Vigenere Cipher - Documentation

Author, Developer: Zoheb Hasan, Stony Brook University, Computer Science - 114687894
September, 17th, 2024

Solution Reference: Vigenere Cipher - Decryption (Unknown Key)

1. Overview

In this documentation, I will explain the implementation of Vigere Cipher. More specifically, I will discuss encryption, decryption, breaking the encryption with the key length, and, lastly, breaking the encryption without knowledge of the key. As I declared the above, I technically did not “Invent” this solution, the solution and the architecture below are a combination of the video I have shared above and a lot of StackOverflow posts.

Main Components

- `Encryption` class.
- `Decryption` class.
- `CrackWithKeyLen` class.
- `CompleteCrack` class.

2. Key Classes & Methods

2.1 Encryption Class

This is the class responsible for encryption. Instead of using a two-dimensional matrix (int[][] board), I chose to use Java’s StringBuilder class, which is a very popular class from building a string.

Attributes

- `None`

Key Methods

- **`encrypt(String plainText, String key)`**
  `return: String`

2.2 Decryption Class

This class is responsible for the decryption of the Vigenere Cipher given the String Encrypted Message and String Key.

Attributes

- none

Key Methods

- decrypt (String cipherText, String key)

2.3. CrackWithKeyLen Class

This class contains some of the most complicated functionalities in this entire project. CrackWithKeyLen class breaks any encrypted message given the length of the key.

Attributes

- **`[]ENGLISH_FREQ:`** A hardcoded array of the current frequencies of English letters starting from A to Z.

Key Methods

- **`divideEncryptedText(String text, int len):`**
  `Return: String[] groups`
  This method is responsible for splitting the text into n groups where n = len, such that, each group[i] contains the characters that are on the text given charAt(i).
- **`shiftedText(String text, int shift):`** This method shifts a string of characters given the number of positions in the Alphabet which can later be used to conduct frequency analysis.
- **`findCharByFrequency(String group):`** Finds the most likely character in the key for a particular group by comparing letter frequencies.
- **`calculateFrequencyMatchScore(input, index):`** This method calculates how well a shifted group matches English letter frequencies.

- **`reinsertSpecialCharacters(String originalCipherText, String decryptedText):`** This method is pretty straightforward, all it does is compare the format of the original string with the current decrypted text and generate a new text that shares the same properties, structure, and format as the original one.

2.4. CompleteCrack Class

This class is responsible for breaking the encryption of the Vigenere Cipher given only the encrypted text. What this class primarily does is that it calculates the index coincidence. The logic of this class is much simpler than I thought it would be. All this class does is put the characters on a map, and look for the coincidences. And, the highest co-incidence is the key.

Dependencies

- CrackWithKeyLeng (2.3)
- Decryption (2.2)

Attributes

- none

Key Methods

- **`calcIndexCoincidence(String text):`** This method calculates the index of coincidence (a measure of how frequently letters repeat) for a given text string.
- **`estimateKeyLen(String text):`** This estimates the key length by analyzing the average index of coincidence across possible key lengths.

3. Key Classes & Methods

3.1. Encryption

The encryption class uses the StringBuilder to construct the encrypted message efficiently. The logic iterates through the characters of the plaintext and key and converts each to it's corresponding character in the Vigenere Cipher. The key is repeated as needed to match the length of the plaintext and alphabetic characters are shifted based on the corresponding key character. My initial plan was to use a two-dimensional array, but then after doing some research on the internet, I realized that I could get a similar behavior using the modulus operator.

3.2. Decryption

The decryption class follows the reverse process of encryption, where each character in the ciphertext is shifted back by the corresponding key character to retrieve the original message. The decrypt method ensures that non-alphabetic characters remain unchanged during the process.

3.2. Breaking the cipher with known key length

In the CrackWithKeyLen class, we utilize the fact each character is in a vigenere cipher by a consistent key. By dividing the ciphertext into groups of characters corresponding to each key position, we can perform frequency analysis on each group. The findCharByFrequency method compares the frequency of letters in each group against known English letter frequencies to find the most likely shift (Key Character). The key is gradually built by combining these characters for each position in the key.

The method reinsertSpecialCharacters ensures that non-alphabetic characters,which were ignored during the encryption and decryption process, are properly restored to their original positions.

3.4. Breaking the cipher with no knowledge of the key

The CompleteCrack class handles the more complex task of breaking the cipher without any prior knowledge of the key or its length. The calcIndexCoincidence method estimates the key length by computing the index of coincidence for different groupings of the ciphertext. The estimated key length is then used with the CrackWithKeyLen class to discover the key.

Once the key is identified, the text is decrypted using the Decryption class, and the original message is reconstructed with all non-alphabetic characters properly restored.