LAB REPORT

September 17, 2019

Mohammed Raihan Ullah
Reg No: 2016831009
Department of Software Engineering
Shahjalal University of Science and Technology

Contents

Chapter 1

Introduction

In cryptography, a substitution cipher is a method of encrypting by which units of plain text are replaced with cipher text, according to a fixed system; the "units" may be single letters (the most common), pairs of letters, triplets of letters, mixtures of the above, and so forth. In this lab task, I decrypted a cipher text analyzing the frequency distribution of English alphabet, monogram, bigram and trigram of English words.

Chapter 2

Knowledge base

Here are the frequency distribution of English alphabet, monogram, bigram and trigram

Frequency distribution English characters

a:	8.05%	b:	1.67%	c:	2.23%	d:	5.10%
e:	12.22%	f:	2.14%	g:	2.30%	h:	6.62%
i:	6.28%	j:	0.19%	k:	0.95%	l:	4.08%
m:	2.33%	n:	6.95%	o:	7.63%	p:	1.66%
q:	0.06%	r:	5.29%	s:	6.02%	t:	9.67%
u:	2.92%	v:	0.82%	w:	2.60%	x:	0.11%
y:	2.04%	z:	0.06%				

Figure 2.1: Frequency Distribution of English Character

```
A :  8.55       K :  0.81       U :  2.68
B :  1.60       L :  4.21       V :  1.06
C :  3.16       M :  2.53       W :  1.83
D :  3.87       N :  7.17       X :  0.19
E : 12.10       O :  7.47       Y :  1.72
F :  2.18       P :  2.07       Z :  0.11
G :  2.09       Q :  0.10
H :  4.96       R :  6.33
I :  7.33       S :  6.73
J :  0.22       T :  8.94
```

Figure 2.2: Monogram Frequency

TH :	2.71	EN :	1.13	NG :	0.89
HE :	2.33	AT :	1.12	AL :	0.88
IN :	2.03	ED :	1.08	IT :	0.88
ER :	1.78	ND :	1.07	AS :	0.87
AN :	1.61	TO :	1.07	IS :	0.86
RE :	1.41	OR :	1.06	HA :	0.83
ES :	1.32	EA :	1.00	ET :	0.76
ON :	1.32	TI :	0.99	SE :	0.73
ST :	1.25	AR :	0.98	OU :	0.72
NT :	1.17	TE :	0.98	OF :	0.71

Figure 2.3: Bigram Frequency

THE :	1.81	ERE :	0.31	HES :	0.24
AND :	0.73	TIO :	0.31	VER :	0.24
ING :	0.72	TER :	0.30	HIS :	0.24
ENT :	0.42	EST :	0.28	OFT :	0.22
ION :	0.42	ERS :	0.28	ITH :	0.21
HER :	0.36	ATI :	0.26	FTH :	0.21
FOR :	0.34	HAT :	0.26	STH :	0.21
THA :	0.33	ATE :	0.25	OTH :	0.21
NTH :	0.33	ALL :	0.25	RES :	0.21
INT :	0.32	ETH :	0.24	ONT :	0.20

Figure 2.4: Trigram Frequency

THE :	6.42	ON :	0.78	ARE :	0.47
OF :	2.76	WITH :	0.75	THIS :	0.42
AND :	2.75	HE :	0.75	I :	0.41
TO :	2.67	IT :	0.74	BUT :	0.40
A :	2.43	AS :	0.71	HAVE :	0.39
IN :	2.31	AT :	0.58	AN :	0.37
IS :	1.12	HIS :	0.55	HAS :	0.35
FOR :	1.01	BY :	0.51	NOT :	0.34
THAT :	0.92	BE :	0.48	THEY :	0.33
WAS :	0.88	FROM :	0.47	OR :	0.30

Figure 2.5: Most common English words

Chapter 3

Encrypted Text

aceah toz puvg vcdl omj puvg yudqecov, omj loj auum klu thmjuv hs klu zlcvu shv zcbkg guovz, upuv zcmdu lcz vuwovroaeu jczoyyuovomdu omj qmubyudkuj vukqvm. klu vcdluz lu loj avhqnlk aodr svhw lcz kvopuez loj mht audhwu o ehdoe eunumj, omj ck toz yhyqeoveg auecupuj, tlokupuv klu hej sher wcnlk zog, klok klu lcee ok aon umj toz sqee hs kqmmuez zkqssuj tckl kvuozqvu. omj cs klok toz mhk umhqnl shv sowu, kluvu toz oezh lcz yvhehmnuj pcnhqv kh wovpue ok. kcwu thvu hm, aqk ck zuuwuj kh lopu eckkeu ussudk hm wv. aonncmz. ok mcmukg lu toz wqdl klu zowu oz ok scskg. ok mcmukg-mcmu klug aunom kh doee lcw tuee-yvuzuvpuj; aqk qmdlomnuj thqej lopu auum muovuv klu wovr. kluvu tuvu zhwu klok zlhhr klucv luojz omj klhqnlk klcz toz khh wqdl hs o nhhj klcmn; ck zuuwuj qmsocv klok omghmu zlhqej yhzzuzz (oyyovumkeg) yuvyukqoe ghqkl oz tuee oz (vuyqkujeg) cmubloqzkcaeu tuoekl. ck tcee lopu kh au yocj shv, klug zocj. ck czm'k mokqvoe, omj kvhqaeu tcee dhwu hs ck! aqk zh sov kvhqaeu loj mhk dhwu; omj oz wv. aonncmz toz numuvhqz tckl lcz whmug, whzk yuhyeu tuvu tceecmn kh shvncpu lcw lcz hjjckcuz omj lcz nhhj shvkqmu. lu vuwocmuj hm pczckcmn kuvwz tckl lcz vueokcpuz (ubduyk, hs dhqvzu, klu zodrpceeuaonncmzuz), omj lu loj womg juphkuj ojwcvuvz owhmn klu lhaackz hs yhhv omj qmcwyhvkomk sowcecuz. aqk lu loj mh dehzu svcumjz, qmkce zhwu hs lcz ghqmnuv dhqzcmz aunom kh nvht qy. klu uejuzk hs kluzu, omj aceah'z sophqvcku, toz ghqmn svhjh aonncmz. tlum aceah toz mcmukg-mcmu lu ojhykuj svhjh oz lcz lucv, omj avhqnlk lcw kh ecpu ok aon umj; omj klu lhyuz hs klu zodrpceeu- aonncmzuz tuvu scmoeeg jozluj. aceah omj svhjh loyyumuj kh lopu klu zowu acvkljog, zuykuwauv 22mj. ghq loj aukkuv dhwu omj ecpu luvu, svhjh wg eoj, zocj aceah hmu jog; omj klum tu dom dueuavoku hqv acvkljog-yovkcuz dhwshvkoaeg khnukluv. ok klok kcwu svhjh toz zkcee cm lcz ktuumz, oz klu lhaackz doeeuj klu cvvuzyhmzcaeu ktumkcuz auktuum dlcejlhhj omj dhwcmn hs onu ok klcvkg-klvuu

Chapter 4

Encryption Procedures

4.1 Considering most frequent monogram 'o'

Here we can see that 'o' is the only monogram used in the text. So it can be either 'a' or 'i'. But 'i' is generally used as a subject and placed at starting point of a sentence. Here 'o' is used at the middle of the sectences and thus we can easily guess 'o' represents 'a'

$$mapper : o = a \tag{4.1}$$

4.2 Considering most frequent word 'omj'

Word 'omj' is used 18 times and which is the highest in the text. Before, we found that **o**=**a** and thus **o**mj=**a**mj. Now the most common trigram that start with 'a' is 'and' and it is one of the most common trigram in English literature with the frequency of 0.73%. So here 'omj' is stands for 'and'.

$$mapper : m = n \tag{4.2}$$

$$mapper : j = d \tag{4.3}$$

4.3 Considering pair 'lu' 'loj'

Form previous mapping we can write 'l**oj**'='l**ad**'. Now if we consider **l**=**h** and assume that **lu**=**he** we can get a pair of subject and auxiliary verb **he had**. The word '**lu**' is used at the beginning of a sentence. So it should be a subject. So we can guess the word would

be '**he**' which is a pronoun and can be used as a subject

$$mapper : l = h \tag{4.4}$$

$$mapper : u = e \tag{4.5}$$

4.4 Considering word 'klu' , 'klok' 'kh'

From our previous assumption we can write 'k**lu**'='k**he**' , 'k**lo**k'='k**ha**k'. Now we can say that the letter '**k**' must represent '**t**'. So the word stands '**the**' and '**that**'. Now in the text 'kh' used 8 times in the text. Here we can write '**k**h'='**t**h'. We know that the highest occurred bigram that starts with '**t**' is '**to**'. So here 'ku' is replaced with 'to'

$$mapper : k = t \tag{4.6}$$

$$mapper : h = o \tag{4.7}$$

4.5 Considering word 'hs'

Word 'hs' is the highest occurred bigram in the text. Form previous guess we can write '**h**s'='**o**s'. We can see from the knowledge base, the most used word in English that starts with '**o**' is '**of**'.

$$mapper : s = f \tag{4.8}$$

4.6 Considering pair 'ck' and 'czm'k'

From previous assumption we can write 'c**k**'='c**t**' and 'cz**m'k**'='cz**n't**'. If we replace 'c' with 'i' and 'z' with 's' the pair stands '**it isn't**'. So here we can easily say that 'c' represents 'i' and 'z' represents 's'

$$mapper : c = i \tag{4.9}$$

$$mapper : z = s \tag{4.10}$$

4.7 Considering sub-string 'klug zocj'

Here '**zocj**'='**said**' which is a verb and thus '**klug**' must be a subject. Now '**klu**g'='**the**g'. So 'g' will definitely be 'y'.

$$mapper : g = y \tag{4.11}$$

4.8 Considering word 'kluvu'

Here '**kluvu**'='**the**v**e**'. Now we can easily guess 'v' must represent 'r'.

$$mapper : v = r \tag{4.12}$$

4.9 Considering sub-string 'loj auum'

We can write from previous mapping, '**loj** a**uum**'='**had** a**een**'. Now we can easily guess '**auum**' must be '**been**'.

$$mapper : a = b \tag{4.13}$$

4.10 Considering word 'puvg'

From previous mapping 'p**uvg**'='p**ery**'. The most common word ends in '**ery**' is '**very**'

$$mapper : p = v \tag{4.14}$$

4.11 Considering word 'zhwu'

From previous assumption we can write '**zh**w**u**'='**so**w**e**'. Now we can easily say that the word is '**some**'

$$mapper : w = m \tag{4.15}$$

4.12 Considering sub-string 'klok toz'

Form previous mapping '**klok** t**oz**'='**that** t**as**'. We saw that the text is in past form. Thus the word '**toz**' must stands for '**was**'

$$mapper : t = w \tag{4.16}$$

4.13 Considering word 'aqk'

Here '**a**q**k**'='**b**u**t**'. So it must be one of the most used trigram '**but**'

$$mapper : q = u \tag{4.17}$$

4.14 Considering word 'dhqzcmz'

Here 'd**hqzcmz**' = 'd**ousins**'. So we can say that the word must be '**cousins**'

$$mapper : d = c \tag{4.18}$$

4.15 Considering sub-string 'aon umj'

Here '**ao**n **umj**' = '**ba**n **end**'. The word is actually the noun '**Bag End**'

$$mapper : n = g \tag{4.19}$$

4.16 Considering the sub-string 'dhwu omj ecpu luvu'

Here '**dhwu omj** e**cpu luvu**' = '**come and** e**ive here**'. Now we can easily complete the sentence as '**come and live here**'

$$mapper : e = l \tag{4.20}$$

4.17 Considering word 'vuwovroaeu'

Here '**vuwov**r**oaeu**'='**remar**r**able**'. Now we can easily guess the word '**remarkable**'

$$mapper : r = k \tag{4.21}$$

4.18 Considering word 'zcbkg'

Here '**z**c**b**k**g**'= '**si**b**ty**'. So the word is '**sixty**'

$$mapper : b = x \tag{4.22}$$

4.19 Considering the sub-string 'kh au yocj shv'

Here '**kh au** y**ocj shv**'='**to be** y**aid for**'. The only way to complete the sentence as '**to be paid for**'.

$$mapper : y = p \tag{4.23}$$

Chapter 5

Key

The key is arranged in English alphabetic order. To be specific, first letter of the key will be replaced for 'a' , second letter of the word will be replaced for 'b' and so on. The blank spaces represent that the alphabet at that position is not used in the above text

key="**bxicl yo dthngavukfwerm ps**"

Chapter 6

Code

```
    class WordFrequency implements Comparable<WordFrequency>{

    String s;
    Integer o;

    public WordFrequency(String s, Integer o) {
        this.s = s;
        this.o = o;
    }

    public int compareTo(WordFrequency o) {
        return o.o.compareTo(this.o);
    }
}
public class DecryptionUsingFrequencyDistribution {
    static void analyzeText () throws IOException
    {
       File fin = new File("data.txt");
       if(!fin.exists())
       {
           fin.createNewFile();
       }
        BufferedReader br = new BufferedReader(new InputStreamReader(new FileInputStr
        String line = null;
```

```
    Map<String, Integer> map = new TreeMap<String, Integer>();
    StringTokenizer st;
    String temp = null;
    while ((line = br.readLine())!=null)
    {
        st = new StringTokenizer(line," ,.:;\"()");

        while(st.hasMoreTokens())
        {
            temp = st.nextToken();
            if(!map.containsKey(temp)){
                map.put(temp,1);
            }
            else {
                int count = map.get(temp);
                map.put(temp, count + 1);
            }
        }
    }
    List<WordFrequency> list = new ArrayList<WordFrequency>();

    for(Map.Entry entry : map.entrySet())
    {
       list.add(new WordFrequency((String) entry.getKey(),((Integer) entry.getVal
    }
    Collections.sort(list);
    for(WordFrequency l: list ){
        System.out.println(l.s+" "+l.o);
    }
    br.close();
}
static void decrypt() throws IOException {
    String keyString = "bxicl yo dthngavukfwerm ps";
    char[] key = keyString.toCharArray();
    File fin = new File("data.txt");
```

```
        File fout = new File("decrypted_data.txt");
        if(!fout.exists())
            fout.createNewFile();
        BufferedReader br = new BufferedReader(new InputStreamReader(new FileInputStr
        BufferedWriter bw = new BufferedWriter(new FileWriter(fout));
        String line = null;
        while ((line = br.readLine())!=null)
        {
            char[] temp = line.toCharArray();
            for(int i=0;i<temp.length;i++) {
                if(temp[i]>='a' && temp[i]<='z') {
                    temp[i] = key[temp[i]-'a'];
                }
            }


            System.out.println(temp);
            bw.write(temp);
            bw.newLine();
        }
        br.close();
        bw.close();


    }
    public static void main(String[] args) throws IOException {

        analyzeText();
        System.out.println("============================================");
        decrypt();

    }
}
```

Chapter 7

Decrypted Text

bilbo was very rich and very peculiar, and had been the wonder of the shire for sixty years, ever since his remarkable disappearance and unexpected return. the riches he had brought back from his travels had now become a local legend, and it was popularly believed, whatever the old folk might say, that the hill at bag end was full of tunnels stuffed with treasure. and if that was not enough for fame, there was also his prolonged vigour to marvel at. time wore on, but it seemed to have little effect on mr. baggins. at ninety he was much the same as at fifty. at ninety-nine they began to call him well-preserved; but unchanged would have been nearer the mark. there were some that shook their heads and thought this was too much of a good thing; it seemed unfair that anyone should possess (apparently) perpetual youth as well as (reputedly) inexhaustible wealth. it will have to be paid for, they said. it isn't natural, and trouble will come of it! but so far trouble had not come; and as mr. baggins was generous with his money, most people were willing to forgive him his oddities and his good fortune. he remained on visiting terms with his relatives (except, of course, the sackvillebagginses), and he had many devoted admirers among the hobbits of poor and unimportant families. but he had no close friends, until some of his younger cousins began to grow up. the eldest of these, and bilbo's favourite, was young frodo baggins. when bilbo was ninety-nine he adopted frodo as his heir, and brought him to live at bag end; and the hopes of the sackville- bagginses were finally dashed. bilbo and frodo happened to have the same birthday, september 22nd. you had better come and live here, frodo my lad, said bilbo one day; and then we can celebrate our birthday-parties comfortably together. at that time frodo was still in his tweens, as the hobbits called the irresponsible twenties between childhood and coming of age at thirty-three