# 未知古典密码分析

## 待解决问题

**分析未知密码密文：**

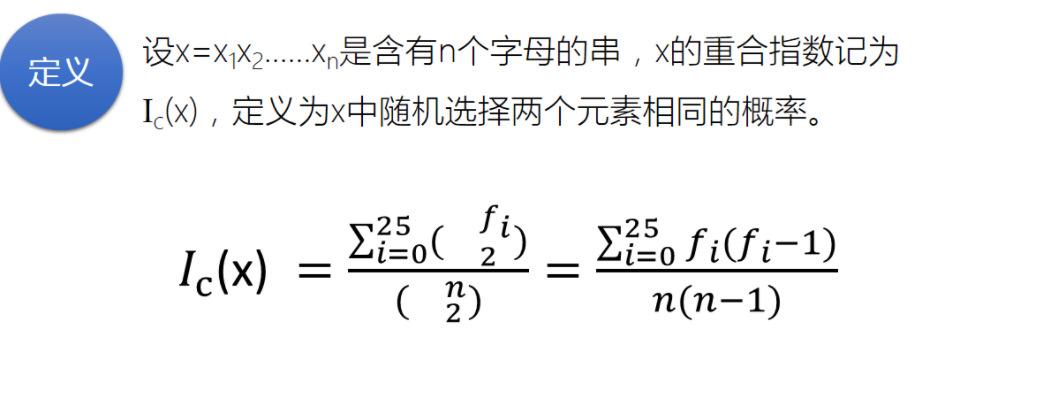
BNVSNSIHQCEELSSKKYERIFJKXUMBGYKAMQLJTYAVFBKVTDVBPVVRJYYLAOKYMPQSCGDLFSRLLPROYGESEBUUALRWXMMASAZLGLEDFJBZAVVPXWICGJXASCBYEHOSNMULKCEAHTQOKMFLEBKFXLRREDTZXCIWBJSICBGAWDVYDHAVEJXZIBKCGJIWEAHTTOEWTUHKRQVVRGZBXYIREMMASCSPBHLHJMBLRFFJELHWEYLWISTFVVYEJCMHYUYRUFSFMGESIGRLWALSWMNUHSIMYYITCCQPZSICEHBCCMZFEGVJYOCDEMMPGHVAAUMELCMOEHVLTIPSUYILVGFLMVWDVYDBTHFRAYISYSGKVSUUHYHGGCKTMBLRX

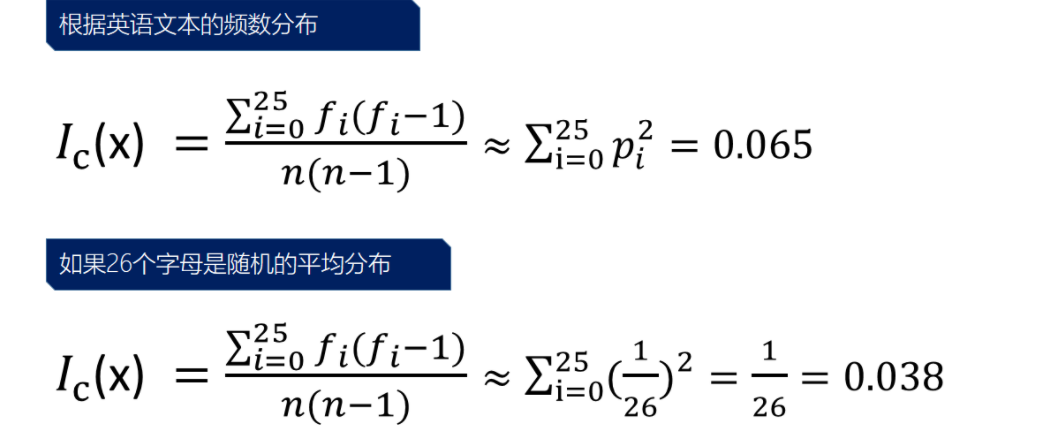
## 题目分析



根据上图，分析前4种常见的古典密码可知，移位、仿射、代换都是单表代换，维吉尼亚密码是多表代换。单表代换不会改变文本的重合指数，多表代换会使重合指数发生变化，由原英文文本的0.065向完全随机文本的重合指数0.038变化。通过验证密文的重合指数可以区分出单表代换和多表代换。其中单表代换中的移位密码和仿射密码又可以通过穷举密钥加明文指数测试的方法求解出明文，如果不能给出明文解。说明不是移位密码或仿射密码，大概率是代换密码，需要通过频数分析的方法来求解。

## 重合指数





## 实验过程

### 1、求密文的重合指数，判断是哪一种加密方式

经计算密文的重合指数mg=0.041814,基本上可以确定加密方式是维吉尼亚密码。

下表为密文中各个字母出现次数和对应频率。

|  |
| --- |
| 字母 |
| A |
| B |
| C |
| D |
| E |
| F |
| G |
| H |
| I |
| J |
| K |
| L |
| M |
| N |
| O |
| P |
| Q |
| R |
| S |
| T |
| U |
| V |
| W |
| X |
| Y |
| Z |

## 2、解密

### (1)确定分组数m。

#### 方法一：Kasiski测试法

根据重复出现的密文串的位置来确定m的值，找出重复出现的长度为3的密文串，计算它们之间的距离，并求解最大公约数来确定分组长度。由于密文长度太短，又或者是该密文是特殊构造的，用Kasiski测试法没有找到重复出现3次以上的长度为3的字符串。

**Kasiski代码**

int Kasiski(string s)
  
{
  
 int m = 0;
  
 int len = s.length();
  
 for (int i = 0; i < len - 2; i++)
  
 {
  
 string chr = "";
  
 chr = chr + s[i] + s[i + 1] +s[i + 2];
  
 int c = 0;//记录出现次数
  
 int index[100];
  
 memset(index, 0, sizeof(index));
  
 for (int j = 0; j < len - 2; j++)
  
 {
  
 string tmp = "";
  
 tmp = tmp + s[j] + s[j + 1] +s[j + 2];
  
   
 if (chr == tmp)
  
 {
  
 index[c++] = j;
  
 }
  
 }
  
 if (c > 2)
  
 {
  
 for (int k = 1; k < c; k++)//计算距离
  
 {
  
 index[k] = index[k] - index[0];
  
 }
  
 int g = gcd(index[1], index[2]);
  
 for (int k = 3; k < c; k++)
  
 {
  
 g = gcd(g, index[k]);
  
 }
  
 if (m == 0)m = g;
  
 else {
  
 if (m == g)m = g;
  
 else m = gcd(m, g);
  
 }
  
 }
  
}
  
return m;
  
}

#### 方法二：穷举m值，用重合指数Ic验证

下表为分组为m时，对应各分组重合指数以及重合指数均值(部分表格)

|  |
| --- |
| 组数m |
| m=1 |
| m=2 |
| m=3 |
| m=4 |
| m=5 |
| **m=6** |
| m=7 |
| m=8 |
| m=9 |
| m=10 |
| m=11 |
| **m=12** |
| m=13 |