Fábio Borges de Oliveira

Análise da segurança de criptografia e esteganografia em seqüências de imagens

Petrópolis, RJ

Fevereiro, 2007

Fábio Borges de Oliveira

Análise da segurança de criptografia e esteganografia em seqüências de imagens

Orientador:
Renato Portugal & Jauvane Cavalcante de Oliveira

LABORATÓRIO NACIONAL DE COMPUTAÇÃO CIENTÍFICA

Petrópolis, RJ

Fevereiro, 2007

ANÁLISE DA SEGURANÇA DE CRIPTOGRAFIA E ESTEGANOGRAFIA EM SEQÜÊNCIAS DE IMAGENS

Fábio Borges de Oliveira

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO LABORATÓRIO NACIONAL DE COMPUTAÇÃO CIENTÍFICA COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM MODELAGEM COMPUTACIONAL.

Aprovada por:

Prof. Renato Portugal, D.Sc.
(presidente)

Prof. Jauvane Cavalcante de Oliveira, Ph.D.

Prof. Eduardo Lucio Mendes Garcia, D.Sc.

Prof. Artur Ziviani, Ph.D.

Prof. Edison Ishikawa, D.Sc.

PETRÓPOLIS, RJ - BRASIL
FEVEREIRO, 2007

Oliveira, Fábio Borges

O48a Análise da segurança de criptografia e esteganografia em seqüências de imagens/ Fábio Borges de Oliveira ; Orientadores : Renato Portugal e Jauvane Cavalcanti de Oliveira - Petrópolis, RJ LNCC, 2007.

xiv, 101 p.; 29 cm.

Dissertação (Mestrado) - Laboratório Nacional de Computação Científica, 2007.

Inclui bibliografia

1. Criptografia. 2. Segurança da Informação. 3. Esteganografia. 4. Processamento de Imagens. 5. Compressão de Dados. I. Portugal, Renato. II. Oliveira, Jauvane Cavalcanti de. III. MCT/LNCC. IV. Título

CDD005.8

DEDICATÓRIA

A Deus, acima de tudo.

A toda minha família, em especial aos meus pais.

AGRADECIMENTOS

Agradeço a todos aqueles que direta ou indiretamente colaboraram para efetivação deste trabalho. Estas margens são muito pequenas para citar todas as pessoas que contribuíram para realizá-lo. No entanto, não posso deixar de citar algumas pessoas que tiveram uma participação toda especial, como por exemplo, meus orientadores. A eles, Norma, Atrv e Racco meu muitíssimo obrigado.

Resumo da Dissertação apresentada ao MCT/LNCC como parte dos requisitos necessários para obtenção do grau de Mestre em Ciências (M.Sc.)

ANÁLISE DA SEGURANÇA DE CRIPTOGRAFIA E ESTEGANOGRAFIA EM SEQÜÊNCIAS DE IMAGENS

Fábio Borges de Oliveira

Fevereiro, 2007

Orientador: Renato Portugal & Jauvane Cavalcante de Oliveira

Modelagem Computacional

A segurança da informação vem sendo considerada de grande importância para as instituições privadas e governamentais. Por este motivo, optamos em realizar um estudo sobre segurança nesta dissertação. Iniciamos com uma introdução à teoria da informação, partimos para métodos de criptografia onde propomos um novo tipo de *Segredo Perfeito* e finalmente fazemos um estudo de esteganografia em uma seqüência de imagens, onde propomos uma esteganografia mais agressiva nos coeficientes da transformada discreta de cosseno.

Abstract of Dissertation presented to MCT/LNCC as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

ANALYSIS OF THE CRYPTOGRAPHY SECURITY AND STEGANOGRAPHY IN IMAGE SEQUENCES

Fábio Borges de Oliveira

February, 2007

Advisor: Renato Portugal & Jauvane Cavalcante de Oliveira

Computational Modelling

Information security is being considered of great importance to the private and governmental institutions. For this reason, we opted to conduct a study of security in this dissertation. We started with an introduction to the information theory, and then we proposed a new kind of *Perfect Secrecy* cryptographic and finally made a study of steganography in an image sequence, in which we suggest a more aggressive steganography in coefficients of the discrete cosine transform.

Sumário

Lista de Figuras

Lista de Tabelas

Tabela de Símbolos

$\lfloor x \rfloor$	Maior inteiro menor do que ou igual a x
(x,y)	Máximo divisor comum dos inteiros x e y
$P(x)$	Probabilidade da ocorrência de x
$\mathbb{N}$	Conjunto dos números naturais
$\mathbb{N}^*$	$\mathbb{N} - \{0\}$
$\mathbb{Z}$	Conjunto dos números inteiros
$\not\emptyset$	Caracter que representa o espaço entre palavras
$G = (G, \oplus)$	Grupo
$R = (R, \oplus, \odot)$	Anel
$\mathbb{F}$	Corpo
$\mathscr{M}$	Conjunto de seqüências finitas
$\#M$	Quantidade de letras de uma seqüência M
$\lvert x \rvert$	Ordem do elemento x
$\lvert G \rvert$	Ordem do grupo G
$\lvert C \rvert$	Cardinalidade do conjunto C
$\varphi(n)$	$\lvert C \rvert$ onde $C = \{x \in \mathbb{N} : x \leqslant m \text{ e } (x,m) = 1\}$
$f(x) = O(h(x))$	Existência de uma constante $c > 0$ e $n_0 \in \mathbb{N}$ tais que $f(x) \leqslant c\,h(x)$ para $x \geqslant n_0$

1 Introdução

Neste trabalho usaremos duas técnicas para proteger a informação. Ambas as técnicas são conhecidas como arte-ciência. A **esteganografia** é a arte de esconder uma mensagem, enquanto a **criptografia** é a arte de cifrar uma mensagem. Apesar de ambas terem a intenção de proteger uma informação, os princípios são bem diferentes.

Com a junção das técnicas, temos a palavra **esteganocriptografia** que é derivada de três palavras gregas, *steganós*-: que cobre; *-kryptós-*: obscuro; e *-graphía* do verbo *gráphein*: escrever.

Imagine que Ana quer mandar uma correspondência para Beth, está correspondência pode ser uma carta, um e-mail, uma foto, ou qualquer outra forma onde se transmite informação. Em geral, chamamos esta correspondência de mensagem e dizemos que a mensagem vai de A para B.

Considere uma mensagem enviada do ponto A ao B, simbolizamos:

$$m : A \rightsquigarrow B$$

a mensagem m pode sofrer quatro **ameaças**:

1. Interceptação, isto é, alguém poderia ler a mensagem durante a transmissão.
2. Alteração, além de ser lida a mensagem poderia ser alterada.
3. Fabricação, poderia ser criada uma mensagem falsa.
4. Interrupção, de forma que a mensagem não chegue ao seu destino.

A criptografia pode proteger *m* das três primeiras ameaças. No entanto, não pode proteger de uma interrupção.

Com a necessidade da mensagem não ser interrompida, a esteganografia foi surgindo de forma natural. Os primeiros relatos sobre seu surgimento, contam sobre tábuas escritas, assim como cobertas com cera e tatuagem na cabeça coberta lentamente com o crescimento do cabelo. Muitos séculos depois, na Segunda Guerra Mundial foi usada a tinta invisível; durante a mesma guerra um espião alemão enviou a seguinte mensagem:

Apparently neutral s protest is thoroughly discounted and ignored. Isman hard hit. Blockade issue affects pretext for embargo on by-products, ejecting suets and vegetable oils

Se for lido apenas a segunda letra de cada palavra temos:

Pershing sails from NY June 1.

Estas histórias (JOHNSON; JAJODIA, 1998) mostram uma esteganografia primitiva. Hoje as mensagens são embutidas em imagens, som, protocolos como TCP/IP (AHSAN; KUNDUR, 2002); em geral meios digitais. Porém, existem estudos de esteganografia até mesmo em DNA (FELDKAMP; BANZHAF; RAUHE, 2000; GEHANI; LABEAN; REIF, 1999).

Caso a interrupção não seja uma ameaça iminente, mesmo assim, em casos de segurança extrema, é recomendado o uso da esteganografia. A questão natural que segue é: se a criptografia atual é segura, por que devemos usar esteganografia?

Um outro problema da criptografia é a falta de conhecimento de suas limitações, pois, com exceção da **cifra de Vernam**[1] (SHANNON, 1949) e do algoritmo 10 que propomos, é difícil medir o quão seguro é um método de criptografia. Em outras palavras somente os algoritmos que garantem um *Segredo Perfeito* são inquebráveis, isto é, a mensagem não

[1] Também conhecida com **One-time-pad**

será descoberta. Nos algoritmos que não possuem esta propriedade a mensagem poderia ser descoberta, se existissem meios computacionais para calculá-la.

Muitas vezes levamos séculos procurando a solução para um determinado problema, que pode surgir de forma inesperada. Durante muito tempo foi procurado um algoritmo que determinasse em tempo polinomial se um número é primo, até que foi encontrado (AGRAWAL; KAYAL; SAXENA, 2004). Não se sabe ainda se pode ser desenvolvido um algoritmo equivalente ao de Shor para um computador clássico.

Uma outra questão surge: Por que não usar apenas a esteganografia?

Primeiramente, porque as técnicas de esteganografia não estão tão bem desenvolvidas como as de criptografia.

É difícil localizar uma mensagem escondida em qualquer lugar, mas ao se ter uma suspeita, fica mais fácil determinar se existe ou não esteganografia do que quebrar a criptografia.

Mesmo que existisse uma esteganografia perfeita, certamente ela seria muito custosa e inviável de ser usada em larga escala. Logo, ela poderia ser usada somente para troca de chaves simétricas, ao invés de ser usada para transmitir a mensagem.

Este trabalho está dividido em três partes. Primeiramente, tratamos de como a mensagem é escrita, depois da criptografia e finalmente da esteganografia.

Na primeira parte, fazemos levantamentos estatísticos, análise de alguns idiomas e localizamos uma taxa média de compressão por idioma. Finalmente, tratamos de compressão em imagens.

Na segunda parte, apresentamos os métodos clássicos de criptografia e propomos um novo método baseado em números irracionais (BORGES; PORTUGAL; OLIVEIRA, 2006), que representa um novo tipo de *Segredo Perfeito*. Fazemos uma análise de sua segurança e introduzimos o conceito de semântica na chave. Redefinimos a palavra

One-time-pad para podermos fazer um melhor estudo sobre métodos classificados como *Segredo Perfeito*. Mostramos que com determinadas hipóteses somente *One-time-pad* é um *Segredo Perfeito*. Fazemos um estudo de onde se baseia a segurança de cada tipo de algoritmo, onde propomos uma métrica para medir o criptossistema.

Na terceira parte, analisamos a natureza da esteganografia e propomos o que seria *Segredo Perfeito* para a esteganografia. Através de testes heurísticos, em seqüências de imagens extraídas de vídeos, mostramos que podemos ter uma esteganografia mais agressiva em imagens, dificultando sua detecção.

As implementações foram compiladas com gcc usando as bibliotecas GNU MP, LiDIA e IJG JPEG library.

2 Escrevendo a mensagem

Neste capítulo tratamos alguns pontos introdutórios da teoria da informação, cujas áreas de estudo são compressão, criptografia, código de correção de erro, sistemas de comunicação e assuntos relacionados. Em especial, estaremos quantificando a informação e tratando da forma que podemos armazená-la.

2.1 Quantificando

Veremos que há sentido em medir a informação que é passada em uma mensagem.

Definição. Alfabeto $\mathscr{A}$ é um conjunto não vazio e finito símbolos, cujos seus elementos são denominados de **letras**.

Se o alfabeto for ordenado podemos considerá-lo como base de um novo sistema numérico, assim as letras são equivalentes aos dígitos. Dado uma mensagem precisaríamos saber qual o espaço utilizado para armazená-la, em outras palavras, a quantidade de dígitos de um dado número em qualquer sistema numérico.

Uma outra forma de escrevermos a mensagem é fazer uma relação biunívoca das letras com um sistema numérico. Em geral, estaremos usando o sistema binário. Usaremos também m_i para a i-ésima letra de uma mensagem m e $P(m_i)$ para a probabilidade da ocorrência de m_i.

Definição. A quantidade de **Informação** contida em m_i é dada por

$$I(m_i) = -\log_2(P(m_i)) \tag{2.1}$$

Com essa definição vemos que a informação é mínima quando m_i tem probabilidade máxima.

Note que a quantidade de dígitos do número m na base $b > 1$ é dada por 1 mais a parte inteira do logaritmo de m na base b,

$$\#(m_b) = \lfloor \log_b m \rfloor + 1,$$

com $m, b \in \mathbb{N}^*$.

Definição. A **Entropia** de Shannon (SHANNON, 1948) é dada por:

$$H(m) = \sum_i P(m_i) I(m_i) = -\sum_i P(m_i) \log_2(P(m_i)) \tag{2.2}$$

Vemos que a entropia é zero quando as letras são iguais, isto é $m_i = m_j \ \forall i, j$, pois $P(m_i) = 1$ e pela equação (2.1) temos que $I(m_i) = 0$. A entropia é máxima quando todas as letras do alfabeto são diferentes e tem a mesma probabilidade de ocorrência. Uma letra natural para informática é o **byte**, composta por oito dígitos binários, **bits**. Considerando esta letra, a entropia máxima é oito. Supondo que todos os bytes têm a mesma probabilidade com $P(m_i) = \xi$, então

$$H(m) = -\sum_{i=1}^{|\mathscr{A}|} \log_2\left(P(m_i)^{P(m_i)}\right)$$

de onde temos

$$H(m) = -\sum_{i=1}^{|\mathscr{A}|} \xi \log_2(\xi) = -|\mathscr{A}| \xi \log_2(\xi).$$

Se todos os bytes têm a mesma probabilidade ξ temos

$$\xi = \frac{1}{256}.$$

Portanto

$$H(m) = -\log_2\left(\frac{1}{256}\right) = 8.$$

2.1.1 Escolha, Incerteza e Entropia

A fórmula da entropia de Shannon surge naturalmente das três imposições abaixo.

Seja $p_i = P(A_i)$ onde A_i é um conjunto de eventos independentes. Queremos que a informação média H tenha as seguintes propriedades:

1. Que H seja uma função contínua das probabilidades p_i.
2. Se os eventos têm probabilidades iguais $p_i = 1/n$, $\forall i$, então H é uma função monotônica crescente.
3. Se uma escolha for dividida em sub-escolhas, então H tem que ser uma soma ponderada dos valores próprios de H, isto garante que a informação média seja a mesma.

Exemplificando a condição 3 temos a figura 2.1 que leva a

$$H\left(\frac{1}{2},\frac{1}{3},\frac{1}{6}\right) = H\left(\frac{1}{2},\frac{1}{2}\right) + \frac{1}{2}H\left(\frac{2}{3},\frac{1}{3}\right).$$

Generalizando, temos

$$H = H(p_1, p_2, p_3).$$

Se dividirmos em dois conjuntos

$$B_1 = \{A_1\},\ \ B_2 = \{A_2, A_3\}$$

As possibilidades dos eventos q_i são dados por

$$q_1 = P(B_1) = p_1,\ \ q_2 = P(B_2) = p_2 + p_3.$$

Logo

$$H = H(q_1, q_2) + q_1 H\left(\frac{p_1}{q_1}\right) + q_2 H\left(\frac{p_2}{q_2}, \frac{p_3}{q_2}\right).$$

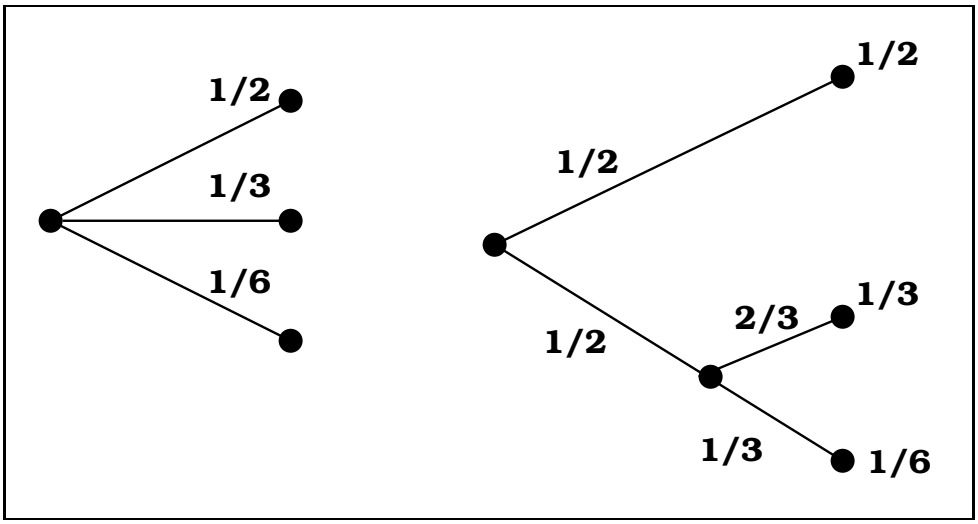


Figura 2.1: *Decomposição de uma escolha de três possibilidades.*

Teorema 1 (Entropia) *O único H que satisfaz as três condições acima é*

$$H = -K\sum_i p_i \log_2 (p_i),$$

onde K é uma constante arbitrária. **Prova.** Seja

$$A(n) = H\left(\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n}\right).$$

da condição 3, podemos decompor uma escolha s^m igualmente provável em uma série de escolhas de m igualmente prováveis, assim

$$A(s^m) = mA(s)$$

analogamente

$$A(t^n) = nA(t).$$

Escolhendo n arbitrariamente grande, encontramos m tal que

$$s^m \leqslant t^n \leqslant s^{m+1},$$

aplicando o logaritmo e dividindo por $n \log s$,

$$\frac{m}{n} \leqslant \frac{\log t}{\log s} \leqslant \frac{m+1}{n},$$

ou seja,

$$\left|\frac{m}{n}-\frac{\log t}{\log s}\right|<\varepsilon$$

com ε arbitrariamente pequeno. Da condição 2 temos

$$A(s^m)\leqslant A(t^n)\leqslant A(s^{m+1})\Rightarrow mA(s)\leqslant nA(t)\leqslant (m+1)A(s).$$

Dividindo por $nA(s)$, temos

$$\frac{m}{n}\leqslant\frac{A(t)}{A(s)}\leqslant\frac{m+1}{n},$$

ou seja,

$$\left|\frac{m}{n}-\frac{A(t)}{A(s)}\right|<\varepsilon$$

logo

$$\left|\frac{A(t)}{A(s)}-\frac{\log t}{\log s}\right|<2\varepsilon,$$

de forma que

$$A(t)=K\log t,$$

onde K deve ser positivo para satisfazer a condição 2. Até agora assumimos eventos igualmente prováveis, para o caso geral, separamos as escolhas em grupos diferentes com tamanho n_i com probabilidade racional. Caso a probabilidade seja irracional, podemos aproximar por um número racional através da condição 1, logo

$$p_i=\frac{n_i}{\sum_{j=1}^{n}n_j}.$$

Usando a condição 3 novamente,

$$K\log\left(\sum n_i\right)=H(p_1,\ldots,p_n)+K\sum p_i\log n_i,$$

i.e.,

$$H=K\left[\sum p_i\log\sum n_i-\sum p_i\log n_i\right]=-K\sum p_i\log\left(\frac{n_i}{\sum n_i}\right),$$

portanto

$$H=-K\sum p_i\log p_i.$$

Escolhe-se $K=1$ por conveniência, pois ele fornece a escala de medida da quantidade de informação. ■

Ao calcular as probabilidades de várias mensagens, começamos a observar padrões.

2.1.2 Padrões

Normalmente as pessoas observam que a letra A é muito freqüente na língua portuguesa e a letra E na língua inglesa. Mas, será que são realmente as letras com maior freqüência? Comparando vários textos, podemos identificar a freqüência média das letras em um idioma. Por outro lado, em um pequeno trecho de um texto ou em um texto técnico, normalmente não se encontra a mesma distribuição de freqüência. É desejado que o padrão se mantenha, ou seja, bem próximo da média, de forma que calculando a freqüência das letras em um texto, podemos identificar o idioma. Estas freqüências serão exploradas na compactação e criptografia.

Quando vamos analisar um idioma desconhecido, podemos formar o alfabeto a partir de uma grande amostra de texto,

$$\mathscr{A} = \bigcup_i m_i.$$

A tabela 2.1 e a tabela 2.2 têm suas freqüências próximas. A primeira foi obtida dos estudos de letras mais freqüentes em alguns idiomas, enquanto a segunda foi obtida através da contagem das letras do livro Amigos de Dios(ESCRIVA, 1977) e algumas de suas traduções. Daqui para frente, quando estivermos comparando idiomas, estaremos usando estes livros como fonte de dados. Normalmente se usa a Bíblia[1] para fazer este tipo de estudo, porém optamos em usar este livro porque atualmente ele se encontra disponível na Internet com boa fidelidade de tradução em maior quantidade de idiomas.

Na tabela 2.2 optamos em diferenciar a tradução brasileira e a portuguesa, para observar suas diferenças.

Com as dez letras mais freqüentes temos aproximadamente 70% do texto.

Observe nas tabelas 2.1 e 2.2 que as letras E, A, N, I, S e R aparecem com alta

[1]Nova Vulgata - http://www.vatican.va

En	%	**Fr**	%	**It**	%	**Es**	%
E	12.31	E	15.87	E	11.79	E	13.15
T	9.59	A	9.42	A	11.74	A	12.69
A	8.05	I	8.41	I	11.28	O	9.49
O	7.94	S	7.90	O	9.83	S	7.60
N	7.19	T	7.26	N	6.88	N	6.95
I	7.18	N	7.15	L	6.51	R	6.25
S	6.59	R	6.46	R	6.37	I	6.25
R	6.03	U	6.24	T	5.62	L	5.94
H	5.14	L	5.34	S	4.98	D	5.58

Tabela 2.1: *Letras com altas freqüências por idioma* (SALOMAA, 1996).

En	%	**Fr**	%	**It**	%	**Es**	%	**Pt**	%	**Br**	%
E	11.52	E	16.61	E	11.44	E	12.61	E	12.76	E	12.81
T	8.58	S	8.15	I	10.38	A	11.36	A	12.32	A	12.36
O	8.11	N	7.06	A	9.86	O	9.13	O	10.27	O	10.28
A	6.89	A	6.78	O	9.07	S	8.03	S	8.85	S	8.91
I	6.80	I	6.69	N	6.78	N	6.89	R	6.20	R	6.16
S	6.46	U	6.35	R	6.19	R	6.36	I	5.47	I	5.42
N	6.13	T	6.34	T	5.64	I	6.04	N	5.02	N	5.01
H	5.71	R	6.33	L	5.23	D	4.92	M	4.86	M	4.90
R	5.61	O	5.59	S	5.03	L	4.40	D	4.81	D	4.77
L	3.96	L	4.54	C	4.55	U	4.02	U	4.15	U	4.20

Tabela 2.2: *Amostra de letras com altas freqüências.*

freqüência nos cinco idiomas.

Na tabela 2.3 temos as maiores freqüências diferenciando os acentos e constando todos os caracteres. Lembramos que o símbolo $\not\emptyset$ significa o espaço entre as palavras.

De posse das freqüências podemos medir a informação (2.1) e a entropia (2.2) média das mensagens em um idioma.

As tabelas 2.4 e 2.5 estão em ordem decrescente. Observe que não há uma relação direta entre a quantidade de letras e a entropia. É natural pensar que quanto mais letras, maior a desordem do texto, isto é, entropia. Mas ao contrário do que poderíamos imaginar isto não ocorre.

En	%	De	%	Fr	%	Nl	%	Pt	%	Br	%	It	%	Es	%
∅	16.94	∅	14.43	∅	15.47	∅	15.69	∅	15.47	∅	15.54	∅	14.95	∅	15.53
e	9.55	e	13.53	e	12.00	e	15.09	e	10.06	e	10.07	e	9.29	e	10.25
t	6.97	n	8.26	s	6.73	n	7.97	a	9.19	a	9.20	i	8.70	a	9.18
o	6.64	i	6.78	n	5.88	i	5.55	o	8.34	o	8.34	a	8.02	o	7.34
a	5.63	r	5.50	i	5.47	a	5.31	s	7.25	s	7.31	o	7.61	s	6.56
i	5.35	t	4.65	r	5.33	t	5.08	r	5.21	r	5.18	n	5.69	n	5.51
s	5.27	s	4.62	u	5.32	d	4.59	i	4.41	i	4.36	r	5.24	r	5.35
n	5.07	d	3.93	t	5.31	o	4.55	n	4.12	n	4.12	t	4.75	i	4.70
h	4.65	h	3.78	a	5.15	r	4.20	m	4.00	m	4.03	l	4.35	d	3.96
r	4.65	a	3.46	o	4.67	l	3.10	d	3.85	d	3.83	s	4.04	l	3.67
l	3.15	u	3.22	l	3.70	s	2.75	u	3.47	u	3.47	c	3.66	t	3.31
d	2.81	l	2.72	d	2.69	g	2.48	t	3.40	t	3.35	d	2.76	u	3.28
u	2.66	c	2.35	c	2.52	h	2.27	c	2.64	c	2.58	m	2.34	c	3.17
f	1.93	g	2.11	m	2.20	v	2.09	p	2.02	p	1.98	u	2.18	m	2.47
c	1.82	m	1.87	p	2.16	m	1.75	l	1.75	l	1.76	p	2.10	p	1.85
m	1.79	,	1.77	↩	1.55	k	1.65	↩	1.58	,	1.59	,	1.60	,	1.64
w	1.78	↩	1.56	é	1.45	↩	1.56	,	1.56	↩	1.56	↩	1.56	↩	1.56
y	1.72	o	1.53	,	1.33	u	1.52	v	1.11	v	1.11	g	1.26	q	1.00
↩	1.55	b	1.34	v	1.32	j	1.49	q	1.08	q	1.11	v	1.26	b	0.93
g	1.37	w	1.17	’	1.02	z	1.45	h	0.84	h	0.82	h	0.99	v	0.78
,	1.31	f	1.00	q	1.01	w	1.31	.	0.76	ã	0.72	f	0.83	g	0.70
p	1.27	z	0.81	f	0.83	,	1.23	f	0.73	.	0.72	z	0.71	.	0.69
v	1.03	k	0.70	.	0.67	b	1.03	g	0.71	f	0.72	.	0.68	h	0.69
b	0.93	.	0.70	h	0.66	p	0.81	ã	0.70	g	0.71	b	0.60	y	0.62
.	0.77	v	0.54	g	0.64	c	0.74	b	0.61	b	0.62	-	0.35	f	0.50
k	0.50	ü	0.52	b	0.53	.	0.74	-	0.52	-	0.53	q	0.34	í	0.38
’	0.40	G	0.46	à	0.41	f	0.60	ç	0.43	ç	0.44	’	0.33	ó	0.35
I	0.30	ß	0.37	-	0.37	H	0.33	é	0.40	é	0.40	à	0.28	z	0.35
G	0.19	ä	0.36	j	0.31	-	0.31	z	0.30	z	0.30	S	0.23	-	0.34
T	0.16	S	0.35	è	0.29	G	0.20	á	0.29	á	0.28	è	0.21	á	0.33
L	0.14	W	0.32	x	0.28	M	0.15	S	0.23	S	0.22	C	0.20	j	0.27
C	0.13	p	0.30	D	0.18	:	0.15	D	0.21	D	0.21	:	0.20	é	0.25
W	0.11	H	0.30	ê	0.18	D	0.14	ó	0.20	E	0.20	D	0.20	S	0.22
-	0.10	E	0.29	:	0.18	)	0.13	í	0.19	í	0.19	ù	0.14	ñ	0.21
O	0.10	M	0.27	C	0.17	(	0.13	j	0.18	j	0.19	M	0.14	:	0.21
S	0.10	D	0.26	S	0.17	1	0.13	E	0.18	ó	0.19	é	0.14	D	0.19
A	0.09	A	0.26	L	0.14	W	0.10	:	0.17	:	0.18	]	0.13	E	0.16
H	0.09	ö	0.26	z	0.14	I	0.09	x	0.15	C	0.15	[	0.13	P	0.14

Tabela 2.3: *Freqüência parcial das letras de $\mathscr{A}$.*

Idioma	**Letras**
Deutsch	551,487
Nederlands	529,936
Français	504,119
Polski	492,326
English	489,593
Russian	484,007
Italiano	481,075
Svenska	473,017
Español	463,359
Português(Br)	458,804
Português(Pt)	456,419

Tabela 2.4: *Quantidade de letras usadas nas traduções.*

Idioma	**Entropia**
Russian	4.8771
Polski	4.7957
Deutsch	4.5736
Svenska	4.4921
Français	4.4392
Italiano	4.4155
Português(Br)	4.4135
Português(Pt)	4.4130
English	4.4079
Nederlands	4.4016
Español	4.3913

Tabela 2.5: *Entropia média por idioma.*

Da mesma forma, observamos que há palavras que são mais usadas enquanto outras são pouco usadas. Com exceção de siglas, não temos palavras com seqüência de caracteres NP e NB, na língua portuguesa. No entanto, temos que a seqüência TH é muito freqüente na língua inglesa. Todas estas características motivam a próxima seção.

2.2 Compressão

Hoje em dia, a quantidade de informação armazenada e transmitida é demasiada grande. O objetivo é armazenar o máximo com o menor custo e transmitir com a maior velocidade. Para atingir estes objetivos, comumente usa-se compressão.

Definição. Compressão C é uma algoritmo que leva uma mensagem m para m_c que requer menos letras. Se C^{-1} leva m_c em m, dizemos que é uma Compressão Sem Perda, caso C^{-1} aproxime m_c de m, dizemos que C é uma Compressão Com Perda.

Podemos usar a compressão com perda em fotos e sons, onde já houve uma quantização, devido aos instrumentos de captação. Neste caso limitamos a quantidade de perda de forma que a percepção humana não perceba as diferenças. Usamos a compressão sem perda para dados que precisam voltar a sua forma original, como um texto.

Dada qualquer mensagem m, é desejado que m_c seja menor que m, porém não pode existir um algoritmo que garanta que isto sempre ocorra. Imagine tal algoritmo aplicado recursivamente.

Em se tratando de compressão, a entropia pode ser interpretada como número médio de bits por símbolo em uma mensagem.

Medindo a entropia H de uma mensagem comprimida, podemos avaliar a qualidade da técnica de compressão. Por outro lado, se a entropia da mensagem é baixa, implica que temos muita redundância de informação, logo a mensagem pode ser comprimida.

Segue uma breve explanação de alguns métodos de compressão. Maiores detalhes podem ser encontrados em (SAYOOD, 2000).

Quando estamos comprimindo a mensagem, ou levando para outro alfabeto, normal-

mente dizemos que estamos **codificando** a mensagem.

2.2.1 Codificação por Dicionário

Dado um idioma, poderíamos ordenar as palavras por freqüência de uso. Considerando que temos 5.000 verbetes, o menos significativo ocuparia

$$\lfloor \log_2(5000) \rfloor + 1 = 13$$

bits, enquanto uma palavra de 5 bytes ocupa 40 bits. Mesmo que reduzissem nosso alfabeto para 64 símbolos teríamos 30 bits, pois com 32 símbolos não podemos representar todos os caracteres da língua portuguesa. Este é um processo de compressão por **dicionário**. Como nem todas as palavras podem estar no dicionário, deve-se conter o próprio alfabeto no dicionário.

Normalmente, o dicionário não é constituído por palavras de um idioma, mas gerado através de seqüências de letras.

Esta técnica também é usada para compressão de imagens, onde se cria um dicionário com as cores predominantes na imagem. Este é o caso do formato GIF[2] (HALSALL, 2001).

2.2.2 Codificação por Carreira

Costuma-se usar mais de uma técnica de compressão para imagens, em geral usa-se compressão com e sem perda. Codificação por carreira é uma compressão sem perda.

Considere uma foto com o céu azul. Temos longos intervalos onde só se encontra o

[2]Graphics Interchange Format

mesmo tom de azul. Para armazenar estes longos intervalos de azul não são necessárias longas áreas de armazenamento, se usarmos codificação por carreira.

A codificação por carreira consiste em contar as repetições e marcar na frente das letras o número de repetições.

Assim, se tivermos a seqüência AAAAABBBABAAAAAAABBBBAAAAAAAA-ABB, usando este método podemos escrever 5A3B1A1B7A4B9A2B que nos fornece uma economia de 50% no tamanho da mensagem.

Note que não adianta calcular apenas a entropia total da seqüência, para verificar se podemos usar compressão, pois $H(\text{“}AB\text{”}) = H(\text{“}AAAABBBB\text{”})$, isto deve ser feito calculando a entropia de pedaços da mensagem.

Este método não serve para ser aplicado quando a mensagem for um texto, pois isto praticamente duplicaria o tamanho da mensagem.

Na comparação dos dois últimos algoritmos apresentados, vemos que a compressão depende muito do tipo de mensagem.

2.2.3 Código de Huffman

Na tabela **ASCII** [3] temos a letra A representada por 01000001 e B por 01000010 e assim sucessivamente. Logo LNCC é codificado como:

$$01001100\,01001110\,01000011\,01000011. \tag{2.3}$$

Podemos ler a seqüência (2.3) porque sabemos que cada letra é representada por oito bits.

[3] American Standard Code for Information Interchange

A idéia principal deste algoritmo de compactação é usar um número de bits variável para cada letra. Como o número de bits por letra não é fixo, é necessário ter um critério para determinar quando começa cada nova letra.

Se codificarmos conforme a tabela 2.6, escrevemos LNCC como

10 110 1 1.

Letra	**Código**
C	1
L	10
N	110

Tabela 2.6: *Código ambíguo.*

Desta forma, não poderemos identificar o começo das letras, em outras palavras, este código não tem uma única decodificação.

Definição. Dizemos que a **Regra do Prefixo** é satisfeita, quando nenhum código é início de outro código.

Para garantirmos um código de decodificação única, basta seguirmos a Regra do Prefixo. Isto nos garante um critério na codificação.

Podemos codificar segundo a tabela 2.7.

Letra	**Código**
C	1
L	01
N	001

Tabela 2.7: *Código de única decodificação.*

Desta forma, teremos LNCC como

01 001 1 1.

Logo, temos um **código com única decodificação**.

O **algoritmo de Huffman** nos apresenta uma forma de codificar garantindo um ótimo código para compactação sem perdas.

Vamos codificar LNCC/MCT, para isto montamos a tabela 2.8.

Letra	**Ocorrência**	**Freqüência**	**Código**
C	3	37.5%	c(C)
L	1	12.5%	c(L)
N	1	12.5%	c(N)
/	1	12.5%	c(/)
M	1	12.5%	c(M)
T	1	12.5%	c(T)

Tabela 2.8: *Codificando com o algoritmo de Huffman.*

Unimos as duas últimas linhas de menor freqüência da tabela 2.8. Antes, considere a operação binária $\odot$ como concatenação. Desta forma, fazemos

$$\begin{cases} c(M) = c(MT) \odot 0, \\ c(T) = c(MT) \odot 1. \end{cases}$$

Ordenando a tabela, temos

Letra	**Ocorrência**	**Freqüência**	**Código**
C	3	37.5%	c(C)
MT	2	25%	c(MT)
L	1	12.5%	c(L)
N	1	12.5%	c(N)
/	1	12.5%	c(/)

Podemos novamente unir as duas últimas linhas, assim

$$\begin{cases} c(N) = c(N/) \odot 0 \\ c(/) = c(N/) \odot 1 \end{cases}$$

Reordenando a tabela, temos

Letra	**Ocorrência**	**Freqüência**	**Código**
C	3	37.5%	c(C)
MT	2	25%	c(MT)
N/	2	25%	c(N/)
L	1	12.5%	c(L)

Este processo é aplicado recursivamente, neste ponto

$$\begin{cases} c(N/) = c(N/L) \odot 0 \\ c(L) = c(N/L) \odot 1 \end{cases}.$$

Reordenando, temos

Letra	**Ocorrência**	**Freqüência**	**Código**
C	3	37.5%	c(C)
N/L	3	37.5%	c(N/L)
MT	2	25%	c(MT)

Fazendo,

$$\begin{cases} c(N/L) = c(MTN/L) \odot 0 \\ c(MT) = c(MTN/L) \odot 1 \end{cases}.$$

Por fim, temos apenas duas linhas

Letra	**Ocorrência**	**Freqüência**	**Código**
MTN/L	5	62.5%	c(MTN/L)
C	3	37.5%	c(C)

Assim, atribuímos

$$\begin{cases} c(MTN/L) = 0 \\ c(C) = 1 \end{cases}.$$

A partir de C(c)=1 podemos substituir recursivamente cada código até formarmos a

tabela 2.9, a partir da tabela 2.8.

Letra	Ocorrência	Freqüência	Código
C	3	37.5%	1
L	1	12.5%	001
N	1	12.5%	0000
/	1	12.5%	0001
M	1	12.5%	010
T	1	12.5%	011

Tabela 2.9: *Código obtido com o algoritmo de Huffman.*

Portanto, podemos escrever LNCC/MCT como

$$001\,0000\,1\,1\,0001\,010\,1\,011.$$

Na figura 2.2 fazemos outra ordenação que também nos leva aos 20 bits. Assim, a mesma mensagem poderia ser escrita como

$$11\,100\,00\,00\,101\,010\,00\,011.$$

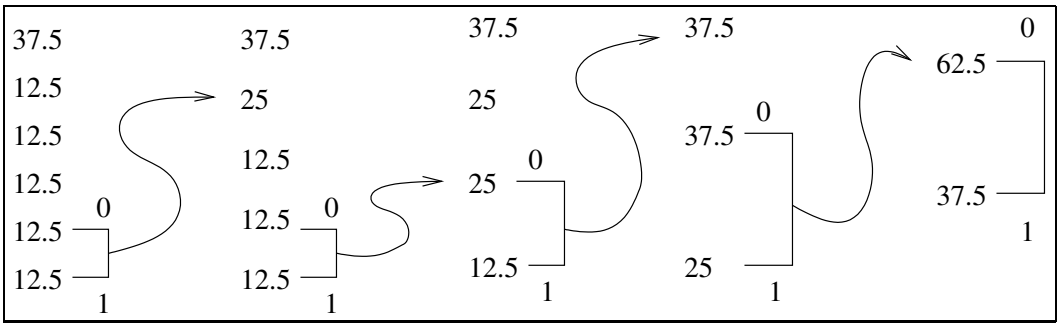


Figura 2.2: *Diagrama do código de Huffman.*

Observe que o tamanho da mensagem codificada é próximo à entropia multiplicada pelo número de letras, $2.4 \times 8 = 19.2$. Isto sempre ocorre, pois em (SAYOOD, 2000), encontra-se a demonstração que

$$H(M) \leqslant \overline{l} < H(M) + 1 \tag{2.4}$$

onde $\overline{l}$ é o comprimento médio de um código, daí temos que

$$\#(c(M)) \approx H(M)\,\#(M).$$

Se a entropia é máxima não poderíamos comprimir mais, sem perda de informações. No entanto, usando codificação por carreira, isto é possível em alguns casos, pois

$$H(\mathscr{A}) = H(k\mathscr{A}),$$

para qualquer $k \in \mathbb{N}^*$.

De posse das freqüências médias das letras por idioma, podemos usar Huffman sem a necessidade de construir uma tabela, desonerando tanto no processamento quanto no armazenamento. Entretanto, a construção da tabela garante uma codificação mais adequada.

Voltando a nossa amostra, podemos ver a entropia das traduções do mesmo texto comprimido na tabela 2.10 e a taxa de compressão em 2.11. Ambas as tabelas estão em ordem decrescente.

Idioma	**Entropia**
Russian	7.9165
Français	7.8811
Deutsch	7.8804
Svenska	7.8779
Polski	7.8698
Italiano	7.8696
Español	7.8661
English	7.8585
Português(Br)	7.8566
Português(Pt)	7.8553
Nederlands	7.8455

Tabela 2.10: *Entropia das traduções compactadas com Huffman.*

Observe que as tabelas 2.11 e 2.5 estão na mesma ordem, conforme esperado.

O algoritmo de Huffman também é usado na compactação de imagens, como por exemplo no formato **JPEG** [4]. Antes da compressão, a imagem passa por processos de quantização.

[4] Joint Photographic Experts Group

Idioma	Compressão
Russian	61.31%
Polski	60.21%
Deutsch	57.52%
Svenska	56.57%
Français	55.80%
Italiano	55.71%
Português(Br)	55.65%
Português(Pt)	55.64%
English	55.54%
Nederlands	55.43%
Español	55.26%

Tabela 2.11: *Taxa de compactação das traduções com Huffman.*

2.3 Quantização

Quantização é um processo de discretizar um sinal contínuo, em processamento de imagens quantização é um tipo de compressão com perdas.

As quantizações não são aplicáveis a texto, mas somente a informações contínuas como imagem e som, pois é a parte onde se produz mais perdas na compressão.

O primeiro processo de quantização é físico, ocorre quando um dispositivo capta o sinal da mensagem. Neste processo, normalmente se usa filtros para eliminar ruídos. Finalmente é feita a discretização.

Usamos quantização para tirarmos proveito das características sensoriais humanas, por exemplo, somos mais sensíveis à luminância do que à crominância, logo podemos descartar mais informações sobre a crominância.

Também temos dificuldades de perceber as mudanças de alta freqüência, por exemplo, a mudança brusca de cores em uma imagem, em outras palavras não se percebe bem o contorno dos objetos de uma imagem. Por isto, foram propostas várias transformadas para retirar esta redundância.

As transformadas devem separar os coeficientes de baixa correlação e devem ser inversíveis e tratáveis computacionalmente.

Na maior parte dos casos é usada a Transformada Discreta de Cossenos (**DCT**) [5], que atua em uma matriz quadrada e retorna uma matriz quadrada de mesma ordem.

2.3.1 DCT

Seja P uma matriz de **pixel** [6] e F a matriz dos coeficientes, então a DCT é dada por

$$F = APA^T, \tag{2.5}$$

com inversa

$$P = A^T FA. \tag{2.6}$$

Onde

$$A_{mn} = C(m-1)\cos\frac{(2n-1)(m-1)\pi}{2N} \tag{2.7}$$

e

$$C(k) = \begin{cases} \frac{1}{\sqrt{2}} \text{ para } k = 0, \\ 1 \text{ para todos os outros valores de } k. \end{cases}$$

Usando a fórmula (2.7) podemos mostrar que $A^T A = I$, isto é uma condição para que a equação (2.6) possa ser obtida de (2.5).

Sendo $N = 8$, ao expandir as matrizes temos

$$F[m+1,n+1] = \frac{C(m)}{2}\frac{C(n)}{2}\sum_{x=0}^{7}\sum_{y=0}^{7} P[x+1,y+1]\cos\frac{(2x+1)m\pi}{16}\cos\frac{(2y+1)n\pi}{16}, \tag{2.8}$$

onde m e n variam de 0 até 7, $P[x,y]$ é uma matriz de pixel.

[5] Discrete Cosine Transform

[6] Picture Element, derivado de pix [pl. de pic(ture)] + el(ement).

A inversa de (2.8) conhecida como **IDCT** é dada por

$$P[x+1,y+1] = \frac{1}{4}\sum_{m=0}^{7}\sum_{n=0}^{7} C(m)C(n)F[m+1,n+1]\cos\frac{(2x+1)m\pi}{16}\cos\frac{(2y+1)n\pi}{16}, \tag{2.9}$$

onde x e y variam de 0 até 7.

Fazendo $P_{ij} = 1\ \forall i,j$ temos

$$F = \begin{bmatrix} a & a & a & a & a & a & a & a \\ b & c & d & e & -e & -d & -c & -b \\ f & h & -h & -f & -f & -h & h & f \\ c & -d & -b & -d & d & b & e & -c \\ a & -a & -a & a & a & -a & -a & a \\ d & -b & e & c & -c & -e & b & -d \\ h & -f & f & -h & -h & f & -f & h \\ e & -d & c & -b & b & -c & d & -e \end{bmatrix}$$

com

$$\begin{cases} a &= 1/4\sqrt{2} \\ b &= 1/2\cos(1/16\,\pi) \\ c &= 1/2\cos(3/16\,\pi) \\ d &= 1/2\cos(5/16\,\pi) \\ e &= 1/2\cos(7/16\,\pi) \\ f &= 1/2\cos(3/8\,\pi) \\ h &= 1/2\cos(1/8\,\pi) \end{cases}.$$

Numericamente,

$$F = \begin{bmatrix} 0.35 & 0.35 & 0.35 & 0.35 & 0.35 & 0.35 & 0.35 & 0.35 \\ 0.49 & 0.42 & 0.28 & 0.10 & -0.10 & -0.28 & -0.42 & -0.49 \\ 0.46 & 0.19 & -0.19 & -0.46 & -0.46 & -0.19 & 0.19 & 0.46 \\ 0.42 & -0.10 & -0.49 & -0.28 & 0.28 & 0.49 & 0.10 & -0.42 \\ 0.35 & -0.35 & -0.35 & 0.35 & 0.35 & -0.35 & -0.35 & 0.35 \\ 0.28 & -0.49 & 0.10 & 0.42 & -0.42 & -0.10 & 0.49 & -0.28 \\ 0.19 & -0.46 & 0.46 & -0.19 & -0.19 & 0.46 & -0.46 & 0.19 \\ 0.10 & -0.28 & 0.42 & -0.49 & 0.49 & -0.42 & 0.28 & -0.10 \end{bmatrix}$$

A matriz F pode ser representada pela figura 2.3, sendo a primeira linha constante representada pelo primeiro gráfico da figura 2.3, e assim sucessivamente cada gráfico representa uma linha.

Podemos representar os produtos da equação (2.5) através da figura 2.4 mostrando a combinação das funções cosseno verticais e horizontais.

Por razões históricas, chamamos o primeiro coeficiente de **DC** [7], onde $m+n=0$, e contém a cor média do bloco. Os demais coeficientes são chamados de **AC** [8].

Observe que o padrão da base mostra um aumento de variação vertical quando m aumenta e horizontal quando n aumenta.

Uma propriedade interessante em F é a concentração da energia perto do coeficiente DC. Assim, os coeficientes de Q tendem a zero quando m ou n tendem a 7.

Aplicamos a DCT (2.8) em cada matriz de pixel de dimensão 8×8 para separarmos as freqüências baixas das altas. Então, podemos aplicar a quantização.

[7] Direct Current
[8] Alternate Current

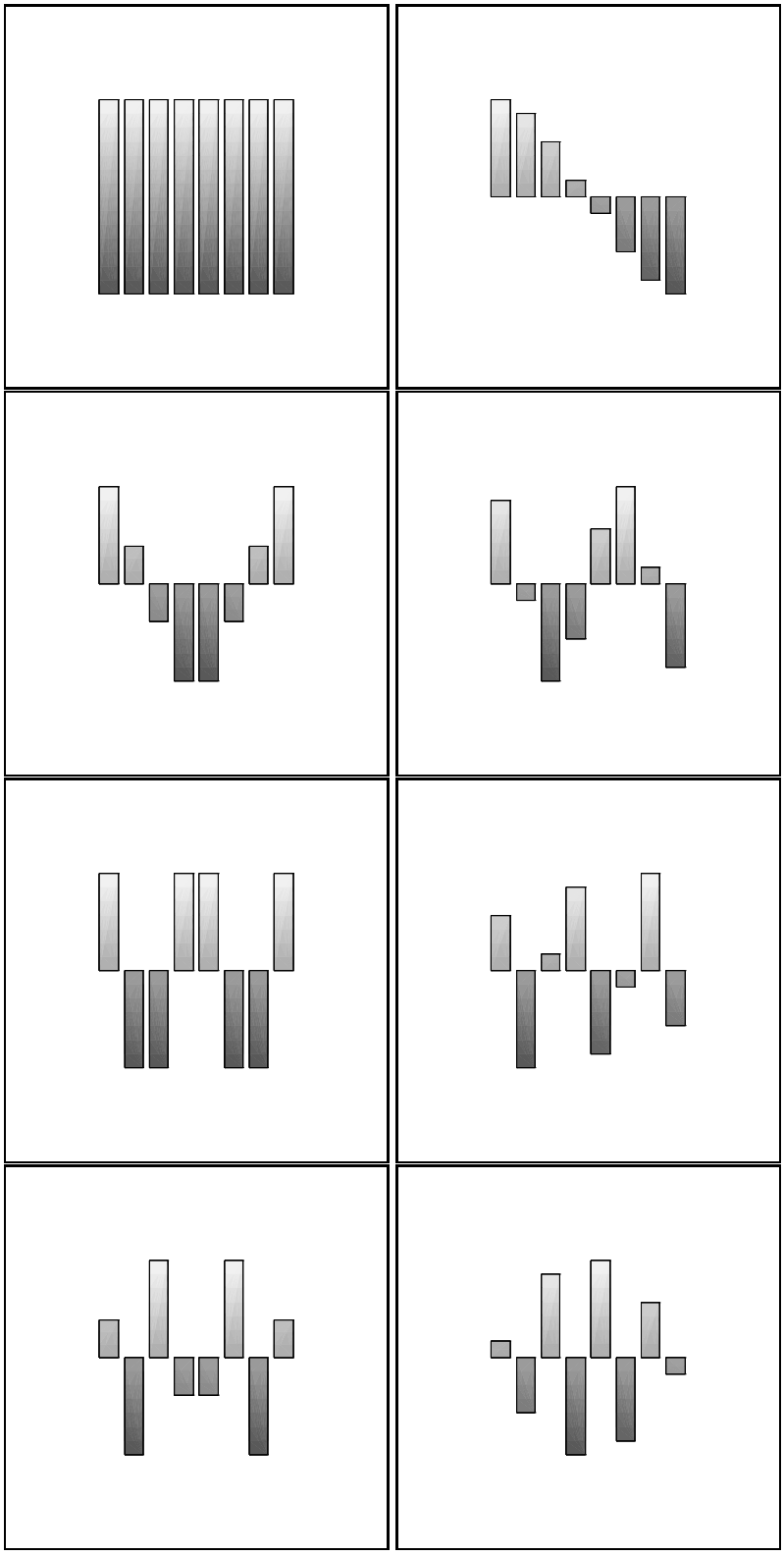


Figura 2.3: *Conjunto de medidas para a base da DCT.*

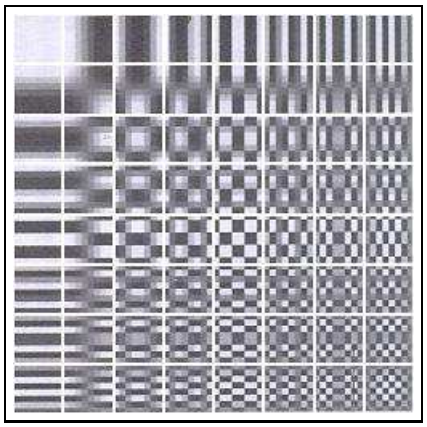


Figura 2.4: *Padrão da base da DCT* 8×8.

Com a quantização

$$F'[m,n] = \frac{F[m,n]}{Q[m,n]} \tag{2.10}$$

podemos controlar a taxa de compressão e a perda de informação da imagem através de ajustes na matriz Q.

Detalhes sobre os protocolos desta seção podem ser encontrados em (ITU-T, 1998b).

Efeito das aplicações nas matrizes

Considere a matriz $P_{8\times8}$ com coeficientes iguais a 1, resolvendo a equação (2.8) temos uma matriz $F_{8\times8}$, cujo coeficiente DC é igual a 8 e os coeficientes AC iguais a zero. Aplicando este resultado em (2.9) obtemos a mesma matriz $P_{8\times8}$, porém isto nem sempre acontece.

As matrizes F têm muitos zeros abaixo da diagonal principal, para que fiquem juntos e assim possamos usar a codificação por carreira, lê-se a matriz em zig-zag, isto é, $F_{1\times1} \rightarrow F_{1\times2} \rightarrow F_{2\times1} \cdots$.

Façamos

$$P = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

logo,

$$F = \begin{bmatrix} 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & -1 & 0 & -1 & 0 & -3 \end{bmatrix}.$$

Mesmo antes de aplicarmos a matriz de quantização Q já obteríamos um ganho sig-

nificativo na compressão. Seja

$$Q = \begin{bmatrix} 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\ 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\ 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 \\ 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 \\ 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 \end{bmatrix} \tag{2.11}$$

então,

$$F' = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

A perda de informação no arredondamento da DCT (2.8) é muito pequena comparada com a quantização (2.10). Podemos multiplicar os coeficientes de Q (2.11) por um fator maior que 1, caso desejemos descartar os coeficientes de alta freqüência. Para termos uma idéia, imagine que os coeficientes são cores de um byte, logo estão entre 0 e 255. Isto significa que a cor 0 é muito semelhante à cor 1.

Aplicando

$$F''[m,n] = F'[m,n]Q[m,n],$$

seguido da inversa da DCT (2.9), temos

$$P' = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Observe que se a DCT não fosse discreta teríamos

$$F = \begin{bmatrix} 4.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & -0.1299457662 & 0.0 & -0.1532814823 & 0.0 & -0.2294019498 & 0.0 & -0.6532814822 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & -0.1532814823 & 0.0 & -0.1808078364 & 0.0 & -0.2705980499 & 0.0 & -0.7705980499 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & -0.2294019498 & 0.0 & -0.2705980499 & 0.0 & -0.4049786010 & 0.0 & -1.153281482 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & -0.6532814822 & 0.0 & -0.7705980499 & 0.0 & -1.153281482 & 0.0 & -3.284267796 \end{bmatrix}.$$

Vejamos mais um exemplo, porém desta vez com mudança brusca de cor, diferente

do exemplo anterior. Seja

$$P = \begin{bmatrix} 0 & 100 & 0 & 100 & 0 & 100 & 0 & 100 \\ 100 & 0 & 100 & 0 & 100 & 0 & 100 & 0 \\ 0 & 100 & 0 & 100 & 0 & 100 & 0 & 100 \\ 100 & 0 & 100 & 0 & 100 & 0 & 100 & 0 \\ 0 & 100 & 0 & 100 & 0 & 100 & 0 & 100 \\ 100 & 0 & 100 & 0 & 100 & 0 & 100 & 0 \\ 0 & 100 & 0 & 100 & 0 & 100 & 0 & 100 \\ 100 & 0 & 100 & 0 & 100 & 0 & 100 & 0 \end{bmatrix}.$$

Logo,

$$F = \begin{bmatrix} 400 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -13 & 0 & -15 & 0 & -23 & 0 & -65 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -15 & 0 & -18 & 0 & -27 & 0 & -77 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -23 & 0 & -27 & 0 & -40 & 0 & -115 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -65 & 0 & -77 & 0 & -115 & 0 & -328 \end{bmatrix}$$

e

$$F' = \begin{bmatrix} 200 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -3 & 0 & -3 & 0 & -3 & 0 & -7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -3 & 0 & -2 & 0 & -3 & 0 & -6 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -3 & 0 & -3 & 0 & -3 & 0 & -8 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -7 & 0 & -6 & 0 & -8 & 0 & -21 \end{bmatrix},$$

pois usamos a mesma matriz Q (2.11).

Por fim, temos a

$$P' = \begin{bmatrix} 11 & 87 & 13 & 87 & 13 & 87 & 13 & 89 \\ 64 & 38 & 63 & 37 & 63 & 37 & 62 & 36 \\ 53 & 48 & 53 & 50 & 50 & 47 & 52 & 47 \\ 37 & 62 & 38 & 65 & 35 & 62 & 38 & 63 \\ 63 & 38 & 62 & 35 & 65 & 38 & 62 & 37 \\ 47 & 52 & 47 & 50 & 50 & 53 & 48 & 53 \\ 36 & 62 & 37 & 63 & 37 & 63 & 38 & 64 \\ 89 & 13 & 87 & 13 & 87 & 13 & 87 & 11 \end{bmatrix}$$

visualmente próxima de P. Observe a diferença nos gráficos 2.5.

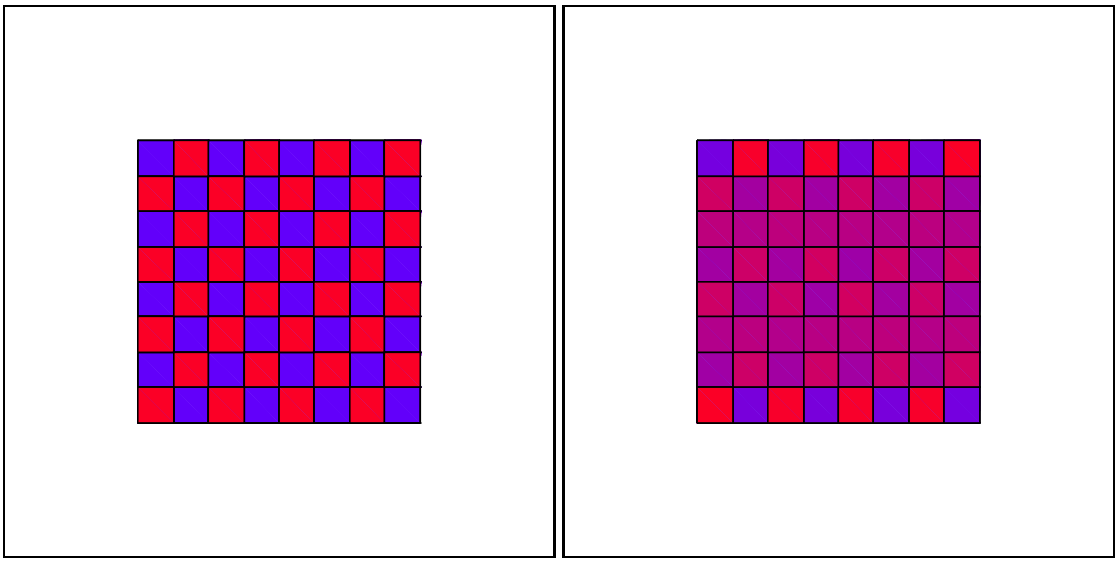


Figura 2.5: *Diferenças entre as matrizes de pixel P e P′.*

Efeito das aplicações nas imagens

Dado uma matriz de quantização Q, podemos aumentar a quantização multiplicando

Q por uma constante k, assim a resolução R é data por $R = 100 - k$. Quando $k = 0$ não há quantização.

Compare a diferença entre as imagens na figura 2.6 e seus respectivos histogramas na figura 2.7. Tais imagens estão com 80%, 50%, 20% e 1% de resolução.

Na figura 2.7 as abscissas representam os valores normalizados das cores, enquanto as coordenadas a freqüência das cores.

Pelo histograma, observamos que a imagem apresentada na figura 2.6 perde resolução mais rápido que a famosa imagem de Lena devido ao maior número de coeficientes de alta freqüência.

2.3.2 Medindo a Distorção

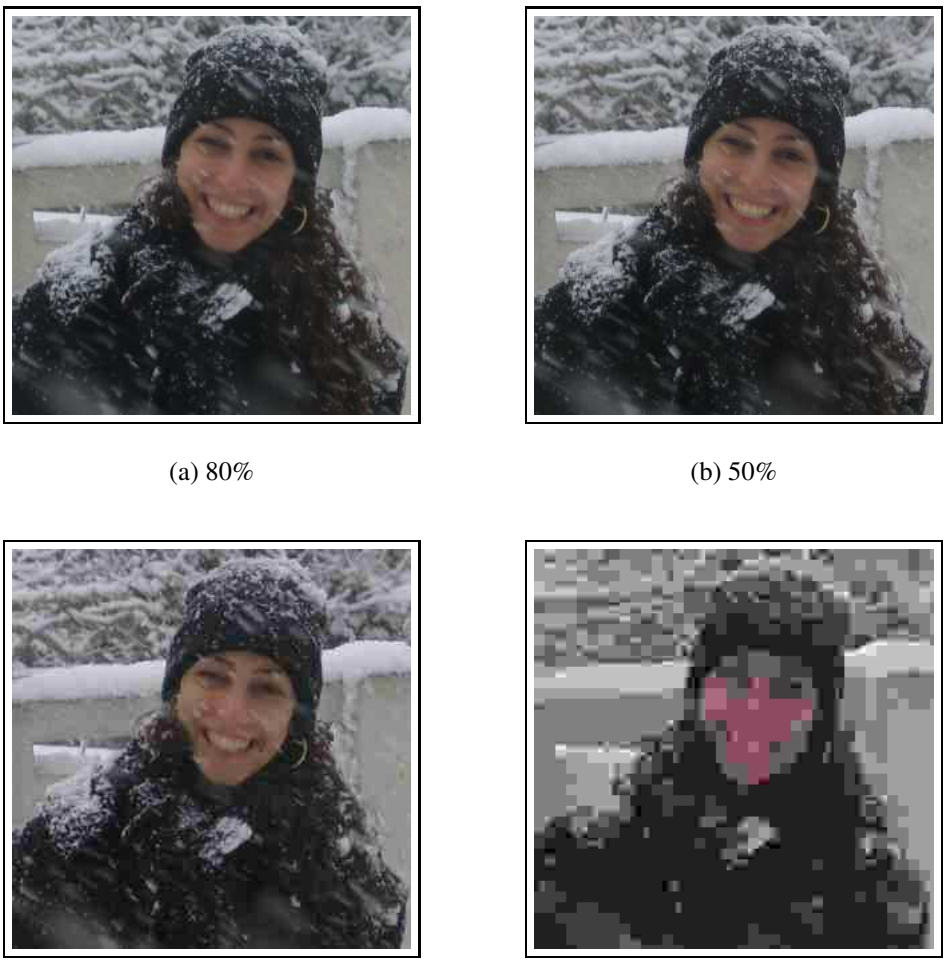
Medindo a entropia das imagens apresentadas na figura 2.6, veremos que ela diminui conforme aumentamos a quantização. Isto nos diz duas coisas: Primeiro, poderíamos comprimir mais ainda a imagem. Segundo, a entropia não nos serve como métrica para distorção do sinal, no nosso caso o quanto a imagem foi alterada.

Existem várias medidas para distorção e dependendo do caso, é utilizada uma métrica mais apropriada. Veremos alguns casos nesta seção.

Seja I a imagem e I' a imagem alterada, então podemos usar como medida a diferença absoluta,

$$d(x,x') = |x - x'|, \tag{2.12}$$

com $x \in I$ e $x' \in I'$.



(a) 80% (b) 50%

(c) 20% (d) 1%

Figura 2.6: *Diferenças da quantização na imagem.*

Poderíamos, também, medir o erro quadrado,

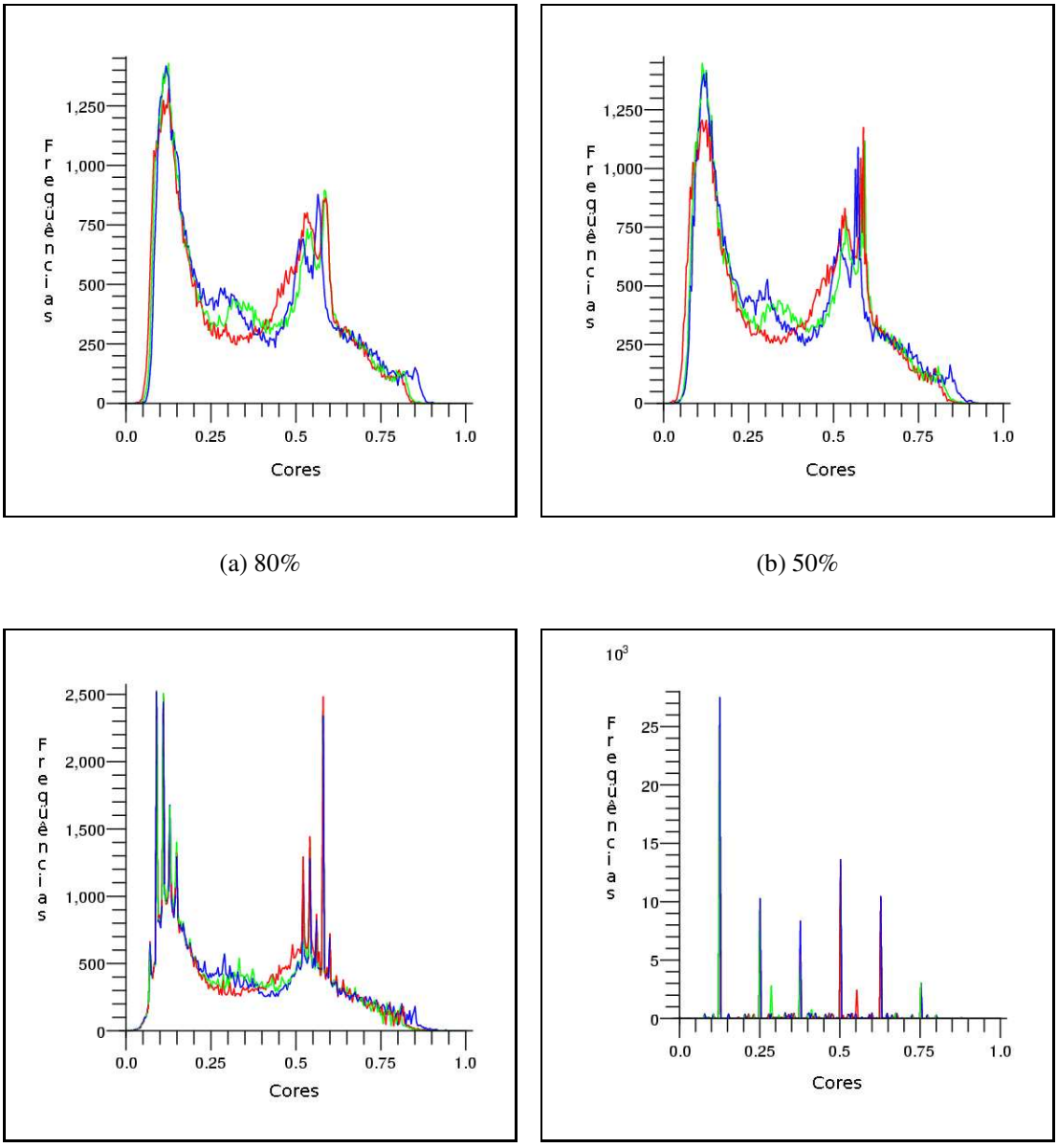
$$d(x,x') = (x-x')^2. \tag{2.13}$$

As expressões (2.12) e (2.13) nos fornecem uma média pontual, por isto é muito mais comum usarmos o erro quadrado médio (**MSE**) [9], dado por

$$\sigma_d^2 = \frac{1}{MN}\sum_{m=1}^{M}\sum_{n=1}^{N}(x_{mn}-x'_{mn})^2.$$

Para medir o erro relativo, podemos calcular a razão do valor do quadrado médio pelo

[9] Mean Square Error



(a) 80%

(b) 50%

(c) 20%

(d) 1%

Figura 2.7: *Diferenças da quantização no histograma.*

MSE. Assim, teremos a relação sinal-ruído (**SNR**) [10], dada por

$$\mathrm{SNR} = \frac{\sigma_x^2}{\sigma_d^2},$$

onde σ_x é o valor do quadrado médio e σ_d o MSE.

Normalmente, medimos o SNR em escala logarítmica e sua unidade de medida é dada em decibéis (dB), logo

$$\mathrm{SNR}_{dB} = 10\log_{10}\frac{\sigma_x^2}{\sigma_d^2}.$$

Podemos também, estar interessados no tamanho do erro relativo para o valor máximo do sinal x_{peak}. Assim, substituindo σ_x por x_{peak} obtemos a taxa de pico da relação sinal-ruído (**PSNR**) [11],

$$\mathrm{PSNR}_{dB} = 10\log_{10}\frac{x_{peak}^2}{\sigma_d^2} = 20\log_{10}\frac{x_{peak}}{\sqrt{\sigma_d}}.$$

Como estamos trabalhando com bits, temos

$$\mathrm{PSNR}_{dB} = 20\log_{10}\frac{2^b - 1}{\sqrt{\mathrm{MSE}}},$$

onde b é o número de bits por amostra da imagem.

Daqui para frente quando nos referirmos a resolução Normal estamos falando de um mapa de bits enquanto 100% de resolução é a imagem comprimida com a resolução máxima de um arquivo JPEG.

Na tabela 2.12 temos os tamanhos dos arquivos versus entropia e PSNR. Usando apenas Huffman na figura 2.6 temos 248 767 bytes de arquivo mais 1 752 bytes de tabela que é um valor próximo da resolução Normal. No entanto a entropia 7.9798 difere.

Os conceitos introduzidos nesta seção serão usados como ferramentas na criptografia ou na esteganografia.

[10] Signal-to-Noise Ratio
[11] Peak signal-to-noise ratio

Resolução	**Tamanho**	**Entropia**	**PSNR**
Normal	269 154	7.3743	∞
100%	74 969	7.9699	46.8611
80%	19 036	7.9668	35.4806
50%	10 971	7.9331	32.4717
20%	5 802	7.8725	29.7059
1%	1 166	5.6117	22.6029

Tabela 2.12: *Comparação de compressão com a figura 2.6.*

3 Cifrando a mensagem

Neste capítulo tratamos de criptografia, cuja técnica impede que uma mensagem seja interceptada, alterada ou fabricada.

De forma abstrata, temos uma transformação T, invertível, que leva a mensagem M em um **criptograma** C, isto é

$$C = T(M)$$

e

$$M = T^{-1}(C).$$

As tentativas de descobrir M a partir de C, são métodos de **criptoanálise**. Cada transformação T gera um sistema de criptografia conhecido como **criptossistema**. Hoje em dia os criptossistemas são conhecidos, sendo secreto apenas uma chave k que possibilita cifrar e decifrar a mensagem.

Shannon (SHANNON, 1949) identificou a difusão como uma propriedade em um criptossistema.

A **difusão** é a propriedade que “dissipa"a redundância estatística de uma mensagem. O ideal seria que todas as letras fossem equiprováveis na relação $M_i \rightarrow C_i$. Isto significa que se cifrarmos uma imagem com longo intervalo de céu azul, no criptograma não aparecerá seqüências de letras.

Uma métrica para difusão d poderia ser dada por

$$d = \frac{H(C)}{H(M)},$$

desde que M tenha informação.

O conceito de difusão fornece duas características de um bom criptossistema:

- Alta difusão quando $H(M) \to 0$;
- Hipersensibilidade a M, pequenas mudanças na mensagem alteram a difusão.

Apresentamos agora nossa métrica para calcular a difusão, ela será usada na secção 3.3.3.

Uma métrica mais refinada deveria usar o conceito de redundância. Pode-se mostrar (BRUEN; FORCINITO, 2004) que a redundância R é dada por

$$R = 1 - \frac{H}{\log_2 |\mathscr{A}|},$$

em um alfabeto binário temos

$$R = 1 - H.$$

Logo, a diferença da redundância de um criptograma e de uma mensagem nos fornece uma métrica para a difusão. Assim,

$$d = R(M) - R(C) = H(C) - H(M).$$

Note que o conceito de difusão na física normalmente é formulado por uma razão. Assim, considere a probabilidade de uma letra em relação a mensagem e ao criptograma, ou seja,

$$p = P(\mathscr{A}_i) \in C$$

e

$$q = P(\mathscr{A}_i) \in M$$

onde i é fixo.

Desta forma,

$$\log\left(\frac{p}{q}\right) = \log p - \log q = I(q) - I(p).$$

Queremos uma razão entre as probabilidades das letras da mensagem e do criptograma. Assim, considere que

$$\log\left(\frac{\sum P(M_i)^{P(M_i)}}{\sum P(C_j)^{P(C_j)}}\right) = \log\left(\sum P(M_j)^{P(M_j)}\right) - \log\left(\sum P(C_j)^{P(C_j)}\right) = H(C) - H(M)$$

é uma boa métrica para a difusão.

Esta métrica nos indica uma boa cifra, no sentido que vai ser difícil de quebrá-la por métodos estatísticos. No entanto, ela não garante que dado T seja difícil encontrar T^{-1}.

Suponha que a tabela do algoritmo de Huffman fosse a chave, tal compressão gera alta difusão. No entanto, dada uma quantidade suficiente de mensagens compactadas, pode-se descobrir a chave-tabela.

Definição. O princípio de **Kerchoff** afirma que a segurança de um criptossistema deve estar apenas na chave (BRUEN; FORCINITO, 2004).

Baseado neste princípio um intruso teria:

- Conhecimento do criptossistema e implementação;
- Grande quantidade de criptogramas;
- O poder computacional máximo.

Logo, a criptografia é considerada segura, quando ninguém consegue quebrá-la, após um exaustivo e extenso estudo de criptoanálise.

3.1 Simétricos

Classificamos os criptossistemas como **simétricos**, quando dada uma chave pode-se cifrar e decifrar a mensagem.

3.1.1 Substituição

Podemos construir um criptossistema baseado na substituição do alfabeto.

Seja $f : \mathscr{A} \to \mathscr{A}$ uma função bijetora. Segundo a tabela 3.1, a mensagem LNCC é levada para o criptograma NXDD.

$\mathscr{A}$	$\overset{f}{\longleftrightarrow}$	$\mathscr{A}$
A	$\leftrightarrow$	Q
B	$\leftrightarrow$	V
C	$\leftrightarrow$	D
D	$\leftrightarrow$	I
E	$\leftrightarrow$	J
F	$\leftrightarrow$	T
G	$\leftrightarrow$	P
H	$\leftrightarrow$	O
I	$\leftrightarrow$	C
J	$\leftrightarrow$	Y
K	$\leftrightarrow$	H
L	$\leftrightarrow$	N
M	$\leftrightarrow$	G

$\mathscr{A}$	$\overset{f}{\longleftrightarrow}$	$\mathscr{A}$
N	$\leftrightarrow$	X
O	$\leftrightarrow$	A
P	$\leftrightarrow$	Z
Q	$\leftrightarrow$	W
R	$\leftrightarrow$	U
S	$\leftrightarrow$	S
T	$\leftrightarrow$	M
U	$\leftrightarrow$	F
V	$\leftrightarrow$	K
W	$\leftrightarrow$	R
X	$\leftrightarrow$	L
Y	$\leftrightarrow$	B
Z	$\leftrightarrow$	E

Tabela 3.1: *Criptossistema por substituição.*

Um inconveniente neste criptossistema é guardar um alfabeto paralelo. Para sanar este problema o exército romano usava um caso particular chamado **Código de César**.

Considere uma função α bijetora que leva os elementos de $\mathscr{A}$ em um anel, ou seja,

$$\begin{array}{rl} \alpha: & \mathscr{A} \to R \\ \alpha(a): & a \mapsto r \end{array}. \tag{3.1}$$

A utilidade de α é converter letras em números para que possamos ter uma álgebra das letras.

Seja f uma função bijetora, tal que

$$\begin{array}{rl} f: & R \to R \\ f(x): & x \mapsto ax+b \mod |R| \end{array} \tag{3.2}$$

Para $\{a,b\} \subset R$, com a condição $(a,|R|) = 1$ satisfeita. Esta condição implica que a tem inverso multiplicativo, ou seja, a pertence ao Grupo das Unidades.[1]

Com o Código de César, pode-se combinar apenas dois números como chave do criptossistema.

Por simplicidade, vamos assumir $\mathscr{A} = \{\not\emptyset, \mathrm{A}, \ldots, \mathrm{Z}\}$ e R o anel $\mathbb{Z}_{27}$. Com a senha 5 e 3 temos $f(x) = 5x+3$. Mas com $5 \times 11 \equiv 55 \equiv 1 \mod 27$, segue que $f^{-1}(x) = a^{-1}(x-b) = 11(x-3)$.

Cifrando a mensagem LNCC temos o criptograma "ISRR", pois

$$\alpha: \text{“LNCC”} \mapsto [12,14,3,3]$$

$$f: [12,14,3,3] \mapsto [9,19,18,18]$$

$$\alpha^{-1}: [9,19,18,18] \mapsto \text{”ISRR”}.$$

Para decifrar basta seguir o caminho inverso.

Na criptoanálise, se soubéssemos que o método é da forma (3.2) e R é um corpo, então o espaço de busca para a é $|R|-1$ para valores de a e $|R|$ para valores de b. Desta

[1] {1, 2, 4, 5, 7, 8, 10, 11, 13, 14, 16, 17, 19, 20, 22, 23, 25, 26}

forma, o total de chaves $|\mathscr{K}|$ é dado por

$$|\mathscr{K}| = |R|(|R|-1).$$

No exemplo acima, como $\mathbb{Z}_{27}$ não é um corpo, temos $27 \times 18 = 486$ chaves diferentes. Mas isto nos fornece um espaço de busca extremamente pequeno, sendo fácil para um intruso ler a mensagem.

No caso de substituição do alfabeto, o número de possibilidades é muito alto, $26! - 1 \approx 4.03\ 10^{26}$. No entanto, $H(M) - H(C) = 0$, isto é, não há difusão.

Para uma quantidade grande de criptogramas é possível ler a mensagem usando os conhecimentos da seção 2.1.2.

Uma forma de barrar os ataques estatísticos, isto é baseados na análise de freqüência, é usar substituição em blocos de letras.

Se substituirmos em blocos de 4 letras, temos $27^4 = 531441$ possibilidades de chaves, porém a entropia começa a mudar.

3.1.2 Método de Hill

Este método (HILL, 1929, 1931) foi apresentado antes do advento da informática, mesmo assim teve grande influência teórica. Além disto, a entropia e a difusão aumentam conforme aumenta a dimensão de uma determinada matriz K.

Seja uma matriz $K_{n\times n}$ invertível sobre um anel R, isto é,

$$(\det K, |R|) = 1.$$

O método consiste em agrupar a mensagem em vetores P_i de comprimento n e aplicar

uma função definida como

$$\begin{aligned} f : R^n &\rightarrow R^n \\ P_i &\mapsto P_i K. \end{aligned}$$

Vejamos um exemplo do método de Hill. Pretendemos enviar a mensagem: LNCC. Previamente, temos que ter combinado uma chave em um canal seguro. Quando combinamos

$$K = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

temos que testar se K é uma chave válida

$$\det(K) = -2 \equiv 25 \mod 27.$$

De posse da chave, podemos cifrar a mensagem, escrevendo-a na forma de vetor aplicando α dada pela equação (3.1),

$$\alpha : \text{ “LNCC” } \mapsto [12, 14, 3, 3],$$

depois separando em vetores, $P_1 = [12, 14]$ e $P_2 = [3, 3]$, então efetuando o produto matricial,

$$P_1 K = [0, 26],$$

$$P_2 K = [12, 18].$$

Finalmente, obtemos o criptograma aplicando a inversa de α,

$$\alpha^{-1} : [0, 26, 12, 18] \mapsto \text{ “}\not{\text{o}}\text{ZLR”}$$

Para decifrar o criptograma basta calcularmos f^{-1} e aplicarmos sobre os vetores do criptograma, ou seja,

$$f^{-1}(C_i) = C_i K^{-1},$$

onde

$$K^{-1} = \frac{1}{\det(K)} (\operatorname{adj} K).$$

Para calcularmos a inversa, precisamos da matriz de cofatores

$$C_{ij} = (-1)^{i+j} d_{ij} \qquad i, j = 1, \cdots, n,$$

onde d_{ij} é o determinante de K, excluindo a linha i e coluna j. Finalmente, a matriz adjunta é definida como a transposta da matriz de cofatores de K,

$$\operatorname{adj} K = \begin{bmatrix} C_{11} & C_{12} & \cdots & C_{1n} \\ C_{12} & C_{22} & \cdots & C_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ C_{1n} & C_{2n} & \cdots & C_{nn} \end{bmatrix}.$$

Vejamos um exemplo de como decifrar uma mensagem. Considere o criptograma, ⊮ZLR, obtido no exemplo anterior e a mesma chave K,

$$K^{-1} = \frac{1}{25}\begin{bmatrix} 4 & -3 \\ -2 & 1 \end{bmatrix} = 13\begin{bmatrix} 4 & 24 \\ 25 & 1 \end{bmatrix} \equiv \begin{bmatrix} 25 & 1 \\ 15 & 13 \end{bmatrix}.$$

Aplicando

$$f^{-1} : [0, 26, 12, 18] \mapsto [12, 14, 3, 3]$$

e

$$\alpha^{-1} : [12, 14, 3, 3] \mapsto \text{ ”LNCC”}$$

obtemos a mensagem.

Deficiência do Método de Hill

Apesar do método aumentar consideravelmente a entropia, conforme aumentamos a dimensão da matriz K, foi encontrada uma grave vulnerabilidade do método.

Suponha que o criptoanalista intercepte o criptograma U⊮NPBUJLPIKAANFRVWRL. Na tentativa de quebrar a cifra, ele aplica α^{-1} e obtém

$$[21, 0, 14, 16, 2, 21, 10, 12, 16, 9, 11, 1, 1, 14, 6, 18, 22, 23, 18, 12].$$

Se ele ficar sabendo ou deduzir que a mensagem termina com LNCC, isto é [12, 14,

3, 3], então certamente tentará encontrar

$$K = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

calculando

$$[12,14]K = [22,23]$$

e

$$[3,3]K = [18,12].$$

Desta forma, calculará

$$\begin{bmatrix} 12 & 14 \\ 3 & 3 \end{bmatrix} \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} 22 & 23 \\ 18 & 12 \end{bmatrix}$$

e tentará resolver o sistema

$$\begin{bmatrix} 12a+14c & 12b+14d \\ 3a+3c & 3b+3d \end{bmatrix} = \begin{bmatrix} 22 & 23 \\ 18 & 12 \end{bmatrix} = S.$$

Porém, como o $(\det S, 27) = 3$ o sistema não tem solução única. Este é o pior caso, no entanto, a primeira solução nos fornece

$$K = \begin{bmatrix} 4 & 3 \\ 2 & 1 \end{bmatrix}.$$

Assim, se obtém que $M =$"CRIPTOGRAFIA⊘NO⊘LNCC".

Método de Hill Generalizado

Vamos descrever agora três formas de prover mais segurança no método: 1. embaralhando o criptograma com a mensagem, 2. embaralhando o criptograma com o próprio criptograma, 3. embaralhando o criptograma com uma seqüência. Seja uma matriz $A_{n\times n}$

invertível sobre um anel R. Agrupe a mensagem em vetores P_i de comprimento n e defina

$$\begin{aligned} f : R^n &\rightarrow R^n \\ P_i &\mapsto P_iA + B_i \end{aligned}$$

1. $B_i = P_{i-1}B$, onde $B_{n\times n}$ está sobre R, dado um vetor inicial P_0.
2. $B_i = C_{i-1}B$, dado um valor inicial C_0.
3. $B_i = (r_i, r_{i+1}, \cdots, r_{i+n-1})$, onde $\{r_j\}$ é uma seqüência recursiva sobre R, dado um valor inicial r_j.

Matrizes Involutórias

Um ponto interessante do método é a facilidade da mesma chave poder ser usada para cifrar e decifrar a mensagem, sem a necessidade de se calcular a matriz inversa. As matrizes que satisfazem a condição $K^2 = I$ são conhecidas como matrizes Involutórias (Levine, Jack; Nahikian, H. M., 1962).

Podemos gerar, facilmente, infinitas matrizes Involutórias, basta escolhermos $A_{r\times s}$ e $B_{s\times r}$ ambos sobre R e calcularmos

$$K = \begin{bmatrix} BA - I & B \\ 2A - ABA & I - AB \end{bmatrix}.$$

Observe que as matrizes Involutórias formam um grupo abeliano. Sejam K e K' Involutórias, assim

$$(KK')^2 = I = K^2K'^2$$

Logo,

$$KK'KK' = KKK'K'.$$

Portanto,

$$K'K = KK'.$$

O uso de matrizes Involutórias facilita muito o método, porém se todos usarem tais matrizes como chave, o criptoanalista pode descobrir as chaves.

Problema das Duas Mensagens

Imagine que uma mensagem foi enviada usando uma matriz Involutória, o destinatário repassa a mesma mensagem para outra pessoa, usando outra matriz Involutória, este processo gera uma vulnerabilidade no método.

Seja K e K' Involutórias, assim teremos dois criptogramas diferentes com a mesma mensagem,

$$C_i = P_i K$$

e

$$C'_i = P_i K'.$$

Calculamos

$$C_i K = P_i$$

e

$$C'_i = C_i K K'.$$

Fazendo $S = [C_{i_1}, \ldots, C_{i_n}]$ e $T = [C'_{i_1}, \ldots, C'_{i_n}]$ temos $T = SKK'$. Se possível, calculamos $KK' = S^{-1}T$ e $T^{-1} = K'KS^{-1}$ e finalmente obtemos

$$(KK')X = X(KK')^{-1}$$

que é equivalente a

$$(KK')X = X(K'K).$$

As matrizes K e K' vão satisfazer as equações acima, possibilitando ao criptoanalista descobrir as duas chaves.

3.1.3 RC4

O RC4 é um método apresentado por Rivest. Atualmente, é muito usado nas redes de computadores, especificamente no protocolo **SSL**[2], por ser considerado muito rápido e seguro (STALLINGS, 2002).

Diferente do método de Hill que trabalha em blocos, o RC4 trabalha com um fluxo. Por isto, são classificados como **cipher block** e **cipher stream**, respectivamente.

Este algoritmo é dividido em três laços responsáveis pela inicialização, permutação e criptografia.

Aqui, apresentamos o algoritmo 1 de forma generalizada, independente do alfabeto. O método foi criado considerando a tabela ASCII como alfabeto.

Algoritmo 1 RC4

Recebe k e M
para $i := 0$ até $|\mathscr{A}| - 1$ **faça**
    $s_i := i$
    $t_i := k_{(i \mod |k|)+1}$
$j := 0$
para $i := 0$ até $|\mathscr{A}| - 1$ **faça**
    $j := (j + s_i + t_i) \mod \mathscr{A}$
    Troque(s_j, s_i)
$i := 0$
$j := 0$
para $n := 1$ até $|M|$ **faça**
    $i := i + 1 \mod \mathscr{A}$
    $j := j + s_i \mod \mathscr{A}$
    Troque(s_j, s_i)
    $t := s_i + s_j \mod \mathscr{A}$
    $C_n := M_n + s_t \mod \mathscr{A}$
retorne C

Compare a figura 3.1 com o algoritmo 1.

[2]Secure Sockets Layer

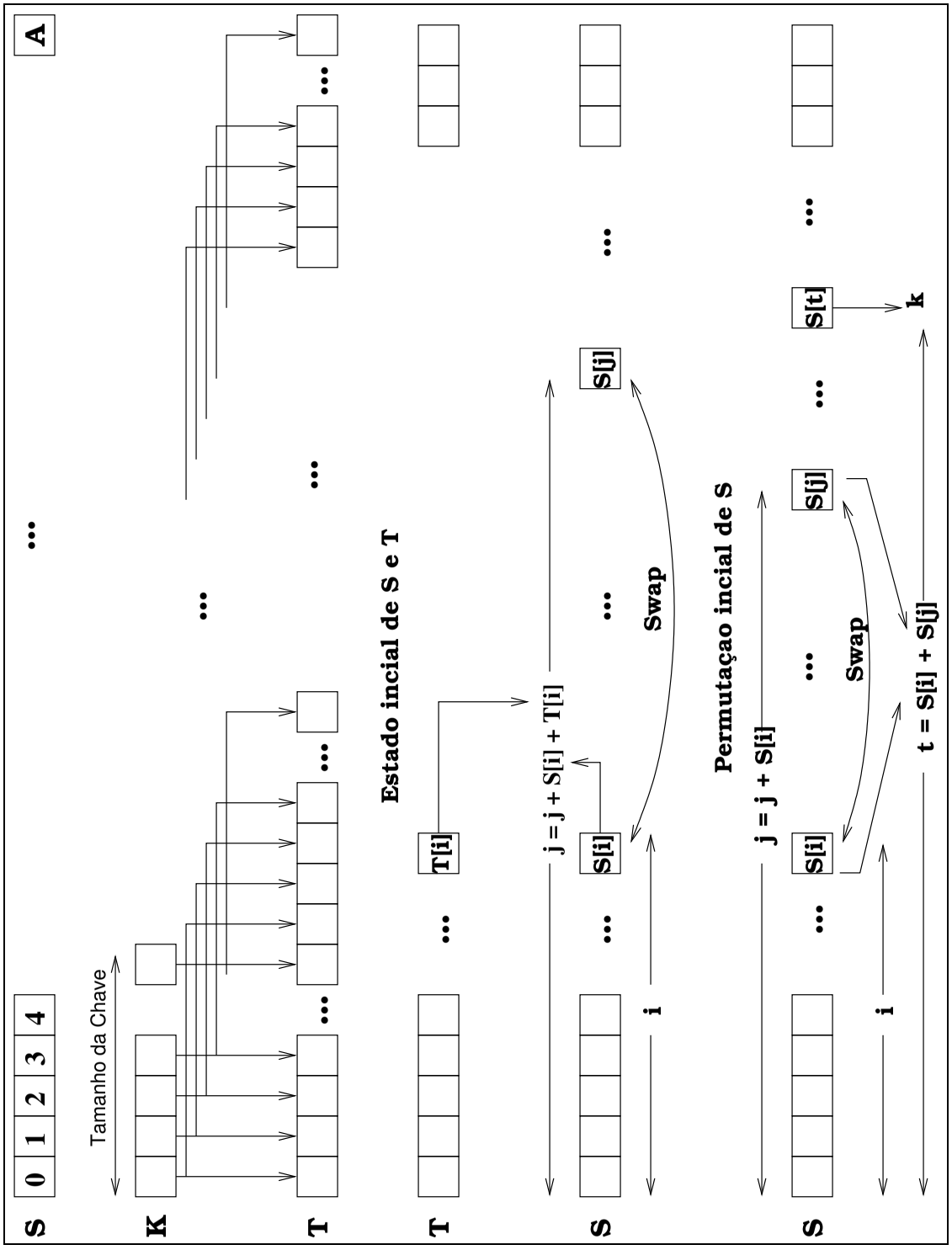


Figura 3.1: *Esquema do RC4.*

Temos aqui um algoritmo rápido e seguro, mas, por ser simétrico, requer que seja combinada uma senha antes de usá-lo. Tal requisição é inviável para comércio eletrônico, ou mesmo, a comunicação segura entre computadores. A próxima seção apresenta o RSA, um método assimétrico muito usado em conjunto com o RC4.

3.2 Assimétricos

Classificamos os criptossistemas como **assimétricos** quando possuem duas chaves, uma secreta e outra pública. Isto é usamos uma chave para cifrar e outra para decifrar a mensagem.

3.2.1 RSA

Este criptossistema se baseia na dificuldade de encontrar os fatores de números grandes. Hoje em dia, tais números são da ordem de 1024 bits, aproximadamente 309 casas decimais.

Diferente dos métodos anteriores, o RSA é um criptossistema assimétrico.

O método seguinte apresentado em 1978 (RIVEST; SHAMIR; ADLEMAN, 1978) é amplamente utilizado na Internet.

Os assimétricos também são conhecidos como criptossistemas de **chave pública**, já que uma das chaves é de conhecimento público. Pode-se ter uma visão geral destes métodos através do artigo (KOBLITZ; MENEZES, 2004).

Os métodos assimétricos se baseiam em funções unidirecionais.

Definição. Uma função unidirecional f é uma função que se encontra $y = f(x)$ facil-

mente e dado y é inviável encontrar $f(x)$.

Vamos construir a assimetria com os seguintes resultados da Teoria dos Números.

Uma propriedade interessante, que facilita as contas no RSA, é garantida pelo teorema abaixo.

Teorema 2 *Sejam $m,n \in \mathbb{N}$ tais que $(m,n) = 1$. Então,*

$$\varphi(nm) = \varphi(n)\varphi(m).$$

Teorema 3 (Teorema de Euler) *Sejam $a \in \mathbb{Z}$ e $m \in \mathbb{N}$, tais que $(a,m) = 1$. Então,*

$$a^{\varphi(m)} \equiv 1 \mod m.$$

Pode-se encontrar as demonstrações dos teoremas 2 e 3 em (SHOKRANIAN; SOARES; GODINHO, 1999) e (KLIMA; SIGMON; STITZINGER, 2000).

O teorema seguinte vai nos garantir que a função construída vai ter inversa, isto é, podemos recuperar a mensagem a partir do criptograma.

Teorema 4 *Sejam p e q números primos com $p \neq q$, seja $\varphi = \varphi(pq)$. Se $a,b \in \mathbb{Z}$ tal que $ab \equiv 1 \mod \varphi$ então $x^{ab} \equiv x \mod pq \; \forall x \in \mathbb{Z}$.*

Prova. Se $ab \equiv 1 \mod \varphi$, então $ab = 1 + k\varphi$ com $k \in \mathbb{Z}$, logo,

$$x^{ab} = x^{1+k\varphi} = x(x^{k\varphi}) = x(x^{p-1})^{k(q-1)}$$

Se $(x,p) = 1$, então $x^{p-1} \equiv 1 \mod p$. Logo, $x^{ab} \equiv x(1)^{k(q-1)} \equiv x \mod p$. Idem para $x^{ab} \equiv x \mod q$. Portanto, $pq \mid (x^{ab} - x) \Leftrightarrow x^{ab} \equiv x \mod pq$ ■

O método

Escolha dois primos p e q grandes e calcule

$$\varphi = \varphi(pq) = (p-1)(q-1).$$

Escolha a inversível, isto é,

$$(a, \varphi) = 1.$$

Com algoritmo Euclidiano Estendido encontre b, tal que

$$ab \equiv 1 \mod \varphi$$

Finalmente, temos que

$$x^{ab} \equiv x \mod pq \quad \forall x \in \mathbb{Z}.$$

Isto significa que a e b são inversas.

Assim, usamos a como chave privada e b como chave pública, também deixamos o produto pq de conhecimento público.

Dada uma chave, fica inviável calcular a outra chave conhecendo apenas o produto dos primos.

Como exemplo, vamos usar nosso alfabeto em $\mathbb{Z}_{27}$.

Se uma mensagem vai de

$$m : A \rightsquigarrow B,$$

então, o destinatário B escolhe $p = 71$ e $q = 97$ e faz o produto $pq = 6887$.

Depois escolhe $a = 9$ e calcula $(9, \varphi) = 3$, isto significa que não é possível encontrar a inversa de a.

O destinatário B escolhe outro valor $a = 151$, calcula $(151, \varphi) = 1$.

De posse de uma chave, B usa o algoritmo Euclidiano Estendido para encontrar $b = 6631$.

O destinatário B envia b e pq para A cifrar a mensagem.

Recebendo $b = 6631$ e $pq = 6887$, o remetente A calcula:

- $P_1 = 1214 \leftrightarrow$ "LN"
- $P_2 = 0303 \leftrightarrow$ "CC"
- $C_1 = P_1^b \mod pq = 6726$
- $C_2 = P_2^b \mod pq = 3306$
- $f : [1214, 303] \mapsto [6726, 3306]$

Então, A envia [6726,3306] para B.

Como somente B conhece $a = 151$, então a mensagem é decifrada calculando:

- $C_1^a \mod pq = 6726^a \mod 6887 = 1214$
- $C_1^a \mod pq = 3306^a \mod 6887 = 303$
- $\alpha^{-1} : [12, 14, 3, 3] \mapsto$ "LNCC"

Enquanto nas cifras por substituição a entropia não muda, é interessante observar que no RSA, com chave de 1024 bits, a entropia de texto é superior a 7.99.

Ataques

Existem muitos ataques ao RSA, porém nenhum deles se mostrou eficaz. Mostramos nesta seção, que uma má escolha dos números primos pode deixar o criptossistema vulnerável.

A escolha dos primos p e q deve ser feita com cuidado. Se forem suficientemente próximos, podemos determiná-los rapidamente a partir do produto $n = pq$.

Seja

$$x = \frac{p+q}{2}$$

e

$$y = \frac{p-q}{2},$$

temos

$$n = pq = x^2 - y^2 = (x+y)(x-y).$$

Para encontrarmos x e y escolhemos $x = \lceil \sqrt{n} \rceil$, então $x^2 - n$ deve ser um quadrado perfeito y^2. Caso não seja, procuramos na vizinhança de x.

Por outro lado, p e q não podem ser muito distantes.

Vejamos um exemplo de ataque.

Queremos determinar p e q a partir de $n = 1520273$. Para encontrar x e y, escolhemos

$$x = \lceil \sqrt{1520273} \rceil = 1233.$$

Então, $x^2 - n = 16 = y^2$.

Portanto, $p = 1233 - 4$ e $q = 1233 + 4$.

No RSA, somente a chave pública e o produto dos primos n podem ser divulgados. Caso outra informação seja descoberta, pode-se determinar a chave privada.

Note que $n-(p-1)(q-1)+1 = p+q$ e que $4n = (p+q)^2-(p-q)^2$. Assim, dado $\varphi(n)$ ou a soma dos primos, podemos encontrar p e q pelas equações:

$$\begin{cases} p+q &= n-\varphi(n)+1 \\ p-q &= \sqrt{(p+q)^2-4n} \end{cases}$$

Padrões

Observe que no exemplo abaixo a cifra mantém um padrão semelhante ao da substituição.

- A quer enviar uma mensagem para B
- A tem $pq = 5353$ e $b = 4591$
- $P_1 = 1214 \leftrightarrow$ "LN"
- $P_2 = 0303 \leftrightarrow$ "CC"
- $C_1 = P_1^b \mod pq = 3665$
- $C_2 = P_2^b \mod pq = 4545$
- $f : [1214, 303] \mapsto [3665, 4545]$
- A envia $[3665, 4545]$

Para nosso alfabeto em $\mathbb{Z}_{27}$ todas as repetições de duas letras levam a uma outra repetição.

Apesar das probabilidades serem baixíssimas, não há garantias que o RSA possa eventualmente cair em uma cifra por substituição.

Pode-se buscar um padrão, também, nos primos ou no produto. A escolha de primos da forma $2^k \pm r$ pode facilitar muito a fatoração, se r for pequeno.

Uma forma de evitar estes números é calcular a entropia. Por exemplo, com

$$p = 10000000000000000000000000000000000000000000000009$$

temos $H(p) \approx 0.28$.

A melhor entropia que podemos ter é $H(\text{“0123456789”}) \approx 3.32$. Se o primo tiver a freqüência de zeros dobrada temos $H \approx 3.28$.

Teste de Primalidade

Distinguir se um número é primo através de um algoritmo determinístico era um problema difícil até 2002, quando surgiu o algoritmo AKS com complexidade polinomial (AGRAWAL; KAYAL; SAXENA, 2004).

Atualmente, os métodos probabilísticos ainda são muito mais rápidos. Por exemplo, no teste de Miller existe apenas um número composto, 3215031751, menor que 2.5×10^{13}, cujos quatro primeiros primos não servem como testemunha. Além disto, Miller mostrou que todo número composto n tem uma testemunha menor que $70(\ln n)^2$, caso a hipótese de Riemann seja verdadeira (MILLER, 1975).

Probabilístico - Miller-Rabin

O Pequeno Teorema de Fermat afirma que se n é primo, então $t^{n-1} \equiv 1 \mod n$, caso $(n,t) = 1$.

Os números compostos que satisfazem o Pequeno Teorema de Fermat são chamados de pseudoprimos.

Por exemplo, $2^{340} \equiv 1 \mod 341$ é pseudoprimo na base 2. No entanto, 341 não é pseudoprimo na base 3, pois $3^{340} \equiv 56 \mod 341$.

Existem 245 pseudoprimos na base 2 menores que um milhão. Além disto, a maioria não é pseudoprimo em outra base.

Os números que são pseudoprimos em todas as bases são chamados de números de Carmichael. Felizmente, existem apenas 2163 números de Carmichael menores que 2.5×10^{10}.

O algoritmo 2 (RABIN, 1980) tem complexidade $O(\log^3 n)$ e responde primo todas as vezes que n for primo, além disto, acerta em mais de $\frac{3}{4}$ das vezes que n é composto. Aplicando o algoritmo recursivamente, temos uma precisão de acerto superior a $\left(\frac{3}{4}\right)^r$, onde r é o número de vezes que o algoritmo foi aplicado. Em outras palavras a taxa de erro é de $\left(\frac{1}{4}\right)^r$.

O algoritmo de Miller-Rabin pode ser considerado determinístico se a Hipótese Generalizada de Riemann for verdadeira e testarmos a para todo o intervalo $1 < a \leqslant 2(\ln n)^2$.

Algoritmo 2 Miller-Rabin

Recebe n
Escolha $0 < a_1 < n$ aleatóriamente
Escreva $n-1 = 2^t q$, com q impar
$k := 1$
Enquanto $k \neq t$ e $a_k \not\equiv 1 \mod n$ **faça**
    $k := k+1$
    $a_k := a_{k-1}^2 \mod n$
Se $k = t$ e $a_k \not\equiv 1 \mod n$ **então**
    retorne Composto
senão Se $k = 0$ **então**
    retorne Primo
senão Se $a_{k-1} \not\equiv -1 \mod n$ **então**
    retorne Composto
senão
    retorne Primo

Determinístico - AKS

Apresentamos aqui as idéias do AKS e enfatizamos que seu interesse é teórico.

Teorema 5 *Suponha a co-primo com n, isto é* $(a,n) = 1$. *Então n é primo se, e somente se,*

$$(x+a)^n \equiv x^n + a \mod n.$$

Este teorema nos fornece um critério para determinar se n é primo ou composto. Mas é impraticável porque a expansão binomial tem $n+1$ termos.

A idéia é acelerar o teste binomial através do polinômio $x^r - 1$ com r primo.

Certamente

$$(x+a)^p \equiv x^p + a \mod (x^r - 1, p).$$

No entanto, é possível que dois polinômios diferentes tenham o mesmo resto quando

$$(x+a)^n \not\equiv x^n + a \mod n,$$

isto é, continua verdadeiro quando n for primo mas pode ser falso quando for composto.

Vemos em (AGRAWAL; KAYAL; SAXENA, 2004) que basta substituir a por uma quantidade pequena de números até encontrarmos

$$(x+a)^n \not\equiv x^n + a \mod (x^r - 1, n).$$

Apresentamos o algoritmo 3.

Algoritmo 3 AKS

Recebe um inteiro $n > 1$
Se n é da forma a^b com $b > 1$ **então**
    retorne Composto
Procure o menor r tal que $o_r(n) > \log^2(n)$
Se $1 <$MDC$(n,a) < n$ para algum $a \leqslant n$ **então**
    retorne Composto
Se $n \leqslant r$ **então**
    retorne Primo
para $a := 1$ até $\lceil 2\sqrt{\varphi(r)}\log n \rceil$ **faça**
    Se $(x+a)^n \not\equiv (x^n + a) \mod (x^r - 1, n)$ **então**
        retorne Composto
retorne Primo

Simétrico	RSA	ECC
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	521

Tabela 3.2: *Número de bits recomendado por chave*

3.2.2 Curvas Elípticas

Curvas Elípticas têm sido usadas em muitas áreas da Matemática. Em criptografia, tal estudo é denominado ECC[3]. Entre as motivações de usar este criptossistema, temos a possibilidade de reduzir o tamanho da chave e, conseqüentemente, reduzir o tempo de processamento. Veja a tabela 3.2 que resume os trabalhos (GUPTA *et al.*, 2004) e (GUPTA *et al.*, 2002).

Nesta seção, temos uma introdução ao método de criptografia com Curvas Elípticas que foi apresentado simultaneamente por (KOBLITZ, 1987) e (MILLER, 1986) em 1985. Uma descrição mais completa pode ser encontrada em (HANKERSON; MENEZES; VANSTONE, 2004) e (WASHINGTON, 2003).

Definição. A **característica** de um corpo $\mathbb{F}$, com identidade multiplicativa 1, é de-

[3]Elliptic curve cryptography

finida como o menor n, tal que, $\underbrace{1+1+\cdots+1}_{n\times} = 0$ e se não existir n que satisfaça esta condição, dizemos que o $\mathbb{F}$ tem característica zero.

Seja $\mathbb{F}$ um corpo de característica diferente de 2 e 3, seja $c,d \in \mathbb{F}$ tal que x^3+cx+d seja livre de raiz, isto é,

$$\Delta = -16(4c^3+27d^2) \neq 0 \tag{3.3}$$

então, o conjunto dos pontos $(x,y) \in \mathbb{F}\times\mathbb{F}$ que são soluções de

$$y^2 = x^3+cx+d$$

junto com um elemento neutro chamado **ponto no infinito** $\overline{O}$ é uma **Curva Elíptica** E.

Com a operação definida abaixo, $(E,+)$ forma um grupo abeliano. Definimos:

- $P+\overline{O} = P \qquad \forall P \in E$
- Se $P=(x,y)$ então definimos $-P=(x,-y)$
- Se $P,Q \in E$ e $P \neq \pm Q$ e a reta $\overline{PQ}$ não é tangente a P ou Q então a reta vai interceptar um ponto R. Definimos $P+Q=-R$
- Se $P \neq \pm Q$ e $\overline{PQ}$ é tangente a P definimos $P+Q=-P$
- Se P não é ponto de inflexão, definimos $P+P=-R$
- Se P é ponto de inflexão $P+P=-P$

Veja os gráficos 3.2 e 3.3.

Tratamos agora de definir a operação no caso de E ser discreto.

Se $P=Q$ definimos:

$$\begin{aligned} x_3 &= \left(\tfrac{3x_1^2+c}{2y_1}\right)^2 - 2x_1 \mod p \\ y_3 &= \left(\tfrac{3x_1^2+c}{2y_1}\right)(x_1-x_3) - y_1 \mod p \end{aligned}$$

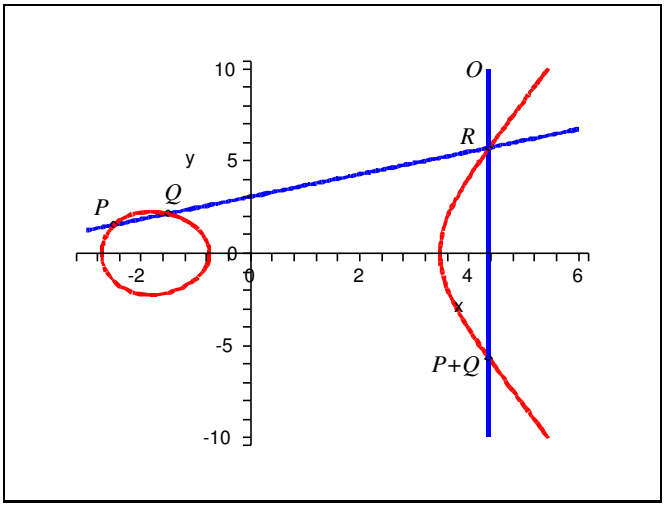


Figura 3.2: $P+Q=-R$.

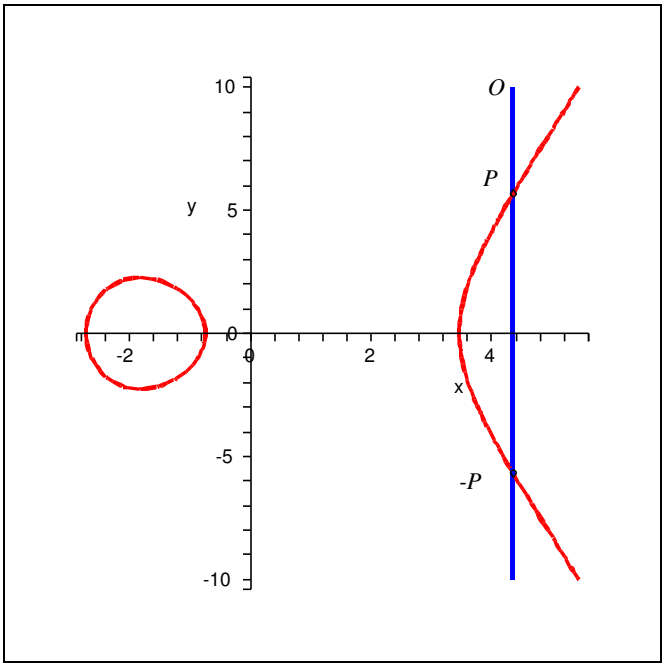


Figura 3.3: $-P$.

Se $P \neq \pm Q$ definimos:

$$x_3 = \left(\frac{y_2-y_1}{x_2-x_1}\right)^2 - x_1 - x_2 \mod p$$

$$y_3 = \left(\frac{y_2-y_1}{x_2-x_1}\right)(x_1-x_3) - y_1 \mod p$$

Como exemplo, vamos considerar uma Curva Elíptica em $\mathbb{Z}_{23}$. Se $c = 1$ e $d = 0$, temos $y^2 = x^3 + x$. Primeiramente, verificamos se a expressão (3.3) é satisfeita,

$$\Delta = -16(4) \mod 23 \equiv 18 \neq 0,$$

depois escolhemos um ponto, por exemplo (9,5), que satisfaz a equação:

$$\begin{aligned} y^2 &= x^3 + x \\ 5^2 &= 729 + 9 \\ 25 &= 738 \\ 2 &= 2 \end{aligned}$$

Existem 23 pontos que satisfazem esta equação.

Poderíamos pensar que $|E| = |\mathbb{F}|$, mas isto nem sempre ocorre, logo uma preocupação importante é garantir que o grupo cresça na ordem do corpo. Isto é garantido pelo Teorema de Hasse cuja demonstração pode ser encontrada em (WASHINGTON, 2003).

Teorema 6 (Hasse) *Se E é uma Curva Elíptica sobre $\mathbb{Z}_p$, então*

$$p + 1 - 2\sqrt{p} \leqslant |E| \leqslant p + 1 + 2\sqrt{p}.$$

De posse destas informações, já podemos apresentar o algoritmo 4. Vamos enviar uma mensagem $m : A \rightsquigarrow B$, o destinatário B começa escolhendo sua chave, envia para o remetente A que cifra a mensagem e envia o criptograma, B decifra.

Algoritmo 4 ECC

B escolhe um primo p grande, c e d
Se $-16(4c^3+27d^2) \equiv 0 \mod p$ **então**
    Volta ao passo anterior
B escolhe $a \in E$ com ordem grande e n
B calcula $b = na$ e envia p, c, d, a e b
A aplica $\alpha : m \rightarrow w \in E$ escolhe k, calcula $y = ka$ e $z = w + kb \in E$, envia y e z
Somente *B* pode ler calculando $z - ny = w + kb - nka = w + kb - kb = w$

3.2.3 Troca de chaves

O conceito de assimetria trouxe novos horizontes para os métodos de criptografia, possibilitando combinar uma chave em um canal de comunicação inseguro.

No modelo simétrico temos uma função

$$E_k(M) = C$$

que leva a mensagem no criptograma. E uma função

$$D_k(C) = M$$

que leva o criptograma na mensagem.

Ambas as funções dependem de k. A substituição de k por outro valor não revela informação sobre a mensagem.

O problema consiste em combinar k, isto deve ser feito por um canal de comunicação seguro.

No modelo assimétrico, temos uma função,

$$E_a(M) = C,$$

que cifra e outra,

$$D_b(C) = M,$$

que decifra.

Ambas as funções dependem de uma chave, porém a chave que cifra é diferente da chave que decifra. A troca de uma das duas chaves impossibilita a comunicação.

Como veremos nos algoritmos desta seção é possível combinar as chaves através de um canal inseguro. Representando uma grande vantagem dos algoritmos assimétricos em relação aos simétricos.

Uma outra vantagem é o número de chaves armazenadas. Observe que o número de chaves K cresce quadraticamente para a criptografia simétrica,

$$K = \frac{n(n-1)}{2},$$

em relação ao número de pessoas n que podem se comunicar. Enquanto que na criptografia assimétrica, K cresce linearmente,

$$K = 2n.$$

Além disto, é possível enviar uma mensagem cifrada que todos possam ler, garantindo que foi escrita por uma pessoa específica. Este é o conceito de **Assinatura Digital** (SCHNEIER, 1996).

Se uma mensagem é cifrada com uma chave pública, então somente o proprietário da chave privada pode ler a mensagem. No entanto, se a mensagem for cifrada com a chave privada, então todos podem ler a mensagem com a chave pública.

A única coisa que não é garantida nos algoritmos é a identidade do remetente e destinatário, isto é, não se garante com quem se está comunicando. Para solucionar este problema, é usada certificação digital (SCHNEIER, 1996). No entanto, esbarramos em outro problema, temos que confiar na certificadora. A solução deste impasse está surgindo com curvas elípticas através de uma técnica chamada emparelhamento. Que possibilita que dados pessoais sejam usados como chave pública. Desta forma um e-mail poderia ser a chave pública de um usuário, sem a necessidade de uma certificadora digital.

O algoritmo 5 de Diffie-Hellman (DIFFIE; HELLMAN, 1976) é uma forma de combinar uma chave do RSA em um canal inseguro. Já o algoritmo 6 de ElGamal (ELGAMAL, 1985) pode ser usado com o RSA e ECC. O algoritmo 7 de Menezes-Vanstone somente pode ser usado com Curvas Elípticas.

Algoritmo 5 Diffie-Hellman

A escolhe dois primos p e q, faz $R = \mathbb{Z}_{pq}$
A escolhe $0 < k \in R$ tal que $(k, pq) = 1$ e envia k e R para *B*
A escolhe $0 < r \in R$, calcula k^r e envia o resultado para *B* mantendo r em segredo
B escolhe $0 < s \in R$, calcula k^s e envia o resultado para *A* mantendo s em segredo
Ambos calculam $a = (k^r)^s = (k^s)^r$
Se $(a, \varphi(pq)) \neq 1$ **então**
    A inicia novamente o processo

Por exemplo, *A* escolhe $p = 83,\ q = 101$ e $k = 256$ calcula $(8383, 256) = 1$ e envia k e R para *B*.

Suponha que *A* escolhe $r = 91$, calcula $k^r = 2908$ e envia o resultado para *B* mantendo r em segredo. No mesmo instante, *B* escolhe $s = 4882$, calcula $k^s = 1754$ e envia o resultado para *A* mantendo s em segredo.

Note que, ambos os pontos *A* e *B* tem $b_A = 2908^s = 1754^r = 6584$, mas somente *A* verifica que b_A não é um expoente válido $(6584, 8200) = 8$ e inicia novamente o processo.

Suponha que *A* mantém $p = 83,\ q = 101$ e $k = 256$, então *A* escolhe $r = 17$, calcula $k^r = 5835$ e envia o resultado para *B* mantendo r em segredo. Do outro lado, *B* escolhe $s = 109$, calcula $k^s = 1438$ e envia o resultado para *A* mantendo s em segredo.

Ambos têm $a = 5835^s = 1438^r = 3439$, e *A* verifica se a chave a é um expoente válido $(3439, 8200) = 1$.

Note que, se $|a| = o$ ou $|G| = o$, então podemos calcular a inversa de forma muito mais fácil, fazendo $y^{-n} = y^{o-n}$.

Algoritmo 6 ElGamal

B escolhe $(G,\oplus)$, $a \in G$ e $n \in \mathbb{N}^*$
B calcula $b = a^n$ e envia a, b e G, escondendo n
A aplica $\alpha : m \to w \in G$, escolhe $k \in \mathbb{N}^*$, calcula $y = a^k$ e $z = wb^k \in G$, depois envia y e z
B calcula $zy^{-n} = wb^k(a^k)^{-n} = w(ba^{-n})^k = w(1)^k = w$

Por exemplo, o destinatário *B* escolhe $G = \mathbb{Z}_p$, com $p = 1000000007$, $a = 419666093$ e $n = 110691024$. Então, calcula $b = a^n \mod p = 215094385$ e envia G, a e b.

Com a posse de G, a e b, *A* aplica $\alpha : m \to w = 12140303$, escolhe $k = 633071297$. Então, calcula $y = a^k \mod p = 295903670$ e $z = wb^k \mod p = 763646857$. Assim, a mensagem pode ser decifrada em *B* calculando $zy^{-n} \mod p = 12140303$ ou $z(y^{(p-1)-n}) \mod p = 12140303$.

Algoritmo 7 Menezes-Vanstone

B escolhe um primo p grande, c e d
Se $-16(4c^3 + 27d^2) \equiv 0 \mod p$ **então**
    Volta ao passo anterior
B escolhe $a \in E$ com ordem grande e $n \in \mathbb{N}^*$, calcula $b = na$ e envia p, c, d, e $a, b \in E$
A $\alpha : m \to w \in \mathbb{Z}_p^* \times \mathbb{Z}_p^*$
A escolhe $k \in \mathbb{N}^*$, calcula $y = ka$, $kb = (c_1, c_2) \in E$ e $z = (z_1, z_2) = (c_1 w_1 \mod p, c_2 w_2 \mod p)$, envia y e z
B calcula $ny = nka = kna = kb$ depois $\left(c_1^{-1} z_1 \mod p, c_2^{-1} z_2 \mod p\right) = \left(c_1^{-1} c_1 w_1 \mod p, c_2^{-1} c_2 w_2 \mod p\right) = w$

Se compararmos o tamanho dos blocos para enviar a mensagem, observamos que a grande vantagem do método de Menezes-Vanstone é poder enviar blocos maiores, pois com uma curva elíptica E sobre $\mathbb{Z}_{19}$ podemos transportar $|E| = 18$ símbolos com o ElGamal e $|\mathbb{Z}_{19}^*|^2 = 324$ com Menezes-Vanstone.

Um intruso de posse de k, pq, k^r e k^s poderia calcular s ou r e depois b_A. Porém, o grau de dificuldade é semelhante ao da fatoração, ou seja, teria que usar algoritmos com complexidade exponencial para encontrar s.

Por exemplo, com $k = 256$, $pq = 8383$, $k^r = 5835$ e $k^s = 1438$, pode-se tentar encontrar $s = 109$ e calcular $b_A = (k^r)^s = 5835^{109} = 3439$. Este problema é conhecido como **Problema do Logaritmo Discreto** e não existe um algoritmo com tempo polinomial para a sua solução.

3.3 Segredos Perfeitos

Classificamos os criptossistemas como *Segredos Perfeitos* quando for impossível descobrir a mensagem sem a chave criptográfica.

3.3.1 One-time-pad

Vigenère-Vernam é conhecido como o único criptossistema inquebrável. Normalmente este criptossistema simétrico não é usada pela dificuldade inerente ao algoritmo. Existe uma variação chamada *Latin Squares* em (BRUEN; FORCINITO, 2004), porém os resultados são os mesmos.

A dificuldade mencionada consiste na necessidade da chave ser do mesmo comprimento ou maior que a mensagem.

Algoritmo 8 Vigenère-Vernam

```
Recebe uma mensagem m
Recebe uma chave k aleatória
Se |k| < |m| então
    retorne Senha muito curta
para i := 1 até |k| faça
    c[i] = k[i] + m[i] mod |𝒜|
retorne c
```

Veja o algoritmo 8, para decifrar podemos usar o mesmo algoritmo usando $-k$ na chave.

Usando o alfabeto em $\mathbb{Z}_{27}$, vamos tentar atacar o método. Suponha que interceptamos

o criptograma "∅TOYNIMCEYVS∅E∅". Então, podemos escrevê-lo como 00, 20, 15, 25, 14, 09, 13, 03, 05, 25, 22, 19, 00, 05, 00. Se tentarmos a chave 01, 07, 25, 07, 00, 00, 01, 25, 22, 04, 23, 17, 14, 25, 01, obtemos a mensagem "A∅MENINA∅BRINCA", isto é 01, 00, 13, 05, 14, 09, 14, 01, 00, 02, 18, 09, 14, 03, 01. Por outro lado, se tentarmos a chave 01, 00, 13, 05, 14, 09, 14, 01, 00, 02, 18, 09, 14, 03, 01, obtemos 01, 20, 01, 03, 01, 18, 00, 04, 05, 00, 13, 01, 14, 08, 01 ou "ATACAR∅DE∅MANHA". Portanto, não sabemos qual mensagem pode ser a verdadeira, uma vez que ambas podem satisfazer o algoritmo e temos um problema de indeterminação.

Com este método não se obtém informações sobre a mensagem, a única informação obtida é que um sinal foi enviado, como em qualquer outro método de criptografia.

Para definirmos *Segredo Perfeito*, precisamos de alguns conceitos. Seja $\mathscr{M} = \{M_1, \ldots, M_n\}$ o conjunto de todas as mensagens possíveis, e $P(M_1), \ldots, P(M_n)$ suas probabilidades de ocorrência, além disto, seja $\mathscr{C} = \{C_1, \ldots, C_n\}$ o conjunto dos criptogramas. Logo

$$C = T_i(M),$$

onde T_i é a transformação que relaciona a mensagem em $\mathscr{M}$ com o criptograma em $\mathscr{C}$.

Definição. Um criptossistema garante um **Segredo Perfeito** quando satisfaz a condição

$$P_C(M) = P(M), \tag{3.4}$$

para todo $M \in \mathscr{M}$ e todo $C \in \mathscr{C}$

A equação 3.4 significa que a probabilidade da ocorrência da mensagem é a mesma conhecendo ou não o criptograma.

Teorema 7 *Vigenère-Vernam é um Segredo Perfeito.*

Prova. Considere um alfabeto com n símbolos, assim a probabilidade de ocorrência de uma letra na posição i no criptograma é

$$P(C_i) = \frac{1}{n},$$

pois a chave k_i é gerada aleatoriamente. Segundo o algoritmo 8, cada letra da mensagem

está unicamente relacionada com uma letra da chave

$$c_i = k_i + m_i \mod \mathscr{A}.$$

Assim, o conhecimento do criptograma não dá informação alguma sobre a mensagem. Portanto,

$$P(M) = P_C(M).$$

■

Vale observar que este criptossistema nos dá uma segurança matemática, garantindo que um criptoanalista não vai obter informações sobre o criptograma, diferente da criptografia assimétrica que nos garante uma segurança computacional.

Uma das tentativas de usar a cifra de Vigenère-Vernam é gerar uma seqüência através de um número pseudo-aleatório. O problema consiste no tamanho da semente e no prefixo pseudo. Desta forma, não garantimos a segurança perfeita, pois não estamos satisfazendo a condição de aleatoriedade.

3.3.2 Números Irracionais

A cifra de Vigenère-Vernam é conhecida como **One-time-pad** e por sua vez como o único sistema matematicamente seguro (BRUEN; FORCINITO, 2004). No entanto, tem o inconveniente que o tamanho da chave deve ser maior ou igual o da mensagem. Estaremos chamando de One-time-pad todos os algoritmos que tenham esta inconveniência e garantam um *Segredo Perfeito*.

Levanta-se a questão, existe algum outro algoritmo que seja um *Segredo Perfeito* sem ser One-time-pad? Se existe, isto é, se a chave for menor, poderíamos combinar uma nova chave a cada troca de mensagem, possibilitando trocar um número ilimitado de mensagens em um algoritmo perfeitamente seguro, sem a necessidade de combinar uma nova chave por outro canal seguro.

Uma outra vantagem de encontrá-lo é entender melhor a segurança dos algoritmos.

Teorema 8 *Dado uma mensagem M fixa e uma chave K, se $M_i, K_j \in |\mathscr{A}|\ \forall M_i, K_j$ e $E = T_k M$ então One-time-pad é o único Segredo Perfeito.*

Prova. Temos que T_k é uma transformação biunívoca. Suponha por contradição que $|K| < |M|$, então existe pelo menos um criptograma E que não é gerado por T_k. Portanto, as mensagens não são equiprováveis. ■

Uma solução é negar uma hipótese do teorema anterior fazendo com que o alfabeto da mensagem seja menor que o alfabeto da chave, isto é,

$$|\mathscr{A}_M| < |\mathscr{A}_K|.$$

Isto seria impraticável para os computadores, pois usam um alfabeto binário. Porém, esta idéia nos mostra que precisamos passar mais informações na chave.

Para atribuir mais informação a uma seqüência de letras, precisamos de um novo paradigma. Uma forma seria atribuir uma semântica à chave, semelhante ao que é feito na codificação por carreira 2.2.2. A linguagem mais natural seria das expressões matemáticas. Ao invés de passarmos uma seqüência de letras, passamos uma seqüência que será interpretada.

Na tentativa de criar criptossistemas seguros, surgiram os algoritmos classificados de **cipher stream** como o RC4. Tais algoritmos tentam gerar, a partir da chave, uma seqüência pseudo-aleatória do tamanho da mensagem. Como a chave é menor que a mensagem e ambas tem o mesmo alfabeto, temos que a igualdade (3.4) nunca é satisfeita.

Um número irracional tem uma seqüência infinita de dígitos que não tem período, além disto, eles formam um conjunto não enumerável. É evidente que quanto maior a seqüência de dígitos, maior o processamento necessário para calculá-la. No entanto, achando um gerador de dígitos de um número irracional, poderíamos ter uma chave menor que a mensagem. Nosso objetivo é transferir o custo do tamanho da chave para um custo computacional.

Teorema 9 *Dado um produto de primos distintos $p_1 \cdots p_n$ e $r > 1$ temos que $\sqrt[r]{p_1 \cdots p_n}$ é um número irracional.*

Prova. Suponha, por contradição, que esta raiz é um número racional na sua forma irredutível, isto é,

$$\sqrt[r]{p_1 \cdots p_n} = \frac{a}{b},$$

logo

$$p_1 \cdots p_n = \frac{a^r}{b^r} \Rightarrow p_1 \cdots p_n b^r = a^r,$$

assim $p_1 | a^r$ desta forma temos que $a = a' p_1$, logo

$$p_1 \cdots p_n b^r = (a' p_1)^r.$$

Portanto

$$p_2 \cdots p_n b^r = a'^r p_1^{r-1}.$$

A contradição consiste em p_1 não dividir fator algum a esquerda da igualdade, uma vez que $(a,b) = 1$. ■

Agora temos um gerador de infinitos números irracionais. Além disto, para $r = 2$, temos que estes números são normais na base 2 (ISAAC, 2005), o que garante que a seqüência é "verdadeiramente aleatória"(BAILEY; CRANDALL, 2002). Isto significa que um bit tem probabilidade $\frac{1}{2}$ de ocorrer na seqüência.

A definição de aleatoriedade tem causado certa divergência. Veja (VOLCHAN, 2002), (KENDALL, 1973), (BAILEY; CRANDALL, 2002) e (RUKHIN, 2000). Tal divergência não altera a segurança dos algoritmos 9 e 10, pois estamos nos baseando que a escolha feita por uma pessoa é aleatória em qualquer definição.

Considere o algoritmo 9 que recebe uma mensagem e uma chave formada por três números e n expressões matemáticas. Através da função *próximo_primo*(e_n) temos o próximo número primo maior que e_n. Se nenhum destes números primos forem repetidos, podemos formar um número irracional e usar as casas décimas da mantissa deste para cifrar a mensagem.

Algoritmo 9 Números Irracionais

Recebe uma mensagem m
Recebe uma chave $r, a, b, e_1, \ldots, e_n$
$p_1 = próximo\_primo(e_1)$
$\vdots$
$p_n = próximo\_primo(e_n)$
Se algum primo está repetido **então**
    Escolha outro primo até não ter repetição
$I = \frac{a}{b}\sqrt[r]{p_1 \cdots p_n}$
k recebe $|m|$ casas decimais da mantissa de I
para $i := 1$ até $|m|$ **faça**
    $c[i] = k[i] \oplus m[i]$
retorne c

O algoritmo 9 se resume no One-time-pad, pois com os números racionais podemos formar qualquer seqüência que queiramos, uma vez que podemos aproximar qualquer número irracional através de um número racional. Neste caso, temos o inconveniente de que a chave pode ficar maior que a mensagem.

Fica a pergunta, podemos aproximar qualquer número através da extração de uma raiz de produtos de primos?

Vamos aproximar o 5 através da raiz cúbica. Primeiramente, fazemos $5^3 = 125$, depois procuramos o primo mais próximo, 127, logo

$$\sqrt[3]{127} \approx 5.02. \tag{3.5}$$

Teorema 10 *Se*

$$\sqrt[r]{p_{n+1}} - \sqrt[r]{p_n} < 1$$

então todo número pode ser aproximado através da raiz de um produto de primos.

Prova. Seja I o número que desejamos aproximar, então

$$I^r = p_1 \ldots p_k (f_1 \ldots f_s),$$

onde f_i são fatores primos com potências maiores que um. Se fizermos

$$f_1 \ldots f_s = p_m + d,$$

com o menor d temos

$$p_m < f_1 \dots f_s < p_{m+1}.$$

Assim

$$\sqrt[r]{f_1 \dots f_s} - \sqrt[r]{p_m} < 1.$$

Quanto maior o r, mais a diferença tende a zero. Portanto, todo número pode ser aproximado por um produto de primos. ■

Para se provar que o algoritmo 9 é um *Segredo Perfeito* é necessário provar a hipótese do teorema 10. No entanto, tal hipótese não deve ser simples de se provar, pois é uma generalização da conjectura (3.6) de Andrica (SMARANDACHE, 1999),

$$\sqrt{p_{n+1}} - \sqrt{p_n} < 1. \tag{3.6}$$

Outra questão que ficou pendente, foi como passar números primos grandes sem aumentar muito o tamanho da chave. As entradas do algoritmo 9 podem receber expressões matemáticas e localizar o menor primo maior que o resultado da expressão. Por exemplo, podemos passar *próximo_primo*(5^{604}), que tem 423 dígitos decimais, isto é, 1403 bits versus 6 bits na expressão.

O algoritmo 9 pode ser bem simplificado. Observe que o produto de um número racional por um irracional resulta um irracional. Desta forma, podemos escrever o algoritmo 10.

Algoritmo 10 Números Irracionais

Recebe uma mensagem m
Recebe uma chave e, r
Se $\sqrt[r]{e} \in \mathbb{N}$ **então**
    Escolha outra expressão
$I = \sqrt[r]{e}$
k recebe $|m|$ casas decimais da mantissa de I
para $i := 1$ até $|m|$ **faça**
    $c[i] = k[i] \oplus m[i]$
retorne c

Observe que toda a aleatoriedade está na escolha das expressões e e r.

Neste caso, o espaço de busca das chaves é maior que a mensagem, além de termos

todas as combinações de criptograma. Assim, se a conjectura (3.6) for satisfeita e tivermos $r = 2$, então teremos um *Segredo Perfeito* (3.4) com o algoritmo 9. Como o algoritmo 10 não depende da conjectura de Andrica, temos que a condição (3.4) é satisfeita, isto é, temos um *Segredo Perfeito* teórico, pois como mostramos na aproximação de π, algumas chaves podem ser computacionalmente custosas.

3.3.3 Fraquezas e comparação

A fraqueza dos algoritmos tipo One-time-pad está no método de escolha da chave. Na cifra de Vigenère-Vernam, algoritmo 8, a fraqueza consiste em usar uma seqüência que não é aleatória como, por exemplo, gerada através de um PRNG[4]. Na cifra de números irracionais, proposto nesta dissertação, a fraqueza consiste em usar sempre expressões baseadas em potências, ao invés de usar funções aritméticas, funções de variáveis complexas ou equações diferenciais. Mais criativo ainda seria criar funções novas, como pegar um intervalo da mantissa de uma constante. Só não seria bom escolher resultados conhecidos, pois possibilitam um ataque por dicionário como, por exemplo, as 547 primeiras casas decimais de π que formam um número primo.

Observamos que os algoritmos assimétricos são fortemente ameaçados pela computação quântica, pois a segurança dos algoritmos assimétricos está no poder computacional (BRUEN; FORCINITO, 2004), uma vez que se baseiam em funções unidirecionais. Os computadores quânticos têm um ganho exponencial justamente nas funções unidirecionais mais usadas em criptografia. Uma boa referência sobre computação quântica pode ser encontrada em (NIELSEN; CHUANG, 2000).

Nos algoritmos simétricos, a segurança já é maior, pois se baseiam em probabilidades. São facilmente quebrados quando a entropia não é alterada, em geral, a dificuldade vai aumentando conforme a entropia muda. No entanto, a entropia e difusão, não são as melhores métricas para a segurança. Em uma mensagem com muita redundância podemos ter difusão máxima, chegando a entropia máxima do criptograma, sem termos a segurança

[4] pseudorandom number generator

Algoritmos	Segurança
Assimétricos	computacional
Simétricos	probabilística
Segredo Perfeito	matemática

Tabela 3.3: *Grau de Segurança*

máxima. Por exemplo, suponha à transformação que leva à repetição de uma letra no alfabeto, isto é,

$$T_k : \underbrace{A \cdots A}_{27\times} \to \mathscr{A}.$$

Tal transformação é simples de ser encontrada e poderia ser dada por

$$\begin{aligned} T_k : \quad & \mathscr{M} \to \mathscr{C} \\ & a \mapsto m_i + m_{i-1} + k \end{aligned} . \tag{3.7}$$

Note que, também, não é muito difícil de quebrar (3.7). Assim, encontramos um criptossistema com baixa segurança enquanto a entropia e a difusão têm valores máximos.

Resumimos a comparação entre o tipo de algoritmo de criptografia e as bases da segurança na tabela 3.3.

A segurança máxima se obtém através de um *Segredo Perfeito* (3.4). Assim, a segurança dos criptossistemas simétricos e assimétricos deve ser medida pela probabilidade de um bit se aproximar de $\frac{1}{2}$ no criptograma.

4 Escondendo a mensagem

4.1 Paradigma

Aqui vale diferenciar esteganografia de marca d´água, apesar de muitas vezes os métodos serem tratados juntos. No primeiro, o foco está em esconder uma mensagem, não se preocupando se o método é robusto e busca a maior quantidade possível de espaço para a mensagem. No segundo, a marca d´água não tem que estar necessariamente escondida, o método deve ser robusto e não majora o espaço.

A esteganografia consegue fornecer uma segurança a mais que a criptografia e com ela a mensagem não é interrompida, se não for detectada.

A esteganoanálise passa a ser um problema de decidir entre a existência ou não de esteganografia, isto é, determinar se em um dado meio existe ou não uma mensagem esteganografada.

Para se transmitir uma mensagem esteganografada, é necessário um meio[1]. Este pode ser uma imagem, um arquivo de áudio ou um arquivo qualquer. Mesmo sem ter um arquivo, pode-se usar a esteganografia, por exemplo, em um protocolo de rede ou meio não eletrônico. Em geral, é necessário muito conhecimento do meio para poder explorar suas redundâncias, onde está embutida a verdadeira mensagem.

Uma boa esteganografia não está baseada na quantidade de meios existentes, mas sim,

[1] Em inglês é usado o termo cover.

na dificuldade de encontrá-la em um meio específico (PROVOS; HONEYMAN, 2003). Para garantir a segurança de um sistema, devemos ter em mente a máxima de Shannon: "o inimigo conhece o sistema".

Uma forma simples de esteganografia em texto, consiste em usar acrônimos como citamos na introdução. É interessante notar que, quanto maior o meio mais fácil de encontrar seqüências que formam mensagens coerentes.

Diferente do *Segredo Perfeito* em criptografia (3.4), na esteganografia, não basta que o método possibilite encontrar todas as mensagens em um meio e as mesmas sejam equiprováveis.

Definição. Um método de esteganografia garante um **Segredo Perfeito** quando satisfaz a condição

$$P_M(W) = P(W), \tag{4.1}$$

onde M é a mensagem, W o meio e P a probabilidade de existência de esteganografia.

Evidente que para enviarmos a mensagem encoberta pela esteganografia, vamos escolher um meio onde $P(W)$ seja baixa. Assim, como na criptografia, não queremos que um *Segredo Perfeito* recaia em uma cifra por substituição.

Note que, no caso da esteganografia, a definição é mais rigorosa, pois normalmente a mensagem é criptografada ou compactada. Assim, se espera que tenha uma entropia alta, possibilitando ataques estatísticos na esteganografia.

Com a condição dada por (4.1), fica difícil saber onde está a mensagem em um meio, mesmo conhecendo o meio e a mensagem. Dado um meio de dois bits e uma mensagem, podemos determinar onde está a mensagem. No entanto, dado um meio com mais que dois bits, não podemos determinar onde está a mensagem.

Um método de esteganografia que garante a condição 4.1 é obtido combinando pontos aleatórios de um meio, onde será embutida a mensagem e ao transmiti-la, geramos um

meio cuja mensagem esteja nos pontos previamente combinados.

A mensagem embutida pela esteganografia pode usar criptografia assimétrica. Neste caso, não é interessante usar uma mídia estática, como uma imagem, mas um fluxo de dados, como em um diálogo. Além disto, nós necessitamos de uma grande quantidade de dados para embutir a mensagem.

É muito interessante observar que a esteganografia explora a redundância de informação em um meio, sendo necessário assim trabalhar com as propriedades do meio para encaixar uma mensagem em tal meio.

Basicamente, nós temos as opções de usar o som ou o vídeo como meio em uma videoconferência. O protocolo ITU-T H.264 usa DCT e Wavelet e tem matrizes com dimensões diferentes, enquanto o ITU-T H.263 usa somente DCT 8×8 (2.8) como o formato jpeg.

No padrão H.263 o fluxo de vídeo contém quadros I, P e B, o primeiro quadro se assemelha ao jpeg, pois não tem estimação e compensação de movimentos. Veja (HALSALL, 2001), (RICHARDSON, 2004) e (ITU-T, 1998a) para maiores detalhes.

Todos estes protocolos são feitos para manter uma interoperabilidade, o que não impede que os dados sejam transmitidos ou armazenados em outro formato com os mesmos algoritmos. Neste trabalho, optamos em usar uma seqüência de jpeg como vídeo.

Encontramos outro trabalho de esteganografia em videoconferência em (WESTFELD; WOLF, 1998).

4.2 Ocultando no Domínio Espacial

Em um mapa de bits, se nós alterarmos o *Least Significant Bit* (**LSB**) de cada pixel, ficamos vulneráveis a ataques visuais. Certamente, se a imagem tem muitos bits por pixel a percepção de granulação será menor ou nula. É interessante imprimir apenas o LSB da imagem para facilitar o ataque visual. Caso a imagem seja colorida, será necessário separar as cores para analisar o LSB. Considere que a imagem da figura 2.6 tem 24 bits de cor no formato RGB, podemos escolher uma das três camadas para formar a figura 4.1.

Figura 4.1: *Mapa de bits com 256 tons referente a figura 2.6.*

Tal processo de separar a imagem, gera uma imagem com 8 bits como a figura 4.2 gerada em tons de cinza.

Ambas as imagens contém 8 bits de cor e ao imprimir cada um dos bits separadamente, obtemos as figuras 4.3 e 4.4. A impressão foi feita começando do LSB em diante.

A inserção de esteganografia em mapas de bits gera interferência na imagem, logo a figura 4.3 indica que a figura 4.1 contém esteganografia e a figura 4.4 indica que a figura 4.2 não tem.

Tanto este ataque visual quanto os estatísticos, não são determinísticos, mas indicam que pode haver presença de algum método de esteganografia. No entanto, estes métodos

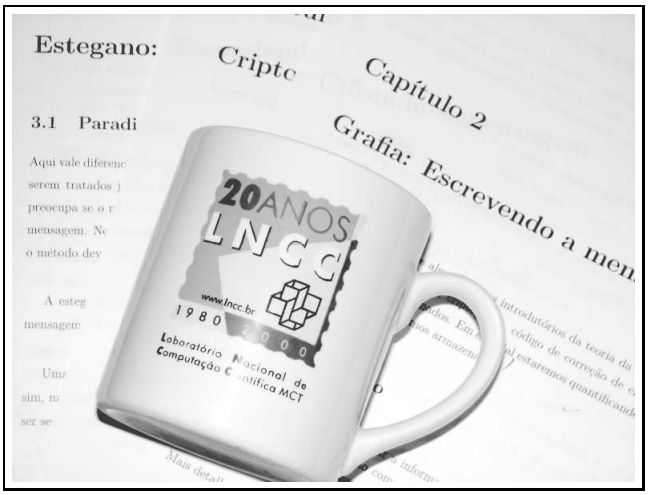


Figura 4.2: *Mapa de bits com 256 tons de cinza.*

podem gerar um falso positivo ou um falso negativo. Por exemplo, utilizando o software Stegdetect[2] em seqüências de imagens formadas de um vídeo de 400 quadros (Foreman[3]), temos 48 falsos positivos e 2 com suspeita de serem falsos positivos.

A figura 4.2 poderia conter uma quantidade pequena de esteganografia, de modo que não percebêssemos. Já a figura 4.1 não contém esteganografia, logo temos um falso positivo.

Um ataque visual tem um número muito maior chances de sucesso se conhecermos o padrão da imagem e em um determinado momento este padrão se altera. Veja um exemplo na figura 4.5.

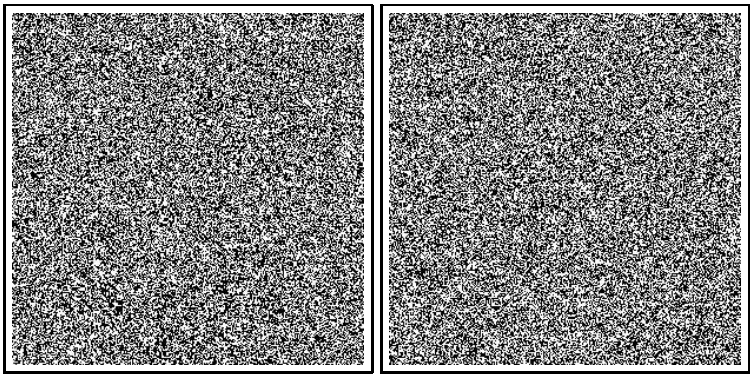
4.3 Ocultando no Domínio de Freqüência

Em uma imagem JPEG, que é similar ao I-frame, se alterarmos os LSB no domínio de freqüência, isto é alterarmos o LSB da matriz resultante da DCT, então um ataque visual se torna ineficaz em uma imagem JPEG(PROVOS; HONEYMAN, 2003). Depois de termos aplicado a DCT (2.8) em cada matriz de pixel 8×8, obtemos um coeficiente DC

[2]http://www.outguess.org/detection.php
[3]http://www.lncc.br/ borges/videos/

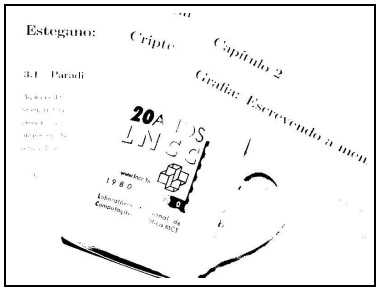
(a) Posição do Bit: 1 (b) Posição do Bit: 2 (c) Posição do Bit: 3

(d) Posição do Bit: 4 (e) Posição do Bit: 5 (f) Posição do Bit: 6

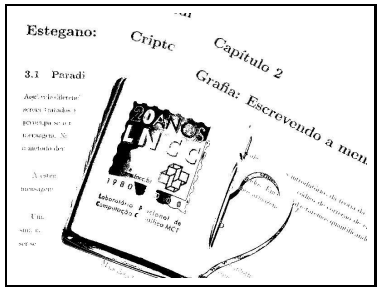


(g) Posição do Bit: 7 (h) Posição do Bit: 8

Figura 4.3: *Impressão das oito camadas de bits da figura 4.1.*

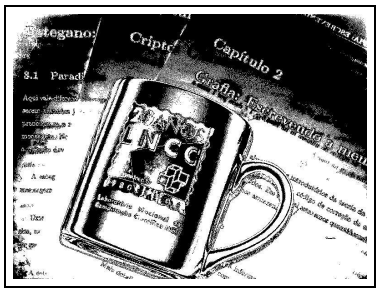


(a) Posição do Bit: 1



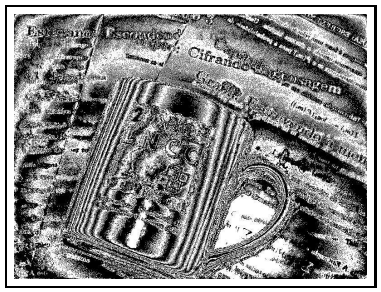
(b) Posição do Bit: 2

(c) Posição do Bit: 3

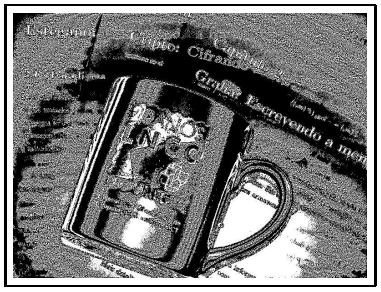


(d) Posição do Bit: 4

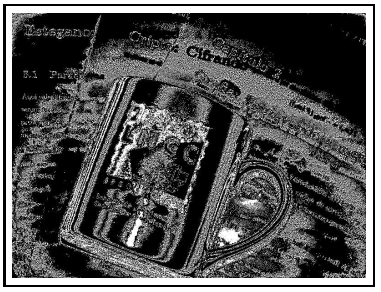
(e) Posição do Bit: 5



(f) Posição do Bit: 6



(g) Posição do Bit: 7



(h) Posição do Bit: 8

Figura 4.4: *Impressão das oito camadas de bits da figura 4.2.*

Figura 4.5: *Mudança no padrão da imagem*

que provê a cor média do bloco e os outros coeficientes conhecidos como AC.

O último estágio da codificação JPEG é a compressão sem perdas, conhecida como entropy encoding, onde se usa Huffman ou compressão aritmética.

O momento interessante para embutir informação é entre a quantização e a entropy encoding. Veja figura 4.6. Até a quantização temos perda de informação e após não há mais perdas, por isto, inserimos a esteganografia entre estas etapas.

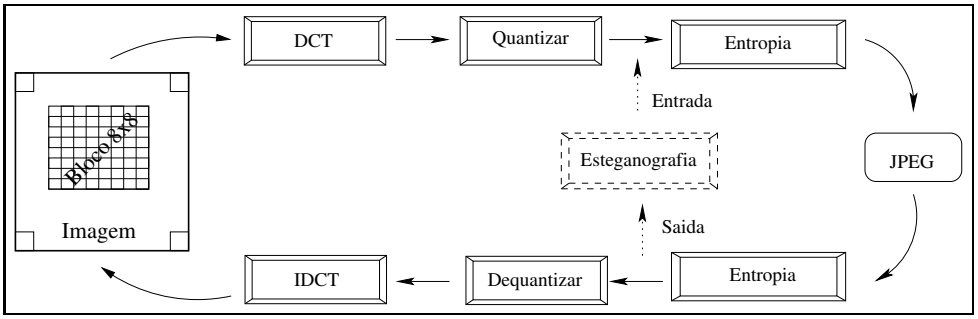


Figura 4.6: *Esquema de esteganografia em JPEG.*

Se os coeficientes AC, diferentes de zero e um, tiverem o LSB alterado de forma seqüencial, um ataque estatístico pode estimar, com boa precisão, o tamanho da mensagem. Veja (PROVOS; HONEYMAN, 2003), (WESTFELD; PFITZMANN, 2000) e (TRIVEDI; CHANDRAMOULI, 2005). Intuitivamente podemos perceber isto conside-

rando que a entropia da mensagem esteganografada é alta e que a entropia do meio não é tão alta, assim fica fácil detectarmos uma região onde a entropia é muito mais alta.

O ataque estatístico é eficiente porque o par de bits que difere somente do LSB tende a ter a mesma freqüência, uma vez que a mensagem transmitida é cifrada ou comprimida.

Entretanto, se a matriz F' é escolhida aleatoriamente, a dificuldade de determinar a presença de esteganografia na mensagem cresce consideravelmente.

Como o processo de detecção da esteganografia é estatístico, temos resultados prováveis. Não é comum que uma imagem tenha entropia alta, mas pode acontecer de ter a mesma probabilidade de zeros e uns. Assim um teste detectaria presença de esteganografia, não entanto é apenas uma característica da imagem causando um resultado falso-positivo. Na tabela 4.1 mostramos uma tentativa de descoberta de esteganografia em quatorze seqüências de imagens extraídas de quatorze vídeos.

Vídeo	**Positivo**	**Negativo**	**Suspeitos**
akiyo	207	91	2
bridge-close	0	2001	0
bridge-far	1118	983	0
carphone	44	317	21
claire	21	473	0
coastguard	55	243	2
container	105	180	15
foreman	48	350	2
highway	375	1625	0
mobile	1	297	2
mother	57	243	0
news	231	68	1
salesman	2	447	0
silent	0	300	0

Tabela 4.1: *Detectando esteganografia nos quadros dos vídeos.*

Seja D a dificuldade de se detectar esteganografia e sejam L_{me} e L_{mi} o tamanho da mensagem e do meio respectivamente. Em geral, a dificuldade de detectar D é inversamente proporcional ao tamanho da mensagem L_{me} e diretamente proporcional ao tamanho

do meio L_{mi} e a forma como a mensagem foi espalhada S no meio. Em suma

$$D = \frac{SL_{mi}}{L_{me}}. \tag{4.2}$$

O nosso objetivo é majorar D, assim nós devemos espalhar a mensagem no meio.

Não existe controle sobre o tamanho da mensagem, mas a equação (4.2) justifica a escolha de um meio grande, onde o único limitante é o tempo de transmiti-la.

Além de escolher as matrizes F' aleatoriamente para serem alteradas, nós também podemos alterar um número pequeno de coeficientes da mesma forma.

4.3.1 Análise da matriz

Considere agora uma matriz de pixel P e a matriz de quantização Q,

$$P = \begin{bmatrix} 0 & 0 & 0 & 200 & 200 & 0 & 0 & 0 \\ 0 & 0 & 200 & 200 & 200 & 200 & 0 & 0 \\ 0 & 200 & 200 & 200 & 200 & 200 & 200 & 0 \\ 200 & 200 & 200 & 200 & 200 & 200 & 200 & 200 \\ 200 & 200 & 200 & 200 & 200 & 200 & 200 & 200 \\ 0 & 200 & 200 & 200 & 200 & 200 & 200 & 0 \\ 0 & 0 & 200 & 200 & 200 & 200 & 0 & 0 \\ 0 & 0 & 0 & 200 & 200 & 0 & 0 & 0 \end{bmatrix},$$

$$Q = \begin{bmatrix} 6 & 11 & 16 & 21 & 26 & 31 & 36 & 41 \\ 11 & 16 & 21 & 26 & 31 & 36 & 41 & 46 \\ 16 & 21 & 26 & 31 & 36 & 41 & 46 & 51 \\ 21 & 26 & 31 & 36 & 41 & 46 & 51 & 56 \\ 26 & 31 & 36 & 41 & 46 & 51 & 56 & 61 \\ 31 & 36 & 41 & 46 & 51 & 56 & 61 & 66 \\ 36 & 41 & 46 & 51 & 56 & 61 & 66 & 71 \\ 41 & 46 & 51 & 56 & 61 & 66 & 71 & 76 \end{bmatrix}.$$

Na seqüência, aplicamos a DCT (2.8) em P e quantizamos o resultado. Então, aplicamos a dequantização e a inversa da DCT (2.9). Como resultado, esperamos uma matriz A e fazemos o processo mais três vezes inserindo esteganografia de forma que no total tenhamos quatro matrizes de pixel.

Considere as quatro matrizes de pixel sendo que:

- A que não sofreu esteganografia,
- B que foi alterada em todos os segundos LSB dos coeficientes AC, cujo módulo é maior que dois,
- C que foi alterada somente no segundo LSB de $F'[0,2]$,
- D que foi alterada em todos LSB dos AC, cujo módulo é maior que um.

Assim, usando a distância Euclidiana como métrica, podemos avaliar o quanto a matriz original foi alterada.

Considerando as matrizes como vetores e calculando a distância Euclidiana, temos:

I. $|P-A| = 35.60898762$

II. $|P-B| = 200.2698180$

III. $|P-C| = 48.98979486$

IV. $|P - D| = 106.5833008$

Como podemos ver, no III caso, alterar o segundo LSB de apenas um coeficiente AC é mais interessante que alterar o primeiro de todos os coeficientes AC, cujo módulo é maior que um, como é feito comumente.

Observe que a distância de I e II são relativamente próximas, isto significa que sofrendo ou não esta esteganografia a imagem tem uma qualidade de resolução próxima.

Isto nos induz a alterar um bit por matriz, podendo ser o primeiro ou o segundo LSB, impedindo que o ataque estatístico mencionado seja bem sucedido.

Se gerarmos gráficos para compararmos as diferenças entre as matrizes, como fizemos na figura 2.5, podemos achar as diferenças grandes. Apesar das diferenças serem aparentemente grandes, veremos na seção seguinte que dado a distância entre um pixel e outro, as alterações são quase imperceptíveis.

4.3.2 Análise da imagem

Na imagem 4.1, com qualidade máxima, temos 29871 coeficientes zerados de um total de 129276 coeficientes AC. O pior caso na esteganografia da imagem acontece quando todos os primeiros LSB maiores que dois são alterados. A tabela 4.2 mostra o total de AC que sofreu as inversões de bits para todos os LSB de determinada ordem que possibilita inversão.

Quando alteramos o segundo LSB em diante, a inversão de todos os bits menores que ele pode não ser a esteganografia mais agressiva na imagem. Considere o número decimal treze e sua representação binária, $13_{10} = 1101_2$, ao invertermos todos os LSB de ordem menor que quatro, isto é os três últimos bits, temos $1010_2 = 10_{10}$. Agrediríamos mais a imagem se invertêssemos apenas o terceiro LSB $1001_2 = 9_{10}$, pois 9 está mais longe do 13 que o 10. Considere o número $15_{10} = 1111_2$, ao invertermos todos os LSB de ordem menor que quatro, temos $1000_2 = 8_{10}$. Agrediríamos menos a imagem se invertêssemos

apenas o terceiro LSB $1011_2 = 11_{10}$. Desta forma, temos que a inversão de todos os bits menores que uma determinada ordem causa uma agressão média na imagem. Logo os testes que fizemos nas seqüências de imagens mostram o que deve acontecer em média.

Ordem LSB	**Maior**	**PSNR**	**Coef. Alterados**
0	1	∞	65920
1	2	48.3191	50632
2	4	44.6541	34795
3	8	41.0672	20952
4	16	38.0045	10522
5	32	35.8443	4260
6	64	34.5841	1343
7	128	34.8148	288
8	256	39.1358	28
9	512	∞	0

Tabela 4.2: *Comparação da alteração dos LSB na figura 4.1.*

A tabela 4.2 mostra o quanto a imagem foi alterada através do **PSNR** e a quantidade de coeficientes da DCT que poderíamos alterar. Observe que conforme aumentamos a ordem do LSB diminuímos o número de coeficientes que podemos alterar. Por um lado estamos agredindo mais a imagem alterando um valor maior nos coeficientes conforme aumentamos a ordem do LSB, assim poderíamos pensara que sempre estaríamos tendo um **PSNR** maior. Por outro estamos agredindo menos a imagem alterando uma quantidade menor de coeficientes. O que vai acontecer com o **PSNR** conforme aumentamos a ordem do LSB depende das propriedades das imagens.

Não mostramos imagens com as alterações referentes a tabela 4.2, pois não são perceptíveis em relação a figura 4.1. Também não mostramos uma figura com a diferença entre a imagem original e as imagens com seus LSB alterados porque teríamos figuras com poucos pontos, praticamente em branco.

O mesmo teste que indica de deformação da imagem e a quantidade de coeficientes que podemos alterar feitos para uma imagem e apresentado na tabela 4.2 foi feito para quatorze seqüências de imagens extraídas de vídeos, os mesmos vídeos da tabela 4.1. Cada seqüência de imagem foi alterada para cada ordem de LSB, assim o vídeo akiyo com

300 imagens gera 2 100 imagens na tabela 4.3. Neste vídeo, cada ordem de LSB tem a média do **PSNR** de 300 imagens e o total de coeficientes da DCT que foram alterados.

Usamos $\overline{PSNR}$ para denotar a média aritmética do valor do PSNR de cada quadro de um vídeo.

Os quatorze vídeos geraram suas respectivas tabelas, de 4.3 até 4.16. Para conseguirmos estas tabelas precisamos gerar 97 seqüências de imagens num total de 71 189 imagens.

Observe que somente nas tabelas 4.5, 4.11 e 4.13, sempre perdemos sinal quando a esteganografia é mais agressiva. Nas outras onze tabelas, em algum momento, usar a posição de um bit mais significativo deforma menos a imagem do que de um bit de posição menos significativa. Na tabela 4.11 a ordem do LSB número 7 tem $\overline{PSNR}$ baixo porque o número de coeficientes alterados é muito menor que o número seqüências de imagens, 2 000, que irão causar a média.

Ordem LSB	**Maior**	$\overline{PSNR}$	**Coef. Alterados**
1	2	40.2153	474643
2	4	37.1055	272634
3	8	35.0927	130660
4	16	32.9069	47109
5	32	36.7156	8181
6	64	5.8914	43
7	128	∞	0

Tabela 4.3: *Comparação da alteração dos LSB no vídeo akiyo.*

Observando que não há diferença visível entre a figura com e sem alterações nos LSB, suspeitamos que podemos alterar qualquer LSB que seja invertível. Esta suspeita pode ser confirmada após aplicar testes para detectar a esteganografia nos quadros dos vídeos. Aplicamos o mesmo programa de detecção de esteganografia que gerou a tabela 4.1 nas seqüências de imagens com coeficientes alterados. No total 61 262 imagens, apresentamos os resultados na tabela 4.17.

Ordem LSB	**Maior**	$\overline{PSNR}$	**Coef. Alterados**
1	2	41.4732	3369807
2	4	39.2139	1829507
3	8	36.1220	924272
4	16	34.4999	372822
5	32	36.2061	72534
6	64	43.2452	4002
7	128	∞	0

Tabela 4.4: *Comparação da alteração dos LSB no vídeo bridge-close.*

Ordem LSB	**Maior**	$\overline{PSNR}$	**Coef. Alterados**
1	2	44.9922	1771732
2	4	42.0735	936201
3	8	38.9428	459238
4	16	38.3336	158958
5	32	35.8321	61017
6	64	34.0594	1723
7	128	∞	0

Tabela 4.5: *Comparação da alteração dos LSB no vídeo bridge-far.*

Ordem LSB	**Maior**	$\overline{PSNR}$	**Coef. Alterados**
1	2	40.7907	718450
2	4	37.7751	433313
3	8	35.6081	223956
4	16	33.4566	92127
5	32	33.0065	26259
6	64	37.2044	2196
7	128	∞	0

Tabela 4.6: *Comparação da alteração dos LSB no vídeo carphone.*

Se compararmos a tabela 4.1 com a tabela 4.17, observamos que somente o vídeo mobile teve alta proporcional de falsos positivos na detecção de esteganografia.

A esteganografia tem uma natureza diferente da criptografia, pois não se baseia em problemas matemáticos, mas somente em problemas heurísticos.

Ordem LSB	**Maior**	$\overline{PSNR}$	**Coef. Alterados**
1	2	41.8397	538619
2	4	38.8667	333118
3	8	36.7643	186012
4	16	34.7089	93886
5	32	32.2477	55527
6	64	36.8229	5793
7	128	∞	0

Tabela 4.7: *Comparação da alteração dos LSB no vídeo claire.*

Ordem LSB	**Maior**	$\overline{PSNR}$	**Coef. Alterados**
1	2	39.7993	666902
2	4	37.0641	372482
3	8	34.8384	179332
4	16	33.1259	69590
5	32	33.2928	19698
6	64	34.8327	2844
7	128	∞	0

Tabela 4.8: *Comparação da alteração dos LSB no vídeo coastguard.*

Ordem LSB	**Maior**	$\overline{PSNR}$	**Coef. Alterados**
1	2	39.2568	570595
2	4	36.5055	348061
3	8	34.1816	181702
4	16	32.8603	80953
5	32	33.3453	20897
6	64	48.9424	1057
7	128	∞	0

Tabela 4.9: *Comparação da alteração dos LSB no vídeo container.*

Ordem LSB	**Maior**	$\overline{PSNR}$	**Coef. Alterados**
1	2	40.5211	867324
2	4	37.2413	523958
3	8	34.5037	270539
4	16	33.1097	109395
5	32	33.1775	28630
6	64	31.0858	687
7	128	∞	0

Tabela 4.10: *Comparação da alteração dos LSB no vídeo foreman.*

Ordem LSB	**Maior**	$\overline{PSNR}$	**Coef. Alterados**
1	2	44.0732	2112319
2	4	41.4695	1107485
3	8	39.5336	457411
4	16	38.6397	137886
5	32	39.8538	33071
6	64	20.6609	3606
7	128	0.1120	7
8	256	∞	0

Tabela 4.11: *Comparação da alteração dos LSB no vídeo highway.*

Ordem LSB	**Maior**	$\overline{PSNR}$	**Coef. Alterados**
1	2	34.5253	1438366
2	4	31.6480	891996
3	8	29.5376	447464
4	16	28.6905	163431
5	32	30.0214	33501
6	64	37.0672	1823
7	128	∞	0

Tabela 4.12: *Comparação da alteração dos LSB no vídeo mobile.*

Ordem LSB	**Maior**	$\overline{PSNR}$	**Coef. Alterados**
1	2	42.4789	422455
2	4	39.2902	242462
3	8	36.9765	112087
4	16	36.0289	33963
5	32	34.1313	9704
6	64	∞	0

Tabela 4.13: *Comparação da alteração dos LSB no vídeo mother.*

Ordem LSB	**Maior**	$\overline{PSNR}$	**Coef. Alterados**
1	2	38.5445	680162
2	4	35.4470	431328
3	8	32.7305	226663
4	16	31.8964	85979
5	32	32.5728	24279
6	64	36.6017	3245
7	128	∞	0

Tabela 4.14: *Comparação da alteração dos LSB no vídeo news.*

Ordem LSB	**Maior**	$\overline{PSNR}$	**Coef. Alterados**
1	2	40.4053	988635
2	4	37.9854	525846
3	8	35.9458	220769
4	16	35.4958	64812
5	32	36.3824	9966
6	64	20.7117	256
7	128	∞	0

Tabela 4.15: *Comparação da alteração dos LSB no vídeo salesman.*

Ordem LSB	**Maior**	$\overline{PSNR}$	**Coef. Alterados**
1	2	40.5174	663133
2	4	37.8862	377431
3	8	35.7030	174705
4	16	33.7630	63324
5	32	35.3055	16210
6	64	39.2841	1317
7	128	∞	0

Tabela 4.16: *Comparação da alteração dos LSB no vídeo silent.*

Vídeo	**Positivo**	**Negativo**	**Suspeitos**
akiyo	553	1243	4
bridge-close	0	12006	0
bridge-far	3515	9091	0
carphone	110	2137	45
claire	71	2893	0
coastguard	125	1671	4
container	240	1520	40
foreman	98	2296	6
highway	1457	12543	0
mobile	181	1615	4
mother	81	1419	0
news	516	1282	2
salesman	7	2687	0
silent	0	1800	0

Tabela 4.17: *Detectando esteganografia nos quadros dos vídeos alterados.*

5 Considerações finais

Foram analisados e experimentados métodos de criptografia em diversas situações e foi testada empiricamente a possibilidade de usar uma esteganografia mais agressiva. Os experimentos em seqüências de imagens, mostraram que podemos usar outros bits diferentes do LSB para aumentar a segurança da informação.

Um outro aspecto interessante é a união das três técnicas já citadas, a esteganocriptografia. Cada uma das três partes desta palavra deve cumprir bem sua obrigação, criando uma segurança por camadas. Por mais atraente que seja um método que compacte, cifre e esconda a mensagem, havendo dependência entre as três etapas, alguma vulnerabilidade pode ser passada para a outra. É bem melhor que elas trabalhem como camadas independentes.

Como principais contribuições deste trabalho nós destacamos um estudo na área de Teoria da Informação, onde introduzimos um conceito de semântica na chave criptográfica na seção 3.3.2. Dividimos os algoritmos de criptografia em Simétricos, Assimétricos e os que contêm a propriedade de *Segredo Perfeito*. Mostramos que os algoritmos apresentados pela literatura que têm a propriedade de *Segredo Perfeito* podem ser classificados como One-time-pad, pois apresentam características semelhantes. Usando o conceito de semântica construímos um *Segredo Perfeito* sem ser do tipo One-time-pad. Isto só foi possível após uma análise da segurança dos algoritmos criptográficos.

Tais contribuições deixam questões interessantes, a primeira: Na criptografia com números irracionais, é possível que sempre se tenha uma chave menor sem perder a propriedade de *Segredo Perfeito*? Se isto for possível, de certa for estaríamos comprimindo a chave criptográfica, levantando a segunda questão interessante: O uso de semântica,

semelhante à criptografia com números irracionais poderia ser usado para altíssima compressão?

No capítulo de esteganografia ampliamos o conceito de *Segredo Perfeito* criando uma definição para esteganografia. Apresentando um tipo *Segredo Perfeito* para esteganografia e deixando a questão se existe outro tipo de *Segredo Perfeito* na esteganografia. Fizemos testes alterando vários coeficientes da DCT em seqüências de imagens, indicando o possível uso de esteganografia em bits diferentes do LSB. Os testes em seqüências de imagens mostram que conforme avançamos nos LSB, tanto a imagem em média sofre menos alterações quanto os testes de esteganografia falham. Desta forma, fica indicado o uso de esteganografia em LSB de maiores ordens.

Referências Bibliográficas

AGRAWAL, M.; KAYAL, N.; SAXENA, N. PRIMES is in P. *Ann. of Math. (2)*, v. 160, n. 2, p. 781–793, 2004. ISSN 0003-486X.

AHSAN, K.; KUNDUR, D. *Practical data hiding in TCP/IP*. 2002. Disponível em: <citeseer.ist.psu.edu/ahsan02practical.html>.

BAILEY, D. H.; CRANDALL, R. E. Random generators and normal numbers. *Experiment. Math.*, v. 11, n. 4, p. 527–546 (2003), 2002. ISSN 1058-6458.

BORGES, F.; PORTUGAL, R.; OLIVEIRA, J. C. Criptografia com números irracionais. *Anais do Congresso de Matemática e suas Aplicações*, Foz2006, v. 1, p. 1–2, 2006. Disponível em: <http://www.mat.ufpr.br/foz2006db/resumos/MS12-0620161759.pdf>.

BRUEN, A.; FORCINITO, M. A. *Cryptography, Information Theory, and Error-Correction: A Handbook for the 21st Century*. [S.l.]: Wiley-Interscience, 2004. ISBN 0471653179.

DIFFIE, W.; HELLMAN, M. E. New directions in cryptography. *IEEE Trans. Information Theory*, IT-22, n. 6, p. 644–654, 1976. ISSN 0018-9448.

ELGAMAL, T. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inform. Theory*, v. 31, n. 4, p. 469–472, 1985. ISSN 0018-9448.

ESCRIVA, J. *Amigos de Dios*. Madrid: Rialp, 1977. Disponível em: <http://www.escrivaworks.org>.

FELDKAMP, U.; BANZHAF, W.; RAUHE, H. A DNA sequence compiler. In: *Proceedings 6th DIMACS Workshop on DNA Based Computers, held at the University of Leiden, Leiden, The Netherlands, 13 - 17 June 2000*. [s.n.], 2000. p. 253. Disponível em: <citeseer.ist.psu.edu/feldkamp00dna.html>.

GEHANI, A.; LABEAN, T. H.; REIF, J. H. DNA-based cryptography. In: WINFREE, E.; GIFFORD, D. K. (Ed.). *Proceedings 5th DIMACS Workshop on DNA Based Computers, held at the Massachusetts Institute of Technology, Cambridge, MA, USA June 14 - June 15, 1999*. American Mathematical Society, 1999. p. 233–249. Disponível em: <citeseer.ist.psu.edu/gehani99dnabased.html>.

GUPTA, V. *et al.* Performance analysis of elliptic curve cryptography for ssl. In: *WiSE '02: Proceedings of the 3rd ACM workshop on Wireless security*. New York, NY, USA: ACM Press, 2002. p. 87–94. ISBN 1-58113-585-8.

GUPTA, V. *et al.* Speeding up secure web transactions using elliptic curve cryptography. In: *11th Ann. Symp. on Network and Distributed System Security – NDSS 2004*. [S.l.]: Internet Society, 2004.

HALSALL, F. *Multimedia Communication: Applications, Networks, Protocols*. [S.l.]: Addison-Wesley, 2001.

HANKERSON, D.; MENEZES, A.; VANSTONE, S. *Guide to elliptic curve cryptography*. New York: Springer-Verlag, 2004. xx+311 p. (Springer Professional Computing). ISBN 0-387-95273-X.

HILL, L. S. Cryptography in an algebraic alphabet. *The American Mathematical Monthly*, Mathematical Association of America, v. 36, n. 6, p. 306–312, jun 1929. ISSN 0002-9890. Disponível em: <http://links.jstor.org/sici?sici=0002-9890ACIAAA

HILL, L. S. Concerning certain linear transformation apparatus of cryptography. *The American Mathematical Monthly*, Mathematical Association of America, v. 38, n. 3, p. 135–154, mar 1931. ISSN 0002-9890. Disponível em: <http://links.jstor.org/sici?sici=0002-9890AO

ISAAC, R. *On the simple normality to base 2 of the square root of s, for s not a perfect square*. 2005. Disponível em: <http://www.citebase.org/cgi-bin/citations?id=oai:arXiv.org:math/0512404>.

ITU-T, I. T. U. *ITU-T Recomendation H.263*. 1998.

ITU-T, I. T. U. *ITU-T Recomendation T.86*. 1998.

JOHNSON, N. F.; JAJODIA, S. Exploring steganography: Seeing the unseen. *IEEE Computer*, v. 31, n. 2, p. 26–34, 1998. Disponível em: <citeseer.ist.psu.edu/johnson98exploring.html>.

KENDALL, M. G. Entropy, probability and information. *International Statistical Review / Revue Internationale de Statistique*, International Statistical Institute (ISI), v. 41, n. 1, p. 59–68, apr 1973. ISSN 0306-7734. Disponível em: <http://links.jstor.org/sici?sici=0306-77343E2.0.CO

KLIMA, R. E.; SIGMON, N.; STITZINGER, E. *Applications of abstract algebra with Maple*. Boca Raton, FL: CRC Press, 2000. xii+256 p. ISBN 0-8493-8170-3.

KOBLITZ, N. Elliptic curve cryptosystems. *Mathematics of Computation*, American Mathematical Society, v. 48, n. 177, p. 203–209, jan 1987. ISSN 0025-5718. Disponível em: <http://links.jstor.org/sici?sici=0025-5718C

KOBLITZ, N.; MENEZES, A. J. A survey of public-key cryptosystems. *SIAM Rev.*, v. 46, n. 4, p. 599–634 (electronic), 2004. ISSN 0036-1445.

Levine, Jack; Nahikian, H. M. On the construction of involutory matrices. *The American Mathematical Monthly*, Mathematical Association of America, v. 69, n. 4, p. 267–272, apr 1962. ISSN 0002-9890. Disponível em: <http://links.jstor.org/sici?sici=0002-9890IM

MILLER, G. L. Riemann's hypothesis and tests for primality. In: *Seventh Annual ACM Symposium on Theory of Computing (Albuquerque, N.M., 1975)*. [S.l.]: Assoc. Comput. Mach., New York, 1975. p. 234–239.

MILLER, V. S. Use of elliptic curves in cryptography. In: *Advances in cryptology—CRYPTO '85 (Santa Barbara, Calif., 1985)*. Berlin: Springer, 1986, (Lecture Notes in Comput. Sci., v. 218). p. 417–426.

NIELSEN, M. A.; CHUANG, I. L. *Quantum computation and quantum information*. Cambridge: Cambridge University Press, 2000. xxvi+676 p. ISBN 0-521-63235-8; 0-521-63503-9.

PROVOS, N.; HONEYMAN, P. Hide and seek: An introduction to steganography. *IEEE Security and Privacy*, IEEE Educational Activities Department, Piscataway, NJ, USA, v. 1, n. 3, p. 32–44, 2003. ISSN 1540-7993.

RABIN, M. O. Probabilistic algorithm for testing primality. *J. Number Theory*, v. 12, n. 1, p. 128–138, 1980. ISSN 0022-314X.

RICHARDSON, I. E. G. *H.264 and MPEG-4 Video Compression*. [S.l.]: Wiley, 2004. ISBN 0470848375.

RIVEST, R. L.; SHAMIR, A.; ADLEMAN, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, ACM Press, New York, NY, USA, v. 21, n. 2, p. 120–126, 1978. ISSN 0001-0782.

RUKHIN, A. L. Approximate entropy for testing randomness. *Journal of Applied Probability*, Applied Probability Trust, v. 37, n. 1, p. 88–100, mar 2000. ISSN 0021-9002. Disponível em: <http://links.jstor.org/sici?sici=0021-9002R

SALOMAA, A. *Public-Key Cryptography*. New York, USA: Springer, 1996.

SAYOOD, K. *Introduction to Data Compression*. [S.l.]: Morgan Kaufmann, 2000.

SCHNEIER, B. *Applied cryptography: protocols, algorithms, and source code in C*. 2nd. ed. New York: Wiley, 1996. ISBN 0–471–12845–7.

SHANNON, C. E. A mathematical theory of communication. *Bell System Tech. J.*, v. 27, p. 379–423, 623–656, 1948. ISSN 0005-8580.

SHANNON, C. E. Communication theory of secrecy systems. *Bell System Tech. J.*, v. 28, p. 656–715, 1949. ISSN 0005-8580.

SHOKRANIAN, S.; SOARES, M.; GODINHO, H. *Teoria dos números*. [S.l.]: Editora Universidade de Brasilia, 1999. ISBN 8523003681.

SMARANDACHE, F. Conjectures which generalize Andrica's conjecture. *Octogon Math. Mag.*, v. 7, n. 1, p. 173–176, 1999. ISSN 1222-5657.

STALLINGS, W. *Cryptography and Network Security: Principles and Practice*. [S.l.]: Pearson Education, 2002. ISBN 0130914290.

TRIVEDI, S.; CHANDRAMOULI, R. Secret key estimation in sequential steganography. *IEEE Trans. Signal Process.*, v. 53, n. 2, part 2, p. 746–757, 2005. ISSN 1053-587X.

VOLCHAN, S. B. What is a random sequence? *The American Mathematical Monthly*, Mathematical Association of America, v. 109, n. 1, p. 46–63, jan 2002. ISSN 0002-9890. Disponível em: <http://links.jstor.org/sici?sici=0002-9890S

WASHINGTON, L. C. *Elliptic Curves: Number Theory and Cryptography*. Boca Raton, FL, USA: CRC Press, Inc., 2003. ISBN 1584883650.

WESTFELD, A.; PFITZMANN, A. Attacks on steganographic systems. In: *IH '99: Proceedings of the Third International Workshop on Information Hiding*. London, UK: Springer-Verlag, 2000. p. 61–76. ISBN 3-540-67182-X.

WESTFELD, A.; WOLF, G. Steganography in a video conferencing system. *Lecture Notes in Computer Science*, v. 1525, p. 32–47, 1998. Disponível em: <citeseer.ist.psu.edu/westfeld98steganography.html>.

Índice Remissivo