrix} \qquad \boldsymbol{A}=\begin{bmatrix} a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 \end{bmatrix}$$

$$\boldsymbol{H}=\begin{bmatrix}1&1&1&0&1&0&0\\1&1&0&1&0&1&0\\1&0&1&1&0&0&1\end{bmatrix}=\begin{bmatrix}\boldsymbol{P} & \boldsymbol{I}_r\end{bmatrix}$$

也可以用矩阵形式来表示：

$$\begin{bmatrix}a_2\\a_1\\a_0\end{bmatrix}=\begin{bmatrix}1&1&1&0\\1&1&0&1\\1&0&1&1\end{bmatrix}\cdot\begin{bmatrix}a_6\\a_5\\a_4\\a_3\end{bmatrix}$$

或表示成为：

$$\begin{bmatrix}a_2 & a_1 & a_0\end{bmatrix}=\begin{bmatrix}a_6 & a_5 & a_4 & a_3\end{bmatrix}\cdot\begin{bmatrix}1&1&1\\1&1&0\\1&0&1\\0&1&1\end{bmatrix}=\begin{bmatrix}a_6 & a_5 & a_4 & a_3\end{bmatrix}\cdot\boldsymbol{Q}$$

这时$\boldsymbol{Q}=\boldsymbol{P}^{\mathrm{T}}$，如果在$\boldsymbol{Q}$矩阵的左边在加上一个$\mathbf{k}\times\mathbf{k}$的单位矩阵，就形成了一个新矩阵$\boldsymbol{G}$：

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{I}_k & \boldsymbol{Q} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

这里G称为生成矩阵，利用它可以产生整个码组：

$$\boldsymbol{A} = \boldsymbol{M} \cdot \boldsymbol{G} = \begin{bmatrix} a_6 & a_5 & a_4 & a_3 \end{bmatrix} \cdot \boldsymbol{G}$$

9.4.3 校验子S

设发送组码$\boldsymbol{A}$，在传输过程中有可能出现误码，这时接收到的码组为$\boldsymbol{B}$。则收发码组之差为：

$$\boldsymbol{B}-\boldsymbol{A}=\begin{bmatrix}b_{n-1} & b_{n-2} & \cdots & b_0\end{bmatrix}-\begin{bmatrix}a_{n-1} & a_{n-2} & \cdots & a_0\end{bmatrix}$$

$$=\boldsymbol{E}=\begin{bmatrix}e_{n-1} & e_{n-2} & \cdots & e_0\end{bmatrix}$$

其中：

$$e_i=\begin{cases}0 & b_i=a_i\\ 1 & b_i\neq a_i\end{cases}$$

则接收端利用接收到的码组B计算校正子：

$$S=BH^{T}=(A+E)H^{T}=AH^{T}+EH^{T}=EH^{T}$$

因此，校正子仅与E有关，即错误图样与校正子之间有确定的关系。

- 汉明码就是一个线性分组码。有以下特点：

（1）最小码距$d_{min}=3$，可纠正一位错误；

（2）码长n与监督元个数r之间满足

$$n=2^{r}-1$$

- 9.1 引言
- 9.2 纠错编码的基本原理
- 9.3 常用的简单编码
- 9.4 线性分组码
- 9.5 循环码
- 9.6 卷积码
- 9.7 网格编码调制

9.5 循环码

9.5.1 循环码原理

- 循环码是线性分组码的一个重要子集，是目前研究得最成熟的一类码，它有许多特殊的代数性质。
- 特点：循环码中任一许用码组经过循环移位后，所得到的码组仍然是许用码组。 描述：许用循环码$A=(a_{n-1}\, a_{n-2} \ldots a_1\, a_0)$，可以将它的码多项式表示为：

$$A(x)=a_{n-1}x^{n-1}+a_{n-2}x^{n-2}+\cdots+a_1x+a_0$$

若一个整数m可以表示为：

$$\frac{m}{n}=Q+\frac{p}{n} \qquad p<n \qquad Q\text{是整数}$$

则在模n运算下，有$m\equiv p$（模n），同样对于多项式而言：

$$\frac{F(x)}{N(x)}=Q(x)+\frac{R(x)}{N(x)}$$

则可以写为：$F(x)\equiv R(x)$ （模$N(x)$）。

在循环码中，若$A(x)$是一个长为n的许用码组，则在按模 x运算下，亦是一个许用码组。

$$x^3\cdot A_7(x)=x^3\cdot\left(x^6+x^5+x^2+1\right)=\left(x^9+x^8+x^5+x^3\right)$$
$$=\left(x^5+x^3+x^2+x\right) \qquad \left(\text{模}x^7+1\right)$$

➢循环码中次数最低的码多项式称为生成多项式，用$g(x)$表示。可以证明生成多项式$g(x)$具有以下特性：

（1） $g(x)$是一个常数项为1的$r=n-k$次多项式；

（2） $g(x)$是x^n+1的一个因式；

（3）该循环码中其它码多项式都是$g(x)$的倍式。

为了保证构成的生成矩阵G的各行线性不相关，通常用$g(x)$来构造生成矩阵，

因此，一旦生成多项式$g(x)$确定以后，该循环码的生成矩阵就可以确定。

$$G(x)=\begin{bmatrix} x^{k-1}\cdot g(x) \\ x^{k-2}\cdot g(x) \\ \vdots \\ x\cdot g(x) \\ g(x) \end{bmatrix}$$

$$g(x)=x^{r}+a_{r-1}x^{r-1}+\cdots+a_{1}x+1$$

显然，上式不符合$G=[I_k \quad Q]$形式，所以此生成矩阵不是典型形式。

利用循环码的特点来确定监督矩阵H：

由于（n,k）循环码中$g(x)$是x^n+1的因式，因此可令：$h(x)=\dfrac{x^n+1}{g(x)}=x^k+h_{k-1}x^{k-1}+\cdots+h_1x+1$

监督矩阵表示为：

$$H(x)=\begin{bmatrix} x^{n-k-1}\cdot h^*(x) \\ x^{n-k-2}\cdot h^*(x) \\ \vdots \\ x\cdot h^*(x) \\ h^*(x) \end{bmatrix}$$

其中：

$$h^*(x)=x^k+h_1x^{k-1}+h_2x^{k-2}+\cdots+h_{k-1}x+1$$

9.5.2 循环码的编解码方法

1、编码过程

首先需要根据给定循环码的参数确定生成多项式$g(x)$，然后，利用循环码的编码特点，即所有循环码多项式$A(x)$都可以被$g(x)$整除，来定义生成多项式$A(x)$。下面就将以上各步处理加以解释：

（1）用x^{n-k}乘$m(x)$。这一运算实际上是把信息码后附加上（n-k）个“0”。

（2）求$R(x)$。由于循环码多项式$A(x)$都可以被$g(x)$整除，也就是：

$$\frac{A(x)}{g(x)} = Q(x) = \frac{x^{n-k} \cdot m(x) + r(x)}{g(x)} = \frac{x^{n-k} \cdot m(x)}{g(x)} + \frac{R(x)}{g(x)}$$

上式也等效于：

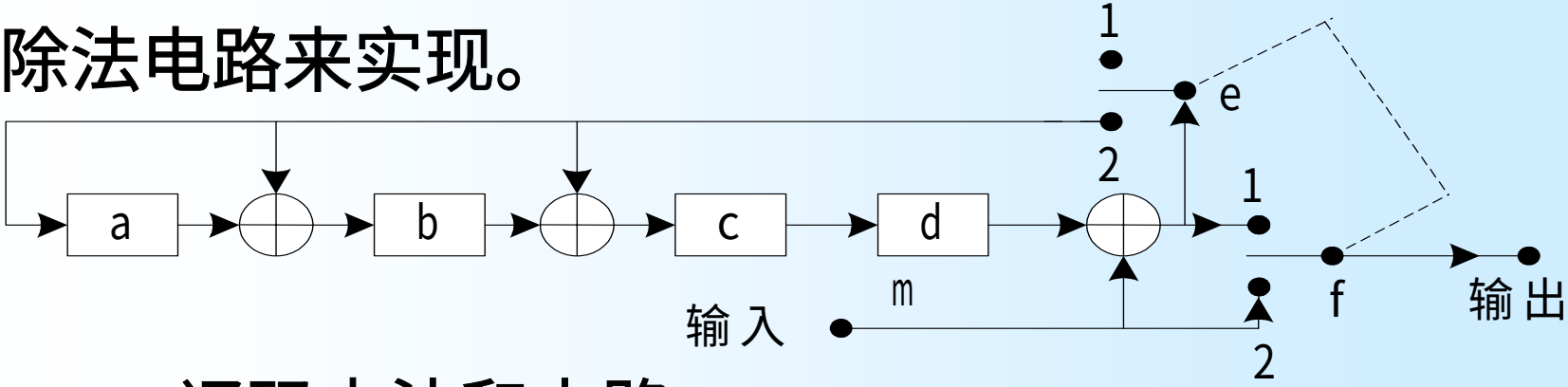
$$\frac{x^{n-k} \cdot m(x)}{g(x)} = Q(x) + \frac{R(x)}{g(x)}$$

这样我们就得到了$R(x)$。

（3）编码输出系统循环码多项式$A(x)$为：

$$A(x) = x^{n-k} \cdot m(x) + R(x)$$

上述三步编码过程，在硬件实现时，可以利用除法电路来实现。



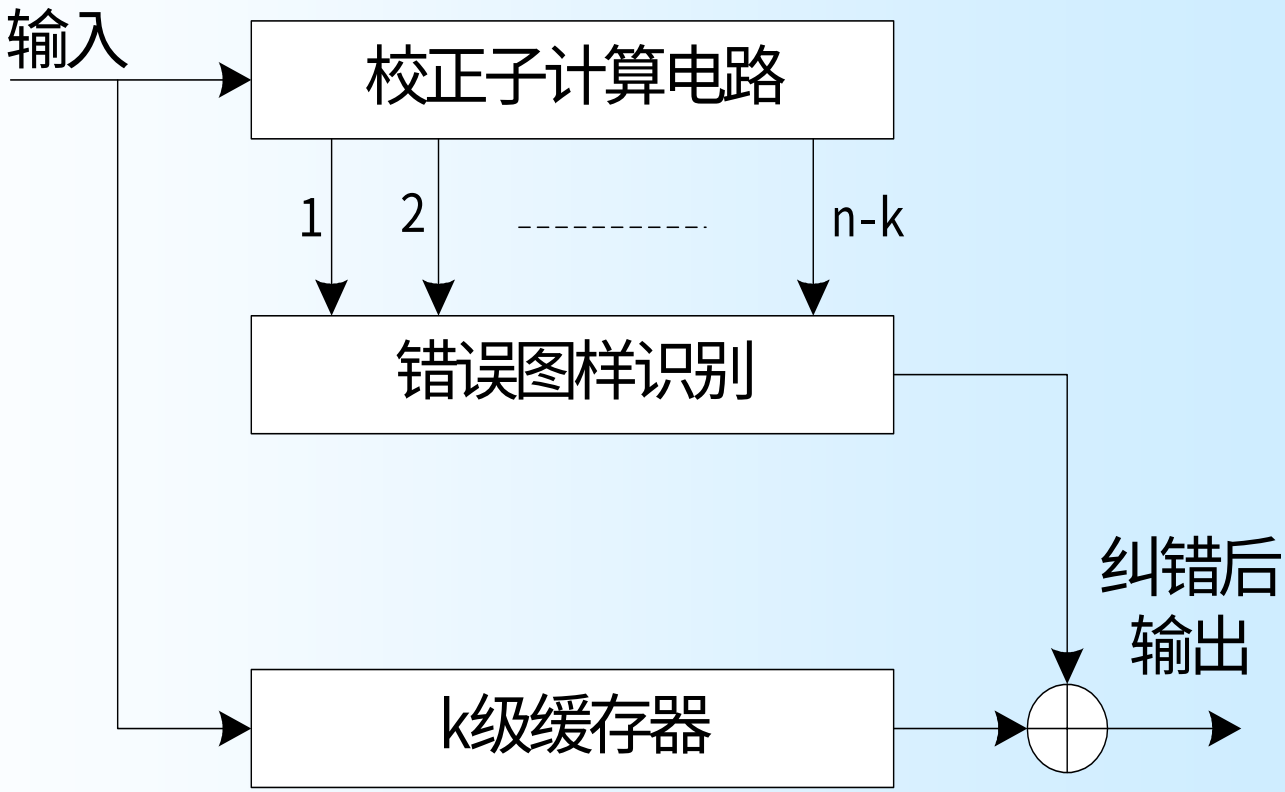
2、 译码方法和电路

- 循环码的译码可以分三步进行：

（1）由接收到的码多项式$B(x)$计算校正子（伴随式）多项式$S(x)$；

（2）由校正子$S(x)$确定错误图样$E(x)$；

（3）将错误图样$E(x)$与$B(x)$相加，纠正错误。



9.5.4 缩短循环码

- 给定一（n，k）循环码组集合，使前i（0<i<k）个高阶信息数字为0，于是得到有 2^k 码组的集合，然后从这些码组中删除这i个零信息位数字，最终得到一种新的（n-i,k-i）的线性码，就称为缩短循环码。

9.5.4 BCH码

- 特点：它的生成多项式$g(x)$与最小码距之间有密切的关系，可以根据所要求的纠错能力t，很容易地构造出BCH码。

相关知识：本原多项式的定义：

(1) $f(t)$为既约多项式；

(2) $f(t)$是(x^p+1)因子，$p=2^n-1$

(3) $f(t)$不是(x^q+1)的因子， $p>q$

9.5.5 RS码

- RS码是q进制BCH码的一个特殊子类，并且具有很强的纠错能力。
  - RS码的参数：码长$n = q - 1$，监督位数目$r = 2t$，其中t是能够纠正的错码数目；其生成多项式为

$$g(x) = (x + \alpha)(x + \alpha 2) \ldots (x + \alpha 2t)$$

    式中，α为伽罗华域GF(2^m)中的本原元。
  - RS码的主要优点：
    - 它是多进制纠错编码，所以特别适合用于多进制调制的场合；
    - 它能够纠正t个q位二进制错码，即能够纠正不超过q个连续的二进制错码，所以适合在衰落信道中纠正突发性错码。

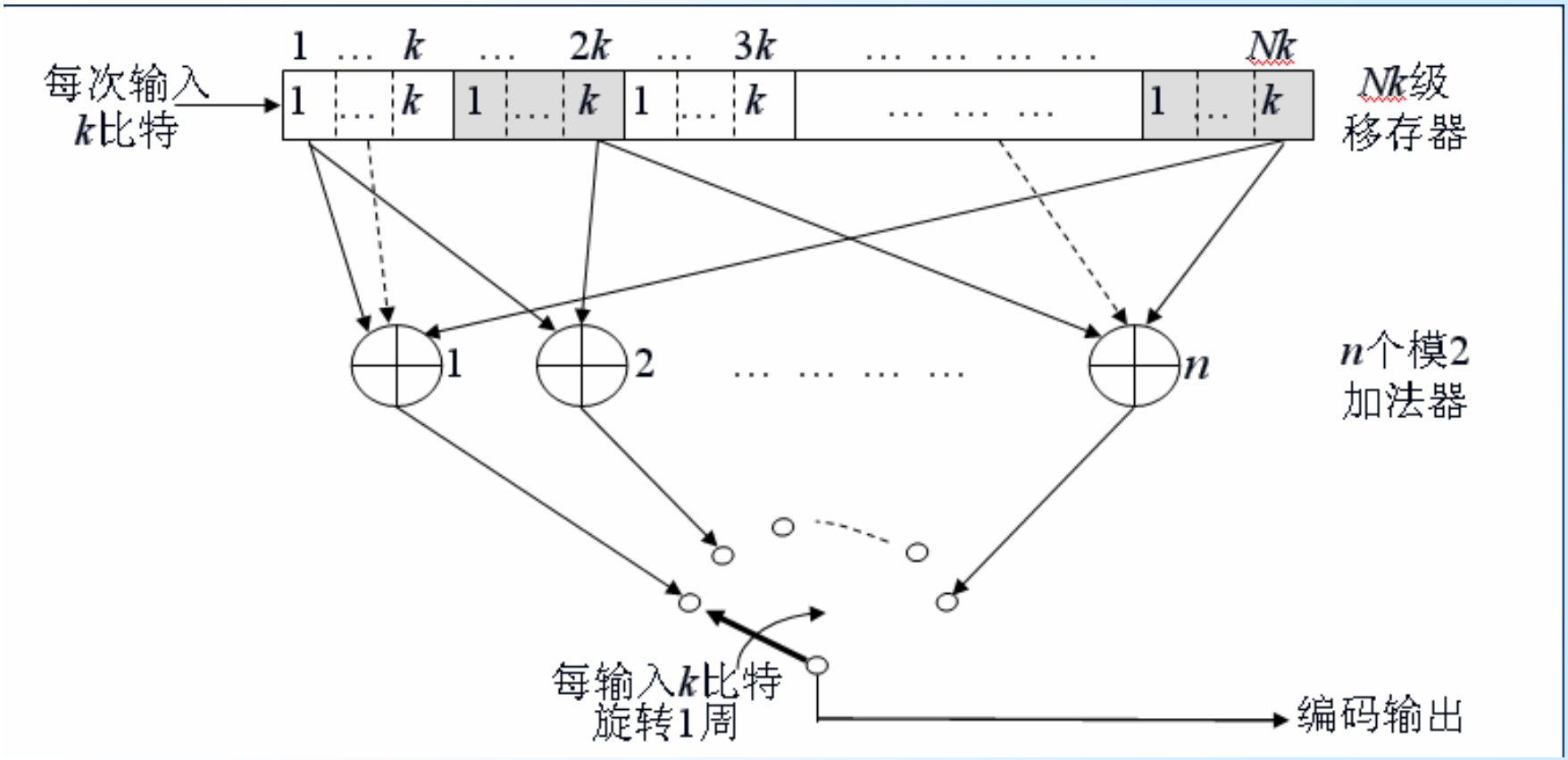
- 9.1 引言
- 9.2 纠错编码的基本原理
- 9.3 常用的简单编码
- 9.4 线性分组码
- 9.5 循环码
- 9.6 卷积码
- 9.7 网格编码调制

9.6 卷积码

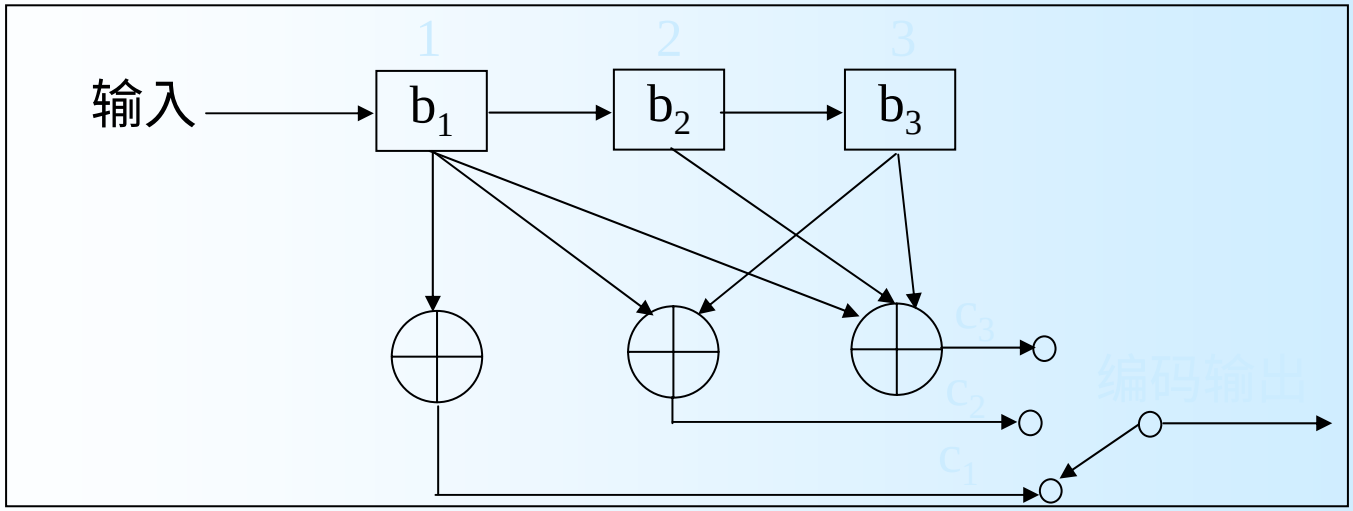
- 卷积码的特点：
  - 监督码元不仅和当前的k比特信息段有关，而且还同前面$m = (N - 1)$个信息段有关。
  - 将N称为码组的约束度。
  - 将卷积码记作(n, k, m)，其码率为k/n。

卷积码的编码

- 一般原理方框图



- 卷积码编码器的实例方框图：(n, k, m) = （3, 1, 2）



– 每当输入1比特时，此编码器输出3比特$c_1c_2\,c_3$：

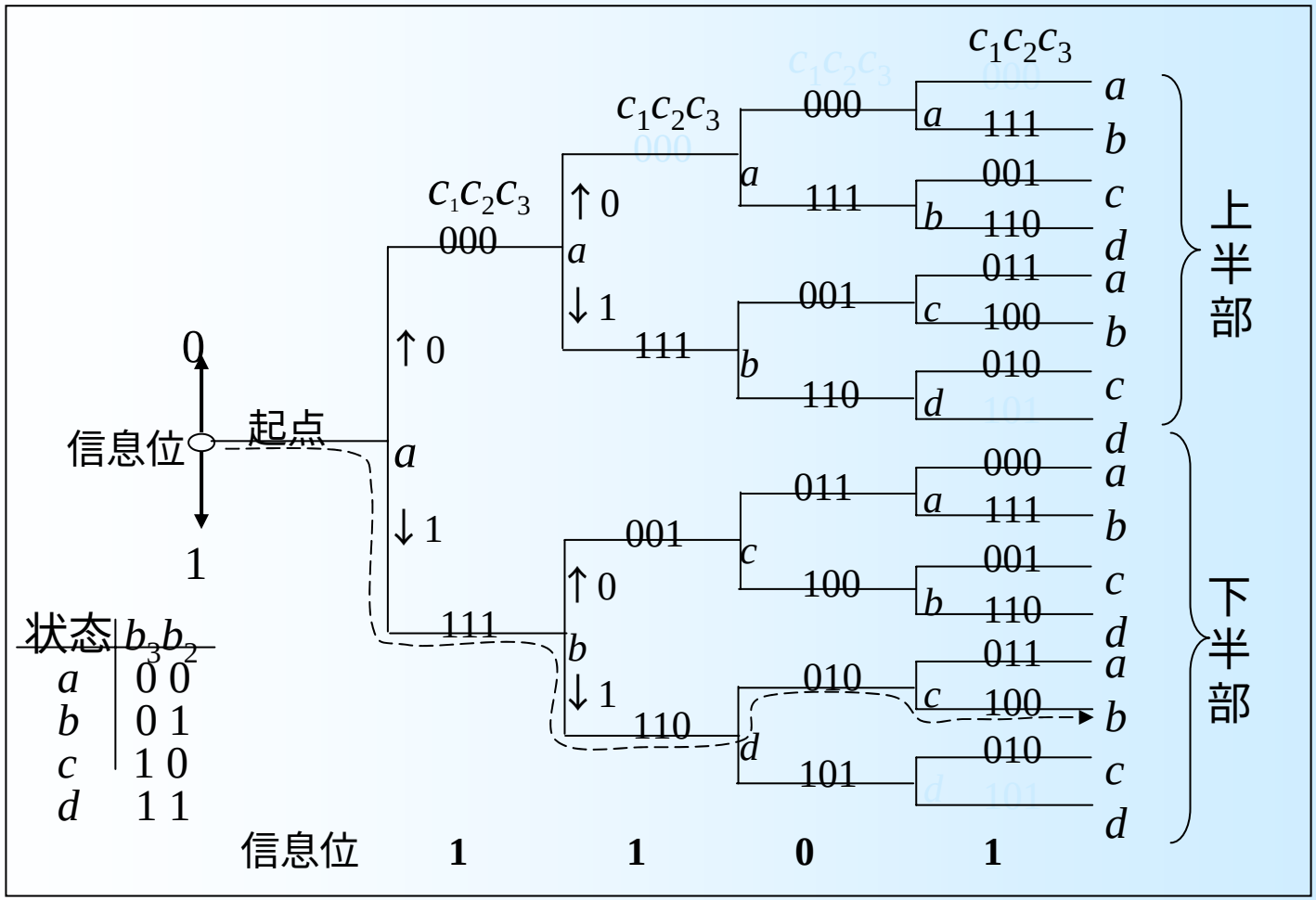
$$c_1 = b_1$$
$$c_2 = b_1 \oplus b_3$$
$$c_3 = b_1 \oplus b_2 \oplus b_3$$

– 编码器的工作状态

b_1	1	1	0	1	0	0	0
b_3b_2	00	01	11	10	01	10	00
$c_1c_2\,c_3$	111	110	010	100	001	011	000
状态	*a*	*b*	*d*	*c*	*b*	*c*	*a*

9.6.2 卷积码的解码

- 码树搜索法：(3, 1, 2)卷积码的码树图



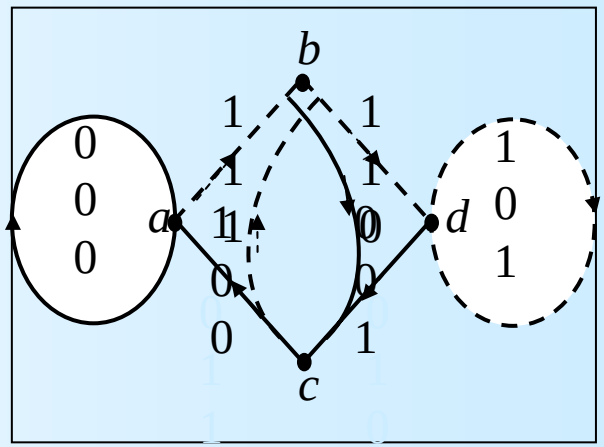
- 此法不实用：因为随信息位增多，分支数目按指数规律增长

状态图和网格图

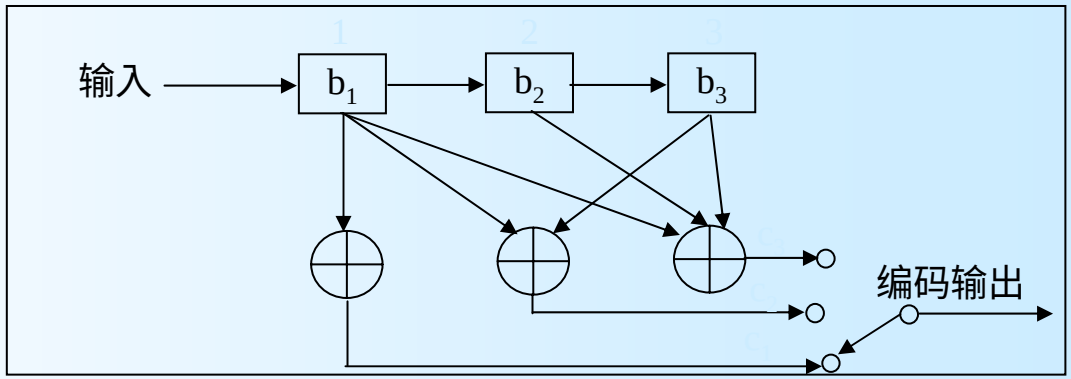
- 移存器状态和输入输出码元的关系

前一状态 $b_3\,b_2$	当前输入 b_1	输出 $c_1c_2c_3$	下一状态 $b_3\,b_2$
a (00)	0	000	*a* (00)
	1	111	*b* (01)
b (01)	0	001	*c* (10)
	1	110	*d* (11)
c (10)	0	011	*a* (00)
	1	100	*b* (01)
d (11)	0	010	*c* (10)
	1	101	*d* (11)

$$c_1 = b_1$$
$$c_2 = b_1 \oplus b_3$$
$$c_3 = b_1 \oplus b_2 \oplus b_3$$

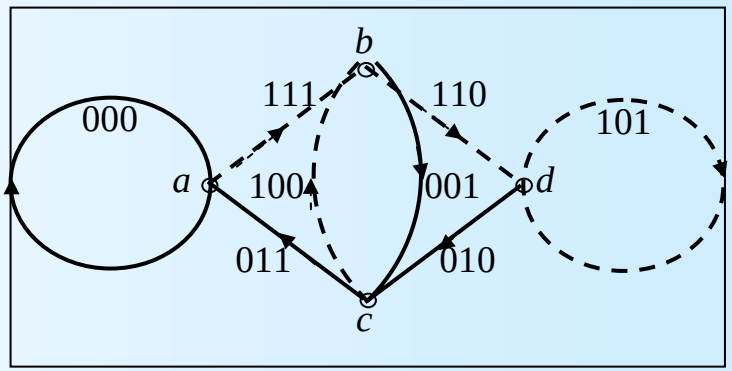


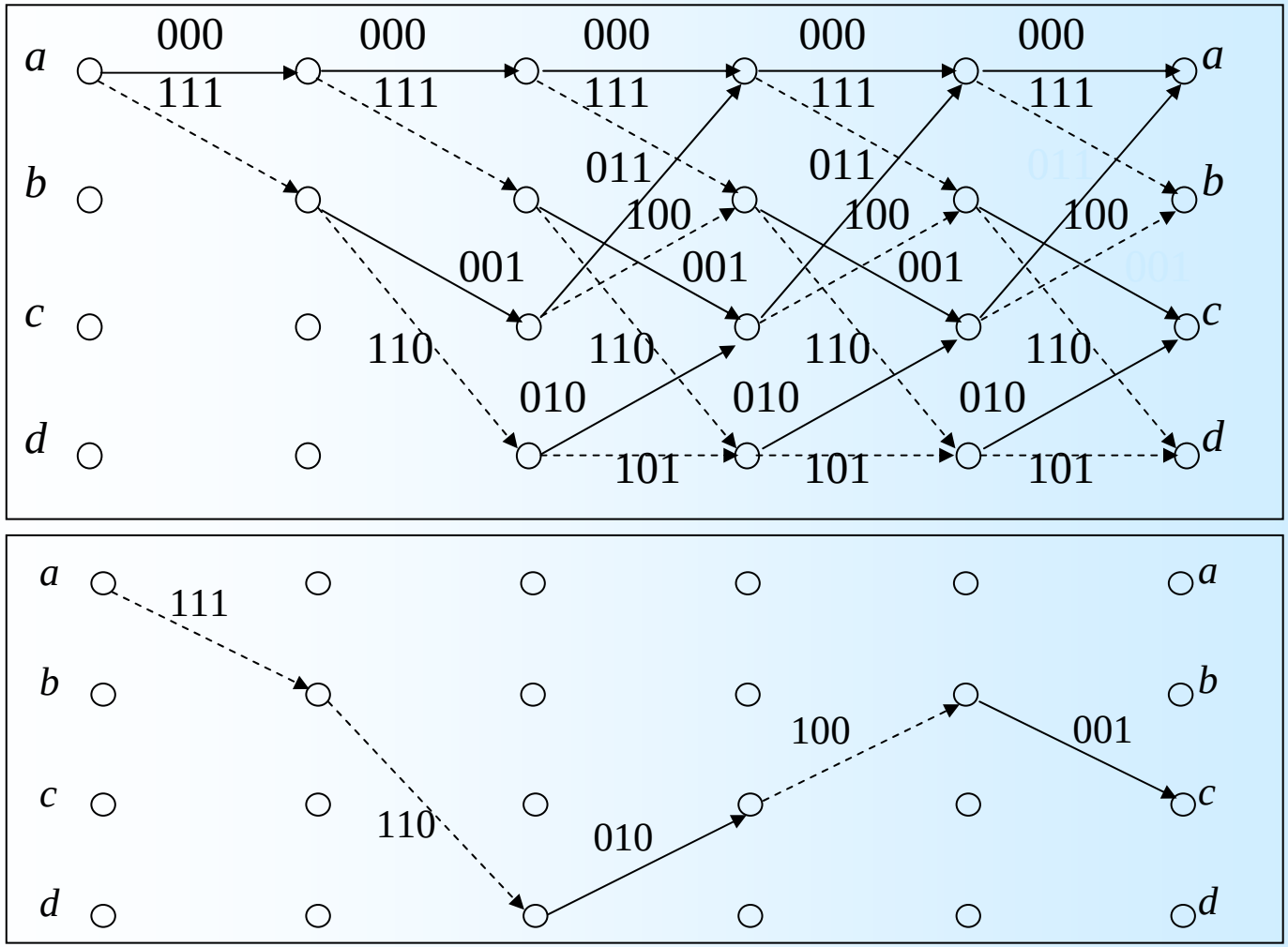
- (3, 1, 2)卷积码网格图



- 网格图中的编码路径举例
  - 输入信息位为1101时
  - 输出编码序列是：

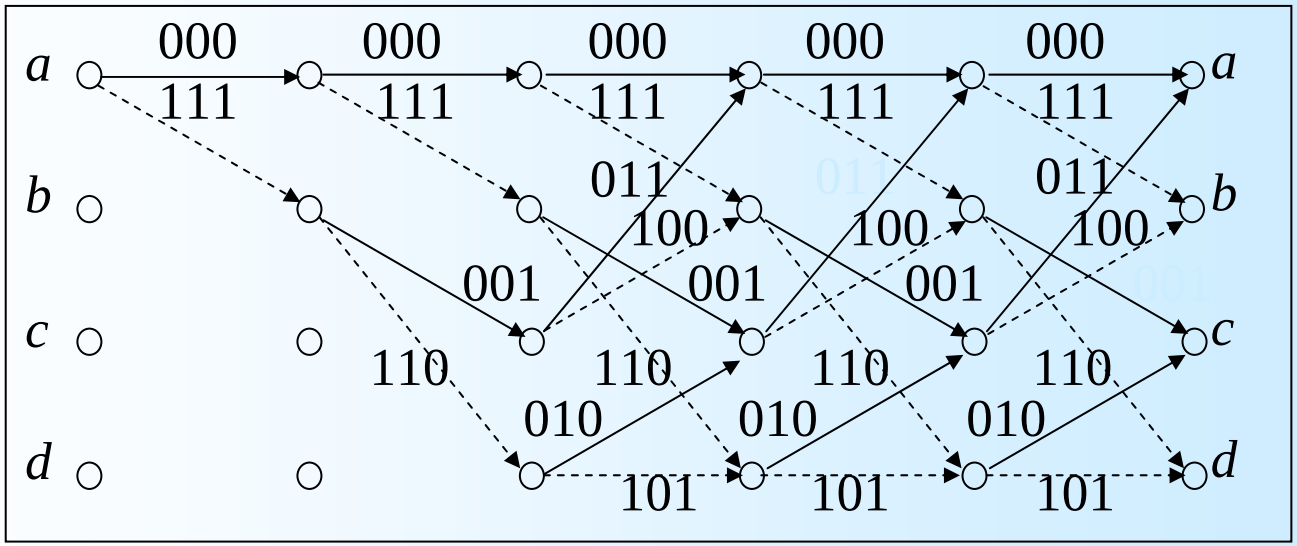
    111 110 010 100 011...




a
b
c
d
000
111
011
100
001
110
010
101
a
b
c
d
111
110
010
100
001

- 维特比算法
  - 基本原理：将接收到的序列和所有可能的发送序列作比较，选择其中汉明距离最小的序列当作是现在的发送序列
  - 例：设卷积码为$(n, k, m) = (3, 1, 2)$码
    - 现在的发送信息位为1101
    - 为了使移存器中的信息位全部移出，在信息位后面加入了3个“0”，即1101000
    - 编码后的发送序列：111 110 010 100 001 011 000
    - 接收序列：111 010 010 110 001 011 000 **(红色为错码)**
  - 由于这是一个 (3, 1, 2)卷积码，发送序列的约束长度为$N = m + 1 = 3$，所以首先需考察3个信息段，即考察$3n = 9$比特，即接收序列前9位“111 010 010”。

- 解码第1步
  - 由网格图可见，沿路径每一级有4种状态a, b, c和d。每种状态只有两条路径可以到达。故4种状态共有8条到达路径。
  - 比较网格图中的这8条路径和接收序列之间的汉明距离。例如，由出发点状态a经过3级路径后到达状态a的两条路径中上面一条为“000 000 000”。它和接收序列“111 010 010”的汉明距离等于5；下面一条为“111 001 011”,它和接收序列的汉明距离等于3。



- 将这8个比较结果列表如下：

序号	路径	对应序列	汉明距离	幸存否？
1	*aaaa*	000 000 000	5	否
2	*abca*	111 001 011	3	是
3	*aaab*	000 000 111	6	否
4	*abcb*	111 001 100	4	是
5	*aabc*	000 111 001	7	否
6	*abdc*	111 110 010	1	是
7	*aabd*	000 111 110	6	否
8	*abdd*	111 110 101	4	是

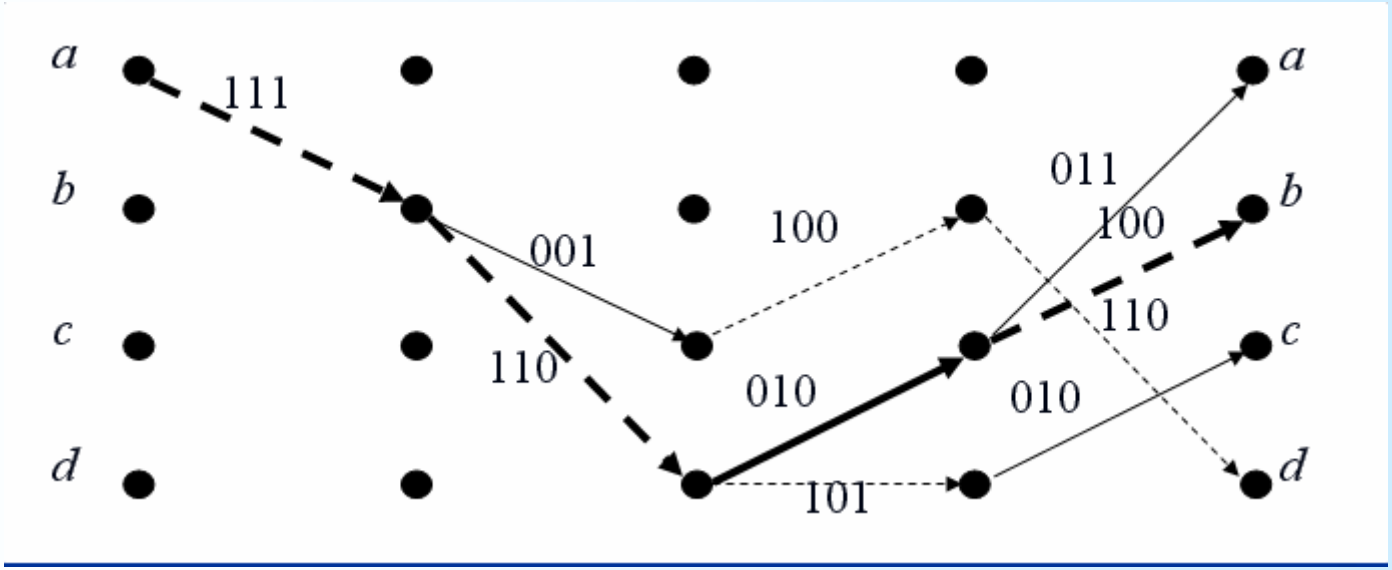
- 比较到达每个状态的两条路径的汉明距离，将距离小的一条路径保留，称为幸存路径。这样，就剩下4条路径了，即表中第2, 4, 6和8条路径。

- 解码第2步：继续考察接收序列中的后继3个比特“110”

计算4条幸存路径上增加1级后的8条可能路径的汉明距离。计算结果列于下表中。表中总距离最小为2，其路径是abdc+b，相应序列为111 110 010 100。它和发送序列相同，故对应发送信息位1101。

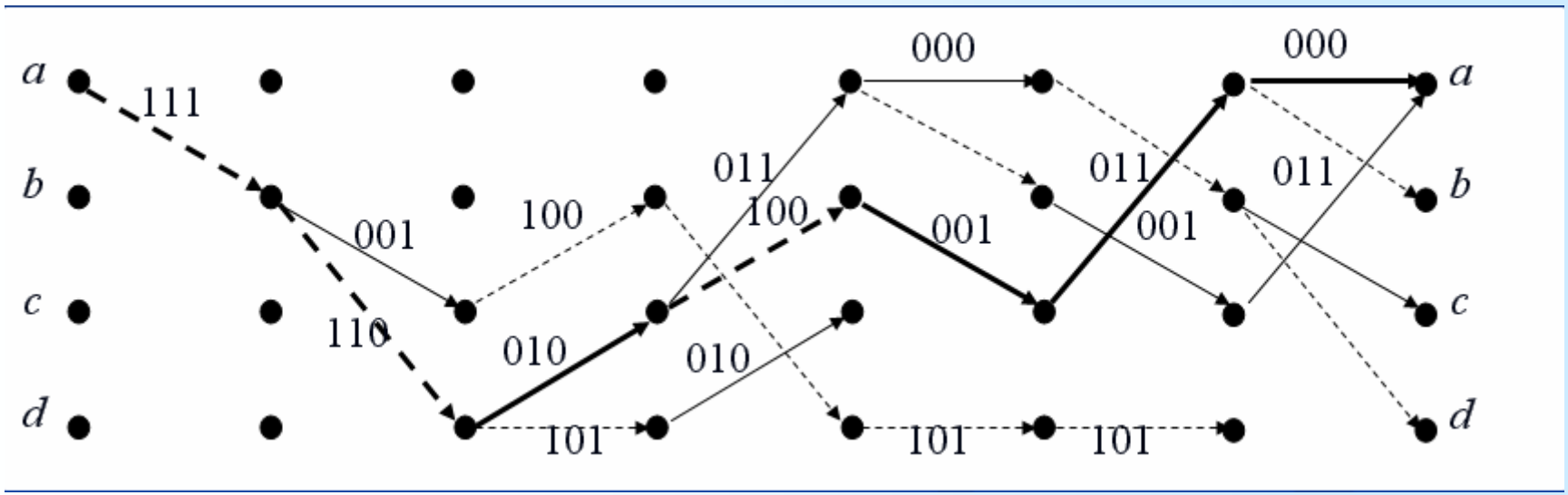
序号	路径	原幸存路径的距离	新增路径段	新增距离	总距离	幸存否？
1	*abca+a*	3	*aa*	2	5	否
2	*abdc+a*	1	*ca*	2	3	是
3	*abca+b*	3	*ab*	1	4	否
4	*abdc+b*	1	*cb*	1	2	是
5	*abcb+c*	4	*bc*	3	7	否
6	*abdd+c*	4	*dc*	1	5	是
7	*abcb+d*	4	*bd*	0	4	是
8	*abdd+d*	4	*dd*	2	6	否

– 按照上表中的幸存路径画出的网格图示于下图中。

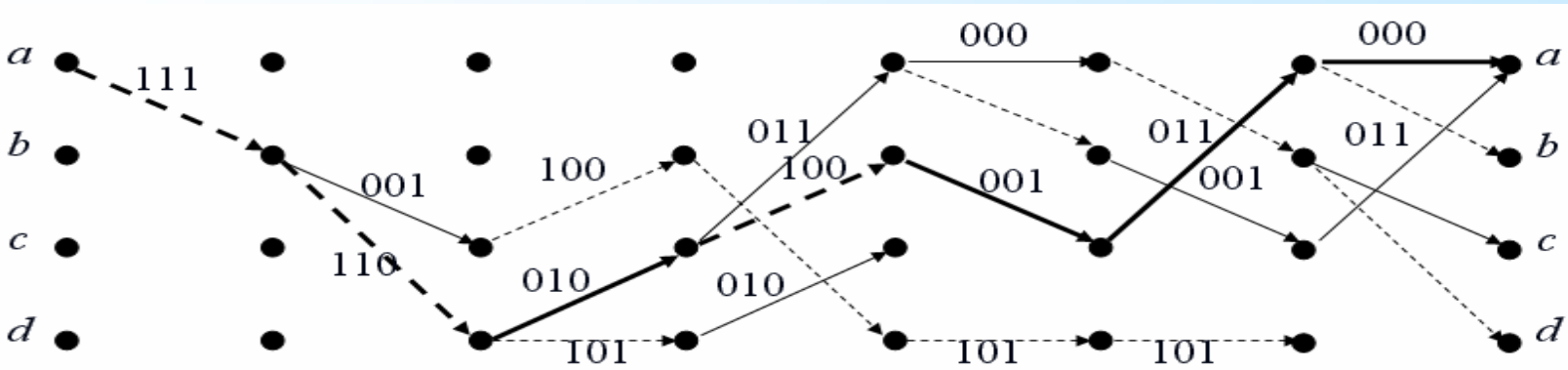


– 图中粗线路径是距汉明离最小（等于2）的路径。

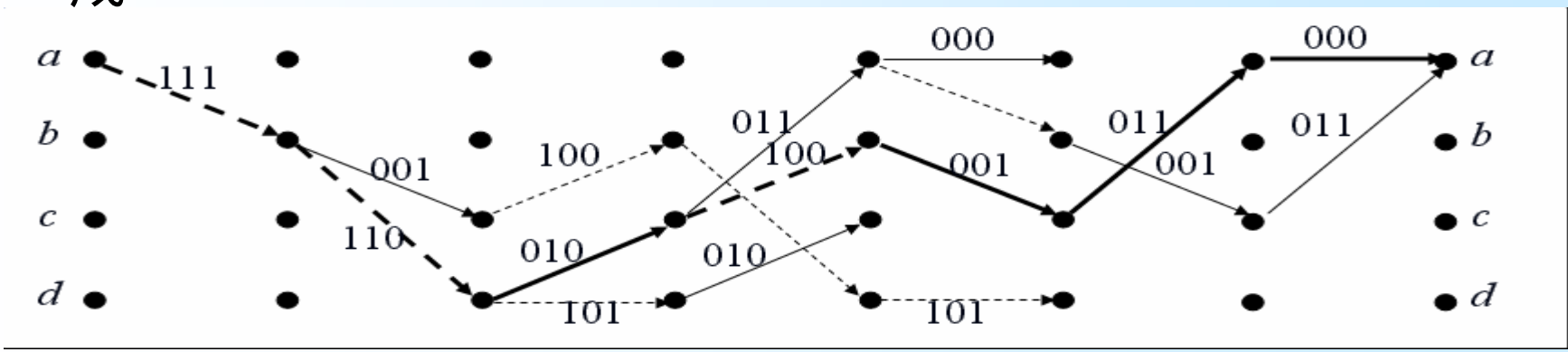
- 在编码时，信息位后面加了3个“0”。若把这3个“0”仍然看作是信息位，则可以按照上述算法继续解码。这样得到的幸存路径网格图示于下图中。图中的粗线仍然是汉明距离最小的路径。



- 若已知这3个码元是（为结尾而补充的）“0”，则在解码时就预先知道在接收这3个“0”码元后，路径必然应该回到状态*a*。而由图可见，



- 只有两条路径可以回到a状态。所以，这时上图可以简化成：



- 在上例中卷积码的约束长度为N = 3，需要存储和计算8条路径的参量。
- 由此可见，维特比算法的复杂度随约束长度N按指数形式2^N增长。故维特比算法适合约束长度较小（$N \leq 10$）的编码。对于约束长度大的卷积码，可以采用其他解码算法，