Um Esquema de Codificação Homofônica Universal Utilizando o Algoritmo LZW

Daniel da R. Simões e Valdemar C. da Rocha Jr.

***Resumo*— Este artigo investiga um esquema de codificação homofônica universal, sugerido por James L. Massey em 1994, ou seja, um esquema que não necessita do conhecimento a priori da estatística da fonte, o que representa uma vantagem na maioria das aplicações práticas em comunicações. Além de uma análise teórica, são mostrados alguns resultados obtidos por simulação em computador envolvendo arquivos de texto e de imagem no formato bitmap.**

***Palavras-Chave*— Criptografia, Codificação Homofônica, Codificação de Fonte.**

***Abstract*— This article investigates a universal homophonic coding scheme, suggested by James L. Massey in 1994, i. e., a scheme that does not need previous knowledge of the source statistics, which is advantageous in most practical applications in communications. Besides a theoretical analysis, some computer simulation results are presented involving text files and bitmap format images.**

***Keywords*— Cryptography, Homophonic Coding, Source Coding.**

I. Introdução

A criptografia é atualmente uma poderosa ferramenta para a proteção de dado sigilosos como fichas médicas de pacientes, resultados de experimentos genéticos, cadastros bancários, etc. A substituição homofônica, ou codificação homofônica, é uma técnica utilizada na criptografia para combater ataques que exploram desvios na estatística do texto claro como, por exemplo, a análise da freqüência relativa dos símbolos. A excessiva redundância do texto cifrado resultou na quebra de alguns sistemas de cifragem, motivando assim o aparecimento da substituição homofônica.

A substituição homofônica consiste na substituição de cada símbolo da mensagem original por um ou mais símbolos, denominados homofonemas (palavra de origem grega, significando "do mesmo som"), pertencentes a um alfabeto maior, de forma a produzir símbolos uniformemente distribuídos e estatisticamente independentes, reduzindo assim a redundância da mensagem. Na sua forma clássica, esse procedimento necessita do conhecimento prévio da estatística do texto claro para realizar a codificação. Em um trabalho pioneiro, Günther [3] descreveu um algoritmo para a realização desse tipo de codificação homofônica, na qual as palavras representando homofonemas podem ter comprimento variável.

Na maioria das aplicações práticas, não se tem a priori o conhecimento da estatística da fonte, de modo que, procedimentos de codificação para fontes específicas tornam-se bastante ineficientes nesta situação. Massey [1] sugeriu um sistema de codificação homofônica baseado em um esquema universal de codificação de fonte, o qual não precisa do conhecimento a priori da estatística da fonte.

Este artigo está organizado como descrito a seguir. Na *Seção II*, são revistas algumas noções básicas sobre codificação de fonte e também algumas propriedades de códigos unicamente decodificáveis. A *Seção III* trata da teoria que envolve o sistema proposto em [1], bem como alguns detalhes da sua implementação. Finalmente, na *Seção IV*, são apresentados alguns resultados envolvendo textos e imagens.

Daniel da R. Simões e Valdemar C. da Rocha Jr., Departamento de Eletrônica e Sistemas, Centro de Tecnologia e Geociências, Universidade Federal de Pernambuco, Recife, Brasil, E-mails: dansim@click21.com.br, vcr@ufpe.br.

II. Alguns Fundamentos da Codificação de Fonte

O objetivo da codificação de fonte é representar a saída de uma fonte de informação com o mínimo possível de dígitos por símbolo da fonte. Seja $U_1, U_2, \cdots$, a seqüência de saída de uma fonte discreta de informação. Essa fonte é dita *estacionária* se, para todo número inteiro positivo L e toda seqüência $u_1, u_2, \cdots, u_L$ de símbolos do alfabeto fonte, temos $P(U_1, U_2, \cdots, U_L = u_1, u_2, \cdots, u_L) = P(U_{i+1}, U_{i+2}, \cdots, U_{i+L} = u_1, u_2, \cdots, u_L)$, para todo $i \geq 0$. Uma fonte estacionária é denominada *ergódica* se o número de vezes que a seqüência $u_1, u_2, \cdots, u_L$ ocorre na seqüência de saída da fonte $U_1, U_2, \cdots, U_{N+L-1}$, de comprimento $N + L - 1$, quando dividido por N é igual a $P(U_1, U_2, \cdots, U_L = u_1, u_2, \cdots, u_L)$, com probabilidade 1 quando $N \rightarrow \infty$ [1]. Serão consideradas neste artigo, em princípio, apenas fontes *discretas estacionárias e ergódicas* (DSES - *Discrete Stationary Ergodic Source*), uma vez que em geral elas são suficientes para modelar qualquer fonte real de informação. Uma exceção será feita no caso de imagens.

Os códigos de fonte utilizados em esquemas de codificação sem perdas são chamados *códigos unicamente decodificáveis*. Uma condição suficiente para que uma concatenação de palavras-código seja unicamente decodificável é que o código seja sem prefixo (*prefix-free*), ou seja, nenhuma palavra código é a primeira parte (ou prefixo) de outra palavra-código. Esta condição é equivalente à condição de um decodificador ser capaz de imediatamente reconhecer o final de uma palavra-código sem precisar ler o começo da próxima palavra-código. Códigos com essa propriedade são chamados de *códigos instantâneos* [4].

Seja W_i o comprimento da palavra-código X_i e Y_i o comprimento, em símbolos da fonte, da mensagem correspondente V_i. A taxa, dada em bits por símbolo da fonte, do esquema de codificação é definida como:

$$R = \lim_{n \rightarrow \infty} \left(\frac{W_1 + W_2 + \cdots + W_n}{Y_1 + Y_2 + \cdots + Y_n} \right). \qquad (1)$$

A taxa R é uma variável aleatória. Dividindo-se tanto o numerador quanto o denominador por n, a ergodicidade da fonte DSES garante que esta seqüência tende a E[W]/E[Y] com probabilidade 1 quando $n \rightarrow \infty$ (lei dos grandes números) [1] e [2].

A *taxa de informação* de uma fonte DSES é definida pela entropia da sua extensão infinita (em bits por símbolo da fonte), expressa como $H_\infty(U) = \lim_{n\rightarrow\infty} H_n(U)$, em que

$$H_n(U) = \frac{1}{n} H(U_1 U_2 \cdots U_n). \quad (2)$$

O teorema da codificação sem perdas [5] estabelece que códigos unicamente decodificáveis com $R < H_\infty(U)$ não existem, mas, para qualquer $\epsilon > 0$, existem códigos com $R < H_\infty(U)+\epsilon$. Um esquema ótimo de codificação de fonte é aquele em que a taxa R é essencialmente igual a $H_\infty(U)$. Para qualquer fonte DSES, temos $H_\infty(U) \leq H(U_1)$ com igualdade se e somente se a fonte é sem memória, ou seja, a sua saída é uma seqüência de símbolos estatisticamente independentes e identicamente distribuídos (i.i.d.). Para uma fonte binária sem memória, temos $H(U_1) \leq$ 1bit, com igualdade se e somente se $P(U_1 = 0) = P(U_1 = 1) = \frac{1}{2}$. Neste caso, a fonte é denominada de fonte binária simétrica, denotada por BSS (*Binary Symmetric Source*). Assim, a fonte BSS é a única fonte binária que não pode ser comprimida. Então, podemos ver a tarefa da codificação de fonte, idealmente, como a tarefa de transformar de forma reversível a saída da fonte numa seqüência de saída de uma fonte BSS, o que a torna bastante atrativa do ponto de vista criptográfico.

Definem-se cifras não-expansivas como cifras em que existe uma sequência de inteiros positivos $n_1, n_2, n_3, \cdots$ tal que os primeiros n_i símbolos $Y_1, Y_2, \cdots, Y_{n_i}$ do texto cifrado junto com a chave secreta determinam unicamente os primeiros n_i símbolos $X_1, X_2, \cdots, X_{n_i}$ do texto claro para $i = 1, 2, 3, \cdots$. Como exemplos de tais cifras podem ser citadas as cifras de fluxo aditivas na qual $Y_i = X_i \oplus Z'_i$, em que $Z'_1, Z'_2, Z'_3, \cdots$ é a chave de seção gerada a partir da chave secreta Z; e as cifras de bloco, em que os blocos de texto claro e de texto cifrado possuem o mesmo comprimento N. Para o caso das cifras de fluxo aditivas, tem-se $n_i = i$ e para o caso das cifras de bloco, tem-se $n_i = iN$. As cifras não-expansivas possuem uma propriedade interessante que é descrita a seguir.

Propriedade *random-in/random-out* das cifras não-expansivas: Se a cifra for não-expansiva e o texto claro for uma fonte BSS, o texto cifrado também será uma fonte BSS, para qualquer escolha z da chave secreta Z. Além disso, o texto cifrado é estatisticamente independente da chave secreta Z para qualquer que seja sua distribuição de probabilidade, ou seja, a cifra é fortemente ideal [6]. A prova dessa propriedade pode ser encontrada em [1]. Esta propriedade implica que em um ataque ao texto cifrado, não se pode obter nenhuma informação sobre a chave secreta Z, não importando a quantidade examinada do texto cifrado. Isso pode ser alcançado aplicando-se uma técnica de substituição homofônica na fonte de informação antes de cifrá-la.

A. O Algoritmo de Lempel-Ziv-Welch (LZW)

O algoritmo de Lempel-Ziv é um esquema de codificação universal de fonte pertencente à família dos códigos de dicionários adaptativos. Variantes desse algoritmo foram utilizadas em programas como *compress* (UNIX), *gzip* (GNU) e *pkzip* (DOS). O princípio básico desse algoritmo é separar os dados a serem transmitidos em segmentos de comprimento variável, que são as menores subseqüências de símbolos não encontradas previamente na seqüência a ser transmitida. A cada segmento é associada uma palavra-código que é transmitida ou armazenada. O último símbolo de cada frase nova é codificado como o primeiro símbolo da próxima frase. Um dicionário contém as associações entre as palavras-código e os segmentos. Em implementações práticas, é comum limitar o tamanho máximo do dicionário a um valor que garanta um bom desempenho e utilize uma porção adequada de memória. Na aplicação considerada, o tamanho máximo do dicionário foi limitado a uma certo número de palavras-código, que denotaremos por N, sendo o seu preenchimento regido pelo algoritmo a ser descrito.

Existem diversas versões do algoritmo de Lempel-Ziv, todas baseadas nas idéias propostas em [7] e [8]. Será considerada a versão descrita por Welch em [9], também conhecida como algoritmo LZW. Este algoritmo é uma extensão do algoritmo LZ78 proposto por A. Lempel e J. Ziv [8], com a diferença de que o dicionário não está vazio no início, e sim contendo todos os símbolos individuais possíveis. Considerando-se um alfabeto de 256 símbolos para a fonte de informação e limitando-se o dicionário a $N = 1000$ palavras, por exemplo, as primeiras 256 palavras-código do dicionário representarão os símbolos individuais do alfabeto da fonte, indexadas de 0 a 255, e as demais 744 palavras-código representarão as subseqüências produzidas pela fonte, indexadas de 256 a 999. Considere a situação em que foram encontradas n subseqüências diferentes na fonte de informação, $n > 744$. As 744 primeiras subseqüências encontradas irão preencher completamente o dicionário, sendo indexadas de 256 a 999, uma vez que o tamanho do dicionário é limitado a 1000 palavras. As subseqüências seguintes serão tratadas como se fossem as primeiras a terem ocorrido, sendo representadas com índices a partir de 256. Assim, o dicionário será alterado, sendo sobrescritas as subseqüências previamente existentes. Esse comportamento se repete até que a seqüência da fonte de informação termine.

Como exemplo, consideremos uma fonte com alfabeto composto pelos símbolos a e b. Seja $\mathcal{L}_1 = (a, b)$ o dicionário inicial. Para cada $i = 1, 2, \cdots$, marca-se o final da i-ésima frase no ponto em que, incluindo o próximo símbolo, resulte em uma seqüência não listada em $\mathcal{L}_i$, e então coloca-se esta seqüência acrescida do próximo símbolo no final do dicionário $\mathcal{L}_i$, formando assim o dicionário $\mathcal{L}_{i+1}$. O dicionário $\mathcal{L}_i$ contém exatamente $i + 1$ seqüências. Considere a saída dessa fonte como sendo $aababababaaa$. A Tabela I ilustra a segmentação da seqüência, o preenchimento do dicionário e a saída resultante da codificação.

Sejam $V_1, V_2, V_3, \cdots$ as frases da seqüência segmentada da fonte. Cada frase V_i é codificada como a palavra binária com

TABELA I
CODIFICAÇÃO DA SEQÜÊNCIA "$aababababaaa$" UTILIZANDO O ALGORITMO LZW.

Entrada:	a	a	b	ab	aba	aa
Saída:	0	0	1	3	5	2

Palavra-código	**Seqüência**	**Derivada da seqüência**
0	a	Inicial
1	b	
2	aa	0 | a
3	ab	0 | b
4	ba	1 | a
5	aba	3 | a
6	abaa	5 | a

comprimento $W_i = \lceil \log(i+1) \rceil$, em ordem lexicográfica, para o índice do dicionário $\mathcal{L}_i$. Para $i > 1$, como a última seqüência da lista $\mathcal{L}_i$ só é colocada depois da segmentação de V_{i-1}, o que requer o primeiro símbolo de V_i, então a decodificação da palavra-código X_i para a frase V_i, não pode ser feita percorrendo o dicionário pois ele terá formado apenas a lista $\mathcal{L}_{i-1}$. Assim, quando $i > 1$ e X_i aponta para a última seqüência de $\mathcal{L}_i$, o primeiro símbolo de V_i deve ser o mesmo do primeiro dígito de V_{i-1}, permitindo o decodificador formar "prematuramente" o dicionário $\mathcal{L}_i$ necessário para decodificar X_i.

Como o comprimento W_i da i-ésima palavra código X_i não depende da seqüência da fonte, o código LZW é sem prefixo (*prefix-free*). Nesse código, para qualquer fonte DSES, a taxa R definida por (1) satisfaz

$$R = H_\infty(U), \tag{3}$$

ou seja, o código LZW é universal para a classe de todas as fontes DSES [1].

III. A CODIFICAÇÃO HOMOFÔNICA UNIVERSAL

Em 1988, C. G. Günther idealizou um esquema de codificação homofônica que transforma uma seqüência de texto claro em uma seqüência de saída de uma fonte BSS, também conhecida como codificação homofônica perfeita [3]. Entretanto, tal codificação requer o conhecimento a priori da estatística da fonte de informação. A fim de lidar com casos em que tal conhecimento a priori não existe, Massey [1] sugeriu um esquema de codificação homofônica que utiliza um código de fonte universal, conforme ilustrado na Figura 1.

Fig. 1. Codificador homofônico universal.

O bloco denominado BSS denota um gerador de símbolos binários aleatórios. Para a implementação do bloco BSS foi utilizado o gerador de números pseudo-aleatórios de Park-Miller-Carta, bastante analisado em [10], [11] e [12]. O multiplexador (n, m) considerado é um dispositivo que tem como saída n símbolos da fonte de informação, seguidos de m símbolos aleatórios da fonte BSS, depois mais n símbolos da fonte de informação, depois mais m símbolos da fonte BSS, e assim sucessivamente, não necessariamente nessa ordem. Observamos que a seqüência de saída do multiplexador não representará em geral uma DSES, uma vez que a multiplexação introduz não estacionaridade. No entanto, blocos de símbolos multiplexados, cujo comprimento seja um múltiplo de $n+m$, dão origem a um processo ciclo-estacionário. Esta condição não é observada pelo codificador LZW, ao processar a saída do multiplexador para gerar a saída do codificador homofônico, porque a segmentação de símbolos pelo algoritmo LZW é de comprimento variável.

A seqüência original produzida pela da fonte de informação pode ser recuperada facilmente a partir da saída do codificador homofônico, sem o conhecimento do gerador de símbolos aleatórios utilizado, simplesmente passando essa saída por um decodificador LZW e então descartando os símbolos aleatórios provenientes da BSS.

A fonte de mensagens possui uma entropia dada por $H_\infty(V) = nH_\infty(U) + m$, em que $H_\infty(U)$ é a entropia da fonte de informação. Como uma representação aproximada de uma fonte de informação usaremos a seguir uma imagem, ilustrando a técnica aqui abordada. Consideremos que o multiplexador inicia sua operação emitindo n_p pixels da fonte de informação e em seguida emitindo m_p pixels de uma "imagem BSS". Os valores de n e m em bytes, denotados por n_b e m_b, considerando-se bytes de 8 bits cada, são dados por:

$$n_b = n_p \cdot b$$
$$m_b = m_p \cdot b$$

em que b é o número de bytes necessários para representar um pixel da fonte de informação. Em imagens no formato RGB (Red-Green-Blue), por exemplo, um pixel é representado por 3 bytes, enquanto em uma imagem em escala de cinza, um pixel é representado por 1 byte. Dividindo-se o número N_{PF} de pixels da imagem fonte por n_p, obtemos $N_{PF} = q_p \cdot n_p + r_p$. Desta forma, o número N_{PMUX} de pixels da imagem gerada pelo multiplexador será:

$$N_{PMUX} = (n_p + m_p) \cdot q_p + r_p \tag{4}$$

O número de bytes N_{BMUX} da imagem gerada pelo multiplexador pode ser obtido calculando-se:

$$N_{BMUX} = N_{PMUX} \cdot b \tag{5}$$

Denotando-se por N_{BHC} o número de bytes da sequência de saída do sistema, e tomando-se o número N_{BF} de bytes da fonte de informação, definimos, em porcentagem, a taxa de compressão ρ relativa à fonte de informação como:

$$\rho = \left(\frac{N_{BHC} - q_p \cdot m_b}{N_{BF}} \right) \cdot 100 \tag{6}$$

Em (6), desconsideram-se no numerador da fração todos os bytes aleatórios oriundos da fonte BSS, pois estes não são comprimidos quando passam pelo codificador de fonte.

Considerando um codificador de fonte ideal, o qual elimina toda a redundância da seqüência de saída do multiplexador,

resulta que os bytes de saída do sistema serão estatisticamente independentes e, equivalentemente, quando estes bytes são considerados como uma imagem, os pixels da imagem de saída do sistema serão estatisticamente independentes. Para realizar a análise deste modelo, foi escolhida como fonte de informação a imagem de Lenna, mostrada na Figura 2. A Figura 3 ilustra a imagem de Lenna codificada pelo sistema com $n_p = 300$ e $m_p = 10$, com um dicionário de $N = 8000$ palavras para o LZW. Como existem 256 bytes distintos de 8 bits, a seqüência de saída de um compressor ideal terá uma entropia dada por $H(V) = \log_2(256) = 8$ bits por byte. Considerando-se que os pixels da imagem de saída não são correlacionados, tome como exemplo a Figura 3, apesar de não garantir a independência estatística, a entropia de primeira ordem, obtida utilizando as freqüências relativas dos bytes, permite uma primeira avaliação da eficácia do sistema.

Fig. 2. Imagem de Lenna com 512 x 512 pixels.

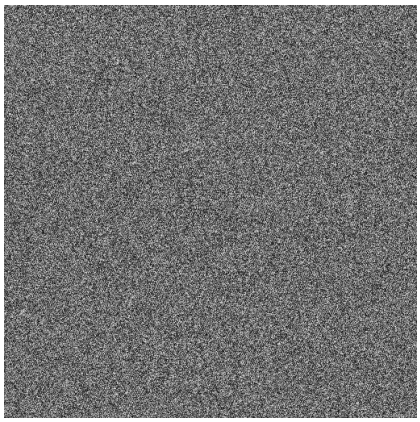

Fig. 3. Imagem de Lenna codificada com $n_p = 300$, $m_p = 10$ e um dicionário de $N = 8000$ palavras para o LZW.

Considerando-se que o multiplexador tem como saída primeiro n_p pixels da imagem, depois m_p pixels aleatórios, o objetivo dos ensaios realizados foi analisar duas estratégias de utilização do sistema a saber.

1) Gerar o dicionário do LZW ignorando a estrutura ciclo-estacionária dos blocos de comprimento $n_p + m_p$ pixels na saída do multiplexador.
2) Desconsiderar o dicionário gerado para o bloco anterior, de comprimento $n_p + m_p$ pixels, reiniciando a geração do dicionário a cada novo bloco.

Na próxima seção são mostrados os resultados dessas duas estratégias.

IV. Resultados

Para encontrar uma relação de $\frac{n}{m}$ com uma entropia de saída próxima de 8 bits por byte, foi fixado o tamanho máximo do dicionário em $N = 8000$ palavras e $m_p = 10$. As Figuras 4 e 5 ilustram ρ e a entropia da seqüência de saída para as duas estratégias, considerando $1 \leq n_p \leq 100$.

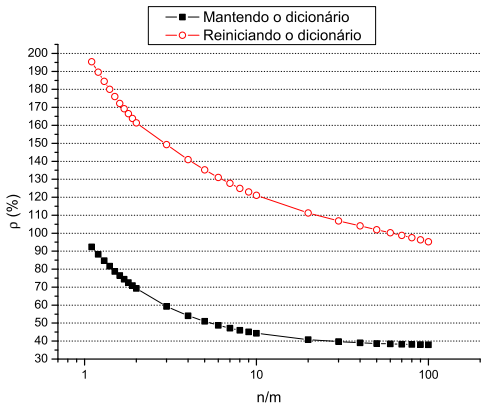

Fig. 4. Taxa de compressão ρ para um dicionário com $N = 8000$ palavras e $m_p = 10$.

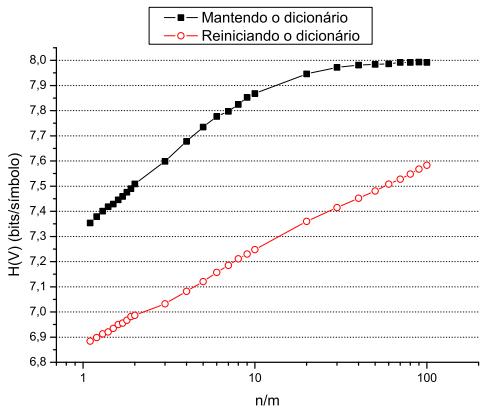

Fig. 5. Entropia da seqüência de saída para um dicionário com $N = 8000$ palavras e $m_p = 10$.

Observando as Figuras 4 e 5, nota-se um melhor desempenho, tanto para a taxa de compressão ρ quanto para a entropia da saída, da estratégia que processa os bytes de saída do multiplexador ignorando a estrutura de blocos de comprimento $n_p + m_p$ ao construir o dicionário. A entropia tende a 8 bits por byte quanto maior for a relação $\frac{n}{m}$. Foi escolhida a relação $\frac{n}{m} = 30$, cuja taxa de compressão e entropia valem,

respectivamente, $\rho = 39,705\%$ e $H(V) = 7,972$ bits/símbolo, e foi mantido o tamanho máximo do dicionário em $N = 8000$ palavras. Fazendo-se $n_p = 30$ e $m_p = 1$, multiplicou-se esses dois parâmetros por a, $1 \leq a \leq 1000$, para novamente observar o comportamento das duas estratégias. As Figuras 6 e 7 ilustram ρ e a entropia da seqüência de saída obtidos.

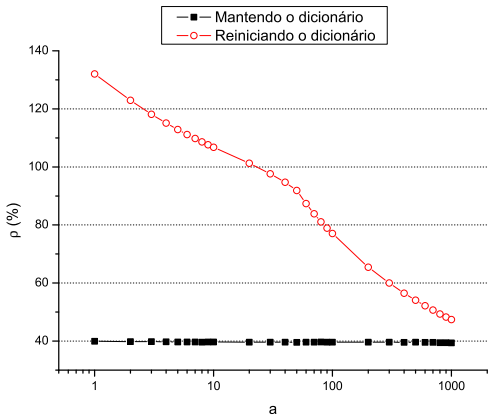

Fig. 6. Taxa de compressão ρ para um dicionário com $N = 8000$ palavras.

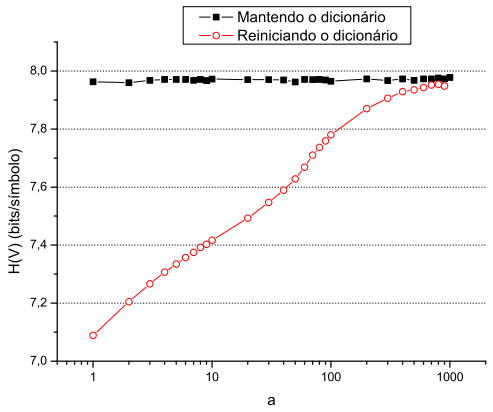

Fig. 7. Entropia da seqüência de saída para um dicionário com $N = 8000$ palavras.

Observando as Figuras 6 e 7, nota-se que não há alterações significativas na taxa de compressão ρ e na entropia quando a estratégia adotada é a de ignorar a estrutura ciclo-estacionária ao gerar o dicionário. Entretanto, para a estratégia de reiniciar o dicionário entre blocos consecutivos, há uma significativa melhora no desempenho do sistema, referente aos dois parâmetros considerados, à medida em que a cresce, atingindo resultados bastante próximos àqueles da outra estratégia.

Finalmente, mantendo-se a taxa $\frac{n}{m} = 30$, procurou-se observar o comportamento do sistema variando-se o tamanho máximo do dicionário no intervalo $300 \leq N \leq 100.000$ palavras. Para a estratégia que reinicia o dicionário entre blocos consecutivos, consideraram-se $n_p = 300$ e $m_p = 10$ ($\rho = 39,666\%$ e $H(V) = 7,972$ bits/símbolo), enquanto que para a estratégia que ignora a estrutura de blocos, consideraram-se $n_p = 15000$ e $m_p = 500$ ($\rho = 54,089\%$ e $H(V) = 7,935$ bits/símbolo). As figuras 8 e 9 ilustram o comportamento das duas estratégias.

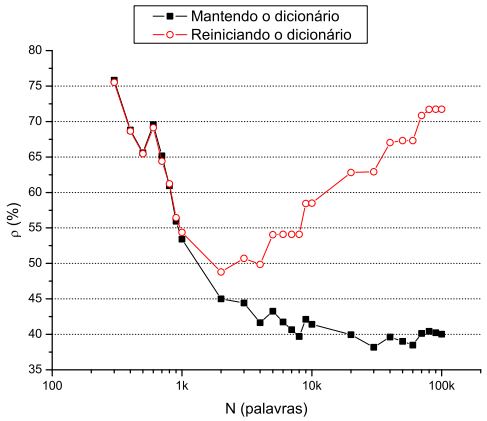

Fig. 8. Taxa de compressão ρ variando-se o número total N de palavras do dicionário.

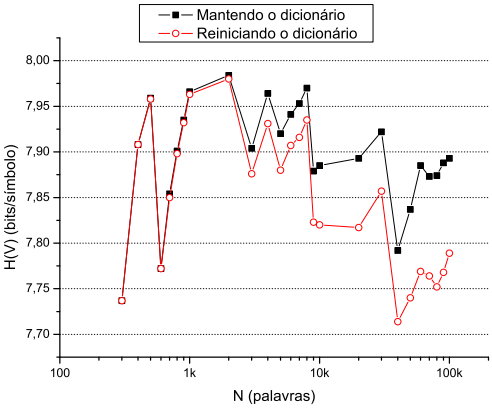

Fig. 9. Entropia da seqüência de saída variando-se o número total N de palavras dicionário.

Observando as figuras 8 e 9, nota-se que a estratégia 1 possui vantagem em relação à estratégia 2, conseguindo uma melhor taxa de compressão ρ à medida em que o tamanho máximo do dicionário aumenta. Para a estratégia 2, a taxa compressão ρ possui resultados semelhantes até um tamanho de aproximadamente $N = 2000$ palavras para o dicionário. A partir daí, observa-se uma piora na taxa de compressão para tamanhos maiores do dicionário, pois não dá informação suficiente para o LZW aprender a estatística em apenas um bloco de $n_p + m_p$ bytes. Em relação à entropia, nota-se um comportamento semelhante entre as duas estratégias, com uma ligeira vantagem para a estratégia 1.

Não foram observadas mudanças significativas invertendo a ordem de alimentação no multiplexador: tendo como saída

primeiro m_p pixels aleatórios, depois n_p pixels da imagem.

V. CONCLUSÃO

Como foi mostrado nas seções anteriores, podemos realizar uma codificação homofônica universal eficiente utilizando o codificador LZW, o qual opera com uma segmentação de comprimento variável. Os resultados obtidos para a imagem considerada mostram que podemos ter uma entropia da seqüência de saída próxima do ideal e, ao mesmo tempo, ter uma boa compressão da seqüência original de entrada. Para se obter resultados mais precisos, quando a fonte de informação é uma imagem, seria desejável proceder testes para determinar o grau de independência estatística dos pixels de saída. No caso em que a fonte de informação é um texto, esta será DSES e portanto a entropia de primeira ordem é sozinha um bom indicativo para a eficácia do sistema. A técnica de codificação homofônica analisada, quando aplicada a uma fonte de informação antes da utilização de um criptograma não-expansivo de chave secreta torna a distribuição de probabilidade dos símbolos da fonte aproximadamente uniforme, de forma universal, não precisando do conhecimento prévio da estatística da fonte. Isto dificulta a obtenção de informação utilizando ataques que exploram desvios da estatística do texto cifrado, o que torna o sistema analisado bastante atrativo, sobretudo devido à sua simplicidade de implementação. É bom lembrar aqui que vários esquemas de substituição homofônica [3] operam com uma expansão do texto original, ao invés de uma compressão, fato que não ocorre no sistema aqui estudado.

AGRADECIMENTOS

Este trabalho recebeu apoio parcial da Fundação de Amparo à Ciência e Tecnologia do Estado de Pernambuco (FACEPE), projeto IBPG-0222-3.04/08, do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), projeto 306612/2007-0, e do Programa de Pós-graduação em Engenharia Elétrica da Universidade Federal de Pernambuco. O primeiro autor agradece a todos os colegas que ajudaram a tornar possível a realização deste trabalho.

REFERÊNCIAS

[1] J. L. Massey, "Some Applications of Source Coding in Criptography", *European Transactions on Telecommunications*, vol.5, no. 4, pp. 7/421-15/429, Julho-Agosto de 1994.

[2] J. L. Massey, "Applied Digital Information Theory I", Lecture notes, Swiss Federal Institute of Technology (ETH), Zurique, Suíça.

[3] C. Günther, "A Universal Algorithm for Homophonic Coding", *Advances in Cryptology - Eurocypt '88*, (Ed. C.G.Günther) LNCS no 330, Springer-Verlag, pp. 405-41, 1988.

[4] N. Abramson, *Information Theory and Coding*, McGraw Hill, New York, 1963.

[5] R. G. Gallager, *Information Theory and Reliable Communications*, John Wiley and Sons, Inc., New York, 1968.

[6] C. E. Shannon, ´´Communication Theory of Secrecy Systems", *Bell Sys. Tech. J.*, Vol. 28, pp. 656-715, Outubro 1949.

[7] J. Ziv e A. Lempel, "A Universal Algorithm for Sequential Data Compression", *IEEE Trans. Inform. Th.*, Vol. IT-23, pp. 337-343, Maio 1977.

[8] J. Ziv e A. Lempel, "Compression of Individual Sequences via Variable-Rate Coding", *IEEE Trans. Inform. Th.*, Vol. IT-24, No. 5, pp. 5306, Setembro 1978.

[9] T. A. Welch, "A Technique for High Performance Data Compression", *IEEE Computer*, Vol. 17, pp. 8-19, Junho 1984.

[10] Stephen K. Park e Keith W. Miller, "Random Number Generators: Good ones are Hard to Find", *Communications of ACM*, Vol. 31, No. 10, Outubro 1988.

[11] David G. Carta, "Two Fast Implementations of the "Minimal Standard"Random Number Generator", *Communications of ACM*, Vol. 33, No. 1, Janeiro 1990.

[12] G. S. Fishman e L. R. Moore, "An Exhaustive Analysis of Multiplicative Congruential Random Number Generators With Modulus $2^{31} - 1$", *SIAM J. Sci. Stat. Comput.*, vol. 7, Janeiro de 1986, pp. 24-45.