|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | 实验项目  名 称 | 实验 1 古典密码体制的统计分析 | 成绩 |  |   **一、实验目的**  (1) 了解古典密码中的基本加密运算。  (2) 了解古典密码体制。  (3) 掌握古典密码的统计分析方法。  **二、实验内容**  1)  用 C\C++语言实现统计 26 个英文字母出现频率的程序。 2)  统计分析 Playfair 算法加密的密文(见附件)得出密钥。(最终使用了Simulated annealing算法) 3)  用 C\C++语言实现解密程序。  **三、实验步骤**  **问题1：**  我使用了两篇英语小说统计词频（一篇为《鸿》，另一篇为《爱丽丝漫游奇遇记》）。  1）为了解决编码的问题，C++统计词频之前，先使用python将文章预处理了一下（“process.py”），去掉了出26个字母以外的字符， 保存在“alice.txt”中。  2）遍历整篇文章，获取每一个字母出现次数与字母总个数sum， 将每个字母的次数与sum相除即可获得词频；  **问题2：**  思路一：   1. 分别统计明文与密文中每个两两组合的字母对中的频率, 将出现频率相同的对相匹配。 2. 破译过程中，有几个trick： 如果明文ab对应密文cd，则明文ba对应密文dc、cd对应ab、dc对应ba；   实际操作中，由于拥有密文数目较少，统计频率后，只能将频率较高的几对互相配对，实验的进展陷入僵局。但是后来我发现还有另一种方法可以破译playfair密码，也就是思路二， [Simulated Annealing](http://en.wikipedia.org/wiki/Simulated_annealing)算法  注：统计频率的代码在“问题2词频统计”文件夹中。实现时，将明文出现的次数转化成与密文出现次数相同的数量级，如下    思路二： [Simulated Annealing](http://en.wikipedia.org/wiki/Simulated_annealing)算法  此算法的核心思想如下：   1. 随机生成一个符合要求足的5\*5密码框，把它叫做key 2. 用key解密密文， 获取明文M 3. 用一个评估函数判断M是真正明文的可能性rate 4. 随机交换key，生成新的秘钥key1 5. 用key1解密得到M1， 6. 用评估函数评估M1是明文的可能性rate1 7. 若rate1>rate, 则用key1代替key。否则用原来的key 8. 重复4）-7），直到rate1大于一个阈值或者迭代的次数达到了一定要求。   虽然思路比较简单，但实现的过程中仍然需要注意一些点：   1. 随机交换生成key可有多种方法，主要是随机交换两个字母的值，还可以随机交换两行、随机交换两列、转置、按行反转、按列反转等 2. 解密函数的实现将在**问题3**中说明 3. 健壮性函数的的使用是一个比较重要的问题，一共有2个方案： 4. 比较解密后M明文词频与真正明文词频的欧几里得距离，距离越小，则rate应当越大。但用代码实现后，发现效果并不理想 5. 使用了参考资料[1]中的健壮性函数实现 6. 用key1代替key时，实际如下实现： 7. 若rate1>rate, key=key1 8. 若rate1<rate, DF= rate1-rate。key=key1以e^(DF/T)的概率发生 9. 这样做是为了避免局部最优值的出现。   问题3：  解密程序思路如下:   1. 将密文字母大小写与key的字母的大小写统一。 2. 获取密文对，分别找到密文对两个字母在秘钥中的的位置，x1，y1，x2，y2 3. 如果x1=x2，则对应明文分别为两个字母循环左移对应字母 4. 如果y1=y2，则对应明文分别为两个字母循环上移对应的字母 5. 如果3）、4）都不满足，则对应明文字母的位置为（x1,y2）与(x2,y1); 6. 如果密文没有解密完，重复2）过程。   **四、实验结果及分析**  问题一：  如下为a-z个字母词频程序统计的结果：    问题二：  程序运行截图如下：    将明文处理后（分词、修改标点,、去掉末尾x、部分i变成j），得到如下文本：  During World War Two, cryptography became an extremely complex science for both the allied and axis powers. The allies managed to secretly crack the code produced by the Nazis enigma machine and thereby may have shortened the war by two years. The axis cryptographer on the other hand never managed to crack the American’s ultimate code. The spoken languages of the Navajo(i变成j) and other American Indians in the age of computers cryptography has become almost unbelievably complex. It is widely used in peace time in such areas as banking tele-communications.  但是发现，将程序运行多次，会获得多个秘钥：  y a b i t  g h k d f  n o p l m  w x z q v  c u r s e  v w x z q  e c u r s  t y a b i  f g h k d  m n o p l  s u r e c  i a b t y  d h k f g  l o p m n  q x z v w  …………………  这令我很疑惑，但思考之后发现是正常的。因为将playfair的5\*5密码方框循环左移、上移、下移、右移是不影响加密和解密结果的。  根据惯例，即使用完秘钥字母后，后面跟随的字母依次是a,b,c，依次到z结束，我们可以确定密码框如下：  s e c u r  i t y a b  d f g h k  l m n o p  q v w x z  因此秘钥是security。其中，密码破译共花费了32秒；  问题三：  实验结果截图如下：其中，解密花费时间为75ms。    **五、实验参考：**  [1] Cryptanalysis of the Playfair cipher： http://practicalcryptography.com/cryptanalysis/stochastic-searching/cryptanalysis-playfair/ |