

实验一 绘制二进熵函数曲线

一、实验目的：

1. 掌握 Matlab 绘图函数
2. 掌握、理解熵函数表达式及其性质

二、实验要求：

1. 提前预习实验，认真阅读实验原理以及相应的参考书。
2. 在实验报告中给出二进制熵函数曲线图

三、实验原理：

1. 信源熵的概念及性质

$$\left[\begin{array}{c} X \\ P(X) \end{array} \right] = \left\{ \begin{array}{cc} x_1 = 0 & x_2 = 1 \\ p & 1-p \end{array} \right\}, 0 \leq p \leq 1$$

$$\begin{aligned} H(X) &= -\sum_i p(x_i) \log p(x_i) \\ &= -[p \log p + (1-p) \log(1-p)] \\ &= H(p) \end{aligned}$$

$$a. H(X) \leq \log n$$

$$b. H[\lambda P + (1-\lambda)Q] \geq \lambda H(P) + (1-\lambda)H(Q)$$

四、实验内容：

用 Matlab 软件制作二进熵函数曲线。

实验二： Huffman 编码软件实现

1、实验目的

- (1) 进一步熟悉 Huffman 编码过程;
- (2) 掌握 C 语言递归程序的设计和调试技术 (或者使用 Matlab)。

2、实验要求

- (1) 输入: 信源符号个数 r 、信源的概率分布 P ;
- (2) 输出: 每个信源符号对应的 Huffman 编码的码字。

3、实验内容

(1) 算法

- 1、从键盘输入组成信源 C 的字符个数 N ;
- 2、从键盘输入信源 C 和组成信源的字符所对应的概率数组 P ;
- 3、用 *Huffman* 函数来对信源进行二进制编码; 先对 P 按从大到小进行排序,

与此同时要把 C 中相应的字符的位置做相应的调换; 用 *count* 数组来记录编码: 在进行记录编码时是从数组 *count* 的最后一个开始存储的, 而且, 每进行一次编码所记录下来两个编码 '0','1' 是按从数组的最后一个元素开始服从 $count[m-k-j]$ 、 $count[m-k-j-1]$, 其中 k 表示编码所进行的次数, j 表示每次编码都只有 '0','1'; 最后用函数 *print()* 来输出编码。

(2) 部分伪代码:

(一) 节点信息结构体

```
struct HuffNode
{
    int weight;//信源符号的概率
    int parent;
    int lchild;
    int rchild;
};
```

(二) 算法

```
void Huffman(int weight[], int n, HuffNode hn[], HuffCode hc[])
{
    for(i = 0; i != 2*n - 1; ++i) //create Huffman Node, step 1
    {}
    for(i = 0; i != n-1; ++i) //create Huffman Node, step 2
    {
```

```

        for(j = 0; j != n+i; j++)
            { if(hn[j].weight < min1 && hn[j].parent == 0)
            {}
else if(hn[j].weight < min2 && hn[j].parent == 0)
            {}else ;
            {}
在此逆序存储 Huffman 编码
int temp[maxlen];
for(i = 0; i != n; ++i)
{
    int parent = hn[i].parent;
    while(hn[child].parent != 0)
    {}
}

```

4、实验报告

- (1) 简要总结 Huffman 编码的原理与特点
- (2) 写出 Huffman 编码的基本步骤，画出实现 Huffman 编码的程序流程图
- (3) 给出 Huffman 编码的源程序，并给出实验过程中的测试结果
- (4) 总结实验过程遇到的问题及解决方法

实验三 循环码的编码和译码程序设计

一、实验目的:

1. 通过实验了解循环码的工作原理。
2. 了解生成多项式 $g(x)$ 与编码、译码的关系。
3. 了解码距 d 与纠、检错能力之间的关系。
4. 分析(7.3)循环码的纠错能力。

二、实验要求:

- 1、编、译码用上述的计算法程序框图编写。
- 2、计算出所有的码字集合,可纠错误图样 $E(x)$ 表和对应的错误伴随式表。
- 3、考查和分析该码检、纠一、二位错误的的能力情况。
- 4、整理好所有的程序清单,变量名尽量用程序框图所给名称,并作注释。
- 5、出示软件报告。

三、实验设计原理

1、循环码编码原理

设有一 (n, k) 循环码, 码字 $C = [C_{n-1} \cdots C_r C_{r-1} \cdots C_0]$, 其中 $r=n-k$ 。码字多项式为:

$$C(x) = C_{n-1}x^{n-1} + C_{n-2}x^{n-2} + \cdots + C_1x + C_0.$$

码字的生成多项式为: $g(x) = gr-1xr-1gr-2xr-2 + \cdots + g_1x + g_0$

待编码的信息多项式为: $m(x) = m_{k-1}x^{k-1} + \cdots + m_0$

$$x^{n-k}m(x) = C_{n-1}x^{n-1} + \cdots + C_{n-k}x^{n-k}$$

对于系统码有: $C_{n-1} = m_{k-1}, C_{n-2} = m_{k-2}, \cdots, C_{n-k} = C_r = m_0$

设监督多项式为: $r(x) = C_{r-1}x^{r-1} + \cdots + C_1x + C_0$

根据循环码的定义, 则有:

$$C(x) = x^{n-k}m(x) + r(x) = q(x) \cdot g(x)$$

$$x^{n-k}m(x) = q(x) \cdot g(x) + r(x)$$

$$r(x) = Rg(x)[x^{n-k}m(x)]$$

即监督多项式是将多项式 $x^{n-k}m(x)$ 除以 $g(x)$ 所得的余式。

编码过程就是如何根据生成多项式完成除法运算求取监督多项式的过程。

设循环码(7.3)码的字多项式为:

$$C(x) = C_6x^6 + C_5x^5 + C_4x^4 + C_3x^3 + C_2x^2 + C_1x + C_0$$

($n=7$)

生成多项式为: $g(x) = x^4 + x^2 + x + 1$

信息多项式为: $m(x) = m_2x^2 + m_1x + m_0$ ($k=3$), 设

$m(x) = x^2 + x$

监督多项式为: $r(x) = C_{r-1}x^{r-1} + \cdots + C_1x + C_0$

根据循环码的定义: 生成多项式的倍式均是码字, 编码实际上是做 $x^{n-k}m(x)$ 除以 $g(x)$ 的

运算求得 $r(x)$ 。

编码程序框图见图 4.1(a)左, 二进制多项式除法示意图见图 4.1(b)。

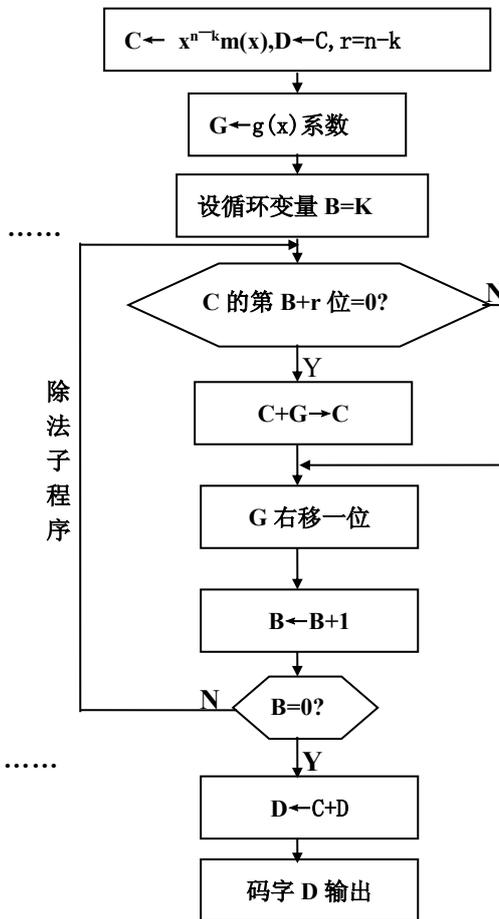
2、译码原理

设 $R(x)$ 为接收码字多项式, $E(x)$ 为错误图样多项式, $S(x)$ 为伴随式, 则根据循环码的性质有:

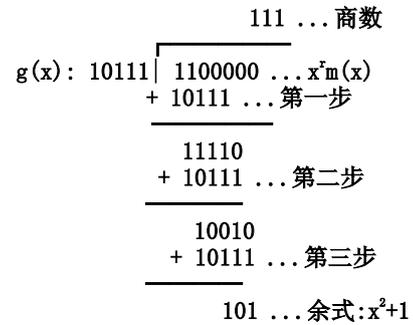
$$S(x) = R g(x) [R(x)] = R g(x) [E(x)]$$

当 $R(x) = C(x)$ 时, 有 $E(x) = 0$, $S(x) = 0$

当 $R(x)$ 不等于 $C(x)$ 时, 有 $E(x)$ 为非 0, $S(x)$ 为非 0



(a) 编码计算程序框图



编码步骤:

- 1、 $n-k=r=7-3=4$, 用 x^4 乘 $m(x)$ 的运算实际上相当于在信息码 110 后附上 4 个 0, 变为 1100000
- 2、用 $x^4m(x) = x^4(x^2+x) = x^6+x^5$ 除以 $g(x)$, 如图 (a) 所示, 得到监督余式 $r(x) = x^2+1$ 。
- 3、编出相应的发送码字为:
 $C(x) = x^4m(x) + r(x)$
 $C = 1100000 + 101 = 1100101$
- 4、按上述步骤, 将得到下述码表:

信息位	监督位
000	0000
001	0111
010	1110
011	1001
100	1011
101	1100
110	0101
111	0010

(b) 二进制多项式除法示意图

图 3.1 编码计算程序框图及多项式除法示意图

译码过程如下:

计算每一种可能被纠的错误图样 $E(x)$ 的伴随式,

$$S_i(x) = R g(x) [E(x)]$$

本地作数据表存储好。

根据已接收码字多项式 $R(x)$, 计算相应的伴随式:

$$S(x) = R g(x) [R(x)]$$

将实际接收码字求出的 $S(x)$ 与本地存储的各 $S_i(x)$ 相比较, 查出两者相等的那个唯一匹配的 $S_i(x)$, 找出并得到相应的错误图样 $E(x)$ 。

(4) 纠错: $C(x)=R(x)+E(x)$

否则由 $S(x)$ 找不出唯一匹配的 $S_i(x)$, 则报出错信息, 表示出现不可纠错的错误图样, 即码元出错的个数超出该循环码的纠错能力。

译码流程图 4.2 所示:

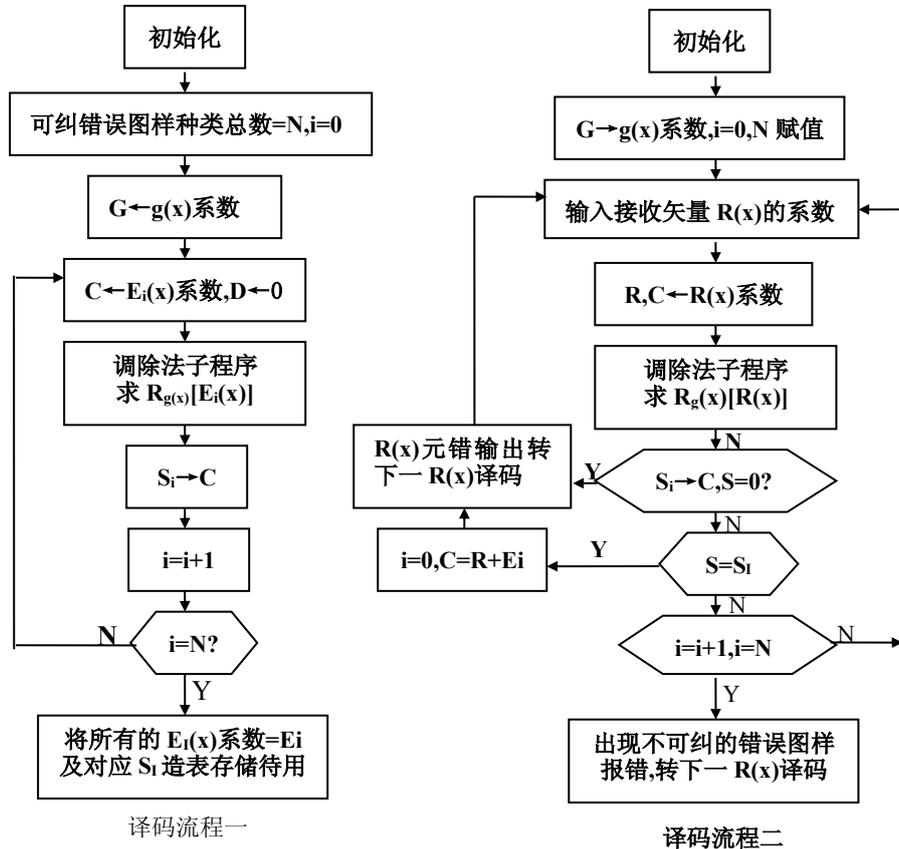


图 3.2 译码程序流程图

四、完成实验报告

- (1) 简要总结循环码编译码原理与特点
- (2) 写出循环码的基本步骤, 画出实现循环码的程序流程图
- (3) 给出循环码的源程序, 并给出实验过程中的测试结果
- (4) 总结实验过程遇到的问题及解决方法

实验四 卷积码的编译码

一、实验目的：

1. 通过实验了解卷积码的工作原理。
2. 了解生成多项式 $g(x)$ 与编码、译码的关系。

二、实验要求：

- 1、编、译码用上述的计算法程序框图编写。
- 2、计算出所有的码字集合，利用 viterbi 算法进行译码。
- 3、整理好所有的程序清单，变量名尽量用程序框图所给名称，并作注释。
- 4、出示软件报告。

三、实验设计原理

卷积码编码原理

(2,1,3) 卷积码的编码结构如图 4-1

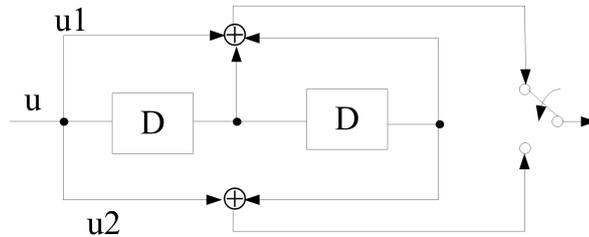


图 4-1 (2,1,3) 卷积码的编码结构

$$\mathbf{G}_\infty = \begin{bmatrix}
 \mathbf{G}_0 & \mathbf{G}_1 & \cdots & \mathbf{G}_m & & & \\
 & \mathbf{G}_0 & \mathbf{G}_1 & \cdots & \mathbf{G}_m & & \\
 & & \mathbf{G}_0 & \mathbf{G}_1 & \cdots & \mathbf{G}_m & \\
 & & & \mathbf{G}_0 & \mathbf{G}_1 & \cdots & \mathbf{G}_m \\
 & & & & \ddots & \ddots & \ddots & \ddots
 \end{bmatrix}$$

则编码器输出可以由下式来确定：

$$\mathbf{c} = \mathbf{uG}$$

说， $k=1$ ， $n=2$ ， $m=2$ ，所以：

$$\mathbf{G}_0 = (11), \mathbf{G}_1 = (10), \mathbf{G}_2 = (11)$$

固有：

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

而输入 $\mathbf{U} = (10111)$ ，所以有：

$$\begin{aligned}
 \mathbf{c} = \mathbf{uG} &= [10111] \begin{bmatrix}
 11 & 10 & 11 & & & \\
 & 11 & 10 & 11 & & \\
 & & 11 & 10 & 11 & \\
 & & & 11 & 10 & 11 \\
 & & & & 11 & 10 & 11
 \end{bmatrix} \\
 &= (11100001100111)
 \end{aligned}$$

难点：矩阵的设置和信息序列的分割。

四、完成实验报告

- (1) 简要总结卷积码编译码原理与特点
- (2) 写出卷积码的基本步骤，画出实现卷积码的程序流程图
- (3) 给出卷积码的源程序，并给出实验过程中的测试结果
- (4) 总结实验过程遇到的问题及解决方法