Pirataria digital na sociedade

Artur Jaeger, Carlos Eduardo Cavalari, João Lucas M. de Oliveira, Miguel V. de Oliveira, Oruam Veiga, Vinicius Floriano

Departamento de Engenharia e Ciência da Computação Regional Integrada do Alto Uruguai e das Missões (URI) – Santo Ângelo, RS- Brasil

artur.jaeger13@gmail.com, caducavalari@gmail.com, joaolucasmartinello@hotmail.com, miguelvid.1909@gmail.com, veigaoruam@gmail.com, viniciusfloriano@aluno.santoangelo.uri.br

Resumo. .

1. Introdução a compressão de dados

Compressão de dados é usado basicamente em todos os lugares. Todas as imagens e vídeos transmitidos na internet são compactadas: em formato JPEG ou PNG para imagens, MP4 para vídeos, por exemplo; vídeos de canais de televisão digital, MPEG-2, enfim. A compressão tem o objetivo de armazenar informações de modo com que elas ocupem menos espaço, isto é, utilizem menos bits que a representação original. Isso é atingido por meio de técnicas que buscam reduzir o tamanho dos arquivos sem comprometimento significativo da qualidade, seja por eliminação de redundâncias ou representação mais eficiente dos dados.

A distinção dos algoritmos de compressão é feita em algoritmos sem perda de dados (*lossless algorithms),* os quais conseguem reconstruir informações sem nenhuma perda de dados da mensagem original, e algoritmos com perda de dados (*lossy algorithms*), os quais reconstroem uma aproximação da mensagem original [Blelloch, Guy E. 2013].

Este artigo irá explorar os tipos de compressão de dados e seus respectivos mecanismos de funcionamento, além de como elas são utilizadas para viabilizar o armazenamento de grades quantidades de dados. Iremos dar atenção especial para a compressão de imagens, aprofundando nas técnicas empregadas nesse contexto.

2. Algoritmos lossless

Como apresentados anteriormente, os algoritmos *lossless* tem como característica a restauração dos dados compressos exatamente como os originais. Esse tipo é necessário para diversas aplicações que exigem que o processo de compressão e descompressão seja livre de perdas de informação, como processos na geofísica, telemetria e imagens médicas digitais, citando alguns exemplos. Esses algoritmos podem ser descritos por um procedimento de dois estágios: descorrelação e codificação de entropia [Yang, Ming, Bourbakis, Nikolaos 2005].

O primeiro estágio, de descorrelação, remove redundância espacial por meio de técnicas preditivas, de transformação, e outros tipos de técnicas de descorrelação. O segundo estágio, codificação de entropia, o qual inclui métodos como codificação de Huffman, LZ77, tem como objetivo remover redundâncias de codificação. Esses termos serão explicados no tópico a seguir que tratará sobre o funcionamento do formato de imagem PNG.

2.1. Portable Network Graphics (PNG)

Portable Network Graphics, mais conhecido como PNG, é um formato de imagem que utiliza uma matriz de bits que especifica a cor de cada pixel em uma matriz retangular de pixels. Esse tipo de mapeamento também é conhecido como *bitmap*. A codificação de imagens PNG é passada, basicamente, por três processos: filtração, LZ77 e codificação de Huffman (PNG utiliza *DEFLATE,* algoritmo de compressão que utiliza esses dois últimos processos citados) [Aguilera, Paula 2006]. Para melhor entendimento, todas vezes que forem citadas, nos parágrafos conseguintes, transmissões de informações, dados, valores ou elementos, lembre-se que essas informações são os bits representando cada pixel de uma imagem.

Primeiramente, o algoritmo de filtração é aplicado antes da compressão propriamente, com o propósito de preparar os dados da imagem para uma compressão otimizada. Ele é separado em 5 tipos básicos de filtro, *None, Sub, Up, Average* e *Paeth,* os quais serão numerados de 0 a 4, respectivamente, a fim de melhor representação. O filtro 0 (*None)*, como diz o próprio nome, os dados são transmitidos não são modificados. O filtro *Sub (1)* transmite a diferença entre cada *byte* e o *byte* correspondente anterior. O segundo filtro, nomeado *Up* (2), funciona com o mesmo princípio do *Sub,* só que ao invés de utilizar o pixel à esquerda, é utilizado o pixel acima. O filtro *Average (3)* utiliza a média entre dois pixels vizinhos (à esquerda e acima) para prever o valor do pixel. Por último, o filtro *Paeth* realiza uma função linear de três pixels vizinhos, à esquerda, acima e esquerda-acima. É somado o acima e esquerda e subtraído do superior-esquerda e salvo numa variável. Após, São feitas três operações verificando a diferença entre a variável e o valor de cada pixel. É selecionado como preditor o pixel vizinho mínimo, isto é, o valor com menor diferença. Por último, subtrai-se o valor do pixel original com o da etapa anterior. [W3C 1996].

Após, o algoritmo LZ77, criado por Abraham Lempel e Jacob Ziv em 1977, é utilizado para analisar a entrada dos dados e determinar uma maneira de como reduzir informações redundantes e substitui-las por metadados. Ele se baseia na leitura sequencial das informações e, quando há uma ocorrência de uma informação com uma mesma informação e sequência já lida, está é substituída pela posição da primeira ocorrência dessa informação. Essa posição é lida como um dicionário, em que os valores são a distância entre a primeira ocorrência e a repetição, e o tamanho da sequência. É utilizada uma estrutura em vetor para computar esse algoritmo, a janela de procura (*sliding window)*, a qual representa todo o vetor sendo lido. Esse nome é dado pois essa janela vai "deslizando" para os valores seguintes, informação nova entra do lado direito e sai pelo esquerdo. A *Sliding window* é separada em duas, o *Look-ahead buffer,* o qual representa o que ainda está sendo processado e possivelmente referenciado, e o *Search buffer,* que armazena foi previamente processado. [Oliveira, Gaspar] [Microsoft 2023].

Por último, a codificação de Huffman é aplicada na imagem. O princípio dessa codificação está na Teoria de Informação de Shannon, a qual diz que, num elemento de informação representado por símbolos, alguns desses símbolos ocorrem mais vezes que outros [Enes, Diogo et. al]. Assim sendo, ao representar os símbolos com maiores repetições com códigos menores (usando menos bits) e os menos frequentes com códigos mais extensos, é obtido uma codificação binária menor, resultando numa diminuição do comprimento médio de cada código [Huffman, David, A. 1952]. Essa representação é feita pela chamada de árvore de Huffman (derivada da estrutura de dados árvore binária).

Os dois valores menos frequentes são pegos e conectados em um nó com a sua soma, e esse processo é realizado recursivamente para todos os valores. Toda vez que vamos para a esquerda da arvore, é adicionado 0, e para a direita, 1.

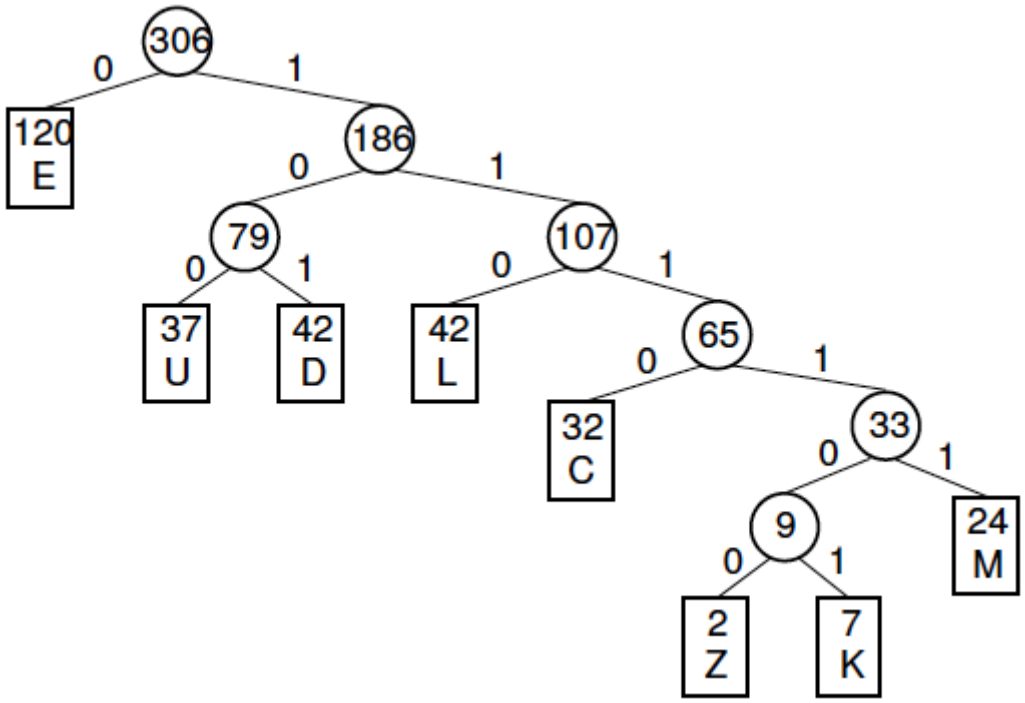

Figura 1. Representação de uma árvore de Huffman.

3. Algoritmos lossy

A compressão com perda de dados, também chamada de *lossy,* envolve perda significativa de informação, dependendo de quanto de redução de tamanho se quer obter. Em retorno por aceitar a distorção da informação na reconstrução, é possível obter proporções de compressão muito maiores que algoritmos *lossless* [Sayood, Khalid 2018]. Ao deletar informações não essenciais da fonte original, é possível salvar muito mais espaço.

Diversos métodos são aplicados para a compressão de informações com alguma perda, como quantizações vetoriais e escalares, as quais são responsáveis por reduzir a precisão dos dados para torna-los mais compressos, e codificações de transformação, em caso de imagens, algoritmos como a Transformada Discreta do Cosseno transforma sinais de domínio espacial para domínios de frequência é particularmente eficaz em compactar energia em menos coeficientes, permitindo taxas de compressão maiores [Marcellin, Michael, W. et. al 2002] [Kurmi, Roopesh, K, Gupta, Sumit 2017].

3.1. JPEG

Joint Photographic Experts Group, vulgo JPEG, batizado em homenagem ao grupo que o desenvolveu, é um formato de imagem que se baseia na eliminação de informações desnecessárias, priorizando o desempenho (pouca memória necessária) mantendo uma qualidade visual relativamente alta. O processo de codificação de imagens JPEG é feito através de 7 processos principais: transformação de RGB para YCbCr, divisão da imagem original em blocos de pixels 8x8, mudança do alcance das variáveis dos pixels, aplicação da Transformada Discreta do Cosseno (DCT), quantização de cada bloco, codificar por

entropia a matriz quantizada e por fim reconstruir a imagem usando o processo reverso [Abu zaher, Mazen & Al Azzeh, Jamil 2017].

O processo de transformação do espaço RGB para YCbCr inicia apartir do espaço dimensional RGB, um modelo aditivo, no qual as cores são formadas pela combinação dos componentes de vermelho, verde e azul. É o espaço de cores mais comumente usado em dispositivos eletrônicos, como monitores e câmeras digitais. Cada pixel em uma imagem RGB possui três componentes, representando a intensidade de cada cor primária [KUMAR, R. Vijaya et al 2016].

No entanto, o espaço de cores YCbCr é um modelo de cores subtrativo, ele separa a informação de luminância (Y), que representa o brilho ou a intensidade da imagem, dos componentes de crominância (Cb e Cr), que representam as diferenças de cor em relação à luminância.

A conversão de RGB para YCbCr é feita por meio de cálculos matemáticos. A partir dos valores de intensidade de cada componente RGB, são calculados os valores de luminância. Os cálculos presentes na figura 2 levam em consideração as diferentes contribuições dos componentes RGB para a percepção humana de brilho e cor [Abu zaher, Mazen & Al Azzeh, Jamil 2017].

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.144 \\ -0.16875 & -0.33126 & 0.5 \\ 0.5 & -0.41869 & -0.08131 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Figura 2. Fórmulas – Coeficientes lineares de transformação RGB para YCbCr .Fonte: AEU 2011.

A separação dos componentes de cor do componente de luminosidade no espaço de cores YCbCr é vantajosa para a compressão de imagens, pois as informações de cor são armazenadas com uma resolução menor do que a informação de brilho, e como a percepção humana é mais sensível às variações de luminosidade do que às variações de coloração, é possível reduzir a resolução dos componentes de crominância sem uma perda perceptível de qualidade na imagem [ROCHA, Eduardo 2020][Yang, Y., Yuhua, P., & Zhaoguang, L. 2007].

Ademais, o DCT (Discrete Cosine Transform), assim como outras transformações, separa as informações de imagem em outro tipo de dado, no caso, como frequências, a partir de métodos matemáticos e em seguida transforma em coeficientes que são tratados de forma independente, sem perder eficiência na compressão [Khayam, S. A. 2003]. Cada bloco de pixels 8x8 é convertido em informações de frequência, através de coeficientes que representam diferentes comprimentos de onda, para cada bloco, a DCT calcula 64 coeficientes obtidos através adição das bases de DCT, que são as representações de ondas que através de somas e transformações lineares, podem formar todas as combinações de pixels do bloco.

Outro detalhe relevante é de que o pixel superior esquerdo é chamado de DC, cujo é o mais importante da matriz, pois é responsável por definir a hierarquia de frequências, partindo da menor para a maior, assim como é a unidade de compressão de energia, que se trata de uma das propriedades fundamentais para garantir que a maior parte das frequências que compõe a imagem permaneçam baixas, possibilitando o descarte de

informação de frequências altas, reduzindo a utilização de bits [Stathaki Tania, Naylor Patrick A (2017)][Fouzi Douak, Redha Benzid, Nabil Benoudjit 2011].

Após obter a matriz transformada do processo de DCT, resta a etapa de quantização, em que os coeficientes de DCT são reduzidos em magnitude para permitir uma maior compactação dos dados. Ppara isso, uma tabela de quantização é utilizada para dividir os valores da matriz, para obter uma terceira matriz, a qual tem seus valores aproximados ao número inteiro mais próximo, nesse processo, propositalmente são frequentemente perdidos dados nas linhas e colunas de frequências mais altas, que representam um impacto menor em imagens ,como fotografias, e consequentemente diminuindo o número de pixels utilizados [R. M. Gray and D. L. Neuhoff (1998)].

A quantidade de redução depende dos valores da tabela de quantização escolhidos, que depende da configuração de qualidade definida, quanto maior a qualidade, menor serão os valores da tabela de quantização, logo, haverá mais informações de frequências mais altas, diminuindo a compressão, no caso contrário, o mesmo ocorre. De modo geral, cada formato de arquivo e/ou codificador possui uma tabela própria para transformar a matriz, que está incluída no cabeçalho do arquivo compactado.

A ultima etapa da compressão de imagens JPEG é realizar a compressão dos valores quantizados pela DCT. Isso é feito a partir de um procedimento de três processos [Roberts, Eric 2009]. Primeiro, realiza-se a conversão do coeficiente DC para um valor relativo: esse coeficiente é mudado de um valor absoluto para um relativo (ao coeficiente do bloco 8x8 anterior). Já que blocos adjacentes em uma imagem mostrar um alto grau de correlação, codificar o coeficiente de dado bloco como a diferença com o coeficiente anterior tipicamente produz um numero pequeno, o qual pode ser armazenado com menos bits.

Segundamente, os blocos que surgem da DCT são reordenados em formato zig-zag*:* como vários coeficientes são truncados a valores zerados durante o processo de quantização, reordenar esses coeficientes numa sequência em zig-zag prepara as informações para o último processo de modo mais eficiente.

Por fim, o mecanismo de codificação por entropia é aplicado. Ele combina os princípios da codificação de Huffman com codificação *Run-length,* processo que comprime informação quando existe uma sequência longa de caracteres repetidos. O resultado consiste numa sequência de 3 *tokens*, repetindo até que o bloco esteja completo. Esses tokens são o *run-length (*número de zeros consecutivos que precedem o atual elemento não zerado na matriz de saída da DCT)*;* a contagem de bits (número de bits usados para codificar a amplitude do valor que segue, conforme determina pelo processo realizado pela codificação de Huffman) e a amplitude do coeficiente da DCT [Silva, Gonçalo et. al].

Referências

Blelloch, Guy E. (2013). Introduction to Data Compression. Carnegie Mellon University. P 3-4.

Yang, Ming, Bourbakis, Nikolaos (2005). An Overview of Lossless Digital Image Compression Techniques. 48th Midwest Symposium on Circuits and Systems. P 1-2.

Aguilera, Paula (2006). Comparison of different image compression formats. ECE 533 Project Report. Wisconsin College of Engineering. P 2-3.

World Wide Web Consortium, W3C (1996). PNG (Portable Network Graphics) Specification.

Oliveira, Gaspar. LZ77. Trabalho desenvolvido no âmbito da disciplina Multimédia II. Universidade Fernando Pessoa.

Microsoft (2023). LZ77 Compression Algorithm.

Enes, Diogo, Domingues, Filipe, Alão, Tiago, M. Código de Huffman. Trabalho desenvolvido no âmbito da disciplina Multimédia II. Universidade Fernando Pessoa.

Huffman, David, A. (1952). A method for the Construction of Minimum-Redundacy Codes. Proceedings of the IRE, 40(9). P 1098-1101.

Sayood, Khalid (2018). Introduction to Data Compression. Morgan Kaufmann (5th edition). Elsevier Inc.

Marcellin, Michael, W. et. al (2002). An overview of quantization in JPEG 2000. Signal processing: image communication 17. P – 73-74.

Kurmi, Roopesh, K, Gupta, Sumit (2017). A Review of Lossless and Lossy Based Image Compression Techniques. International Journal of Engineering and Advanced Technology. Volume-6 Issue-6. P 1-2.

Rocha, Eduardo (2020). Olhos reconhecem a luz por causa do trabalho dos cones e bastonetes. Jornal da USP. Disponível em: https://jornal.usp.br/?p=367404. Acesso em: 19 de jun de 2023.

Kumar, R. Vijaya et. al (2016). Gray Level to RGB Using YcbCr Color Space Technique. International Journal of Computer Applications, v. 975, p. 8887.

Yang, Y., Yuhua, P., & Zhaoguang, L (2007). A fast algorithm for YCbCr to RGB conversion. IEEE Transactions on Consumer Electronics, 53(4), 1490-1493.

Kerr, Douglas A (2012). Chrominance subsampling in digital images. The Pumpkin, v. 2012, n. 3, p. 1-15.

Roberts, Eric (2009). Lossless Compression of Quantized Values. CS10SC. Intellectual Excitement of Computer Sciente. Stanford University.

Silva, Gonçalo, Galocha, Pedro, Alves, João. Run-length Encoding. Trabalho desenvolvido no âmbito da disciplina Multimédia II. Universidade Fernando Pessoa.

Khayam, S. A. (2003). The discrete cosine transform (DCT): theory and application. Universidade Estadua do Michigan, 114(1), 31.

Fouzi Douak, Redha Benzid, Nabil Benoudjit (2011). Color image compression algorithm based on the DCT transform combined to an adaptive block scanning. AEU - International Journal of Electronics and Communications. Volume 65, pg 16-26.

Stathaki Tania, Naylor Patrick A (2017). Discrete Cossine Transform. Digital Signal Processing and Digital Filters (E4-13, EE9SC1), pg 48 – 53.

R. M. Gray and D. L. Neuhoff (1998). Quantization. IEEE Transactions on Information Theory, vol. 44, no. 6, pp. 2325-2383.