Cifra de Vigenere

Thiago Tokarski 190096063

May 6, 2023

Abstract

Este documento é um relatório que descreve o algoritmo de Vigenere, sua implementação como também a implementação de ataque à cifra.

1 Introdução

O algoritmo de cifra de Vigenere é uma cifra clássica de substituição que utiliza séries de cifras de César para criar uma substituição polialfabética. Charles Babbage é conhecido por ter quebrado a cifra, mas nunca publicou sobre isso. Kasiski, por sua vez, quebrou completamente a cifra e publicou a respeito no século 19.

Neste documento, examinamos sua implementação por meio da linguagem de programação Python e também a implementação de um ataque utilizando análise de frequência - a maior fraqueza dessa cifra.

2 Funcionamento da cifra

2.1 Cifração

Como citado anteriormente, se trata de uma cifra de substituição e portanto temos que uma mensagem qualquer (M) que será mapeada de acordo com uma chave (K) para uma nova mensagem cifrada (C). Para melhor entendimento podemos trocar as letras por números, onde A = 0, B = 1, C = 2 ... Então, é somado o valor do carácter puro M_i com o carácter da chave K_i, divide-se esse valor por 26 de forma a obter o resto C_i e mantermos o mapeamento para letras do alfabeto.

$$C_i = M_i + K_i \ mod \ 26$$

Como nem sempre a chave terá o tamanho da mensagem, podemos reescrever da seguinte forma:

$$C_i = M_i + K_{i \ mod \ len(K)} \ mod \ 26$$

2.2 Decifração

A decifração segue de forma bem intuitiva a seguinte equação:

$$M_i = C_i - K_{i \ mod \ len(K)} \ mod \ 26$$

2.3 Exemplo

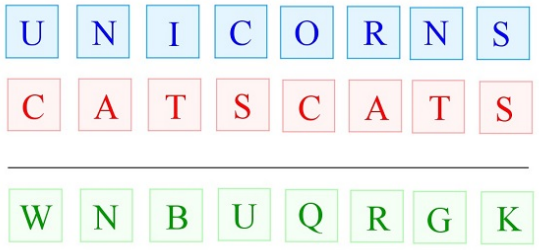

Figure 1: Exemplo de cifração.

3 Ataque

Para a quebra da cifra, primeiro foi necessário obter o tamanho da chave. Para isso, foi utilizado o método do índice de coincidência. Em seguida, o texto cifrado foi dividido em grupos e, para cada grupo, foi feita uma análise da frequência dos caracteres daquele grupo. Essa frequência foi rotacionada até que coincidisse com a frequência dos caracteres em português ou em inglês, dependendo do desafio. Então, o número de rotações seria o carácter da chave que estamos buscando.

3.1 Tamanho da chave

Como mencionado, a técnica para obter o tamanho da chave foi a do índice de coincidência. Este índice representa a probabilidade de quaisquer dois caracteres em um texto serem iguais. Na equação a seguir, n_i representa o total de aparições da letra de valor i no texto e N representa o total de caracteres.

$$IoC = 26 * \sum_{0}^{25} n_i(n_i - 1)/N(N - 1)$$

Cada língua tem seu índice de coincidência. Para este projeto, foi suficiente considerar o índice de coincidência como 1.6 tanto para o português quanto para o inglês.

Então, foram testados diferentes tamanhos de chave dividindo a cifra em grupos de acordo com esse tamanho. Para cada grupo, foi mensurado seu índice de coincidência e, se a média dos índices de coincidência daquele tamanho fosse maior que o valor determinado (1.6), seria considerado que aquele tamanho é o tamanho da chave.

Segue exemplo da divisão dos grupos para um teste de tamanho 3:

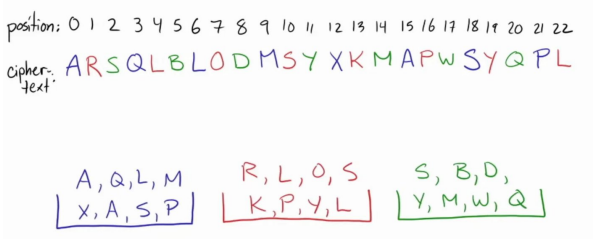

Figure 2: Exemplo de divisão dos grupos.

A divisão ocorre dessa forma porque, se a chave tivesse de fato tamanho 3, significaria que a cada três letras de uma determinada posição se repetiria o deslocamento utilizado e, portanto, seria possível verificar a frequência dos caracteres para aquele grupo.

3.2 Análise de frequência

Para cada grupo foi então obtida a frequência dos caracteres no grupo e então essa frequência é rotacionada e comparada com a frequência disponibilizada pelo professor até que esteja próxima o suficiente.

Para fazer essa comparação, é utilizado como medida o produto escalar entre o vetor das frequências do grupo e o vetor das frequências do português ou inglês

$$dot\_product = \sum_{0}^{25} P_i * G_i / \sum_{0}^{25} (\sqrt{P_i * P_i})(\sqrt{G_i * G_i})$$

Se esse dot_product for maior que 0.9, então consideramos que está próximo o suficiente e, portanto, a quantidade de rotações que fizemos para o grupo chegar a essa proximidade é o valor do caractere da chave

4 Implementação

Para a implementação, foram criadas duas classes: Vigenere, que contém o código para a cifração e decifração de um texto com uma chave; e a classe VigenereCrack, que implementa a quebra da cifra e todos os resultados auxiliares, como o índice de coincidência e o produto escalar.

Além desses códigos, tem-se também uma função auxiliar que limpa o texto recebido, retirando pontuações, caracteres especiais, números e espaços.

A implementação em linguagem Python foi bem tranquila, principalmente devido à sua flexibilidade de tipos e tratamento de dados.

5 Conclusão

A cifra e sua quebra foram implementadas com sucesso. A parte mais desafiadora deste trabalho foi definitivamente obter o tamanho da chave, já que as noções de grupos e índice de coincidência estavam bastante abstratas. Inicialmente, pretendia-se que a implementação fosse feita em Java, a fim de ter a oportunidade de aprender a linguagem, mas a falta de experiência tornou o trabalho mais difícil. Então, por falta de organização e tempo, optou-se por implementar em Python. Em conclusão, a oportunidade de trabalhar na implementação e quebra de uma cifra certamente trouxe novas perspectivas e curiosidade sobre os mecanismos de criptografia disponíveis atualmente

References

[noa] Five Ways to Crack a Vigenère Cipher.

[noa15] Breaking the Vigenère cipher. . . , April 2015.

[noa22a] Frequência de letras, September 2022.

[noa22b] Index of coincidence, December 2022.

[Sta20] Katherine Stange. Cryptanalysis of Vigenere cipher: not just how, but why it works, August 2020.

[Wik23] Wikipedia. Vigenère cipher, March 2023.