CSC3621 Cryptography - Exercise 1

Aim - To understand the non-uniform distribution of English letters and how that may be exploited in cryptanalysis.

Introduction:

Frequency Analysis is the analysis of letter/character frequencies in a given text. I have created a programme to complete a frequency analysis of a given plain text and cipher text, to analysis the cipher text and make an assumption of the encryption algorithm used on it. Using this information my programme can be used to decrypt the cipher text and retrieve the plain text.

My understanding of a frequency analysis is that; it should be used to count the frequency of each character in a given text. From this you can find the most common characters. It is stated that the most common letter distributed in the English language is ‘e’. - *Robert Lewand's Cryptological Mathematics.*

This distribution is commonly assessed among experts. The following commonly distributed letters are ’t’ and ‘a’.

This distribution was used by historical cryptographers to create Encryption algorithms. For example the “Caesar Cipher” is a shift cipher which uses English distribution to encrypt plain text by simply shifting each letter by three spaces in the alphabet (a-z). Julius Caesar used this cipher to prevent enemies from intercepting messages he sent to officials and generals throughout his rule of the Roman empire.

The cipher works for example by shifting the letters three places:

CAESAR -> FDHVDW

The distribution however is also the key to breaking the cipher and retrieving the plain text. To decrypt the cipher text, you can complete a frequency analysis of the cipher text and find the most commonly distributed letter. So in the example above the most frequent letter is ‘D’. By the rules of a Caesar cipher we can assume that it relates to ‘A’ as ‘A’ is the third most commonly distributed letter. By trial and error we can see that ‘D’ does not relate to ‘E’ or ’T’ as the shift is too large. by the third attempt we can see that ‘D’ is three characters from ‘A’. We can then attempt the decipher by shifting each letter back three positions. Then if the text becomes clear we can conclude it is indeed a Caesar cipher.

Shift Cipher:

A shift cipher is an adaptation of the Caesar cipher which allows a shift of a larger number, increasing the range required to decrypt the cipher text.

For this programme I have assessed the cipher text using the frequency analysis. I did this by completing a frequency analysis of a given plain text and then using the most common distribution so that the system can compare them to work out the shift pattern.

```
Most Frequent letter in "pg1661.txt" e
Frequencies of each letter:
{a=36142, b=6638, c=11103, d=19100, e=54944, f=9363, g=8299, h=29579, i=31248, j=544, k=3681, l=17636, m=12155, n=29731, o=34869, p=7284, q=437, r=25684, s=27965, t=40511, u=13636, v=4572,
w=11534, x=577, y=9760, z=153}
```

When running the analysis of “pg1661.txt” you can see that the most commonly distributed letter is ‘e’ with a frequency of ‘54944’.

```
Most Frequent letter in "Exercise1Ciphertext.txt" i
Frequencies of each letter:
{a=40, b=4, c=25, e=117, f=21, g=44, h=55, i=205, j=24, k=32, l=86, m=109, n=3, o=9, p=56, q=43, r=103, s=99, t=36, u=4, v=101, w=113, x=120, y=44, z=12}
```

The second screenshot shows the frequency analysis of the cipher text “Exercise1Ciphertext.txt”. This shows that the letter ‘i’ is the most commonly distributed letter in the cipher text. We can then assume from this that ‘i’ in fact is ‘e’. So by assumption we move the text back 4 places and check the decrypted text which shows it is in fact the plain text.

Implementation:

To complete the frequency analysis, I used a HashMap to map the characters in the text to a counter. The counter for a letter increments every time said letter is read by the system. The map is then passed to a most frequent function to assess which character is the most frequent. It does this by setting a most frequent point and comparing it to the counter at each point while iterating through the map. If the counter is larger than the most frequent then the most frequent is set to the counter. This function is called twice for the plain text and the cipher text and returned to the calculation method which calculates the shift offset.

Manually frequency analysis is quite an easy concept although on a large scale it is much more efficient to use a program. To do this however I had to understand how to make the system understand how to complete the task. As a computer cannot read letters but can read ASCII etc, I was able to complete the functions using the character values. For example in the calculation used to work out the shift offset.

```
Shift number for ciphertext: 4
```

This is the output from the calculation where the plain text most frequent letter is subtracted from the most frequent of the cipher text. The range of lower case alphabetic characters is 97 - 122 for a-z. This means that ‘e’ = 101, ‘i’ = 105 so 105 - 101 = 4.

Once the shift offset has been discovered, it can be passed to the encryption/decryption function to encrypt/decrypt the text.

To encrypt the plain text it is passed into the encrypt function along with the shift number. The shift is set to the remainder left from 26 letters of alphabet plus 26. This shift along with the arithmetic from ‘a’ or ‘A’ allows for the characters to be moved to the right until they have been shifted sufficiently to the given offset. The text is then split to a char array where each character is checked to be a letter if it is a letter then a string is built using the shift. I.e:

(shift = shift % 26 + 26; // where shift = 4) = 30
((‘a' + (i - 'a' + shift) % 26) // where i = e and shift now = 30) = 8 hence shift is to ‘i’ which is 8 positions right of ‘a’.

This means the shift from ‘a’ is 8 spaces using the shift of 4 so letter becomes ‘i’.

What happens is a letter is missing?

“Exercise1Ciphertext.txt” has no letter ‘d’ in the text. My program is able to adapt to this occurrence and function without issue. This is because the map simply does not add the letter in the frequency analysis and does not record anything for the letter as it does not exist.

Testing:

To test the compatibility of my programme I also tested other text to compare the results and test functionality of the system.

I first added the word “Supercalifragilisticexpialidocious” and completed the frequency analysis on it. The analysis shows that the most common letter is ‘a’ which is the third most common in the english language.

```
Most Frequent letter in "pt2" a
Frequencies of each letter:
{a=3, c=3, d=1, e=2, f=1, g=1, i=7, l=3, o=2, p=2, r=2, s=3, t=1, u=2, x=1}
```

The encryption produces the cipher text “Wytivgepmjvekmpmwxmgibtmepmhsgmsyw” it is clear by manually checking that this has been correctly encrypted.

I then did a frequency analysis on the cipher text which produced:

```
Most Frequent letter in "ct2": e
Frequencies of each letter:
{b=1, e=3, g=3, h=1, i=2, j=1, k=1, m=7, p=3, s=2, t=2, v=2, w=3, x=1, y=2}
```

This shows that ‘e’ is the most common and as we know it is a cipher text we can assume that ‘e’ is not the most common in the plain text. We can find from trial and error that the shift is 4 once shift to ’t’ is tried which does not show accurate results.

I then decrypted it using the decrypt function which reproduced the plain text.

```
Supercalifragilisticexpialidocious
```

A final test that I tried was the testing of using a plain text that does not have an ‘e’ in the text. The results show an incorrect shift pattern as the position is calculated wrongly due to the program being trained to use ‘e’ as the most common reference. Because of this the file added doesn't contain an ‘e’ so the programme calculates it wrong.

After testing this it is safe to assume that my programme works for the frequency analysis. However I am not preparing for all cases as mentioned above. To do this an approach of measuring distances between patterns would be more appropriate but is difficult for me at my current level of programming skill.

References

Asciitable.com,. 'Ascii Table - ASCII Character Codes And Html, Octal, Hex And Decimal Chart Conversion'. N.p., 2015. Web. 30 Oct. 2015.

KAHN, David. *The Codebreakers. The Story Of Secret Writing. (Seventh Printing.)*. New York: Macmillan Co., 1972. Print.

Lewand, Robert. *Cryptological Mathematics*. Washington, DC: Mathematical Association of America, 2000. Print.