Project 1 (Cryptanalysis of a class of ciphers)

Robert Frost Michael Chen Yin Tian

Table of Contents

[1. Introduction 3](#_Toc116286758)

# Introduction

## 1.1 Team Members

Robert Frost implemented our solution for the known plain text attack and our attempt to use statistical analysis for the second decryption approach. Michael formulated a solution utilizing a hash table to track character differences and a brute force approach to deal with random insertions. Yin Tian helped Robert design the statistical analysis approach.

## 1.2 Approach

Our group designed to methods for extracting the plaintext, one naïve method that relies upon known plaintexts and a second statistical method. The naïve approach works for the first test when one does not consider the random insertions. The second approach, while we believe has merit, did not fully solve test two.

# 2. Informal Explanation

## 2.1 Naïve Approach

The naïve approach relies on knowing the full universe of potential plain texts that might be enciphered to create the cipher text. It also relies on the encryption key being linearly and cyclically applied. The algorithm iterates through all potential key lengths and all potential plain texts. For a given key length k, it then selects each “kth” character from both the plain text and cipher text. It compares how much each character in the plain text would have to shift to achieve the cipher text. If all shifts are the same, we have found a potential key length. All possible key lengths are then used to build a candidate key from each plain text and the cipher text. That key is then applied to the entire message to determine if we have found the actual correct plain text. This approach scales only moderately well as the universe of plain text candidates grows. The algorithm completes in four seconds at a thousand candidates, 21 seconds at ten thousand and jumping to nearly 5 minutes for a hundred thousand.

## 2.2 Statistical Approach

Our second approach attempts to use the normal distribution of characters within an English text and repeated N-grams within the cipher text to attempt to identify the key without relying on knowledge of the plain text. This is because there are far too many combinations of words (roughly 4\*10^106) from the second dictionary to allow the first approach to be successful. First, our algorithm finds all character sequences of length three that repeat in the cipher text. It then calculates the gaps between those characters repeating, theorizing that a repeat might indicate the same three-character plain text being enciphered by the same part of the key. All found gaps are then compared to all possible key lengths, assuming that key lengths that “fit” more gaps are more likely to be the key length and returning a prioritized list of key lengths to attempt.

Each potential key length is then used to build groupings of characters which are then brute forced for potential key values. The generated decryption attempts are analyzed to determine rough similarity to English and each grouping of characters adds potential key values. Finally, each individual key value possibility is attempted to generate full decrypted texts which are again analyzed. The candidate plain text which scores the most similar to English is returned as the plain text guess.

# 3. Rigorous Explanation

## 3.1 Naïve Approach

Pseudo code / explain

## 3.2 Statistical Approach

Pseudo code/explain

# 4. References

Sweigart Al. *freqAnalysis.Py*. Github: 2013. Retrieved October 3, 2022. <https://github.com/asweigart/codebreaker/blob/master/freqAnalysis.py>