Universidade de Brasília

FACULDADE DE TECNOLOGIA

Trabalho de Implementação 1

Cifra de Vigenère

AUTORES:

Lorena Gomes de Freitas Nascimento - 200040278
Vitor Kohara Guerra - 211027349

Versão 1.0

Brasília – DF
2024

Sumário

1. A Cifra de Vigenère

A cifra de Vigenère é um método de criptografia que aplica uma sequência de diferentes cifras de César baseada nas letras de uma senha previamente combinada entre o remetente e o destinatário da mensagem. A cifra é baseada no conceito de chaves, o que a tornou um marco no campo da criptografia, uma vez que as mensagens cifradas com sua ajuda são muito mais seguras.

Para ser aplicada, uma tabela de alfabetos específica, chamada de "grade de Vigenère" [Figura 1], é usada. Ela é composta por letras maiúsculas de A a Z, cada linha representa uma variação do alfabeto, sendo deslocada ciclicamente em relação à linha anterior. Dessa forma, a tabela contém 26 possíveis cifras de César, uma para cada linha.

Durante a aplicação da cifragem, cada letra da mensagem é associada a uma letra da palavra-chave, o ponto de interseção entre a coluna da letra da mensagem e a linha da letra da chave resulta na letra cifrada. Por exemplo, se a mensagem tem a letra "G" e na chave a letra correspondente for a "L", a letra criptografada será a "R" que é a inteserção entre as duas na tabela.

Este método é relativamente fácil de implementar e fornece um nível extra de complexidade em comparação com cifras diretas, tornando a mensagem menos propensa a ataques diretos. Contudo, a cifra de Vigenère está aberta a ataques de análise de frequência, pois a repetição da chave permite a identificação de padrões e decifragem do código.

Algebricamente, a cifra de Vigenère pode ser expressa usando operações de módulo. Supondo que cada letra do alfabeto mapeia para o intervalo de inteiros de 0 a 25, a criptografia de uma mensagem pode ser expressa como $Ci = Pi + Ki\ (mod\ 26)$, onde Pi é a letra da mensagem e Ki é a letra da chave. O processo de descriptação é realizado usando a fórmula inversa: $Pi = Ci - Ki + 26\ (mod\ 26)$. Embora não seja amplamente usada atualmente devido à sua vulnerabilidade a ataques criptográficos, a cifra de Vigenère continua sendo um método clássico e influente na história da criptografia e fornece a base conceitual para cifras modernas baseadas em chaves.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Tabela de Vigenère. Figura 1. Fonte: https://pt.wikipedia.org/wiki/Cifra_de_Vigen%C3%A8re

Para quebrar a cifra de Vigenère, a primeira coisa a se fazer é encontrar o tamanho da chave localizando blocos de caracteres que se repitam na mensagem. Em seguida, através da diferença de

posição entre as ocorrências de repetição, calcula-se o maior número que tem divisão exata com essas ocorrências, assim encontra-se o tamanho da chave. Com o tamanho da chave é possível separar a mensagem em "fatias" e determinar alguns pontos da criptografia. A partir disso é aplicada a Análise de Frequência que tem como objetivo deduzir a letra a partir da frequência de repetição da letra criptografada. Sabendo o idioma que a mensagem foi escrita, sabe-se a letra que é usada com mais frequência e com isso, a chave pode ser descoberta.

A Figura 2 mostra a frequência das letras no Português e no Inglês. Nota-se que as letras com maior frequência no Português são A e E. Já no inglês são as letras E e T.

Letra	Frequência
a	14.63%
b	1.04%
c	3.88%
d	4.99%
e	12.57%
f	1.02%
g	1.30%
h	1.28%
i	6.18%
j	0.40%
k	0.02%
l	2.78%
m	4.74%

Letra	Frequência
n	5.05%
o	10.73%
p	2.52%
q	1.20%
r	6.53%
s	7.81%
t	4.34%
u	4.63%
v	1.67%
w	0.01%
x	0.21%
y	0.01%
z	0.47%

Letra	Frequência
a	8.167%
b	1.492%
c	2.782%
d	4.253%
e	12.702%
f	2.228%
g	2.015%
h	6.094%
i	6.966%
j	0.153%
k	0.772%
l	4.025%
m	2.406%

Letra	Frequência
n	6.749%
o	7.507%
p	1.929%
q	0.095%
r	5.987%
s	6.327%
t	9.056%
u	2.758%
v	0.978%
w	2.360%
x	0.150%
y	1.974%
z	0.074%

Frequência de Letras no Português e Inglês. Figura 2. Fonte: Frequência de letras – Wikipédia, a enciclopédia livre

2. Implementação da Cifra

Para realizar as operações de cifragem e decifragem, foi utilizada a tabela ASCII (American Standard Code for Information Interchange) a qual possui 128 caracteres e símbolos os quais estão associados a um certo byte, como é mostrado na figura 4. Sendo que só foram utilizadas as letras do alfabeto minúsculas.

ASCII Table

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	[space]	64	40	100	@	96	60	140	`
1	1	1		33	21	41	!	65	41	101	A	97	61	141	a
2	2	2		34	22	42	"	66	42	102	B	98	62	142	b
3	3	3		35	23	43	#	67	43	103	C	99	63	143	c
4	4	4		36	24	44	$	68	44	104	D	100	64	144	d
5	5	5		37	25	45	%	69	45	105	E	101	65	145	e
6	6	6		38	26	46	&	70	46	106	F	102	66	146	f
7	7	7		39	27	47	'	71	47	107	G	103	67	147	g
8	8	10		40	28	50	(	72	48	110	H	104	68	150	h
9	9	11		41	29	51	)	73	49	111	I	105	69	151	i
10	A	12		42	2A	52	*	74	4A	112	J	106	6A	152	j
11	B	13		43	2B	53	+	75	4B	113	K	107	6B	153	k
12	C	14		44	2C	54	,	76	4C	114	L	108	6C	154	l
13	D	15		45	2D	55	-	77	4D	115	M	109	6D	155	m
14	E	16		46	2E	56	.	78	4E	116	N	110	6E	156	n
15	F	17		47	2F	57	/	79	4F	117	O	111	6F	157	o
16	10	20		48	30	60	0	80	50	120	P	112	70	160	p
17	11	21		49	31	61	1	81	51	121	Q	113	71	161	q
18	12	22		50	32	62	2	82	52	122	R	114	72	162	r
19	13	23		51	33	63	3	83	53	123	S	115	73	163	s
20	14	24		52	34	64	4	84	54	124	T	116	74	164	t
21	15	25		53	35	65	5	85	55	125	U	117	75	165	u
22	16	26		54	36	66	6	86	56	126	V	118	76	166	v
23	17	27		55	37	67	7	87	57	127	W	119	77	167	w
24	18	30		56	38	70	8	88	58	130	X	120	78	170	x
25	19	31		57	39	71	9	89	59	131	Y	121	79	171	y
26	1A	32		58	3A	72	:	90	5A	132	Z	122	7A	172	z
27	1B	33		59	3B	73	;	91	5B	133	[	123	7B	173	{
28	1C	34		60	3C	74	<	92	5C	134	\	124	7C	174	|
29	1D	35		61	3D	75	=	93	5D	135	]	125	7D	175	}
30	1E	36		62	3E	76	>	94	5E	136	^	126	7E	176	~
31	1F	37		63	3F	77	?	95	5F	137	_	127	7F	177	

Tabela ASCII. Figura 3. Fonte: https://br.pinterest.com/pin/620370917392853851/

A função de cifra () realiza a operação de cifragem de uma mensagem a partir de uma certa chave. Para fazer isso foi feito um loop for, mostrado na figura 5, com a seguinte fórmula: $Ci = (Mi + Kz)\ mod\ 26$. Sendo que c é o criptograma, m é a mensagem, k é a chave, z é o índice que é calculado por i mod tamanho da chave e o número 26 representa o tamanho do alfabeto.

```
void cifra() {

    for (i = 0; i < tam_mensagem; i++) {
        dec_m[i] = static_cast<int>(mensagem[i]) - 'a';
        z = i % tam_senha;
        dec_cript[i] = (dec_m[i] + dec_s[z]) % 26; // Cifragem
        criptograma[i] = static_cast<char>(dec_cript[i] + 'a'); // Transformando de decimal para char
    }

    cout << "Criptograma:" << endl;
    cout << criptograma << endl;
}
```

Implementação da função de cifragem. Figura 4.

Já a função decifra realiza a operação de decifragem de um criptograma dada uma certa senha. Para implementar isso, novamente, foi usado um loop for, apresentado na figura 6, com esta fórmula: $Mi = (ci + Kz + 26)\ mod\ 26$. A soma de 26 serve para evitar um resultado negativo.

```
void decifra(string criptograma,string senha) {
    for (i = 0; i < tam_senha; i++) {
        dec_s[i] = static_cast<int>(senha[i]) - 'a'; // 97
    }

    for (i = 0; i < tam_criptograma; i++) {
        dec_cript[i] = static_cast<int>(criptograma[i]) - 'a'; // 97
        z = i % tam_senha;
        dec_m[i] = (dec_cript[i] - dec_s[z] + 26) % 26; // Decifragem
        mensagem[i] = static_cast<char>(dec_m[i] + 'a'); // Transformando de decimal para char
    }

    cout << "Mensagem:" << endl;
    cout << mensagem << endl;
}
```

Implementação da função de decifragem. Figura 5.

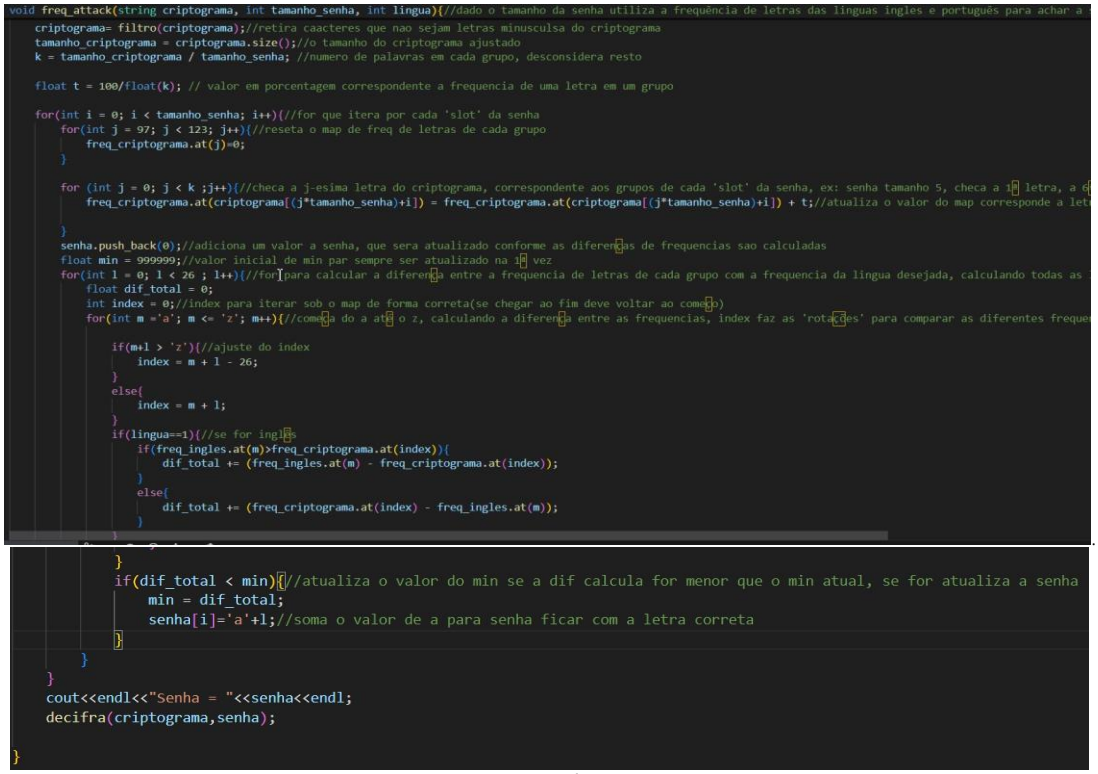
3. Implementação do Ataque

Além da cifragem e decifragem, também é feito um ataque de recuperação de senha por análise de frequência. O usuário entra com a mensagem cifrada e o programa devolve a mensagem decifrada e sua respectiva senha. O programa é capaz de decifrar mensagens em português e inglês. O usuário informa o idioma da mensagem para que a análise de frequência das letras seja feita de forma correta. Além disso, o usuário pode fornecer o tamanho da chave ou pedir que o próprio programa calcule de forma automática.

A função freq_attack() faz esse ataque usando as frequências de letras dos dois idiomas. Primeiramente é feito um mapeamento de frequências que armazena as frequências de cada letra para o inglês e português e um mapeamento para registrar as frequências do criptograma que está sendo analisado (freq_criptograma). Além disso, faz uma filtragem de forma que somente letras minúsculas

fiquem no criptograma, em seguida calcula o tamanho do criptograma e o divide pelo tamanho da senha para que se tenha o número de palavras em cada grupo.

Em seguida é feito o cálculo da frequência das letras em cada grupo, a frequência das letras é atualizada em "freq_criptograma" e é feita a comparação das letras no criptograma com a frequência das letras no idioma para que seja determinada a senha. Com a senha descoberta, a função decifra() é chamada para decifrar o texto em questão.



```
void freq_attack(string criptograma, int tamanho_senha, int lingua){//dado o tamanho da senha utiliza a frequência de letras das línguas inglês e português para achar a
    criptograma= filtro(criptograma);//retira caacteres que nao sejam letras minusculsa do criptograma
    tamanho_criptograma = criptograma.size();//o tamanho do criptograma ajustado
    k = tamanho_criptograma / tamanho_senha; //numero de palavras em cada grupo, desconsidera resto

    float t = 100/float(k); // valor em porcentagem correspondente a frequencia de uma letra em um grupo

    for(int i = 0; i < tamanho_senha; i++){//for que itera por cada 'slot' da senha
        for(int j = 97; j < 123; j++){//reseta o map de freq de letras de cada grupo
            freq_criptograma.at(j)=0;
        }

        for (int j = 0; j < k ;j++){//checa a j-esima letra do criptograma, correspondente aos grupos de cada 'slot' da senha, ex: senha tamanho 5, checa a 1ª letra, a 6ª
            freq_criptograma.at(criptograma[(j*tamanho_senha)+i]) = freq_criptograma.at(criptograma[(j*tamanho_senha)+i]) + t;//atualiza o valor do map corresponde a let

        }
        senha.push_back(0);//adiciona um valor a senha, que sera atualizado conforme as diferenças de frequencias sao calculadas
        float min = 999999;//valor inicial de min par sempre ser atualizado na 1ª vez
        for(int l = 0; l < 26 ; l++){//for para calcular a diferença entre a frequencia de letras de cada grupo com a frequencia da lingua desejada, calculando todas as
            float dif_total = 0;
            int index = 0;//index para iterar sob o map de forma correta(se chegar ao fim deve voltar ao começo)
            for(int m ='a'; m <= 'z'; m++){//começa do a até o z, calculando a diferença entre as frequencias, index faz as 'rotações' para comparar as diferentes freque

                if(m+l > 'z'){//ajuste do index
                    index = m + l - 26;
                }
                else{
                    index = m + l;
                }
                if(lingua==1){//se for inglês
                    if(freq_ingles.at(m)>freq_criptograma.at(index)){
                        dif_total += (freq_ingles.at(m) - freq_criptograma.at(index));
                    }
                    else{
                        dif_total += (freq_criptograma.at(index) - freq_ingles.at(m));
                    }
                }
                }
                if(dif_total < min){//atualiza o valor do min se a dif calcula for menor que o min atual, se for atualiza a senha
                    min = dif_total;
                    senha[i]='a'+l;//soma o valor de a para senha ficar com a letra correta
                }
            }
        }
        cout<<endl<<"Senha = "<<senha<<endl;
        decifra(criptograma,senha);

}
```

Implementação da função de ataque. Figura 6

4. Referências

[1] A cifra de Vigenère - criptografia e esteganografia, disponível em: https://www.eduardopopovici.com/2011/04/cifra-de-vigenere-criptografia-e.html, acesso em 06/11/2024

[2] Cifra de Vigenère, disponível em: https://pt.wikipedia.org/wiki/Cifra_de_Vigen%C3%A8re, acesso em 06/11/2024

[3] Criptografia – Cifra de Vigenère, disponível em: https://danieldonda.com/criptografia-cifra-de-vigenere/#:~:text=A%20cifra%20de%20Vigen%C3%A8re%20consiste,do%20anterior%20por%20uma%20posi%C3%A7%C3%A3o., acesso em 06/11/2024

[4] Quebrando a Cifra de Vigenère, disponível em: https://informatabrasileiro.blogspot.com/2013/04/quebrando-cifra-de-vigenere.html, acesso em 06/11/2024