Patrick Walter
EECS 765: Cryptography
Project 1: Vigenere

Vigenere Cipher and Brute Force Attack

Abstract

A Vigenere cipher was written to take in a string and a key and encode it using the Vigenere shift cipher. Next a brute force algorithm was developed so that a user could enter a cipher text, first word length and key length to the algorithm. C++ was used to code the algorithm for maximum efficiency.

Results – Rudimentary Implementation:

1. **Cipher Text 1:**
    a. MSOKKJCOSXOEEKDTOSLGFWCMCHSUSGX
    b. **Key 1:** KS
    c. **Timer1:** 746 ns (Optimized Dictionary: 506 ns)
    d. **Plain Text:** Caeser's wife must be above suspicion.

2. **Cipher Text 2:**
    a. OOPCULNWFRCFQAQJGPNARMEYUODYOUNRGWORQEPVARCEPBBS CEQYEARAJUYGWWYACYWBPRNEJBMDTEAEYCCFJNENSGWAQRTSJ TGXNRQRMDGFEEPHSJRGFCFMACCB
    b. **Key:** JAY
    c. **Timer:** 12 ms (Optimized Dictionary: 10.2 ms)
    d. **Plain Text:** Fortune which has a great deal of power in other matters but especially in war can bring changes in a situation on through very slight forces.

3. **Cipher Text 3:**
    a. MTZHZEOQKASVBDOWMWMKMNYIIHVWPEXJA
    b. **Key:** IWKD
    c. **Timer:** 361.5ms (Optimized Dictionary: 316 ms)
    d. **Plain Text:** Experience is the teacher of all things.

4. **Cipher Text 4:**
    a. HUETNMIXVTMQWZTQMMZUNZXNSSBLNSJVSJQDLKR
    b. **Key:** ZIENF
    c. **Timer:** 28.5 seconds (Optimized Dictionary: 24.9 seconds)
    d. **Plain Text:** Imagination is more important than knowledge.

5. **Cipher Text 5:**

a. LDWMEKPOPSWNOAVBIDHIPCEWAETYRVOAUPSINOVDIEDHCDSEL HCCPVHRPOHZUSERSFS
b. **Key:** HACKER
c. **Timer:** 192 seconds (Optimized Dictionary: failed)
d. **Plain Text:** Education is what remains after one has forgotten what one has learned in school.

6. **Cipher Text 6:**
    a. VVVLZWWPBWHZDKBTXLDCGOTGTGRWAQWZSDHEMXLBELUMO
    b. Unsuccessful

Results – Intelligent Implementation:

1. **Cipher Text 1:**
    a. Timed out after 797 second having not found a solution
2. **Cipher Text 2:**
    a. 843 ms
3. **Cipher Text 3:**
    a. 8.1 seconds
4. **Cipher Text 4:**
    a. 135 seconds
5. **Cipher Text 5:**
    a. Timed out after 794 seconds
6. **Cipher Text 6:**
    a. Unsuccessful

Design

The original design was to enumerate all of the possible keys for a given key length and try each other against the given dictionary. This was the most straightforward and rudimentary approach. After deciphering all but one of the cipher texts, algorithm was modified to optimize time.

First Optimization:

Since the first word length was given, the dictionary loader was modified to only load words that were the same length as the first word. This only affected the execution time minimally. Since I used a *set* to hold and search the dictionary, the complexity of the find algorithm was O(nlog(n)). Thus a smaller dictionary only slightly decreased time.

Second Optimization:

Since no word in the English dictionary has three consecutive equivalent letters, when enumerating the keys, such keys were immediately discarded. This

made the decryption process faster but in turn made the key enumeration algorithm slower. The full effect of this change has not been identified.

Third Optimization:

The third optimization deals with anticipating the usefulness of a key given the first few letters in the key. For example, if “qu” is the beginning of an English word but “quu” is not, then obviously the key that generates this sequence of letters is invalid.

The idea was to recognize when a key was generating an impossible word and immediately through it out. This optimization would not work if the keys were already generated. Thus, a recursive algorithm was used and its function is descried below

1. Generate keys of length 2
2. Decipher the first 2 letters with each key
3. If the deciphered word results in the start of a valid word then enumerate this key, otherwise discard the key
4. Once the length of the enumerated keys is the same as given key length then use each key to decipher the first word.

Comments

As it turns out, the third optimization actually resulted in longer decryption times. But the first optimization resulted in better times. Part of the reason why the optimized times are slower is because they include the time for enumerating the keys while the other tests did not.