Lab1 (Report)

VATSAL AGRAWAL

Master Of Technology
COMPUTER SCIENCE and ENGINEERING
2nd Sem

Entry Number - 2021MCS2157

NETWORK & SYSTEM SECURITY

SIL 765

Introduction

My programme is working correctly in most of the run. The screenshot of working is in the end. Both ciphertexts decrypted well. Code is not hardcoded; parameters can be passed. I have solved these from seeing various papers and sites. For the initial key to hill climb, I have done frequency analysis and matched it with the usual frequency of letters in English. quadram.txt file is taken from:-http://practicalcryptography.com

The basic idea of code can also be found here. I have used the n-gram technique, specifically, quadgram statics fitness prediction technique to solve this problem along with the hill climb algorithm. The average time taken to run the program is 17 sec for ciphertext1 on my machine (it can go up to 50 sec on some machines) and 14 sec for ciphertext2 (it can go up to 40 sec on some machines) for high accuracy.

To lessen the time, parameters can be passed to low iteration. But this will sometimes give a false result. A detailed description is given below. I have used hill climb algo with some minor modifications to accommodate frequency analysis. Rest hill algo is same to same as given in one screenshot below.

INDEX

Extracted Key

For CipherText - 1

y5n8@p7q1rwu09$342vos6#txz

For CipherText - 2

8ot64spnrxzqwy$193vu205@#7

Note:- Keys can vary for the letter which does not appear in the ciphertext given. This is because these texts never appear, so their position cannot be known.

Decrypted Plain Text

For CipherText - 1

india, officially the republic of india, is a country in south asia. it is the seventh largest country by area, the second most populous country, and the most populous democracy in the world. bounded by the indian ocean on the south, the arabian sea on the southwest, and the bay of bengal on the southeast, it shares land borders with pakistan to the west; china, nepal, and bhutan to the north; and bangladesh and myanmar to the east. in the indian ocean, india is in the vicinity of sri lanka and the maldives; its andaman and nicobar islands share a maritime border with thailand, myanmar and indonesia. good, now turn for the second part of the question, good luck!

For CipherText - 2

defeated and leaving his dinner untouched, he went to bed. that night he did not sleep well, having feverish dreams, having no rest. he was unsure whether he was asleep or dreaming. conscious, unconscious, all was a blur. he remembered crying, wishing, hoping, begging, even laughing. he floated through the universe, seeing stars, planets, seeing earth, all but himself. when he looked down, trying to see his body, there was nothing. it was just that he was there, but he could not feel anything for just his presence.

Approach / Observation

1. I have consulted several papers and sites (mentioned in Reference).
2. The best approach I found to solve this problem is using the hill climb algorithm (described below) using quadgram statistic as a fitness score.
3. For the initial key, I have done frequency analysis and matched it with **“etaoinshrdlcumwfgypbvkjxqz”** these are ordered according to the frequency in English text from left to right, as mentioned on WIKIPEDIA.
4. I have used the os library to find the absolute path of quadgram.txt.
5. Also, the code is not hardcoded, and up to 3 (or less than three or none) arguments can be passed.
6. The first argument is the absolute path of ciphertext to be decrypted.
7. The second Argument is the no of iteration the hill climb should run.
8. The third argument is the maximum no of times a hill climb algorithm should swap characters.
9. Time library is for seeding random
10. Math library is for importing log 10 so that in multiplication part of calculating fitness score float do not overflow (to reduce overflow error)
11. First, I have read the ciphertext file and stored it in a variable.
12. Then I have opened the quadgram file and read the keys and values and stored them in the dictionary.
13. Also, for finding the probability of the key, I have found the sum of all values of frequency of key and divided each of them by this.
14. If some word in ciphertext cannot be found, then instead of assigning them probability 0 (which will cause an error in the log), I have assigned them very low probability, assuming their frequency to be 0.01.
15. Also, we have calculated the log probability of key as it will help us to multiply two probabilities easily without overflowing in finding scores.
16. The prediction score is calculated using the n-gram technique as described below. In this, if a quad is found in the quadgram file, then its log probability is added to the total score; otherwise, its quadgram score is taken to be almost 0, and 0 is added to the total score.
17. We use log probability to use addition instead of multiplication which we may have used when using probability only
18. In the frequency analysis part, I have maintained a dictionary of cipher letters and read ciphertext, and increased word count as they appear.

19. Then I have matched with the above given standard frequency and made an initial key. (In the order from a to z).
20. Now I have passed this initial key to our hill climb iteration loop algo.
21. In each main iteration of this algo, I first decrypt the ciphertext with the key given to it and find its fitness score (Prediction score).
22. Then this algo has sub iteration (child) in which we randomly pick two numbers and swap the letter in key corresponding to them; then, we again find the prediction score using this new key.
23. If fitness score of this sub iteration is more than that of parent, then we replace parent key with these new key and restart the sub iteration from initial and if not greater, then we proceed this child iteration till fixed no of times.
24. After a child operation has run a fixed no of times without replacing the parent key, then we check that original key score and parent key score and replace the original key with the parent key if the score is high.
25. Also, we shuffle our parent key and start the next main iteration following the same steps.
26. Finally, we print our desired result after the final iteration
27. Output can be written to file by uncommenting lines 166, 24, and 35 for decryption
28. On the frequent run, it is found that no of main iteration greater than 14 always give the right answer for our ciphertext given
29. The main iteration above 8 gives the correct answer with probability of 95%
30. Child iteration should be ranging from 1000 to 1200.
31. High child iteration leads to predicting other quads that can be possible and consuming more time.
32. Low child iteration may not find all possible swaps.
33. I have chosen a number of main iterations as eight and child iteration as 1000. These values are similar to some of the basic hill climb programs available online as they give high accuracy.
34. Running time of decrypting/ extracting key of ciphertext2 is on average 14 seconds on terminal and 30 sec on IDE (some system may take upto 1 minute).
35. Running time of decrypting/ extracting key of ciphertext 2 is on average 17 second on terminal and 50 sec on IDE (some system may take upto 1 minute).
36. Keys can vary for the letter which does not appear in the ciphertext given.
37. I have also tried an approach where keys are swapped in a left to the right manner, but time was very high for high accuracy. So I have used the random method as mentioned in hill climb algo.

38. I have also tried a naïve method of only frequency analysis using 1 and 2 letter words, but accuracy was very low as compared to hill climb algo.
39. Also working of hill climb algo can be checked easily by going on a website like guballa.com and changing a non-English letter to English with Monosubstitution

Hill Climb Algorithm

In this algorithm, when a person wants to climb a hill, start from some initial point and analyze all paths available to him and take less descent path and then repeat the same step after going some short path. Similarly, in this, we first find fitness score using the initial key and see swaps that can be made in our key, which can make our fitness score higher from the initial. If no such swap can be found, then we change our initial key. Each main iteration is like starting a hill climb from some different initial point, and each sub iteration is like finding a direction to go from some point. Each sub child will restart a new child iteration if it finds some swap that increases fitness score and updates key. Otherwise, it will continue and find a new swap.

Three screenshots of algo from 3 of the research paper are given below in the screenshot section.

Quadgram Statics Fitness Score

For matching the similarity of decrypted text with English words, we use this. It is a famous NLP technique. ABCDE has ABCD, BCDE as its quadgram. It is based on the technique to find the probability/frequency of all quadgram appearing in English text. To detect a word is of English, we took that word and found all its quadgram and multiplied their probability. The higher the probability, the more is the chance that word is in English. As multiplication is a costlier/faulty process so we can use the log probability of it, which uses addition. Also, as the log is not defined for 0, that is, when a word does not seem to be of English, then it is taken to be very small.

Frequency analysis

While seeing ciphertext for mono-substitution, we can see some letters occur more than others. This is a similar trend seen in English text. So we can map the most frequently occurring word in English to the frequently occurring word in cipher. Also, we can see the single letter and double letter words to find some similarities with the English word.

Pre-Requisite

How To Make Executable and Run Programme

1) We can use the **make** and **make all** command to see our program running
2) **make all1** and **make all2** will extract the key and decrypt program for ciphertext 1 and 2, respectively
3) **make extract1** and **make extract2** will extract the key for ciphertext 1 and 2, respectively
4) **make decrypt1** and **make decrypt2** will extract the key for ciphertext 1 and 2, respectively
5) You can also run **python extractKey.py** to extract key
6) You can also run **python decryptText.py** to extract key
7) Up to 3 arguments (0 or 1 or 2 or 3) can be passed to the above two command
8) The first Argument is the Absolute path of CipherText
9) The second argument is the number of the main iteration
10) The third argument is the number of child iteration

Structure Of Programme

Program is made up of directory structure which includes

1) decryptText – code to decrypt the cipher text
2) extractKey – code to extract key
3) Makefile – Makefile command
4) INPUT – folder containing input ciphertext
5) OUTPUT – folder containing output plaintext
6) Env.txt – Containing Environment info on which program was tested.

Screenshot of Result with time

```
baadalvm@vatsal:~/NSS/Lab1/2021MCS2157-assignment-1$ time python decryptText.py
8ot64spnrxzqwy$1@3vu2057#9
defeated and leaving his dinner untouched, he went to bed. that night he did not sleep
 well, having feverish dreams, having no rest. he was unsure whether he was asleep or
dreaming. conscious, unconscious, all was a blur. he remembered crying, wishing, hopin
g, begging, even laughing. he floated through the universe, seeing stars, planets, see
ing earth, all but himself. when he looked down, trying to see his body, there was not
hing. it was just that he was there, but he could not feel anything for just his prese
nce.

real    0m12.638s
user    0m12.007s
sys     0m0.188s
baadalvm@vatsal:~/NSS/Lab1/2021MCS2157-assignment-1$ time python decryptText.py /home/
baadalvm/NSS/Lab1/2021MCS2157-assignment-1/INPUT/ciphertext-2.txt
8ot64spnrxzqwy$173vu205@#9
defeated and leaving his dinner untouched, he went to bed. that night he did not sleep
 well, having feverish dreams, having no rest. he was unsure whether he was asleep or
dreaming. conscious, unconscious, all was a blur. he remembered crying, wishing, hopin
g, begging, even laughing. he floated through the universe, seeing stars, planets, see
ing earth, all but himself. when he looked down, trying to see his body, there was not
hing. it was just that he was there, but he could not feel anything for just his prese
nce.

real    0m13.218s
user    0m12.983s
sys     0m0.113s
```

Fig1. Time and plaintext for ciphertext -2

```
baadalvm@vatsal:~/NSS/Lab1/2021MCS2157-assignment-1$ time python decryptText.py /home/
baadalvm/NSS/Lab1/2021MCS2157-assignment-1/INPUT/ciphertext-1.txt
y5n8@p7q1twu09$342vos6#zxr
india, officially the republic of india, is a country in south asia. it is the seventh
 largest country by area, the second most populous country, and the most populous demo
cracy in the world. bounded by the indian ocean on the south, the arabian sea on the s
outhwest, and the bay of bengal on the southeast, it shares land borders with pakistan
 to the west; china, nepal, and bhutan to the north; and bangladesh and myanmar to the
 east. in the indian ocean, india is in the vicinity of sri lanka and the maldives; it
s andaman and nicobar islands share a maritime border with thailand, myanmar and indon
esia. good, now turn for the second part of the question, good luck!

real    0m16.901s
user    0m15.947s
sys     0m0.317s
baadalvm@vatsal:~/NSS/Lab1/2021MCS2157-assignment-1$ time python decryptText.py /home/
baadalvm/NSS/Lab1/2021MCS2157-assignment-1/INPUT/ciphertext-1.txt
y5n8@p7q1zwu09$342vos6#txr
india, officially the republic of india, is a country in south asia. it is the seventh
 largest country by area, the second most populous country, and the most populous demo
cracy in the world. bounded by the indian ocean on the south, the arabian sea on the s
outhwest, and the bay of bengal on the southeast, it shares land borders with pakistan
 to the west; china, nepal, and bhutan to the north; and bangladesh and myanmar to the
 east. in the indian ocean, india is in the vicinity of sri lanka and the maldives; it
s andaman and nicobar islands share a maritime border with thailand, myanmar and indon
esia. good, now turn for the second part of the question, good luck!

real    0m16.580s
user    0m16.089s
sys     0m0.145s
baadalvm@vatsal:~/NSS/Lab1/2021MCS2157-assignment-1$
```

Fig2. Time and plaintext for ciphertext -1

Screenshot of Hill Climb Algo

```
plaintext = decrypt (ciphertext, parent)
bestfit = fitness (plaintext)
bigcount = 0

while bigcount < BIGLIMIT do
        for i in 1, ..., period do
                parent[i] = randomkey()
                plaintext = decrypt (ciphertext, parent)
                parentfit = fitness (plaintext)
                count = 0
                while count < LIMIT do
                        child = parent
                        j = random() mod 26
                        k = random() mod 26
                        child[i][j], child[i][k] = child[i][k], child[i][j]
                        plaintext = decrypt (ciphertext, child)
                        childfit = fitness (plaintext)
                        if childfit > parentfit then
                                parent = child
                                parentfit = childfit
                                count = 0
                        else
                                count++
                        if childfit > bestfit then
                                bestfit = childfit
                                bestkey = child
```



```
                        bigcount = 0
                else
                        bigcount++

output bestkey, decrypt (ciphertext, bestkey)
```

```
for a pre-selected number of iterations do
key = getRandomPermutation(0,25)
        bestIterationKey = key #currently best it. key
        fitnessOfkey = getFitness(decrypt(cipherText, key))
        while true do
                newKey = generateNewKey()
                fitOfNewKey = getFitness(decrypt(cipherText,newKey))
                if fitOfNewKey > fitnessOfKey then
                        #new key and its fitness are memorized
                        #as the best ones for this iteration
                        fitnessOfKey = fitOfNewKey
                        bestIterationKey = newKey
                end if
                #break if better key hasn't been found in T steps
        end while
        #Update best ever key if bestIterationKey is better
end for
```

Fig3,4,5. Hill Climb Algo – Implemented with minor modification

The hill-climbing algorithm looks like this:

1. Generate a random key, called the 'parent', decipher the ciphertext using this key. Rate the fitness of the deciphered text, store the result.
2. Change the key slightly (swap two characters in the key at random), measure the fitness of the deciphered text using the new key.
3. If the fitness is higher with the modified key, discard our old parent and store the modified key as the new parent.

References

I have completed my assignment by learning from the following sources

1 http://practicalcryptography.com/cryptanalysis/text-characterisation/quadgrams/#a-python-implementation

2 http://practicalcryptography.com/cryptanalysis/stochastic-searching/cryptanalysis-simple-substitution-cipher/https://iitd-plos.github.io/os/2020/lec/l25.html

3 https://machinelearningmastery.com/calculate-bleu-score-for-text-python/

4 https://gitlab.com/guballa/SubstitutionBreaker

5 https://eprint.iacr.org/2020/302.pdf

6 https://www.montis.pmf.ac.me/vol44/11.pdf

7 https://www.researchgate.net/publication/340633483_Hill-climbing_cipher

Future Scope

The algorithm can be modified by some new methods like simulated annealing, particle swarm optimization, etc.

The algorithm can use trigram, bigram also with some weightage.

Some patterns can be found in my work that can enhance the stability of work.