

信息量

$$I_A = \log_2 \frac{1}{P_A} = -\log_2 P_A$$

对于一个概率为 P_A 的事件 A , 其信息量可以用公式 $I_A = \log \frac{1}{P_A}$ 或 $I_A = -\log P_A$ 来表示。这里的
信息量表示的是事件 A 发生所提供的信息量的大小。当 P_A 越小 (即事件发生的可能性越小), I_A
的值就越大, 表示该事件的发生提供的信息量越多。

For a binary (digital) system we use base 2, i.e. log2, giving information in bits

平均信息熵

$$H = \sum_{i=0}^{N-1} P_i \log_2 \frac{1}{p_i}$$

该公式是离散无记忆源的平均信息熵的计算公式。在离散无记忆源中, 有 N 个可能的事件, 每个事件
发生的概率为 P_i , 该公式计算了每个事件发生时所传递的平均信息量, 其单位为比特 (bit) 。

其中:

\log_2 表示以 2 为底的对数,

P_i 表示事件 i 发生的概率,

$1/P_i$ 表示发生事件 i 所传递的信息量,

H 表示所有事件所传递的平均信息量。

该公式中求和符号表示对所有可能的事件的信息量进行求和。

- DNA is a molecule which contains the genetic information for life. The information is stored as a sequence of 4 possible nucleotides (bases): Adenine (A), Cytosine (C), Thymine (T) and Guanine (G), all with equal probability $P_i = 1/4$.
- The complete genome (all the genetic DNA) for E coli (a form of bacteria) has 4×10^6 of these bases. What is the information content?

$$I_{E\text{ Coli}} = 4 \times 10^6 \times \log_2 4 = 8 \times 10^6 \text{ bit}$$

- The complete genome for a human being has approximately 3.2×10^9 of these bases. What is the information content?

$$I_{\text{human}} = 3.2 \times 10^9 \times \log_2 4 = 6.4 \times 10^9 \text{ bit}$$

a single-layer DVD-ROM has a capacity of $4.7 \text{ GB} = 38 \times 10^9 \text{ bit}$

Source Entropy

源熵是指离散随机变量的不确定性度量，表示一个信息源（信息发射器）所提供的信息量的平均值。源熵越大，意味着源提供的信息越不确定，即每个符号携带的信息量越多，需要更多的信息才能准确描述该源的行为。源熵的单位是比特（bit）或香农（Shannon，缩写为Sh），表示对该离散随机变量进行编码时，需要多少比特的信息量来表示每个符号。

Source Entropy

Example: the 4 symbols A, B, C and D form a complete set occurring with probabilities $1/2, 1/4, 1/8$ and $1/8$ respectively.

1. The information in the 3 symbol message $X = BDA$ (assuming that the symbols are statistically independent) is

$$I_X = \log_2 4 + \log_2 8 + \log_2 2 = 2 + 3 + 1 = 6 \text{ bit}$$

2. The source entropy for this set is

$$H = \frac{1}{2} \log_2 2 + \frac{1}{4} \log_2 4 + \frac{1}{8} \log_2 8 + \frac{1}{8} \log_2 8 = \frac{1}{2} + \frac{2}{4} + \frac{3}{8} + \frac{3}{8} = 1.75 \text{ bit}$$

编码

Huffman Code

这个可能会考需要掌握

[Huffman Coding](#)

Huffman编码是一种常用的无损数据压缩的最优前缀编码方法。

1. 按概率从大到小对源输出进行排序。
2. 将两个概率最小的合并成一个输出，其概率为它们的概率之和。
3. 如果剩余的输出数量>2，则返回步骤1。
4. 任意将0和1分配为两个剩余输出的编码。
5. 如果一个输出对应于一个合并输出，则将当前编码单词添加0或1，然后重复，直到所有原始输出都分配了编码。

例如：四个符号A、B、C和D，它们以 $1/2$ 、 $1/4$ 、 $1/8$ 和 $1/8$ 的概率分别出现，形成完整的集合。

如果使用概率分别为0.4、0.3、0.2和0.1，我们将得到相同的结果，但是由于概率不再是理想的 $2-n$ ，效率提高到了1.9比特/符号。

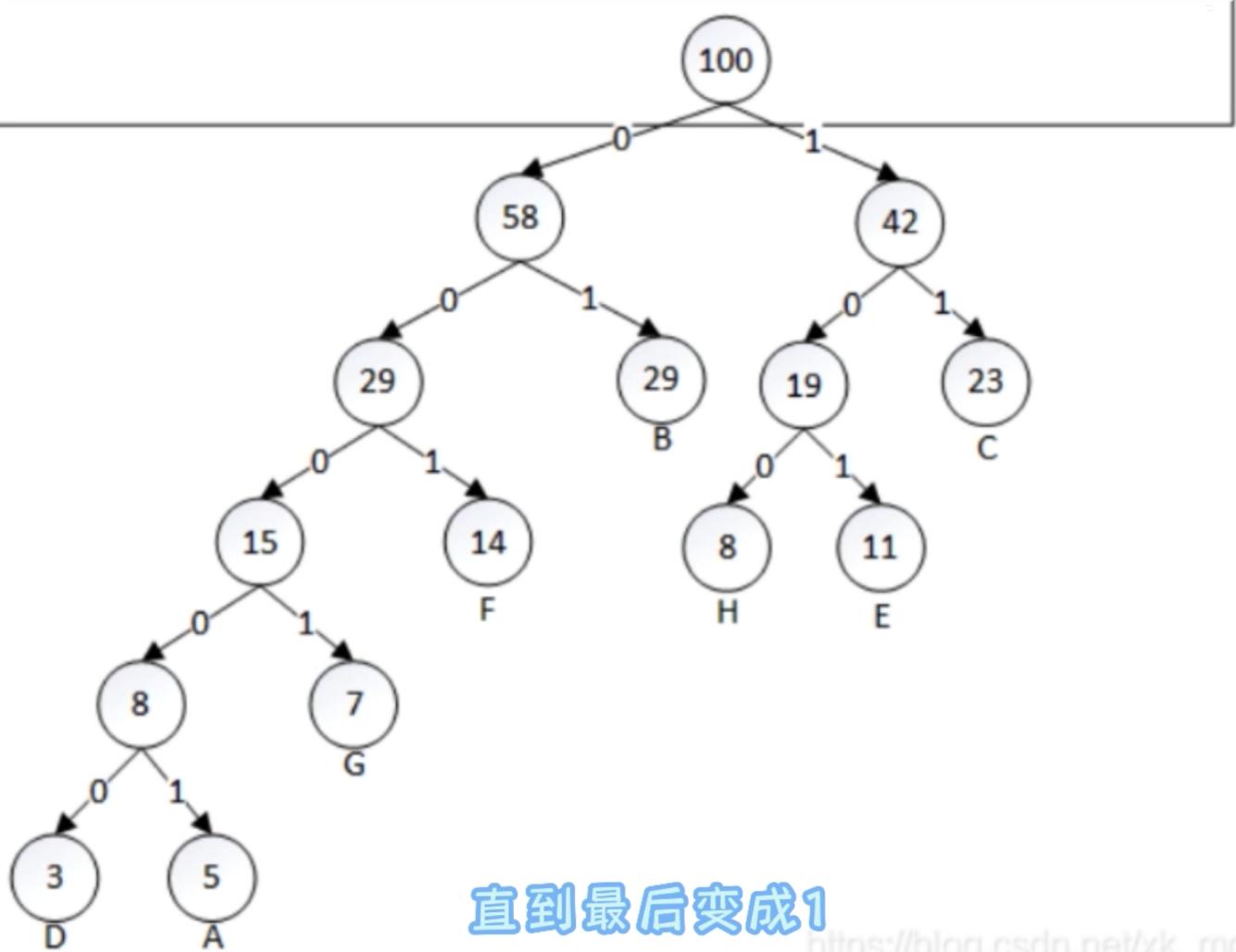
简单来说，Huffman编码利用符号出现的概率来构造短的二进制码字，以实现对数据的压缩。出现概率较高的符号被赋予较短的编码，而出现概率较低的符号被赋予较长的编码。在解压缩时，根据编码表将压缩后的数据转换回原始数据。该编码方法通常用于无损压缩，因为编码后可以完全恢复原始数据，不会有任何信息丢失。

记住，先加上最小的！

赫夫曼编码的具体方法：先按出现的概率大小排队，把两个最小的概率相加，作为新的概率和剩余的概率重新排队，再把最小的两个概率相加，再重新排队，直到最后变成1

例子：

3 5 7 8 11 14 23 29



直到最后变成1

https://blog.csdn.net/xk_mov

Lempel-Ziv-Welch (LZW) coding

LZW压缩编码

这个可能会考需要掌握

Lempel-Ziv-Welch (LZW) 编码是一种通用的无损数据压缩算法。它避免了Huffman编码中需要预先知道源概率的问题。其编码算法的高级视图如下：

1. 初始化字典以包含所有长度为一的字符串。
2. 查找字典中与当前输入匹配的最长字符串W。
3. 将W的字典索引发射到输出并从输入中删除W。
4. 将W后跟输入的下一个符号添加到字典中。
5. 转到步骤2。

解码相对简单，因为解码器构建的扩展字典与编码器相同。

Lec2

$$M = \frac{\sqrt{S+N}}{\sqrt{N}} = \sqrt{1 + \frac{S}{N}}$$

在公式 $M = \sqrt{1 + \frac{S}{N}}$ 中， M 表示在存在噪声的情况下，量化器可以分辨的不同电平的数量，也可以理解为量化器的分辨率。其中， S 表示信号功率， N 表示噪声功率。

Therefore the maximum entropy H is

$$H_{max} = \sum_{i=0}^{M-1} \frac{1}{M} \log_2 M = \log_2 M = \frac{1}{2} \log_2 \left(1 + \frac{S}{N}\right)$$

$$C_{max} = R_{samp} H_{max} = B \log_2 \left(1 + \frac{S}{N}\right)$$

香农定理

$$C_{max} = B \log_2 \left(1 + \frac{S}{N}\right)$$

$\frac{S}{N}$ 表示的就是SNR,信噪比.不是简单的Signal 除以 Noise.

$\frac{S}{N}$ 的单位为dB. 与简单的S除以N的换算关系为 $SNR = \frac{S}{N} = 10 \lg \left(\frac{P_S}{P_N}\right)$, 注意是以10为底

Tutorial_1

1.

- (a) Assuming that letters occur with equal probability, calculate the information (in bits) for the message “the quick brown fox jumps over the lazy dog” (ignore spaces). You may find the following identity useful if you are using a calculator:

$$\log_2 x = (\log_{10} x) / (\log_{10} 2)$$

- (b) The table shows the probability distribution of letters in the English alphabet. How do you think these values are arrived at?

k	x_k	$P(x_k)$	k	x_k	$P(x_k)$	k	x_k	$P(x_k)$
1	a	8.167%	10	j	0.153%	19	s	6.327%
2	b	1.492%	11	k	0.772%	20	t	9.056%
3	c	2.782%	12	l	4.025%	21	u	2.758%
4	d	4.253%	13	m	2.406%	22	v	0.978%
5	e	12.702%	14	n	6.749%	23	w	2.360%
6	f	2.228%	15	o	7.507%	24	x	0.150%
7	g	2.015%	16	p	1.929%	25	y	1.974%
8	h	6.094%	17	q	0.095%	26	z	0.074%
9	i	6.966%	18	r	5.987%			

- (c) Using this probability distribution, calculate the information in the given message.
(d) Again using this probability distribution, calculate the entropy of the English alphabet.
(e) Is this a true Discrete Memoryless Source (DMS)?

2. Two six-sided dice are thrown and the numbers shown are added together.

- (a) Is this a true Discrete Memoryless Source (DMS)?
(b) Calculate the probabilities for each sum total.
(c) Hence, determine the information (in bits) for each sum total.
(d) Hence, calculate the entropy for the symbol set consisting of the sum totals.
(e) Devise a Huffman code for this alphabet.

3. Repeat the previous question, but now take the absolute value of the difference between the two thrown dice.

4. The LZW algorithm was used in the lecture notes to compress the first line of the following tongue-twister. Continue with applying the LZW source coding to the rest of the verse. Remember the space is a character in the initial dictionary.

she sells sea shells on the sea shore
the shells she sells are sea shells im sure
and if she sells sea shells on the sea shore
then im sure she sells seashore shells

What compression ratio did you achieve? Check your answer by attempting to decode your result.

$$\log_2 x = (\log_{10} x) / (\log_{10} 2)$$

1(a)

要计算给定消息的信息量（以比特为单位），我们可以使用以下公式：

$$\text{信息量} = -\log_2(P)$$

其中，P是字符出现的概率。假设字母出现的概率是相等的，英文字母有26个，所以每个字母出现的概率是 $1/26$ 。

给定的消息是：“the quick brown fox jumps over the lazy dog”。忽略空格，该消息包含35个字符。要计算整个消息的信息量，我们需要将每个字符的信息量相加。

$$I_A = \log_2 \frac{1}{P_A} = -\log_2 P_A$$

$$\text{每个字符消息量 } \log_2 \left(\frac{1}{26} \right)$$

整个消息的信息量就是：字符数 * 每个字符携带的信息量

$$35 * \log_2 \left(\frac{1}{26} \right)$$

1(b)

这些值是通过对英文文本的统计分析获得的。可以通过对大量英文文本进行扫描，统计每个字母在文本中出现的频率来计算这些值。然后，这些频率可以归一化，得到每个字母的概率分布。这种方法是一种常见的语言模型建立方法，

These values are obtained through statistical analysis of English text. By scanning a large amount of English text and counting the frequency of each letter appearing in the text, these values can be calculated. Then, these frequencies can be normalized to obtain the probability distribution of each letter.

(c)

和a一样,只不过每个字母的概率变了.

以字母x为例子:

先统计x出现的次数为3,然后 $3 * -\log_2(P)$,然后每个字母都这么做,再加起来.

(d)

计算字母表的熵

$H = \sum_{i=0}^{N-1} P_i \log_2 \frac{1}{p_i}$,用这个公式把字母表过一遍就行了.

(e)

不是。离散无记忆源 (DMS) 指的是一个符号的出现与之前的符号无关。然而，在自然语言（如英语）中，字母并非独立出现。它们通常以特定的组合（如单词）出现，而某些字母组合的概率要高于其他组合。因此，英语并不是一个真正的离散无记忆源。

8.323% * 2



两个六面骰子被掷出，显示的数字相加。

- (a) 这是一个真正的离散无记忆源 (DMS) 吗？
- (b) 计算每个总和的概率。
- (c) 因此，确定每个总和的信息量 (以比特为单位)。
- (d) 因此，计算由总和组成的符号集的熵。
- (e) 为这个字母表设计一个哈夫曼编码。

(a) 是的，这是一个真正的离散无记忆源 (DMS)，因为每次掷骰子的结果与之前的结果无关。

(b) 两个六面骰子的和可能在 2 到 12 之间。计算每个总和的概率如下：

$$P(2) = 1/36$$

$$P(3) = 2/36$$

$$P(4) = 3/36$$

$$P(5) = 4/36$$

$$P(6) = 5/36$$

$$P(7) = 6/36$$

$$P(8) = 5/36$$

$$P(9) = 4/36$$

$$P(10) = 3/36$$

$$P(11) = 2/36$$

$$P(12) = 1/36$$

(c) 计算每个总和的信息量 (以比特为单位)：

$$\text{信息量} = -\log_2(P)$$

$$I(2) \approx 5.170$$

$$I(3) \approx 4.170$$

$$I(4) \approx 3.585$$

$$I(5) \approx 3.170$$

$$I(6) \approx 2.585$$

$$I(7) \approx 2.585$$

$$I(8) \approx 2.585$$

$$I(9) \approx 3.170$$

$$I(10) \approx 3.585$$

$$I(11) \approx 4.170$$

$$I(12) \approx 5.170$$

(d) 计算由总和组成的符号集的熵：

$$\text{熵 } (H) = - \sum [P(x_k) * \log_2(P(x_k))]$$

$$H \approx -(1/36 * 5.170 + 2/36 * 4.170 + 3/36 * 3.585 + 4/36 * 3.170 + 5/36 * 2.585 + 6/36 * 2.585 + 5/36 * 2.585 + 4/36 * 3.170 + 3/36 * 3.585 + 2/36 * 4.170 + 1/36 * 5.170) \approx 2.598$$

(e) 为这个字母表设计一个哈夫曼编码：

根据概率对总和进行排序，然后构建哈夫曼树。最终得到如下哈夫曼编码：

2: 00000

3: 00001

4: 0001

5: 001

6: 01

7: 10

8: 110

9: 1110

10: 11110

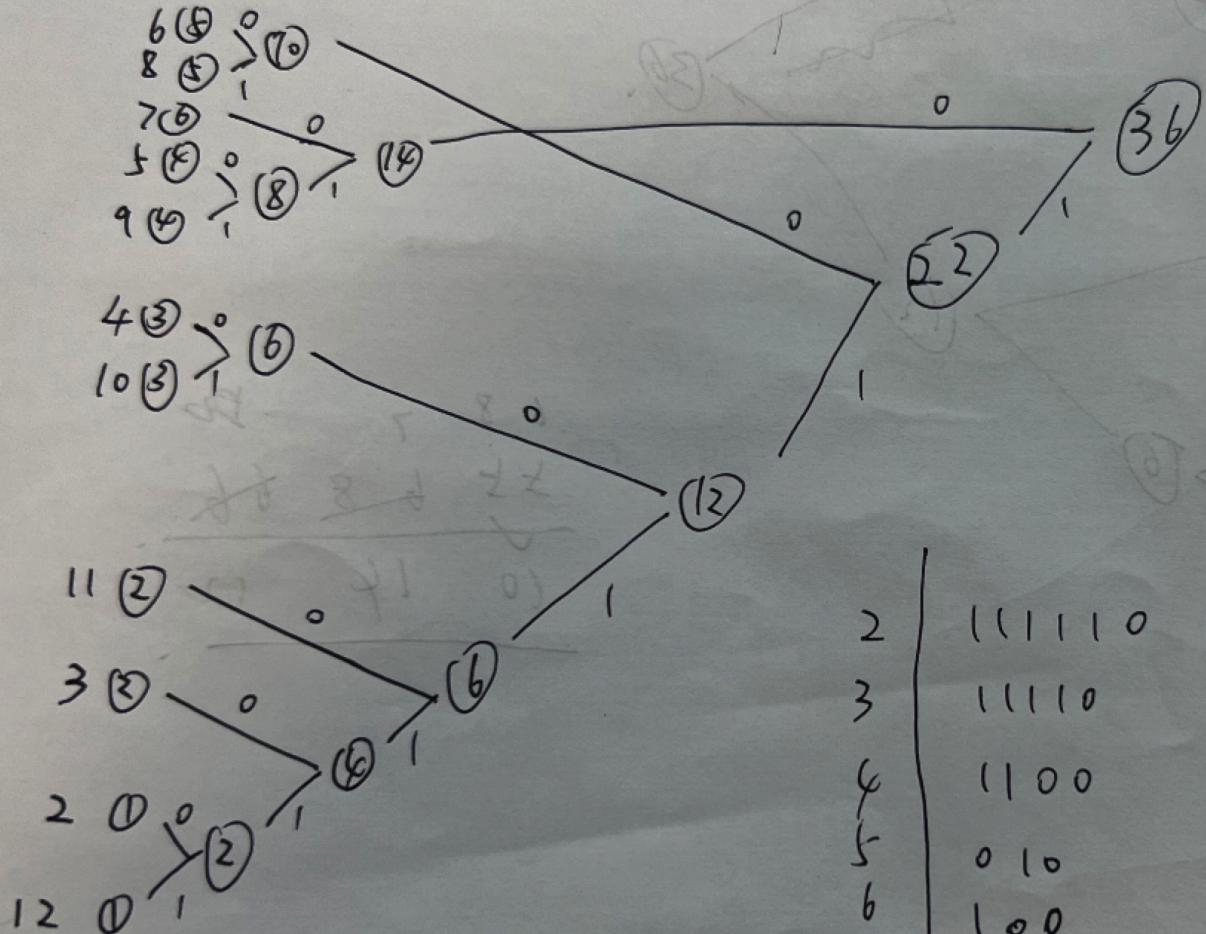
11: 111110

12: 111111

这是一种可能的哈夫曼编码，具体编码可能因

构建哈夫曼树的顺序而有所不同。但是，不同的哈夫曼编码应具有相同的平均码长，这是由于其构建原理决定的。

通过使用这个哈夫曼编码对由两个六面骰子总和组成的符号集进行编码，我们可以获得接近熵的平均编码长度，从而实现高效的数据压缩。请注意，实际应用中，哈夫曼编码通常应用于更大的数据集，以便充分利用概率分布特征以实现更高的压缩效果。



2	11110
3	11110
4	1100
5	010
6	100
7	00
8	101
9	011
10	110
11	1110
12	11111

PC不在码表里,先加入码表,再编p的码到code,p指向C

pc在码表里,不加码表,不写code,但是p指向pc

Lec2

An SD terrestrial digital TV requires a bit rate of 24 Mbit s⁻¹. If the power to noise ratio is 30 dB, what bandwidth is required?

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

$\frac{S}{N}$ 就是 SNR , 单位就是 dB

一个SD地面数字电视需要24 Mbit/s的比特率。如果信噪比为30 dB, 那么所需的带宽是多少?

要解决这个问题, 我们需要使用香农定理来计算所需的带宽。香农定理如下:

$$C = B * \log_2(1 + SNR)$$

其中 C 是通道容量 (比特率) , B 是带宽, SNR 是信噪比。

首先, 我们需要将信噪比从分贝 (dB) 转换为线性比值。使用下面的公式进行转换:

$$SNR(\text{线性}) = 10^{(SNR(dB) / 10)}$$

将信噪比 30 dB 代入公式, 得到:

$$SNR(\text{线性}) = 10^{(30 / 10)} = 1000$$

现在, 我们需要找到满足所需比特率的带宽。从香农定理中解出带宽 B :

$$B = C / \log_2(1 + SNR)$$

将比特率 ($C = 24,000,000 \text{ bit/s}$) 和信噪比 ($SNR = 1000$) 代入公式, 得到:

$$B = 24,000,000 / \log_2(1 + 1000) \approx 7,972,359 \text{ Hz} \approx 7.97 \text{ MHz}$$

因此, 所需的带宽约为 7.97 MHz。

Lec3

奇校验 : 1 的个数为奇数为 0

偶校验 : 1 的个数为偶数为 0

在一个包含 n 个比特的数据块中, 有 k 个比特出现错误的概率由概率质量函数给出:

$$Pr(X = k) = \binom{n}{k} p^k (1-p)^{n-k}$$

pisbiterrorration(BER), binomialcoefficient($\binom{n}{k}$) = $\frac{n!}{(n-k)!}$

Hamming code

$$n - k = r$$

n 码的总长度 k 有效字符的长度 r 纠错码的长度

block length $2^r - 1$ 这个是汉明码的总长度

message length $k = 2^r - r - 1$ 这个是汉明码有效位的长度

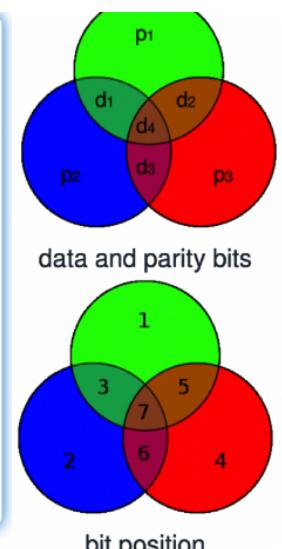
$$\text{code-rate efficiency } \frac{k}{n} = 1 - \frac{r}{2^r - 1}$$

(7,4) -> n = 7 r = 3 k = 4, 所以(7,3)码是不可能出现的

如何得到生成矩阵G和校验矩阵H

$$\mathbf{G} = (I_k | A^T) = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$\mathbf{H} = (A | I_{n-k}) = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$



取小海叫此商：一个码的取小海叫此商定江息网下小问码子之问时

(7,4)码

|||||||

| ----- | --- | --- | --- | --- | --- | --- |

| 位置 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

| 二进制码表 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| G第一行 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |

| 表示的有效位位置 | 3 | 5 | 6 | 7 | 1 | 2 | 4 |

1 2 4 位置是校验位

3 5 6 7 位置是有效位置

1. 左边先写4*4的单位矩阵.

2. 数据位3,与校验位1 2 有关. 数据位3的位置上是1,为了分别与1 2 捂成偶数个1,所以1 2 碗也写1,就变成了: 1000 110

一共7个码字,4个有效位,3个校验位.

生成矩阵 :

这里也需要再看看:怎么得到的H矩阵

Hamming DIstance

海明距离 : 在信息理论中, 两个相等长度字符串之间的海明距离指的是对应位置上符号不同的数量。

最小海明距离 : 一个码的最小海明距离是任意两个不同码字之间的最小海明距离

侦测到e个错误: 最小汉明距离至少为 $e+1$; 纠正e个错误, 最小汉明距离至少为 $2e+1$

hamming code的最小海明距离一定是3,这是由于hamming code的特性决定的.所以汉明码的纠错能力就是1,不会再改变了.

block length $2^r - 1$

message length $k = 2^r - r - 1$

code-rate efficiency $\frac{k}{n} = 1 - \frac{r}{2^r - 1}$

parity bits	total bits	data bits	name	code-rate efficiency
2	3	1	Hamming(3,1)	$\frac{1}{3} \approx 33\%$
3	7	4	Hamming(7,4)	$\frac{4}{7} \approx 57\%$
4	15	11	Hamming(15,11)	$\frac{11}{15} \approx 73\%$
5	31	26	Hamming(31,26)	$\frac{26}{31} \approx 84\%$

Hamming(3,1) code is equivalent to Triple Modular Redundancy

Hamming Code

H parity check code matrix 奇偶校验矩阵

G Generator Matrix 生成矩阵

syndrome 校验和

检验单bit错误的步骤和例子

- Here binary matrix multiplications are modulo 2 (i.e. result 0 if even, 1 if odd)
- The Hamming matrices have the property $\mathbf{H} \cdot \mathbf{G}^T = \mathbf{0}$
- The Hamming code word is obtained $\mathbf{x} = \mathbf{p} \cdot \mathbf{G}$ where \mathbf{p} is the original data code word.
- If no error occurs during transmission, the received code word $\mathbf{r} = \mathbf{x}$
- The syndrome $\mathbf{z} = \mathbf{H} \cdot \mathbf{r}^T$, is the null vector if there is no error and \mathbf{r} is a valid code word.
- For a single bit error, $\mathbf{r} = \mathbf{x} + \mathbf{e}_i$, where \mathbf{e}_i has a 1 in the i th position and 0 otherwise.
- The syndrome $\mathbf{z} = \mathbf{H} \cdot \mathbf{r}^T = \mathbf{H} \cdot \mathbf{e}_i^T$ identifies the single bit error from matching the appropriate column of \mathbf{H}
- The General Algorithm identifies the bit error as the binary value of \mathbf{z} .
- The correction is by flipping the i th bit of \mathbf{r}
- Hamming codes will also detect two-bit errors, but these are indistinguishable from single-bit errors and cannot be corrected.

对于单比特错误 (single bit error) , 接收到的码字 \mathbf{r} 可以表示为 $\mathbf{r} = \mathbf{x} + \mathbf{e}_i$, 其中 \mathbf{e}_i 表示一个只在第 i 位上有 1, 其它位上均为 0 的向量。

在通信中, 传输过程中可能发生了单比特错误, 即在传输过程中只有一个位发生了错误。在这种情况下, 接收到的码字 \mathbf{r} 与发送的码字 \mathbf{x} 只有在一位上不同, 可以表示成 \mathbf{x} 的某个比特位上的取反, 用 \mathbf{e}_i 表示。因此, 可以将接收到的码字 \mathbf{r} 表示为发送的码字 \mathbf{x} 加上 \mathbf{e}_i , 即 $\mathbf{r} = \mathbf{x} + \mathbf{e}_i$ 。

例如, 假设发送的码字为 $\mathbf{x}=[1, 0, 1, 1]$, 接收到的码字为 $\mathbf{r}=[1, 1, 1, 1]$, 那么可以将接收到的码字 \mathbf{r} 表示为发送的码字 \mathbf{x} 加上 \mathbf{e}_2 , 即:

css

 Copy code

```
e_2 = [0, 1, 0, 0]
r = x + e_2 = [1, 0, 1, 1] + [0, 1, 0, 0] = [1, 1, 1, 1]
```

注意, 这里的 \mathbf{e}_i 是一个只在第 i 位上有 1 的向量, 在其它位上均为 0。这个向量在二进制中只有一位为 1, 其它位均为 0, 因此被称为单比特错误。通过将接收到的码字 \mathbf{r} 表示为发送的码字 \mathbf{x} 加上 \mathbf{e}_i , 就可以判断出发生错误的位置, 并进行纠错。

纠正单bit错误的例子:

您提到的“综合征 $z = H \cdot r^T = H \cdot e_i^T$ 通过匹配H的适当列来识别单比特错误”描述了线性方程组的解码过程，通常用于纠错编码。让我们详细解释这个过程。

当使用线性纠错码（如汉明码）时，会有一个生成矩阵G和一个校验矩阵H。要发送的信息通过生成矩阵G进行编码，然后接收端接收到编码后的信息。为了检测和纠正错误，接收端使用校验矩阵H计算综合征z。

综合征z是接收到的信息r与校验矩阵H的转置相乘得到的，即： $z = H \cdot r^T$ 。

如果没有错误，综合征z将为零。如果有单比特错误，综合征z将与H的某一列相等，这一列对应于发生错误的比特位置。这样，通过匹配综合征z与H的适当列，可以找到并纠正单比特错误。纠正错误的方法是简单地翻转错误位。

需要注意的是，这种方法仅适用于单比特错误检测和纠正。如果发生多比特错误，综合征z可能与H的多列匹配，从而导致误判。

```
makefile
G = | 1 0 0 0 |   H = | 1 0 1 1 1 0 0 |
      | 0 1 0 0 |       | 0 1 1 1 0 1 0 |
      | 0 0 1 0 |       | 0 0 0 1 1 1 1 |
      | 0 0 0 1 |
```

1. 假设要发送的4比特信息为：`m = [1, 0, 1, 1]`
2. 使用生成矩阵G对信息进行编码：`c = m + G = [1, 0, 1, 1, 0, 1, 0]`
3. 在传输过程中，第5位发生错误，接收端收到的信息为：`r = [1, 0, 1, 1, 1, 1, 0]`
4. 使用校验矩阵H计算综合征z：`z = H + r^T = [0, 1, 1]`
5. 综合征z与H的第5列相等，因此我们知道第5位发生了错误。
6. 翻转第5位，修复错误：`corrected_r = [1, 0, 1, 1, 0, 1, 0]`
7. 解码修复后的信息：`corrected_m = [1, 0, 1, 1]`

$$z = H \cdot r^T = H \cdot e_i^T$$

Low-density parity-check code, LDPC码)

Low-density parity-check code, LDPC码) 是最接近香农极限的编码.

模2计算就是异或:相同为0,想异为1

Lec4

To generate a (7,4) cyclic code we need a generator polynomial of degree $(n - k)=3$, which has to divide $(x^7 + 1)$. Since, $x^7 + 1 = (x + 1)(x^3 + x^2 + 1)(x^3 + x + 1)$ we could choose either $(x^3 + x^2 + 1)$ or $(x^3 + x + 1)$. We will use $g(x) = (x^3 + x^2 + 1)$ and multiply it by all P(x) polynomials of the form $P(x) = p_1x^3 + p_2x^2 + p_3x + p_4$ where the pis can be 0 or 1. This generates 16 code word polynomials from $c(x) = P(x)g(x)$ and hence the complete set of code words.

为了生成一个(7,4)循环码，我们需要一个生成多项式，其次数为 $(n-k)=3$ ，该多项式必须能够整除 $(x^7 + 1)$ 。因为 $x^7 + 1 = (x + 1)(x^3 + x^2 + 1)(x^3 + x + 1)$ ，我们可以选择 $(x^3 + x^2 + 1)$ 或 $(x^3 + x + 1)$ 。我们将使用 $g(x) = (x^3 + x^2 + 1)$ ，并将其与所有形式为 $P(x) = p_1x^3 + p_2x^2 + p_3x + p_4$ 的P(x)多项式相乘，其中pis可以是0或1。这将通过 $c(x) = P(x)g(x)$ 生成16个码字多项式，从而生成完整的码字集。

这段话解释了如何生成一个(7,4)循环码。首先，我们需要找到一个生成多项式 $g(x)$ ，它的次数是 $(n-k)$ 。我们从 $x^7 + 1$ 的因式中选择一个合适的生成多项式。接下来，将生成多项式与所有可能的P(x)多项式相乘，得到一组码字多项式 $c(x)$ 。这样，我们就得到了一个(7,4)循环码的完整码字集。

BCH

For any m and t, there exists a BCH code with $n = 2m - 1$, $(n - k) \leq mt$ and $d_{min} = 2t + 1$.

For example, $t=2$, $m=4$ provides a $(15,7)$ BCH code with a generator polynomial corresponding to 111010001 . $(15,7)$: $7 = 15-mt = 15-8 = 7$

Tutorial_2

Digital Communications 4: Tutorial on Block Codes

1. A microwave digital signal is received from a satellite by a ground station. The noise spectral density at the receiver is $3 \times 10^{-20} \text{ W Hz}^{-1}$, the bandwidth is 32 MHz and the required bit rate is 64 kbit s^{-1} . What is the minimum received energy per bit in order to achieve error free transmission?
2. Write down a generator matrix and a corresponding parity check matrix for the Hamming $(7,4)$ code in systematic form, where the parity bits p_1, p_2 and p_3 obey:

$$\begin{aligned} p_1 \oplus d_1 \oplus d_2 \oplus d_4 &= 0 \\ p_2 \oplus d_1 \oplus d_3 \oplus d_4 &= 0 \\ p_3 \oplus d_2 \oplus d_3 \oplus d_4 &= 0 \end{aligned}$$

- (a) Obtain the 16 valid code words.
 - (b) Identify a mapping between single bit errors and the syndrome vector by considering the syndrome for the 7 codes \mathbf{e}_i consisting of a 1 in the i th position and 0 elsewhere.
 - (c) Verify your result with a few examples by taking a valid codeword and introducing a single bit error.
 - (d) If the single bit error ratio is p , and you can assume $p \ll 1$, then what is the probability of an error in a transmission of the original 4 bit code word?
 - (e) What is the probability of bit errors resulting in a valid but incorrect received 7 bit code word?
 - (f) What is the probability that there is an error in the decoded received 7 bit code word?
3. Consider an $(8,4)$ block code by adding an additional overall parity bit p_4 to a set of Hamming $(7,4)$ valid code words, e.g. taken from the previous question and in addition:

$$p_1 \oplus p_2 \oplus p_3 \oplus p_4 \oplus d_1 \oplus d_2 \oplus d_3 \oplus d_4 = 0$$

- (a) What is the minimum Hamming distance for this code?
- (b) Therefore identify the number of errors which can be detected for this code.
- (c) Therefore identify the number of errors which can be corrected for this code.
- (d) Write down the generator and the parity check matrices for this code.
- (e) Verify the application of this code with a few examples by taking a valid 8 bit codeword and introducing single bit errors and two bit errors.

1.

$$C_{max} = B \log_2 \left(1 + \frac{S}{N} \right)$$

$$SNR = \frac{S}{N} = 10 \lg \left(\frac{P_S}{P_N} \right)$$

$\frac{S}{N}$ 表示的就是 SNR, 信噪比. 不是简单的 Signal 除以 Noise.

$\frac{S}{N}$ 的单位为 dB. 与简单的 S 除以 N 的换算关系为 $SNR = \frac{S}{N} = 10 \lg \left(\frac{P_S}{P_N} \right)$, 注意是以 10 为底

2

(a)

注意从右向左为低位到高位

data code(7 6 5 3)	parity code(4 2 1)	encoded data(7 6 5 4 3 2 1)
0000	000	0000000
0001	011	0000111
0010	101	0011001
0011	110	0011110
0100	110	0101010
0101	101	0101101
0110	011	0110011
0111	000	0110100
1000	111	1001011
1001	100	1001100
1010	010	1010010
1011	001	1010101
1100	001	1100001
1101	010	1100110
1110	100	1111000
1111	111	1111111

Hamming(7, 4)的生成矩阵 G 是

$$G = \begin{bmatrix} 1000 & 111 \\ 0100 & 110 \\ 0010 & 101 \\ 0001 & 011 \end{bmatrix}$$

其校验矩阵H是

$$H = \begin{bmatrix} 1110 & 100 \\ 1101 & 010 \\ 1011 & 001 \end{bmatrix}$$

注意干扰可以写成 $e(x) = 0000010$, 即一个1表示干扰位置在第二位上, 其余全零的形式。如果原码为 $m(x)=1100101$, 则 $m(x)$ 经过干扰 $e(x)$, 应为 1100111 。

(b)

(b) Hamming (7,4)码的校验矩阵可以写为:

$$H = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

用这个校验矩阵

001 第一位有错

010 第二位有错

011 第三位有错

...

111代表第7位有错

(c)

改一位 得到然后乘以校验矩阵

(d)

For a 7-bit word with no parity this gives

$$\Pr_7(X = 1) = \binom{7}{1} p^1 (1 - p)^6 \approx 7p$$

For an 8-bit word with parity, single bit errors are detected, so the probability of error is dominated by the probability of 2 bit errors

$$\Pr_8(X = 2) = \binom{8}{2} p^2 (1 - p)^6 \approx 28p^2$$

把7换成4

(e)?

$$\frac{16 * 15}{16 * 2^7}$$

16种能变成其他 2^7 种

(f)

解码存在错误

因为只有一个错误是能够检测并纠正的,不会在解码中体现出来

$$\text{存在错误} = 1 - \text{无错误} - \text{只有一个错误} = 1 - (1 - p)^7 - C_7^1 p^1 (1 - p)^6$$

3

(a)

e表示错误 最小汉明距离

检错 $e+1$

纠错 $2e+1$

4

(b)

$$3 e+1 = 4 e = 3$$

(c) $2e+1 e = 1 3 e = 2 5 5 > 4$ 所以还是只能纠错1

能纠正2个错误. 因为他有4个校验位,所以能纠错两个错误. $2t$ 个校验位纠正 t 个错误.

(d)

$$G = \begin{bmatrix} 1000 & 1101 \\ 0100 & 1011 \\ 0010 & 0111 \\ 0001 & 1110 \end{bmatrix}$$

$$H = \begin{bmatrix} 1101 & 1000 \\ 1011 & 0100 \\ 0111 & 0010 \\ 1110 & 0001 \end{bmatrix}$$

Tutorial3

1. Consider the (7,4) cyclic code defined by the generator polynomial $x^3 + x^2 + 1$, equivalent to Hamming (7,4). Construct a lookup table to identify the bit error from the non-zero syndrome. Hint: since 0000000 is a valid codeword you can calculate the syndrome for the seven error codewords 1000000, 0100000, ...
 2. A cyclic $n = 6, k = 4$ code has the generator polynomial $x^2 + x + 1$.
 - (a) Construct a table of all possible 4-bit input data and the corresponding codewords for this cyclic code
 - (b) What is the minimum Hamming distance for this code?
 - (c) Hence, identify how many bit errors this code is able to detect
 - (d) Can this code be used for forward error correction?
 3. Find a generator polynomial and the corresponding code words for a (7,3) cyclic code. Hint: consider the factorisation of $x^7 + 1$ in your lecture notes. What is the minimum Hamming distance for this code?
 4. Using the generator polynomial $x^4 + x + 1$, find the generator matrix and the parity check matrix (in systematic form) of a (15,11) cyclic code.
-
5. A convolutional encoder is shown in Fig. 1, where the two outputs shown are bit interleaved into a single output stream.

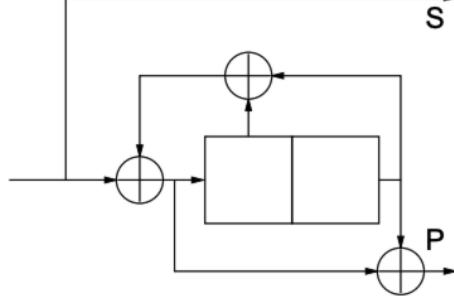


Figure 1: A convolutional encoder

- (a) What is the rate of this convolutional code?
- (b) Is this code recursive or non-recursive?
- (c) Is this code systematic or non-systematic?
- (d) Draw a state transition diagram for this convolutional encoder.
- (e) Using a Trellis diagram, or otherwise, find the convolutional code corresponding to the digital input 1001. You can assume the shift register is initially in the default all-zero state.
- (f) A received code, which may contain an error, is 11110001. Using the Viterbi algorithm and a Trellis diagram, determine the most likely binary data sequence that generated this code.
- (g) What is the rate 2/3 punctured code corresponding to part (e) using the puncturing

(g) What is the rate 2/3 punctured code corresponding to part (c) using the puncturing matrix $\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$

模2加法

$0+0=0$ $0+1=1$ $1+0=1$ $1+1=0$ 就是异或

| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 这才是正确的位数表示 |

| --- | --- | --- | --- | --- | --- | ----- |

| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

如何求循环码的生成矩阵

生成矩阵的例子

(7,4) 循环码的一个码字为 0011101, 请写出他的生成矩阵:

$c(x) = x^4 + x^3 + x^2 + 1 = (x^3 + x + 1)(x + 1)$, 所以 $g(x) = x^3 + x + 1$

(1) 码字0011101写成多项式

$c(x) = x^4 + x^3 + x^2 + 1 = (x^3 + x + 1)(x + 1)$, 所以 $g(x) = x^3 + x + 1$

(2)

$$G = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

(7,4) 4行7列

$$g(x) = x^3 + x + 1 \rightarrow 000\ 1011$$

$g(x)$ 改写成 \rightarrow 写在最下面, 然后左移. 就是循环码的生成矩阵, 然后用矩阵的初等变化得到最终的生成矩阵.

初等变幻:

矩阵的任意一行加减其他行, 矩阵不变. 这里的加减是模2除法, 也就是异或.

1. 错误的码字 校验和 第几位有错

错误的码字	校验和	第几位有错		
0000 001	001	第一位有错		
0000 010	010	第二位有错		
1000 000	111	第七位有错		

生成多项式的最高次幂是等于那个校验位的个数,也等于校验矩阵的行数

2. (a) 写(6,4)循环码
 3. 写出生成矩阵
 4. 简化生成矩阵
 5. 根据生成矩阵写出码表
- (6,4)循环码的单位矩阵应该是 4×4
-

Cyclic code (6, 4), $g(x) = x^2 + x + 1$

先写出生成矩阵 G , 然后化简成系统生成矩阵, 再根据该矩阵获取全部的编码表, 使用相应行进行亦或运算即可。

$$G = \begin{bmatrix} 111000 \\ 011100 \\ 001110 \\ 000111 \end{bmatrix} = \begin{bmatrix} 1000 & 11 \\ 0100 & 10 \\ 0010 & 01 \\ 0001 & 11 \end{bmatrix}$$

data	parity	codeword
0000	00	000000
0001	11	000111
0010	01	001001
0011	10	001110
0100	10	010010
0101	01	010101
0110	11	011011
0111	00	011100
1000	11	100011
1001	00	100100
1010	10	101010

data	parity	codeword
1011	01	101101
1100	01	110001
1101	10	110110
1110	00	111000
1111	11	111111

(b)

3

2e+1

(d)

是 前向纠错码 (Forward Error Correction, FEC) 和后向纠错码 (Backward Error Correction, BEC) 是纠错编码的两种主要方式。

前向纠错码是在数据传输前通过编码技术添加一定的冗余信息，使得接收方在接收到数据时，可以通过解码技术检测和纠正一定数量的传输错误，从而提高数据传输的可靠性。前向纠错码需要在发送端对数据进行编码处理，在接收端进行解码处理。

后向纠错码则是在数据接收过程中通过检测和纠正传输错误，提高数据传输的可靠性。后向纠错码需要在接收端对数据进行检测和纠正处理，需要大量的处理时间和计算资源。

因此，前向纠错码和后向纠错码的主要区别在于，前向纠错码在数据传输前通过编码技术添加冗余信息，提高了数据传输的可靠性；而后向纠错码在数据接收后通过检测和纠正传输错误，提高了数据传输的可靠性，但需要大量的处理时间和计算资源。

3.

$$x^7 + 1 = (x + 1)(x^3 + x^2 + 1)(x^3 + x + 1)$$

生成多项式的最高次幂是等于那个校验位的个数,也等于校验矩阵的行数

n-k是 生成多项式的最高次幂

选

$(x + 1)(x^3 + x^2 + 1)$ 或者 $(x + 1)(x^3 + x + 1)$ 都行.

4.

5.

Cyclic Code Implementation

Can implement with shift registers

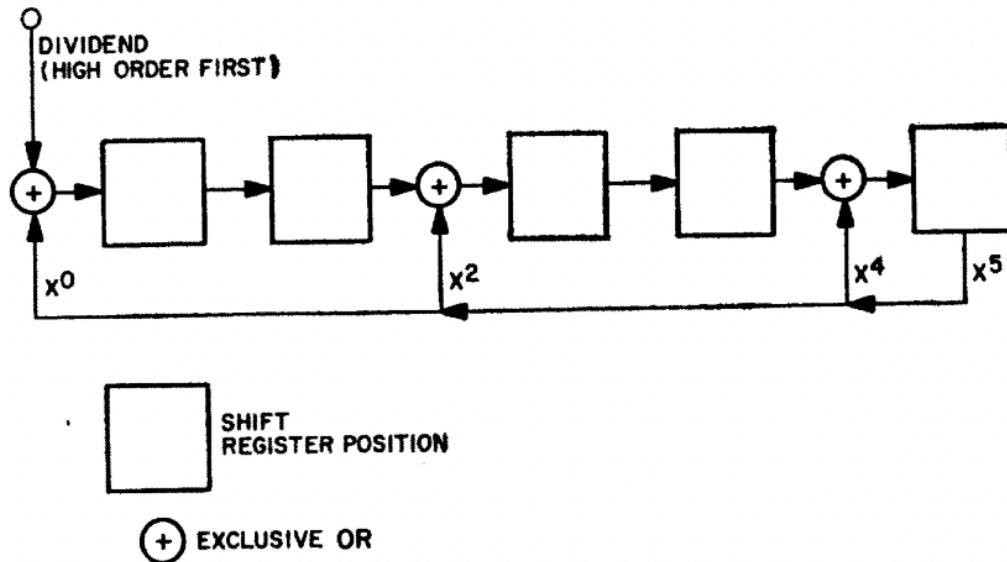


Fig. 1—A shift register for dividing by $1 + X^2 + X^4 + X^5$.

Can implement with shift registers

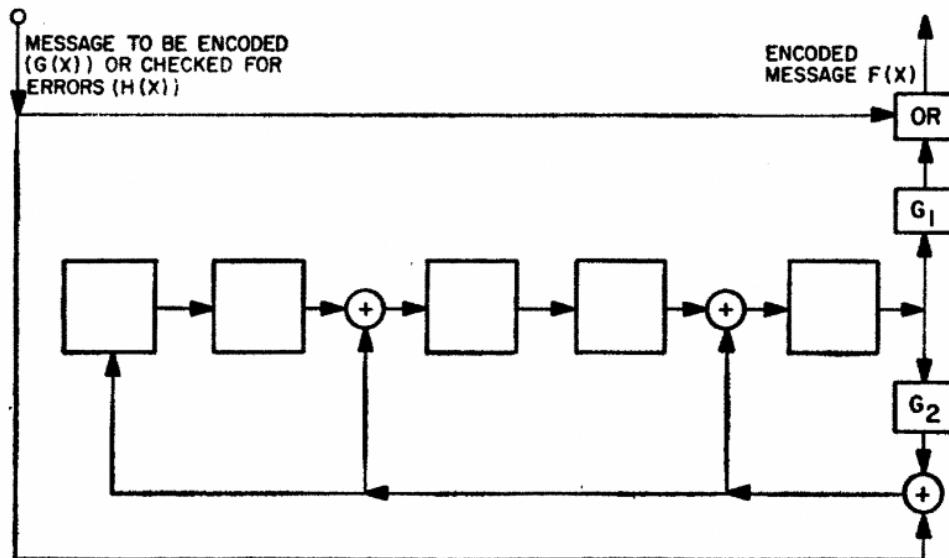


Fig. 3—A more efficient circuit for encoding and error detection.
(In this example, $P(X) = 1 + X^2 + X^4 + X^5$.)

5. (a)

$\frac{1}{2}$ 卷积码使用 (n, k, m) 表示

n 输出码字 k 输入的比特信息 m 表示寄存器个数 码率则为 $R = \frac{k}{n}$, 这一题两个输出码字 s p, —

个输入信息,所以 $\frac{1}{2}$

(b)

在卷积码中, 如果编码器的输出信号反馈回到它自身的输入端, 则该卷积码被称为递归码

(Recursive code)。反之, 如果编码器的输出信号不反馈回到它自身的输入端, 则该卷积码被称为非递归码 (Non-recursive code)。因此, 可以通过观察卷积码的编码器结构来判断卷积码是否是递归码。

所以这一问是: 递归的

(c)

是系统码

(d)

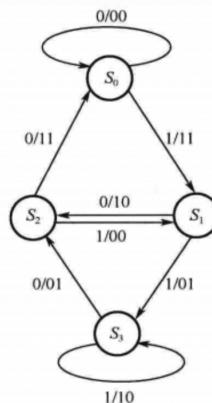
状态图

例如前面所述的 (2, 1, 2) 卷积码, 其编码寄存器状态如下表所示:

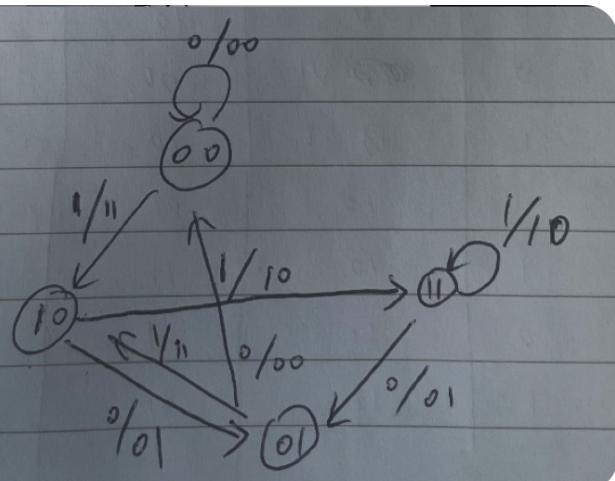
状态 S	D_1D_2	状态 S	D_1D_2
S_0	00	S_1	10
S_2	01	S_3	11

随着信息序列的输入, 编码器中寄存器的状态在上述4个状态之间转移, 并输出相应的码序列, 将编码器随输入而发生状态转移的过程用状态转移图表示, 如下图所示, 若: 码器处于 S_0 状态, 下一时刻输入 1 时, 编码器从 S_0 状态转移到 S_1 状态, 同时输出 11。

1. S_0 00 给 1 到状态 S_1 , 右边的 0 拿走 -> 1 写到左边 -> 10 S_0 给 1 到状态 S_1 . 1/11 -> S_1



input	status		output.	
	当前状态	下一个状态	输出 S	输出 P
0	00	00	0	0
0	00	00	0	0
0	10	01	0	1
0	11	01	0	1
1	00	10	1	1
1	01	10	1	1
	10	11	1	0
1	11	11	1	0



(e)

(f)

(g)

卷积码

1. 卷积码简介

卷积码将 k 个信息比特编成 n 个比特，但 k 和 n 通常很小，特别适合以串行形式进行传输，时延小。与分组码不同，卷积码编码后的 n 个码元不仅与当前段的 k 个信息有关，还与前面的 m 段信息有关。在编码器复杂性相同的情况下，卷积码的性能优于分组码。

卷积码使用 (n, k, m) 表示

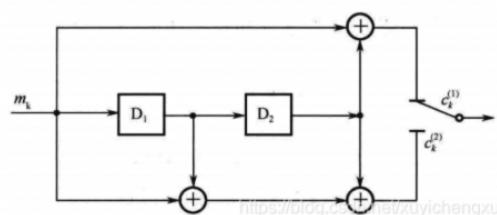
n 为输出码字；

k 为输入的比特信息；

m 表示寄存器个数；

码率为 $R = k/n$ 。

下图为 $(2, 1, 2)$ 卷积码，码长 $n=2$ ，寄存器个数 $m=2$ ，输入比特 $k=1$ 。

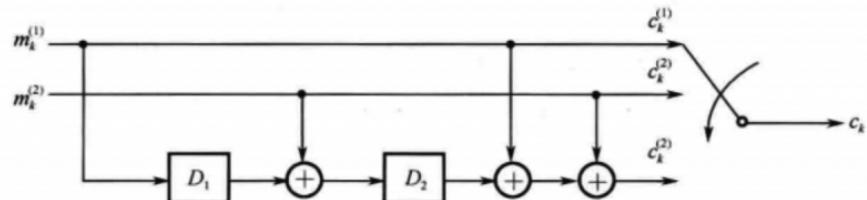


其生成多项式为：

$$g1(x) = 1 + x^2;$$

$$g2(x) = 1 + x + x^2;$$

下图为 $(3, 2, 2)$ 卷积码，码长 $n=3$ ，寄存器个数 $m=2$ ，输入比特 $k=2$ 。



其生成多项式为：

$$g11(x) = 1;$$

$$g12(x) = 0;$$

$$g13(x) = 1 + x^2;$$

$$g21(x) = 0;$$

$$g22(x) = 1;$$

$$g23(x) = 1 + x;$$

- 看箭头初始的箭头与最终输出箭头之间差多个寄存器,就是最终的生成多项式的最高次幂.
- 寄存器后面如果有加和的话就有对应项,否则就没有对应项,以上面的图为例.

Conclusion

生成多项式的最高次幂是等于那个校验位的个数,也等于校验矩阵的行数

生成矩阵的行数是嗯~信息位的个数

From Hamming distance concept, a code with minimum distance $2t+1$ can correct any t errors.

A linear block code that corrects all burst errors of length t or less must have at least $2t$ check symbols. 线性分组码:也就是说 $2t$ 个校验位,能纠正 t 个错误. -> 我们学的都是线性分组码

RS 码是线性分组循环码. QR(二维码就是应用了RS码)

Example: A popular Reed-Solomon code is RS(255,223) with $s=8$ -bit symbols. Each codeword contains $n=255$ code word bytes, of which $k=223$ bytes are data and $n - k = 2t = 32$ bytes are parity. For this code the decoder can correct any $t=16$ symbol errors in the code word: i.e. errors in up to 16 bytes anywhere in the codeword can be automatically corrected. Reed-Solomon codes are particularly well suited to correcting burst errors.

系统码就是指信息位和校验位 (也就是信道编码产生的冗余位) 分开而非系统码的信息位与校验位则相互交叉, 如信息位为 : (a_0, a_1, a_2, a_3) ,编码产生的位数为 (b_1, b_2) ,则利用系统码可能生成为 $(a_0, a_1, a_2, a_3, b_0, b_1)$;而利用非系统码则可能为 $(a_0, b_0, a_1, a_2, b_1, a_3)$ (当然还有其他可能)。例如汉明码就是 非系统码

如何得到生成多项式和校验矩阵

(n,k)

生成矩阵 : k 行 n 列

校验矩阵 : $n-k$ 行 n 列

1. 根据生成多项式写好最后一行,然后向左移位,移动 $k - 1$ 次
2. 最后通过初等行变化,改写为单位矩阵的形式
 - (7,4)码的生成矩阵中包含 4×4 的单位矩阵.,右边是校验位
 - H矩阵除去单位矩阵的行是

$$\mathbf{G} = (I_k | -A^T) = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$\mathbf{H} = (A | I_{n-k}) = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

这是(7,4)码的生成和校验矩阵,你可以反推,nk与校验矩阵之间的关系.

- 除去G的单位矩阵,列转为H除去单位矩阵的行

例子

请自己推一遍

Cyclic code (6, 4), $g(x) = x^2 + x + 1$

先写出生成矩阵G, 然后化简成系统生成矩阵, 再根据该矩阵获取全部的编码表, 使用相应行进行亦或运算即可。

$$G = \begin{bmatrix} 111000 \\ 011100 \\ 001110 \\ 000111 \end{bmatrix} = \begin{bmatrix} 1000 & 11 \\ 0100 & 10 \\ 0010 & 01 \\ 0001 & 11 \end{bmatrix}$$

生成矩阵的最高次幂,应该是 $n-k = r$. (7,4)的最高次幂是3,(15,11)的最高次幂是4
(15,11)的生成多项式就是: $x^4 + x + 1$

(7,4)码

Data (d_1, d_2, d_3, d_4)	Hamming(7,4)		Hamming(7,4) with extra parity bit (Hamming(8,4))							
	Transmitted ($p_1, p_2, d_1, p_3, d_2, d_3, d_4$)	Diagram	Transmitted ($p_1, p_2, d_1, p_3, d_2, d_3, d_4, p_4$)	Diagram						
0000	0000000		0000000		0010		0101010		01010101	
1000	1110000		1110001		1010		1011010		10110100	
0100	1001100		1001101		0110		1100110		11001100	
1100	0111100		0111100		1110		0010110		00101101	
3	1101001		11010010		0011		1000011		10000111	
	0011001		00110011		1011		0110011		01100110	
	0100101		01001011		0111		0001111		00011110	
	1010101		10101010		1111		1111111		11111111	