Transmissão de arquivos médicos por compressão de Huffman e RLE

Arthur N. Guedes, Diego Vinicius da Silva, Eduardo M. de Souza, Gustavo Murayama, Hiago Lucas C. de Melo, Lucas Damazio da Costa, Lucas Tornai
Centro de Matemática, Computação e Cognição – Universidade Federal do ABC (UFABC) - Santo André – SP – Brasil

guedes.arthur@aluno.ufabc.edu.br, diego.vinicius@aluno.ufabc.edu.br, e.maciel@aluno.ufabc.edu.br, g.murayama@aluno.ufabc.edu.br, jomi@inf.furb.br, damazio.costa@aluno.ufabc.edu.br, lucas.tornai@aluno.ufabc.edu.br

Resumo. Este artigo é parte do método avaliativo da disciplina de Algoritmos e Estrutura de Dados II com o tema de Transmissão de arquivos de exames médicos on-line de emergência (telemedicina) em redes de comunicação. Apresenta a pesquisa e processo de aplicação dos conceitos da disciplina para a produção de um software de compressão de arquivos, que utiliza dos algoritmos de Huffman e alternativamente o método de Run-Length Encoding.

Abstract. This article is part of the evaluative method of Data Structure and Algorithms II with the topic Compression and Transmission for Telemedicine Networks. Presents the research and application process of the subject concepts for the production of a file compression software, which uses the Huffman algorithms or the Run-Length Encoding method.

1. Introdução

Telemedicina é o conceito de exercitar a medicina mediada através de conceitos tecnológicos como comunicação audiovisual e de dados[1]. Um dos formatos mais difundidos atualmente é o do prontuário eletrônico do Paciente (PEP), que é uma tecnologia criada para integrar os históricos médicos de forma digital, com o intuito de facilitar a análise do paciente e aumentar a eficiência dos atendimentos e consultas. Além disto, apesar de com menor difusão, existe o método de se transmitir dados de exames para que possam ser analisados remotamente por um perito, por exemplo, em estados diferentes do país. No Brasil, a implantação de ambos é discutida desde 2002 e tem progredido desde então, embora num ritmo lento causado pela infraestrutura precoce.

Além disso, as tecnologias destinadas às consultas costumam receber maior prioridade, o que defasa ainda mais o sistema computacional dos consultórios. Dito isso, diversas estratégias devem ser tomadas para tornar o PEP e a leitura de exames remota útil no dia-a-dia dos agentes da área da saúde, como a compressão das grandes quantidades de dados(geradas pela precisão e minuciosidade dos exames), visando a transmissão dos mesmos, que agiliza o acesso das informações.

Durante o processo, devem ser utilizados métodos de compressão sem perdas de dados, pois uma pequena distorção nas informações de um exame ou uma consulta pode alterar consideravelmente o seu diagnóstico.

Portanto, para este trabalho, foram escolhidos, para o desenvolvimento de pesquisa, o algoritmo de Huffman e o Método do Run-Length Encoding, pois ambos são técnicas conhecidas e consideravelmente simples de compressão sem perdas.

2. Fundamentos

2.1 Algoritmo *Run-Length Encoding*

O algoritmo Run-Length Encoding, como previamente mencionado, é um algoritmo de compressão sem perdas. É um algoritmo utilizado desde a década de 70[2], e foi patenteado pela empresa Hitachi em 1983[3].

É um algoritmo extremamente simples, e possui uma eficiência melhor sabendo-se de antemão que o objeto a ser comprimido possui muitas repetições. Isto acontece pois o algoritmo funciona de forma a reduzir uma sequência de caracteres repetidos em um número representando o número de repetições seguido do caracter que formava a sequência. Portanto, em casos gerais, não é um algoritmo bom para compressão de palavras, pois dificilmente se há a repetição de mais de 3 letras iguais sequencialmente, enquanto que para imagens é mais promissor, pois imagens costumam ter sequências grandes das mesmas cores. Também pode ser útil para dados binários, pois sequências de zeros e uns ocorrem com frequência.

Para exemplificar seu funcionamento, podemos pegar a seguinte sequência de caracteres: "AAABBCCCCD". Neste caso, o algoritmo converteria essa sequência em: "3A2B4CD". Nos casos onde se existam números nos caracteres (por exemplo nos casos de binários, por exemplo) pode-se utilizar um caractere especial para sinalizar que o número logo em seguida é um caractere literal, e não um número de vezes que um outro caractere aparece. Por exemplo, no caso de: "AAA55CCCD" podemos obter algo como: "3A2@53CD", onde deve-se de antemão sinalizar que o caractere "@" é o que indica um número como caractere literal.

2.2 Algoritmo de Huffman

O algoritmo de Huffman foi criado por David A. Huffman e publicado no paper "A Method for the Construction of Minimum-Redundancy Codes"em 1952.[5]

É um algoritmo de compressão que além de não possuir perdas, possui a característica de produzir uma saída que possui tamanho variável para diferentes símbolos, dependendo da frequência que tal símbolo aparece, e também de ser um algoritmo entrópico, ou seja, não depende das características do que está sendo comprimido.

É um algoritmo muito utilizado pois é de relativa simples implementação, não possui patente e é também rápido. É utilizado em CODECs populares como JPEG e MP3.[6]

O seu método de funcionamento começa pela ordenação dos símbolos do objeto a ser codificado. Após isso, soma-se as menores probabilidades para construir um novo símbolo hipotético, e repete-se essa fórmula até se chegar a apenas um símbolo hipotético. Isto acaba na forma de construção de uma árvore conhecida como Árvore de Huffman.

A Árvore de Huffman é uma árvore binária onde a principal informação é o símbolo das arestas, que pode ser 0 ou 1, geralmente 0 para as arestas ligadas aos filhos à esquerda e 1 para as arestas ligadas ao filho da direita. As folhas possuem os símbolos que foram codificados, de forma que a combinação das arestas para chegar até o mesmo consiste na forma comprimida deste símbolo.

A imagem abaixo mostra a implementação de forma visual, onde si é o símbolo a ser codificado e pi é a probabilidade deste símbolo aparecer.

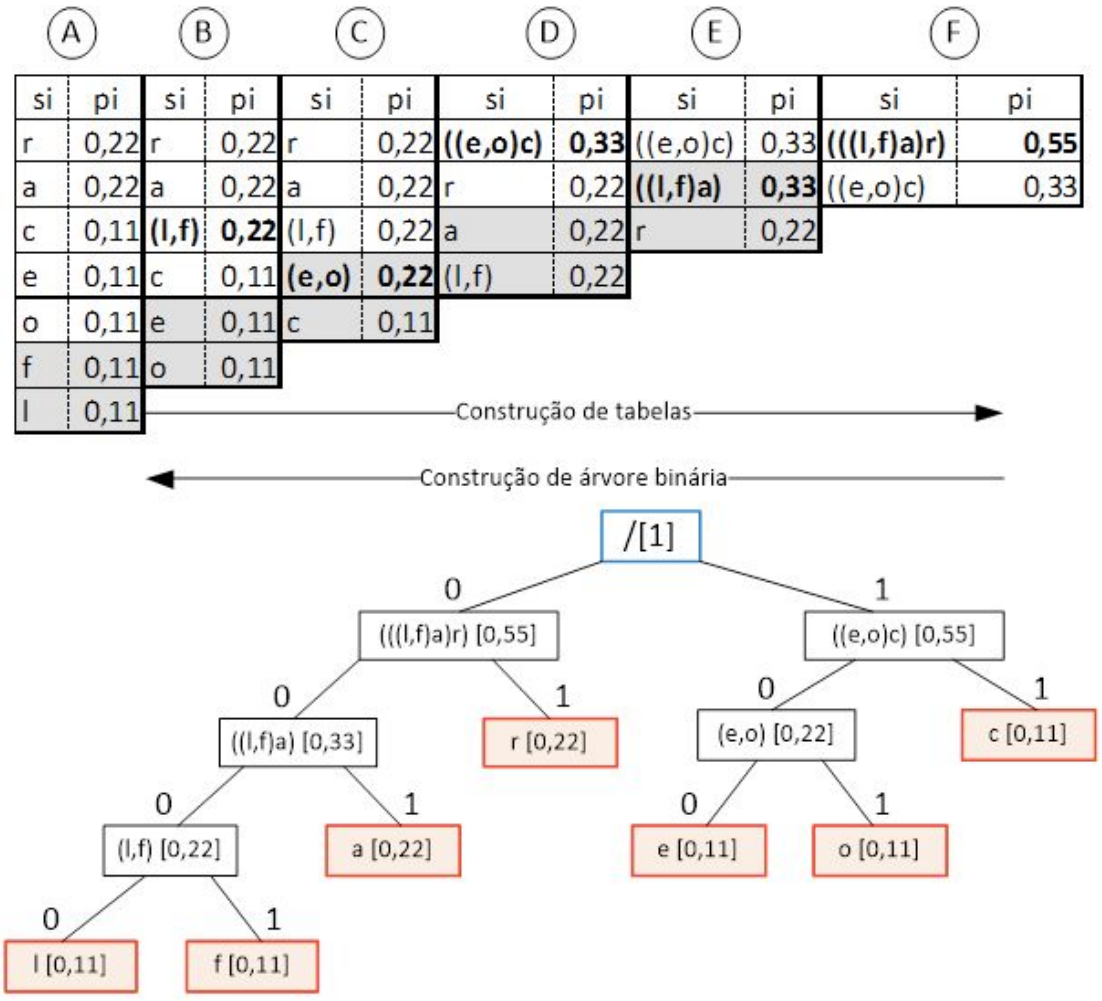

Imagem 1 - Implementação do Algoritmo de Huffman.[6]

2.3 Arquivos DICOM

DICOM (*Digital Imaging and Communications in Medicine*) é a padronização internacional para transmissão, armazenamento, impressão, processamento e apresentação de informações visuais médicas, permitindo interoperabilidade de informações e integrando equipamentos médicos de aquisição de imagem[12].

3. Implementação

Foi implementado uma aplicação para realização da compressão e transmissão de arquivos médicos, para que fosse possível fazer uma comparação entre os dois algoritmos. Essa aplicação possui uma interface gráfica para que seja possível escolher qual é o arquivo que se deseja comprimir e enviar e uma interface para receber os arquivos.

A aplicação foi desenvolvida utilizando-se o .NET Framework versão 4.6 utilizando-se a linguagem de programação C#. A escolha dessa linguagem e framework se devem a possibilidade de construir interfaces gráficas e sua alta performance e robustez. Os algoritmos de compressão também também fazem parte dessa aplicação e por isso foram escritos na mesma linguagem.

A aplicação utiliza o paradigma cliente servidor e existe uma interface específica para cada um dos envolvidos. O cliente nesse caso é a interface no qual é possível realizar a compressão e o envio dos arquivos, enquanto o servidor recebe os arquivos e faz sua persistência. A comunicação entre o cliente e servidor foi feita através do protocolo HTTP e o conteúdo da mensagem foi serializado para o formato JSON.

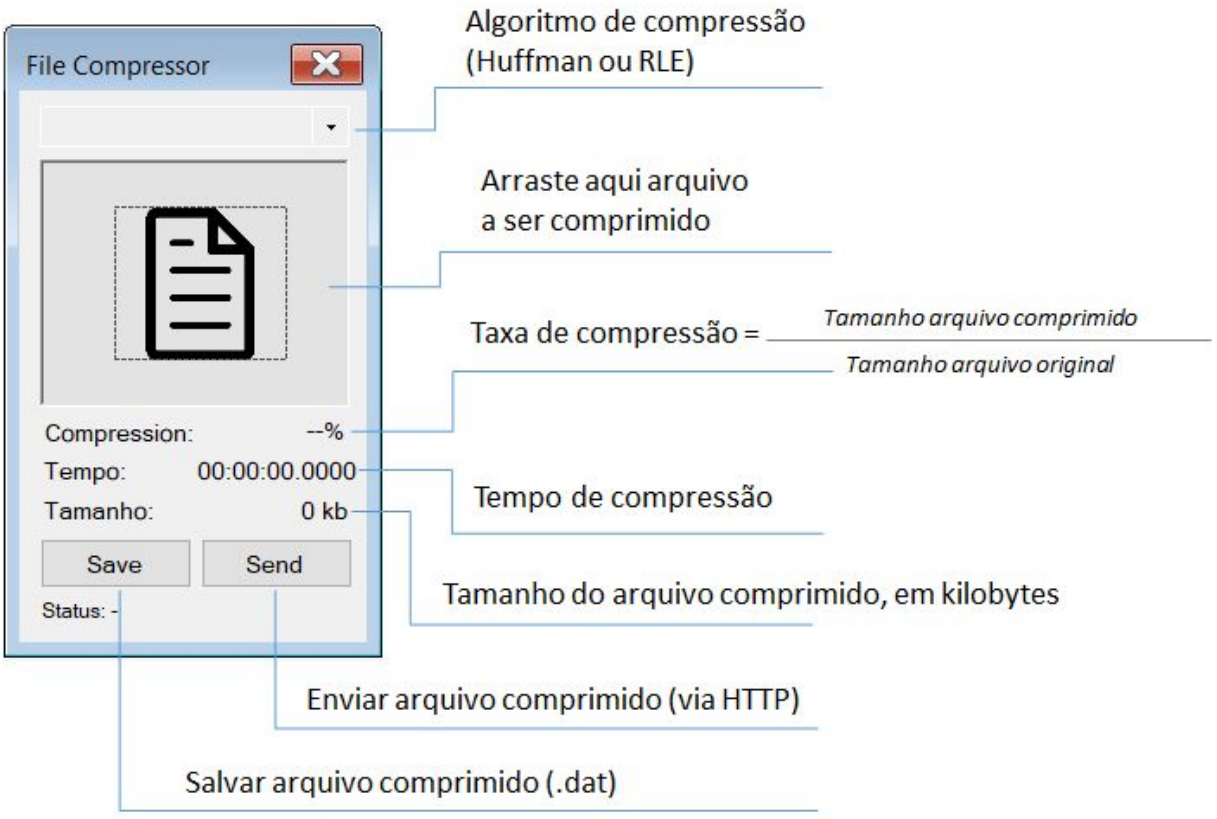

Imagem 2 - Guia da interface do programa File Compressor

3.1 Implementação do algoritmo de Huffman

Para se realizar a compressão a unidade mínima de informação utilizada foram os bytes do arquivo, ou seja, o arquivo foi lido byte a byte e no caso do algoritmo de Huffman realizou-se a contagem das frequências de repetição de cada um dos bytes presentes no arquivo.

Além disso, para realizar a construção da árvore de Huffman, seguiu-se a vasta literatura presente na área que recomenda a utilização da estrutura da dados HEAP mínimo para garantir boa performance do algoritmo [7].

As etapas do algoritmo estão descritas abaixo:

1. Leitura do arquivo byte a byte e construção de um vetor contendo informações sobre o byte e sua frequência;
2. Construção do HEAP mínimo utilizando-se o vetor de frequências;
3. Construção da árvore de Huffman utilizando-se o HEAP mínimo;
4. Com base na árvore de Huffman obtida, foi criado um dicionário que mapeia um byte para sua nova codificação em bits.
5. Criação de vetor de bits vazio para armazenar o arquivo comprimido.
6. Leitura byte a byte do arquivo mapeando cada byte com o dicionário criado no passo 4 para seu novo prefixo e seu armazenamento no vetor criado no passo 5.
7. Transformação do vetor de bits que armazena o arquivo comprimido para bytes.

Com base nessa descrição podemos fazer uma análise da complexidade do algoritmo baseando-se na quantidade de bytes do arquivo. O algoritmo então tem complexidade O(n*log(n)), pois é realizada uma iteração completa em todos os bytes do arquivo e uma operação de custo log(n) na Heap mínima no passo 3 para um dos bytes;

3.2 Implementação do algoritmo Run Length Encoding

Para se realizar a compressão a unidade mínima de informação utilizada foram os bytes do arquivo, ou seja, o arquivo foi lido byte a byte e no caso do algoritmo de RLE (Run Length Encoding) foi necessário apenas realizar uma iteração sobre os bytes desse arquivo.

As etapas do algoritmo estão descritas a seguir:

1. Criação de vetor vazio para armazenar a tupla contendo a quantidade de vezes que um byte aparece e o literal do byte;
2. Leitura do arquivo byte a byte do arquivo;
3. Ao encontrar um byte diferente do lido anteriormente adição no vetor da quantidade de repetições do byte juntamente com o literal do byte.

Com base nessa descrição podemos fazer uma análise da complexidade do algoritmo baseando-se na quantidade de bytes do arquivo. O algoritmo então tem complexidade O(n), pois apenas uma iteração sobre o arquivo é necessária e apenas operações de tempo constante são realizadas durante a iteração.

4. Medição dos algoritmos

Para verificar a taxa e o tempo de execução, utilizamos 5 amostras de tomografias médicas[11] monocromáticas no formato DICOM. Ao comprimi-las com os algoritmos de Huffman e RLE, obtivemos os seguintes resultados:

Tabela 1 - Tempo e taxa de compressão da amostra.

	Tamanho	T_HUFF	Taxa_HUFF	T_RLE	Taxa_RLE
CT-MONO2-8-abdo	257KB	48ms	47,23%	2ms	70,06%
CT-MONO2-16-ankle	514KB	62ms	33,46%	9ms	199,83%
CT-MONO2-16-brain	514KB	112ms	68,51%	8ms	199,66%
CT-MONO2-16-chest	142KB	47ms	99,77%	2ms	198,32%
CT-MONO2-16-ort	514KB	104ms	63,59%	9ms	199,53%

De acordo com os resultados, pode-se verificar que o método de compressão RLE é significantemente mais rápido que o algoritmo de Huffman, o que já era esperado devido a sua menor complexidade, porém o arquivo final é também bem maior e, muitas vezes, maior até que o arquivo DICOM original, o que torna esse método inviável para comprimir esse tipo de arquivo. Em contrapartida, o algoritmo de Huffman obteve resultados satisfatórios para esse tipo de arquivo, não demorando mais que 0,2 segundos em nenhuma tentativa e comprimindo a maioria dos arquivos para resultados bem menores que o original.

Portanto, para compressão de arquivos DICOM, o algoritmo de Huffman apresenta resultados satisfatórios de compressão e tempo.

5. Trabalhos semelhantes

O primeiro estudo escolhido, intitulado “Compressão sem Perdas de Sinais Eletrocardiográficos”[8] também analisa métodos de compressão de dados sem perdas para o uso em telemedicina, porém focado nos exames de eletrocardiogramas, combinando três funções de descorrelação aos quatro algoritmos escolhidos pelo autor, dentre eles o de huffman. De fato, o melhor resultado foi obtido pela combinação da função de descorrelação denominada “preditor 2” com o algoritmo de huffman, alcançando uma razão de compressão média de 2,69:1, ou seja, uma taxa de compressão de cerca de 37% do tamanho original, sendo superior aos utilitários de compressão famosos. No estudo, concluiu-se que variações como ruídos dos sinais captados podem alterar substancialmente a eficiência da Compressão.

O segundo artigo, denominado “Compressão de Imagens Mamográficas Utilizando Segmentação e o Algoritmo PPM”[9] percorre compressões de mamografias pela necessidade de muitos exames para a população com o intuito preventivo. Os algoritmos de huffman e RLE foram analisados, porém não obtiveram o melhor desempenho (conquistado pelo padrão JPEG sem perdas), provavelmente por causa da escolha do autor em separar as imagens em

planos de bits. Com esse processo, o autor alcançou uma taxa de compressão média de 36,7% do tamanho inicial das imagens.

Por fim, a pesquisa de “Análise e Implementação de Algoritmos de Compressão de Dados”[10] foi escolhida por sua semelhança teórica, onde o autor discorre sobre o uso do algoritmo de huffman e propõe uma construção alternativa, baseada no original. Embora a aplicação prática seja mais genérica no estudo, assemelha-se ao presente trabalho pelo interesse na eficiência de algoritmos de compressão, já que o algoritmo alternativo proposto conseguiu alcançar uma taxa de compressão máxima de 13%, mas, pela estrutura proposta, perde eficiência conforme o tamanho dos dados aumenta, o que é um problema consideravelmente grave, já que o estudo de compressão tem como principal objeto de análise grandes quantidades de dados.

6. Conclusão

Entre todos os arquivos utilizados no estudo, o algoritmo de Huffman apresentou melhores resultados se comparado ao RLE, que inclusive aumentou o tamanho do arquivo na maioria dos casos. O RLE normalmente é recomendado para a compressão de imagens, mas as imagens contidas no DICOM já são comprimidas em JPEG 2000[13], o que reduz a eficiência da compressão RLE.

Pelos resultados obtidos com o algoritmo de Huffman, foi alcançada uma taxa de compressão média de cerca de 62,51%e o tempo de compressão apresenta uma possível correlação inversa com a eficiência do processo, já que os arquivos onde mais bits por segundo foram processados apresentaram melhor taxa de compressão. Uma maior exploração do funcionamento da estrutura dos arquivos DICOM pode ser um trabalho futuro que resulte em melhores compressões, assim como a possibilidade de enviar o arquivo comprimido via email pode ser implementado em próximas atualizações.

7. Bibliografia

[1] ”Resolução CFM n° 1643/2002”. Conselho Federal de Medicina, Santo André, 01 de Dezembro de 2019. Disponível em <https://sistemas.cfm.org.br/normas/visualizar/resolucoes/BR/2002/1643>. Acesso em:01/12/2019

[2] Robinson, A. H.; Cherry, C. (1967). "Results of a prototype television bandwidth compression scheme". *Proceedings of the IEEE*. IEEE.

[3] "Run Length Encoding Patents". Internet FAQ Consortium. Santo André, 21 de Março de 1996. Disponível em <http://www.ross.net/compression/patents_notes_from_ccfaq.html>. Acesso em: 14/07/2019.

[4] ”ME-06_AED-II-CompDados”. São Bernardo do Campo, 01 de Dezembro de 2019. Disponível em <https://docs.google.com/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxjYXJsb2tsZWJlcnxneDo0MWZlODExMjI2NjQ1NzFi>. Acesso em:01/12/2019

[5] "A Method for the Construction of Minimum-Redundancy Codes". *Proceedings of the IRE*. São Bernardo do Campo, 01 de Dezembro de 2019. Disponível em <http://compression.ru/download/articles/huff/huffman_1952_minimum-redundancy-codes.pdf>. Acesso em: 01/12/2019

[6] “Algoritmo de Huffman”. Universidade Fernando Pessoa. São Bernardo do Campo, 01 de Dezembro de 2019. Disponível em <http://multimedia.ufp.pt/codecs/compressao-sem-perdas/codificacao-estatistica/algoritmo-de-huffman>. Acesso em 01/12/2019

[7] “*Huffman Coding Technique for Image Compression*”. Compusoft. Santo André, 01 de Dezembro de 2019. Disponível em <http://docshare02.docshare.tips/files/31563/315639086.pdf>. Acesso em: 01/12/2019

[8] “Compressão sem Perdas de Sinais Eletrocardiográficos”. *Research Gate*. São Bernardo do Campo, 01 de Dezembro de 2019. Disponível em <https://www.researchgate.net/publication/265884416_Compressao_sem_Perdas_de_Sinais_Eletrocardiograficos>. Acesso em 01/12/2019

[9] “Compressão de Imagens Mamográficas Utilizando Segmentação e o Algoritmo PPM”. *Research Gate*. São Bernardo do Campo, 01 de Dezembro de 2019. Disponível em <https://www.researchgate.net/publication/237797052_Compressao_de_Imagens_Mamograficas_Utilizando_Segmentacao_e_o_Algoritmo_PPM>. Acesso em 01/12/2019

[10] “Análise e Implementação de Algoritmos de Compressão de Dados”. Revista e-Fatec. São Bernardo do Campo, 01 de Dezembro de 2019. Disponível em <http://revista.fatecgarca.edu.br/index.php/efatec/article/view/15>. Acesso em 01/12/2019

[11] “*Medical Image Samples*”. barre. Santo André, 01 de Dezembro de 2019. Disponível em <https://barre.dev/medical/samples/>. Acesso em 01/12/2019

[12] “*DICOM Standard*”. *DICOM Standard.* Santo André, 01 de Dezembro de 2019. Disponível em <https://www.dicomstandard.org/>. Acesso em 01/12/2019

[13] “*8.2.4 JPEG 2000 Image Compression*”. *DICOM.*. Santo André, 01 de Dezembro de 2019. Disponível em <http://dicom.nema.org/medical/dicom/2016c/output/chtml/part05/sect_8.2.4.html>. Acesso em 01/12/2019