Compressão de dados

Prof. André Backes | @progdescomplicada

O que é compressão de dados?

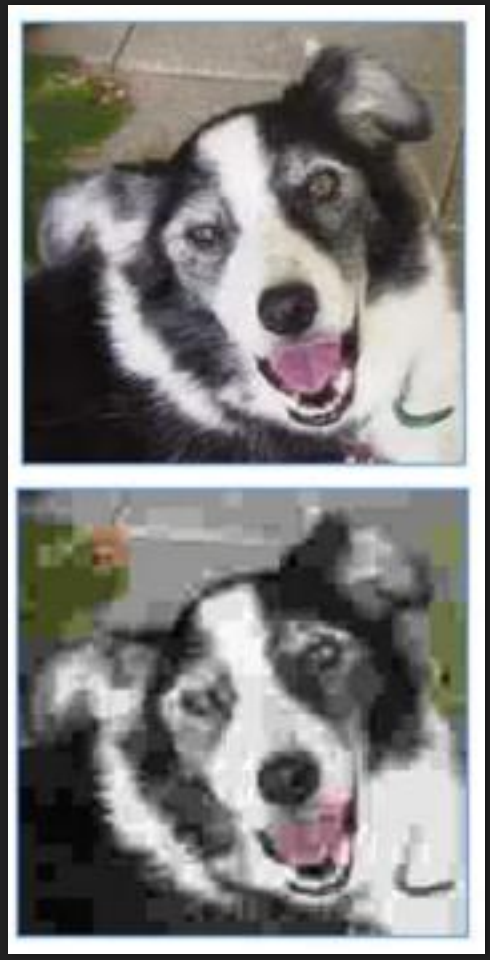
- Conjunto de métodos que visam reduzir o tamanho dos arquivos e economizar espaço de armazenamento
  - Envolve a codificação da informação
- Benefícios
  - Transferência mais rápida pela internet: diminui o tempo de acesso, permite transmissão com rede com largura de banda menor (economia de rede)
  - Economia de recursos de espaço de armazenamento (HD)
  - Processamento sequencial mais rápido

O que é compressão de dados?

- A variedade de técnicas é enorme
  - Algumas técnicas são gerais, e outras específicas para certos tipos de dados, como voz, imagem ou texto
- Técnicas reversíveis vs. irreversíveis
  - A compressão pode ser **com** ou **sem perda**

Compressão com Perdas

- Durante o processo de compressão, o método elimina detalhes menos perceptíveis para reduzir o tamanho do arquivo
- Exemplos de uso
  - Compressão de imagens (JPEG) e áudio (MP3)
- Vantagens
  - Maior taxa de compressão
  - Economia de espaço
  - Adequado para dados menos sensíveis à perda de qualidade



Compressão sem Perdas

- Reduz o tamanho do arquivo sem perder informações, permitindo a recuperação exata dos dados originais
- Exemplos de uso
  - Compressão de documentos (ZIP) e imagens (PNG)
- Vantagens
  - Preserva a qualidade dos dados, ideal para dados críticos e sensíveis à perda de informações

Taxa de compressão

- É a razão entre o tamanho final dos dados comprimidos e o tamanho sem compressão
- Podemos interpretar de duas formas
  - Redução de x% no tamanho do arquivo

$$taxa\ de\ compressão = \frac{(tamanho\ original) - (tamanho\ comprimido)}{(tamanho\ original)} = \frac{100\ - 30}{100} = 0{,}7\ ou\ 70\%$$

  - Redução do arquivo a x% do tamanho original

$$taxa\ de\ compressão = \frac{(tamanho\ comprimido)}{(tamanho\ original)} = \frac{30}{100} = 0{,}3\ ou\ 30\%$$

Compressão sem Perdas

Requisitos desejados

- Manipular o arquivo compactado (busca, processamento, etc.) sem que seja necessário descompactar todo o arquivo e/ou a partir do início
- Manipular o arquivo compactado não deve ser mais lento do que manipular o arquivo original
  - A vantagem de processar um volume menor de informações deve ser maior do que o custo computacional de codificação/decodificação eventualmente necessários

Métodos fundamentais

- Redução de redundâncias
  - Notação diferenciada: atribui uma notação diferente para dados repetitivos
  - Omissão de sequências repetidas: identificação e remoção de padrões repetitivos nos dados para economizar espaço
- Código de Huffman
  - Atribui códigos de tamanho variável a cada símbolo com base em suas frequências
  - Alta eficiência de compressão

Notação diferenciada

- Atribui uma notação mais compacta para dados repetitivos
- Exemplo
  - Para armazenar o campo UF, são necessários 2 bytes (ASCII)
  - Brasil: 26 estados + distrito federal = 27 UFs
  - É possível codificar todas as UFs usando apenas 5 bits!

Notação diferenciada

- Se trabalharmos com valores binários, 5 bits permitem codificar 32 valores
- Ao invés de armazenar a UF (ASCII), podemos usar apenas 1 byte para codificar estados
  - Economia de 50% (no mínimo)

UF	Código
RS	00001
SC	00010
PR	00011
SP	00100
MG	00101
RJ	00110
ES	00111
MT	01000
MS	01001

Notação diferenciada | Problemas

- Isso torna arquivo ilegível quando aberto por editor de textos simples
- Possui um custo computacional para codificação/decodificação
  - Consultas na tabela
- Aumenta complexidade dos programas

UF	Código
RS	00001
SC	00010
PR	00011
SP	00100
MG	00101
RJ	00110
ES	00111
MT	01000
MS	01001

Omissão de sequências repetidas

- Busca identificar e remover padrões repetitivos nos dados para economizar espaço
- Exemplo: como podemos reduzir o tamanho da sequência abaixo, formada exclusivamente por letras do alfabeto (A a Z)?

BBBEAAAAFFHHHHHCBMMALLLCDDBBBBBBBCC

- Podemos armazenar o número de repetições ao invés das repetições

Omissão de sequências repetidas

- Arquivo original

BBBEAAAAFFHHHHHCBMMALLLCDDBBBBBBBCC

- Arquivo compactado

3B1E4A2F5H1C1B2M1A3L1C2D7B2C

- Dá para melhorar um pouco
  - Se uma letra não se repete, não há necessidade de armazenar o número de repetições dela

Omissão de sequências repetidas

- Arquivo original

BBBEAAAAFFHHHHHCBMMALLLCDDBBBBBBBCC

- Arquivo compactado (versão 1)

3B1E4A2F5H1C1B2M1A3L1C2D7B2C

- Arquivo compactado (versão 2)

3BE4A2F5HCB2MA3LC2D7B2C

Omissão de sequências repetidas

- Essa abordagem depende do alfabeto de entrada
  - Conjunto de símbolos usados
- Dá para ser mais genérico: considera um valor de byte especial para indicar a repetição de uma sequência
  - codificação de comprimento-de-carreira (*run-length encoding*)

Omissão de sequências repetidas

- Codificação
  - Compressão de imagem com cada pixel representado por 8 bits
    - Valores variam de 0 a 255
  - Ler pixels em sequência, copiando os valores para o arquivo, exceto quando o valor do pixel se repete
  - Se o valor do pixel se repete, substitua o valor por
    - O indicador do código *run-length*
    - O valor do pixel
    - O número de repetições (até 256)

Omissão de sequências repetidas

- Para a sequência hexadecimal

22 23 24 24 24 24 24 24 24 25 26 26 26 26 26 26 25 24

- Usando 0xFF como código indicador de repetição (código de *run-length*)

22 23 FF 24 07 25 FF 26 06 25 24

Omissão de sequências repetidas

- Decodificação
  - Percorremos o arquivo compactado gerando um outro arquivo conforme o original, com os mesmos símbolos
  - Replicamos os símbolos repetidos, quando for o caso, conforme indicado pelo trio

FF 24 07  **24 24 24 24 24 24 24**

Omissão de sequências repetidas

- Pesquisa no Arquivo Compactado
  - Percorremos o arquivo compactado em busca dos mesmos símbolos
  - O número de ocorrências, caso maior que 1, será indicado pelo trio

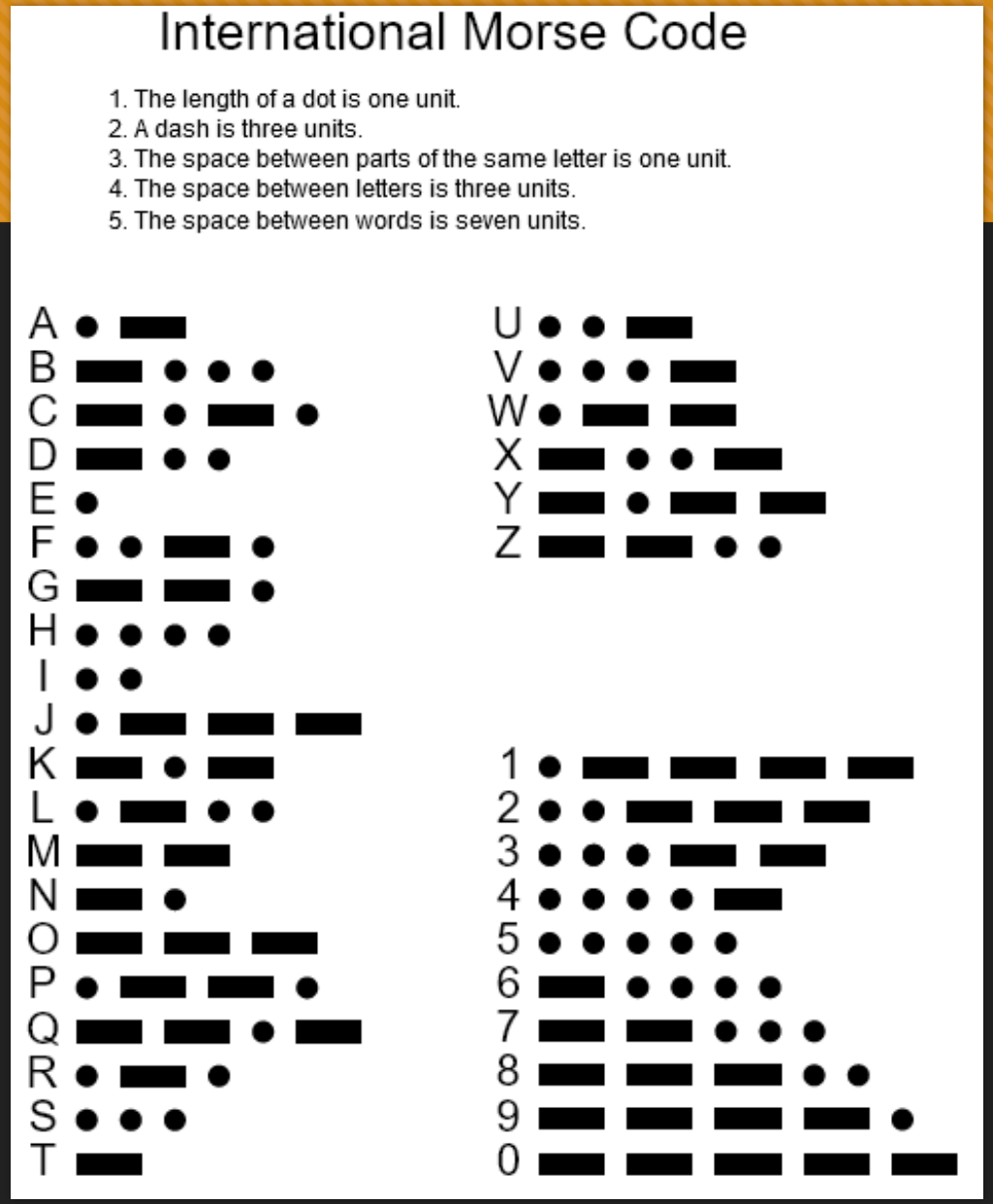
FF 24 07

Omissão de sequências repetidas | Aplicações

- Pode ser usado em aplicações que possuem dados esparsos ou com muita repetição
  - Esta técnica é mais promissora na compressão de imagens pois imagens apresentam maiores áreas contínuas de uma mesma cor
    - Imagens de céu
  - Desenhos e outras imagens com número limitados de cores tendem a ser melhor comprimíveis usando esta técnica
- Nem sempre garante redução de espaço
  - Não pode ser usado em imagens com muita variação, por exemplo

Problema

- A maioria dos dados não possui distribuição de frequência previsível
- Usar o mesmo esquema para representar repetições pode não ser vantajoso em algumas situações
- Uma solução é utilizar um código de tamanho variável
  - Código Morse: dot (.) ou dash (-)
  - Tabela que associa símbolos e valores



Problema

- Em um texto não compactado, um caractere é representado por um byte (ASCII)
- Todo caractere é representado pelo mesmo número de bits

Caractere	Decimal	Binário
A	65	01000000
B	66	01000001
C	67	01000010
...		
X	88	01011000
Y	89	01011001
Z	90	01011010

Problema

- No entanto, alguns caracteres ocorrem com mais frequência em um texto do que outros

- Frequência de alguns caracteres do texto ao lado
  - ' ': 13
  - 'o': 9
  - 'e': 8
  - 'i': 2
  - 'f': 1

Código de Huffman

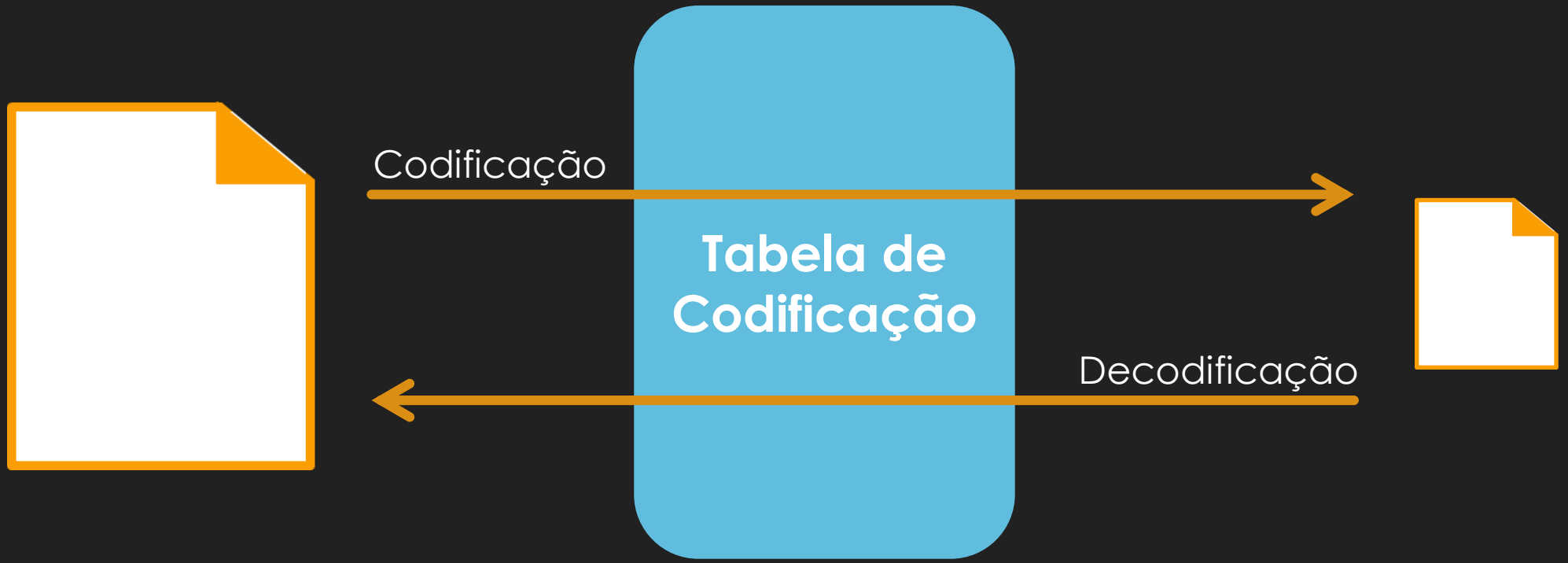
- É um exemplo de código de tamanho variável
- Ideia: valores mais frequentes são associados a códigos menores
  - Se s símbolos que ocorrem com mais frequência têm códigos menores, as cadeias tendem a ficar mais curtas
  - Requer informação sobre a frequência de ocorrência de cada símbolo a ser codificado
- Muito usado para codificar texto

Código de Huffman | Funcionamento

- Ideia geral: reduzir o número de bits que representam os caracteres mais frequentes
  - Usar caracteres (símbolos) com número variável de bits
  - Dada uma mensagem, encontrar a melhor (menor) codificação para cada caractere
  - Construir a Árvore de Prefixos de Huffman
    - Maior frequência: códigos pequenos
    - Menor frequência: códigos grandes

Código de Huffman | Funcionamento

- Codificação e decodificação



Código de Huffman | Funcionamento

- Árvore de Prefixos de Huffman
  - Nós folhas representas os símbolos
  - O código de Huffman constrói uma árvore binária, onde um percurso representa o código para um símbolo
  - A cada iteração, as duas árvores com menores frequências são combinadas e a soma das frequências é associada à raiz
  - Assim, um código único e de tamanho variável é atribuído a cada símbolo diferente
  - **Nenhum código pode ser o prefixo de qualquer outro código**
    - Sem ambiguidades, código canônico

Código de Huffman | Passo a passo

- Considere um texto contendo apenas os símbolos: A B C D E
  - Calcular a frequência de cada símbolo
  - Cada símbolo é um nó folha
- Passo 1: unir os dois símbolos de menos frequência
  - D e E

Símbolo	Contagem
A	39
B	21
C	19
D	12
E	9

Código de Huffman | Passo a passo

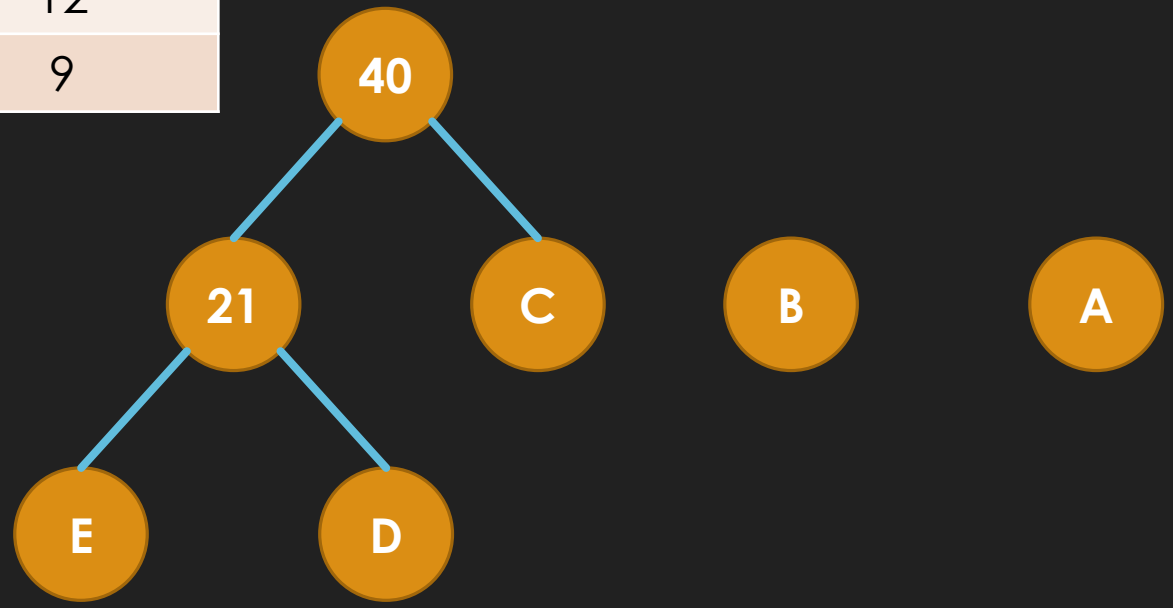
Símbolo	Contagem
A	39
B	21
C	19
D	12
E	9

- Passo 2: unir os dois símbolos (ou árvores) de menos frequência
  - C com (D e E)
  - Outra opção seria unir B e C

Código de Huffman | Passo a passo

Símbolo	Contagem
A	39
B	21
C	19
D	12
E	9

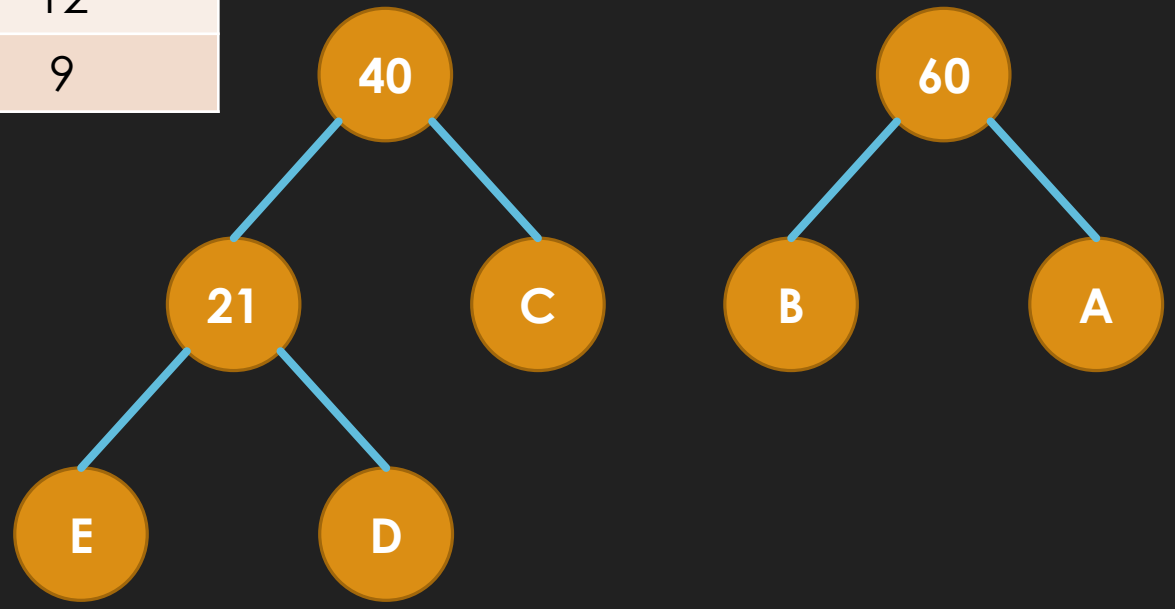
- Passo 3: unir os dois símbolos (ou árvores) de menos frequência
  - B e A



Código de Huffman | Passo a passo

- Passo 4: unir os dois símbolos (ou árvores) de menos frequência
  - (C e D e E) com (B e A)

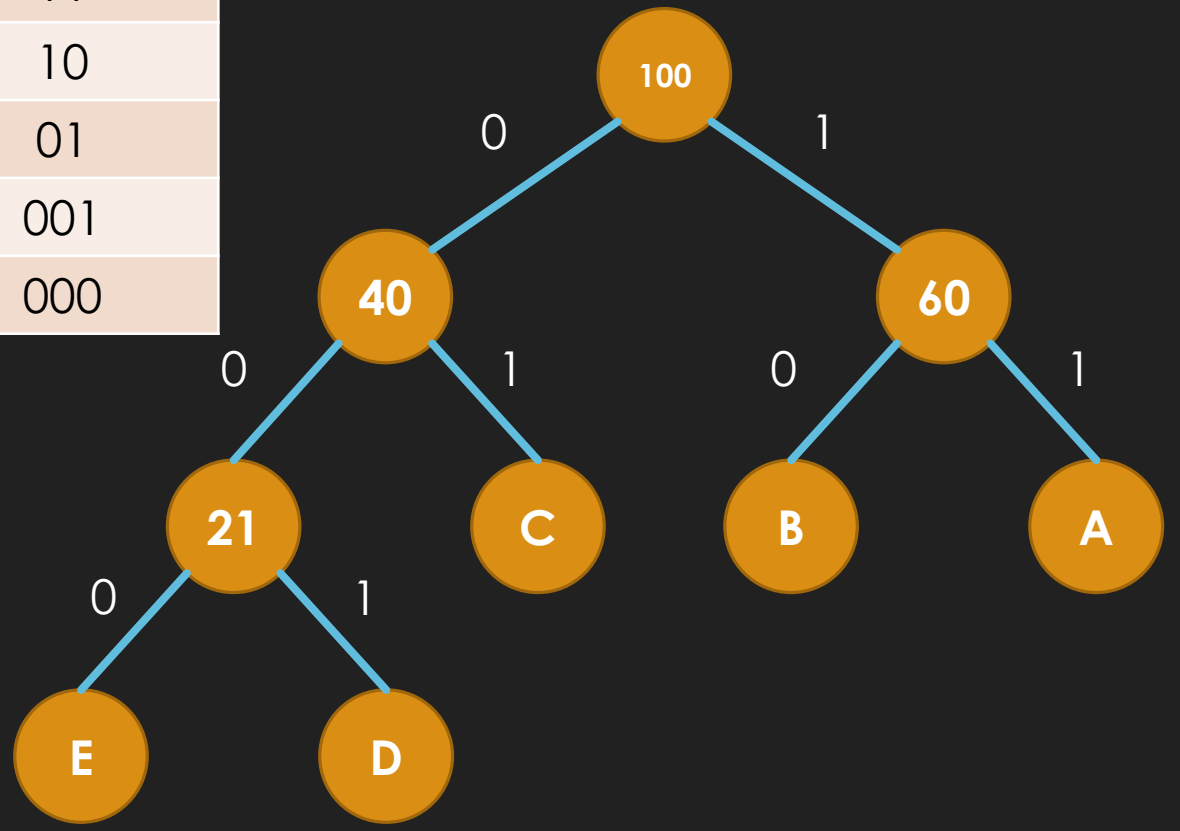
Símbolo	Contagem
A	39
B	21
C	19
D	12
E	9



Código de Huffman | Passo a passo

- Fim do processo: não existem mais nós folhas soltos
- Atribuir um código para cada nó folha de acordo com o percurso
  - Esquerda: 0
  - Direita: 1

Símbolo	Código
A	11
B	10
C	01
D	001
E	000



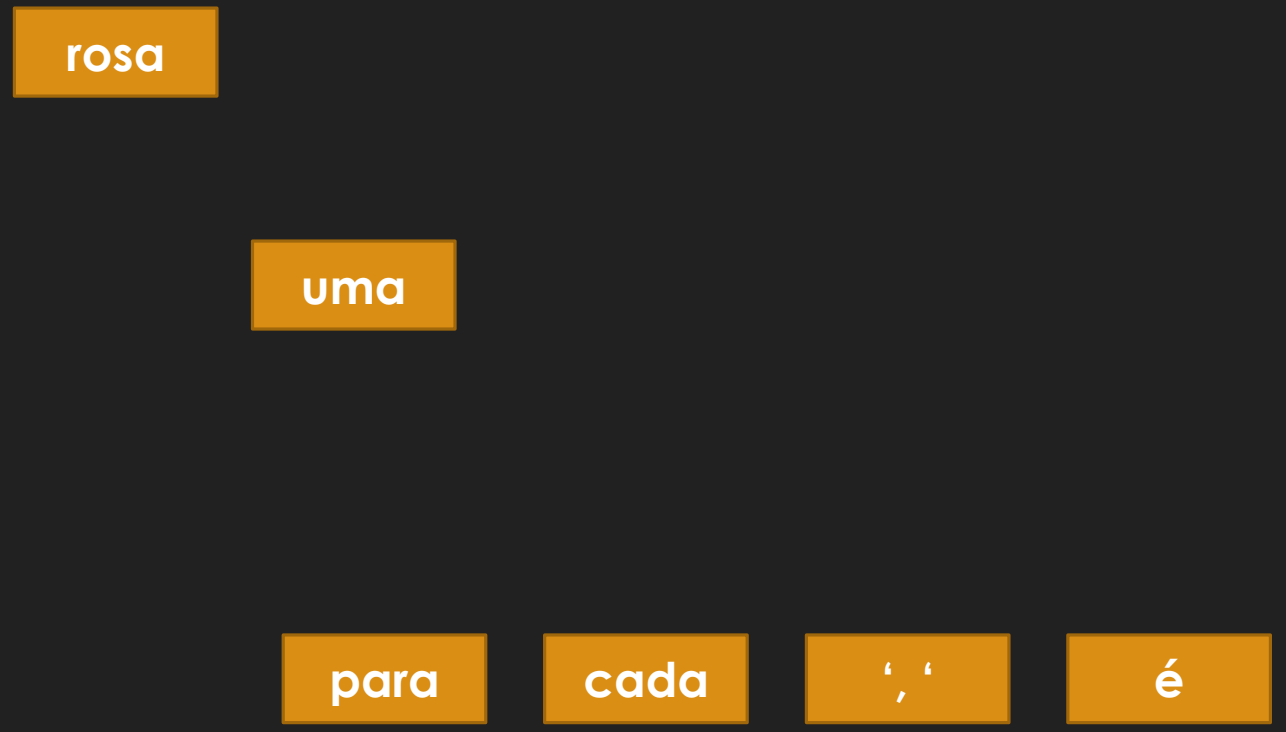
Código de Huffman

- A compressão com o código de Huffman pode ser feita
  - Caractere por caractere: compressão pode chegar a 60% do tamanho original
  - Palavra por palavra: compressão pode chegar a 25% do tamanho original
    - O princípio é o mesmo
  - É o método que produz melhores resultados no processamento de linguagem natural

Código de Huffman | Passo a passo

Símbolo	Contagem
para	1
cada	1
rosa	4
‘, ’	1
uma	2
é	1

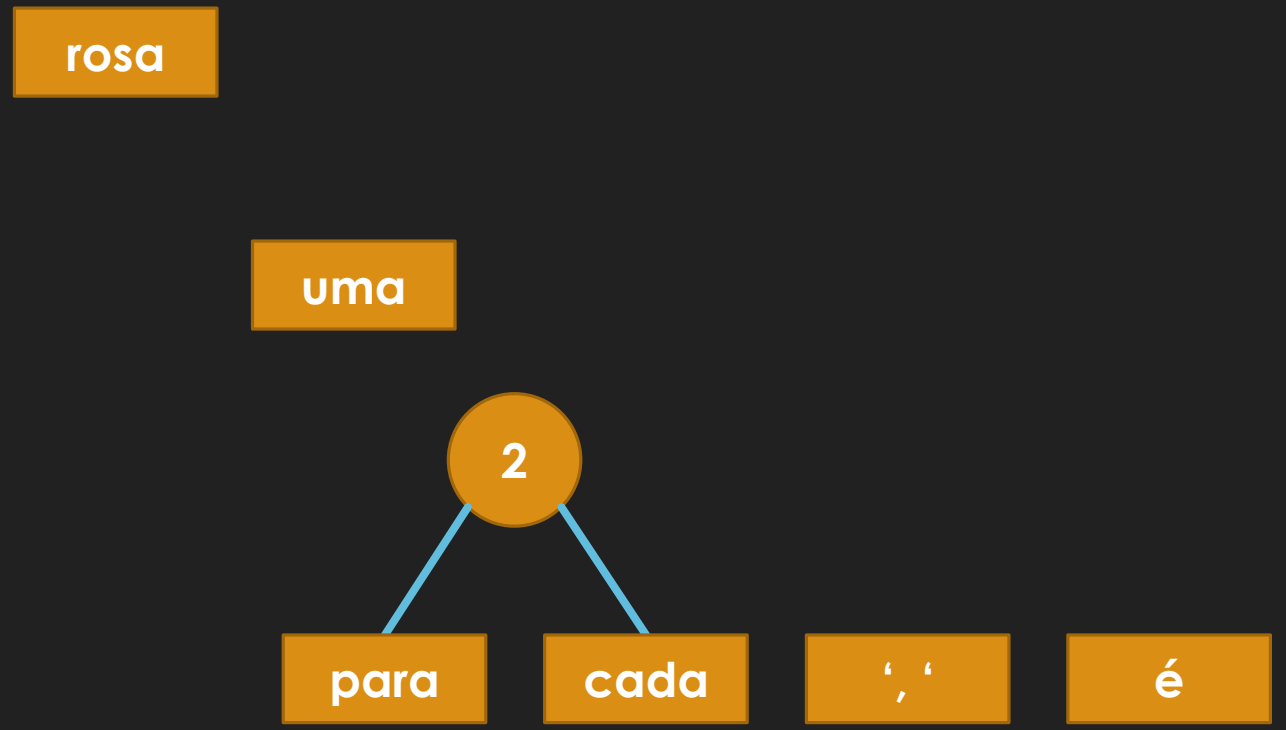
- “Para cada rosa rosa, uma rosa é uma rosa”
- Passo 1: unir “para” e “cada”



Código de Huffman | Passo a passo

Símbolo	Contagem
para	1
cada	1
rosa	4
‘, ’	1
uma	2
é	1

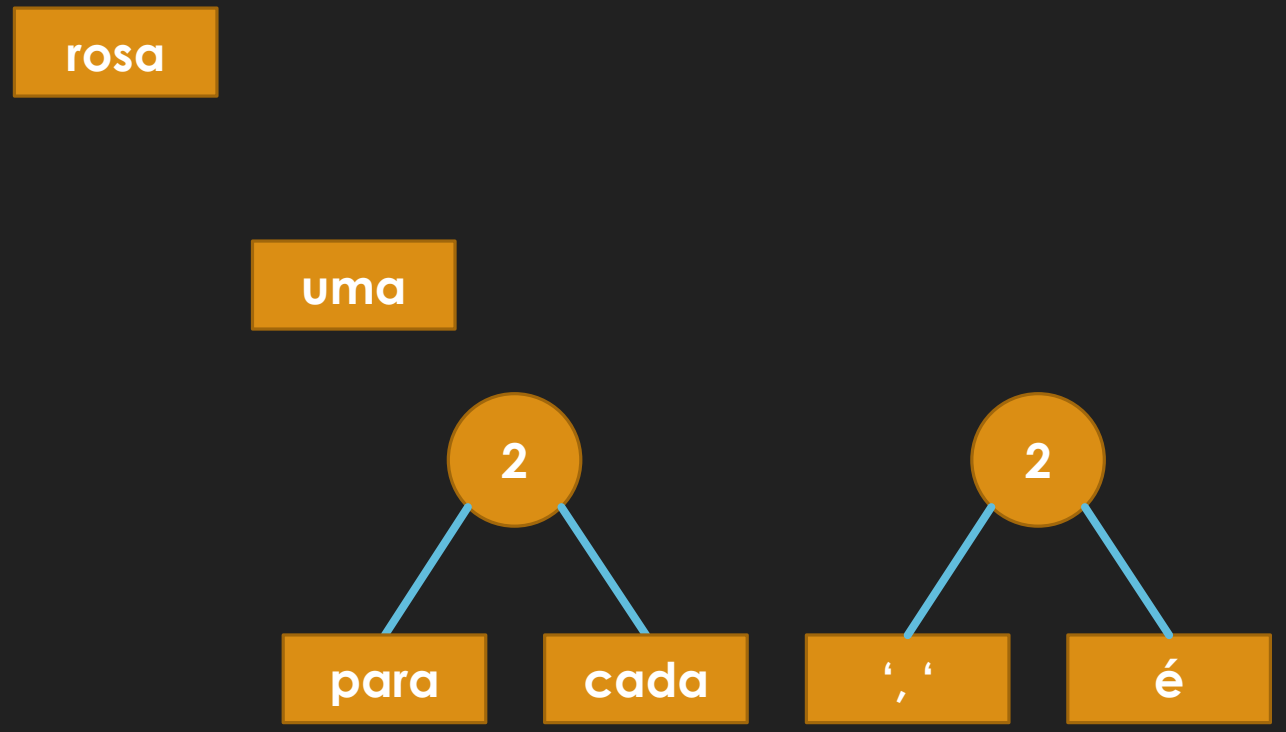
- Passo 2: unir “,” e “é”



Código de Huffman | Passo a passo

Símbolo	Contagem
para	1
cada	1
rosa	4
‘, ’	1
uma	2
é	1

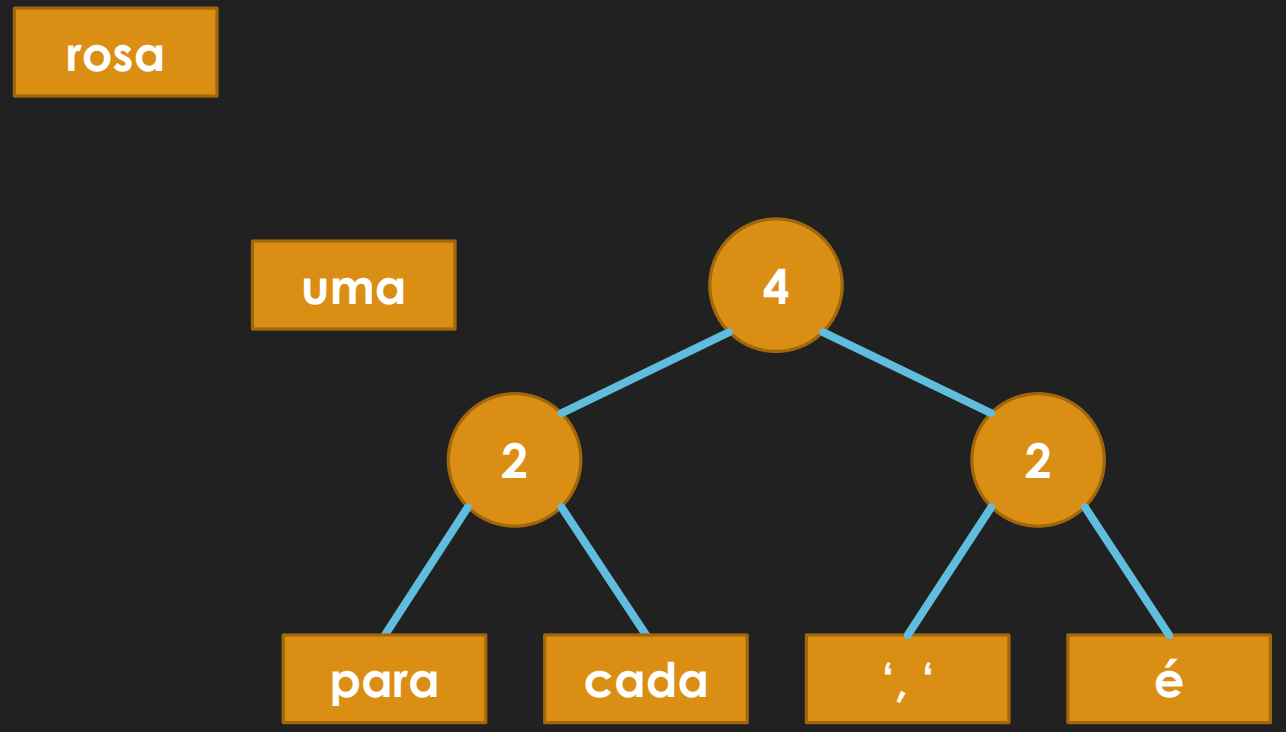
- Passo 3: unir (“para” e “cada”) e (“, “ e “é”)



Código de Huffman | Passo a passo

Símbolo	Contagem
para	1
cada	1
rosa	4
‘, ’	1
uma	2
é	1

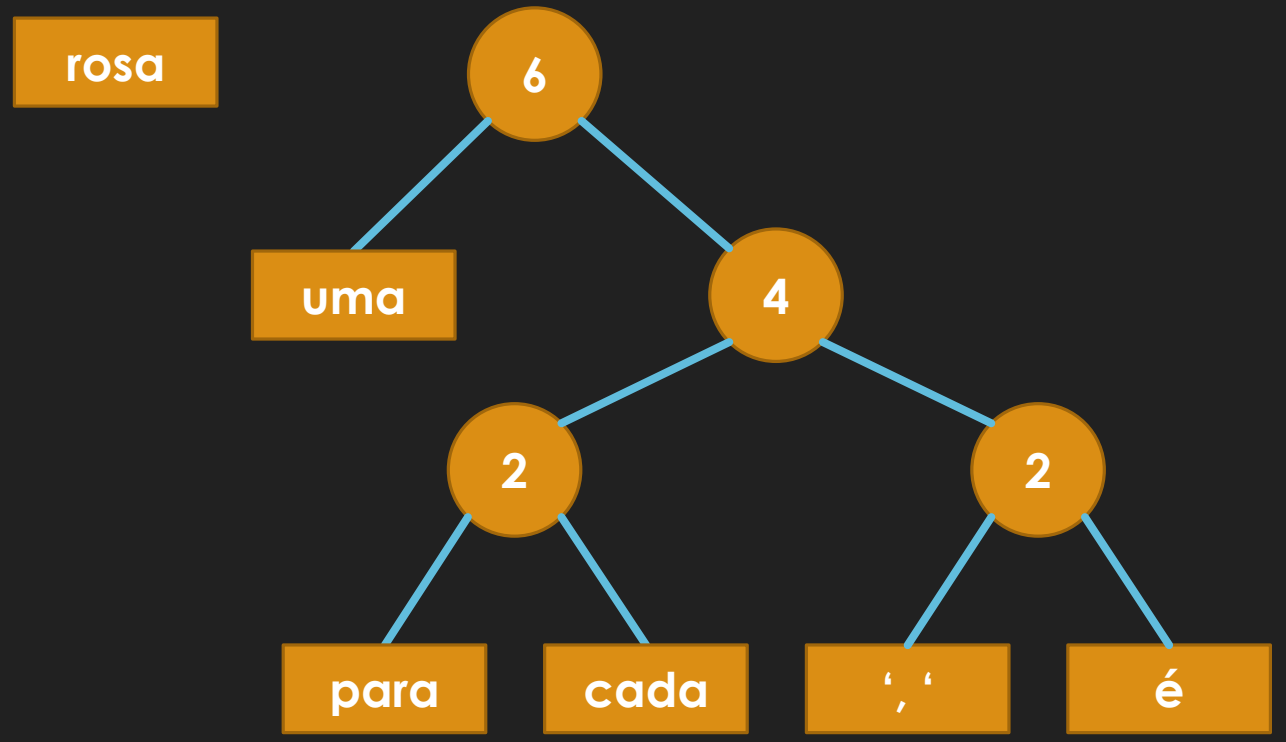
- Passo 4: unir a árvore “4” com “uma”



Código de Huffman | Passo a passo

Símbolo	Contagem
para	1
cada	1
rosa	4
‘, ’	1
uma	2
é	1

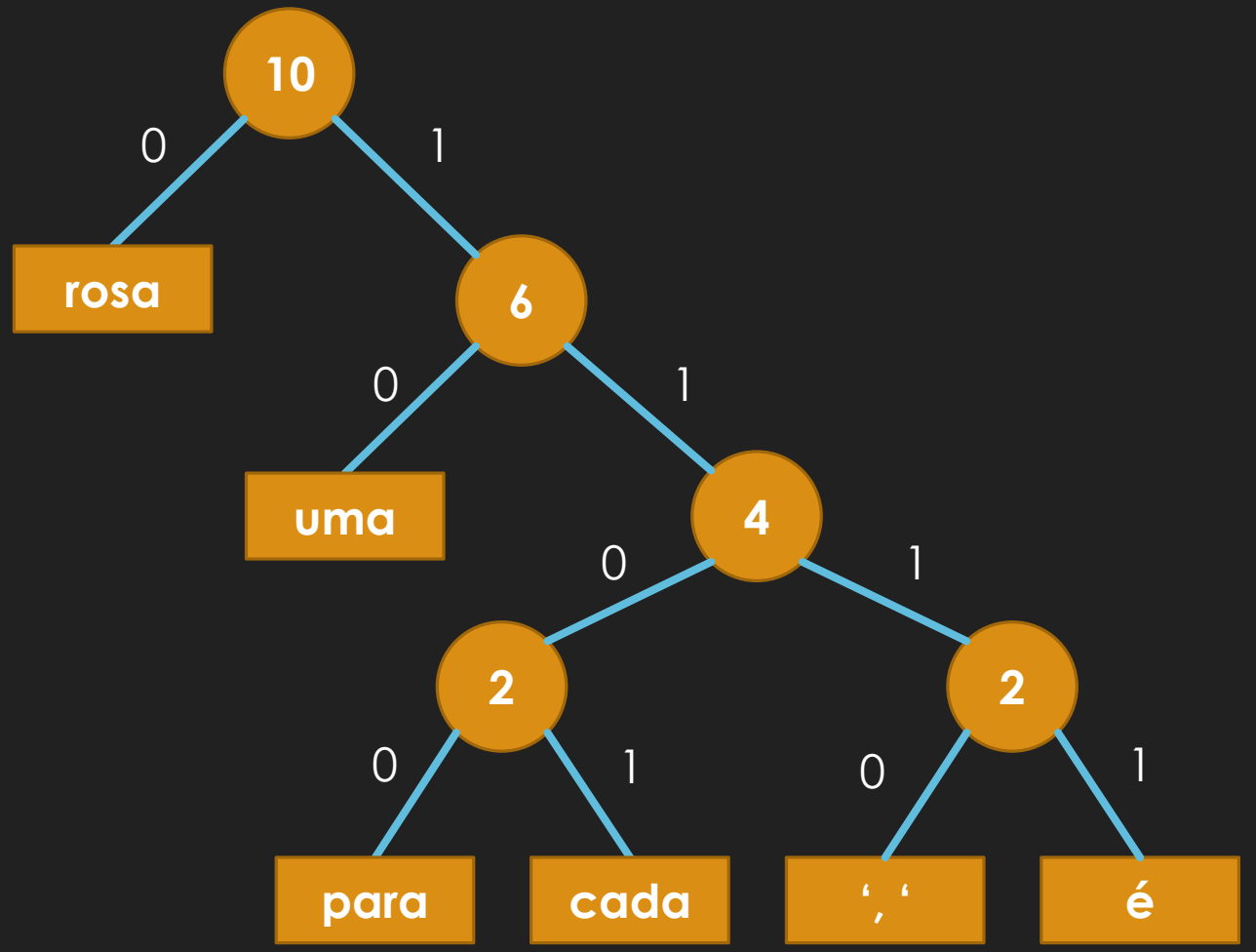
- Passo 5: unir a árvore “6” com “rosa”



Código de Huffman | Passo a passo

Símbolo	Código
para	1100
cada	1101
rosa	0
‘, ’	1110
uma	10
é	1111

- Fim do processo
- Gerar o código de acordo com o percurso



Código de Huffman

- Arquivo compactado: dados + dicionário
  - Texto compactado não precisa de separadores
  - ”para cada rosa rosa, uma rosa é uma rosa”: 40 bytes
  - 110011010011101001111100: 24 bits
  - Resultado: 3 bytes + dicionário
  - Dicionário: 18 bytes + separadores

Símbolo	Código
rosa	0
uma	10
para	1100
cada	1101
‘, ‘	1110
é	1111

dicionário

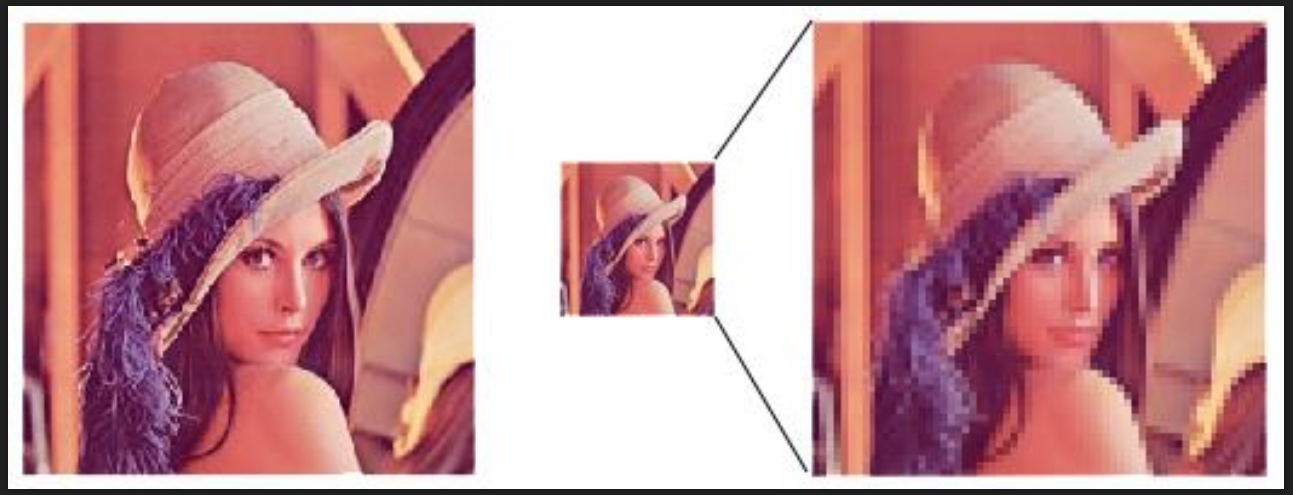
Compressão com Perdas

Definição

- As técnicas apresentadas anteriormente preservam todos os dados originais
  - É possível recuperar o arquivo original a partir do compactado
  - A compactação se dá pelo fato de haver redundância nos dados
- Outro tipo de compressão, denominada irreversível, baseia-se na suposição de que parte dos dados podem ser sacrificados (perdidos)
  - Muito usado em imagens, vídeos e áudio

Exemplo de compressão com perdas

- Comprimir uma imagem de 400 x 400 pixels para uma imagem de 100 x 100 pixels
  - Cada 4 x 4 pixels (16 pixels) será representado por 1 pixel
  - Imagem de 100 x 100 pixels é muito semelhante a imagem de 400 x 400 pixels
  - Entretanto, a partir dela, pode não ser possível obter a imagem original



Tipos de imagens

- Geradas por computador (Gráficos)
  - Armazenadas (e transmitidas) como um conjunto de instruções que geram a imagem
    - Pontos em um gráfico, equação de reta, etc.
  - Um esquema de compressão sem perdas tem de ser utilizado

- Imagens digitalizadas (Fotos escaneadas, etc.)
  - Armazenadas em formato matricial (pixels)
  - Em geral, usam complexos algoritmos de processamento de imagens
  - A codificação é feita por transformadas, diferenças e por repetição de série (*run-length*)

Padrões de compressão de imagem

Formato	Sistema de Cor	Compressão
GIF	RGB com tabela de até 256 cores	LZW
TIFF	RGB*, CMYK,YCbCr, Lab, Luv	RLE, LZW, JPEG, JBIG, Huffman ou nenhuma e outros
JPEG	RGB, YCbCr, CMYK	DCT , Huffman
PCX	RGB*	RLE
BMP	A BGR*	RLE ou nenhuma
TGA	RGB*	RLE ou nenhuma
PNG, MNG	RGBA (alfa em 256 tons)	LZ77 + Huffman = deflate
JPEG2000	RGB, YCbCr	DWT (wavelets) ou nenhuma

GIF: *Graphics Interchange Format*

- Cada pixel é codificado com um elemento de uma tabela de cores (8 bits) ao invés de 24, com uma compressão 3:1
- A Tabela de Cores pode ser
  - Global: é utilizada na imagem inteira
  - Local: é utilizada apenas em parte da imagem
- A codificação LZW pode ser utilizada para obter maior compressão

PNG: *Portable Network Graphics*

- Surgiu em 1996 como substituto para o formato GIF
- Permite comprimir as imagens sem perda de qualidade com muitas cores
- Usa uma variação do algoritmo Lempel-Ziv 77 e também a compressão Huffman, depois da compressão LZ, numa forma denominado LZH ou de Deflate/Inflate
- Assim como o GIF, permite armazenar animações
  - MNG: *Multiple Image Network Graphics*

JPEG: *Joint Photographic Experts Group*

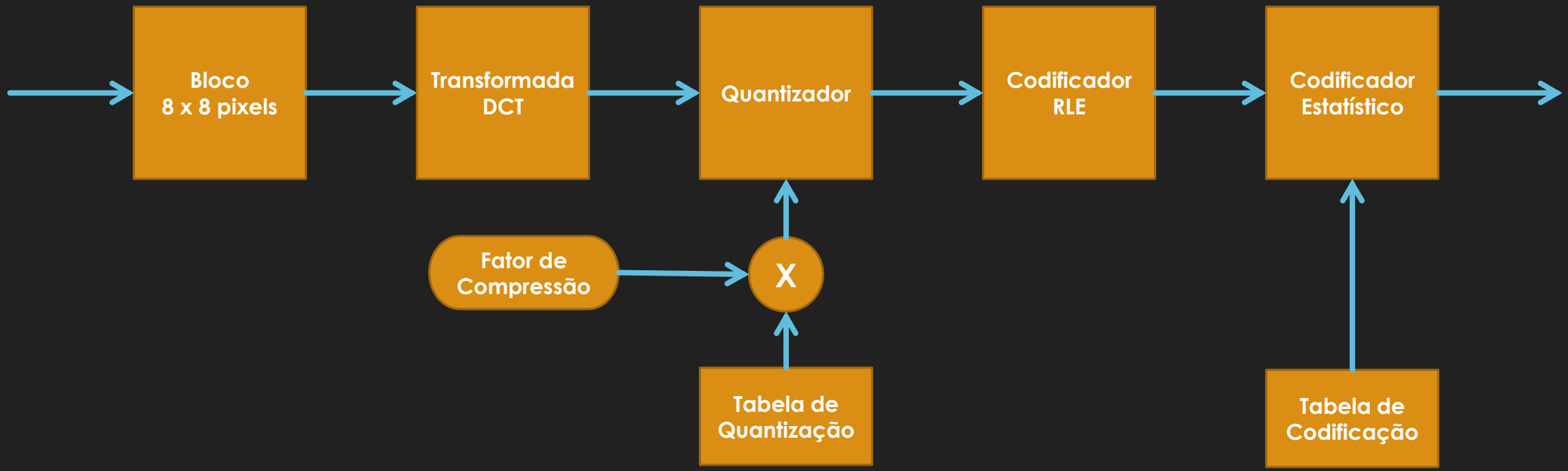
- Permite gravar imagens sem usar tabelas de cores usando toda a informação de Tons de Cinza ou RGB
  - Permite o uso de até 16 milhões de cores
  - O tamanho dos seus arquivos de imagens costuma ser bem pequeno
- Usa a compressão DCT (e depois Huffman)
- A compressão é realizada em três passos
  - Computação
  - Quantização
  - Atribuição do código de tamanho variável

JPEG: *Joint Photographic Experts Group*

- O processo de compactação JPEG é composto das seguintes fases
  - A imagem é divida em blocos de 8 x 8 pixels e em cada um destes blocos é calculada a DCT (*discrete cossine transform*)
    - Isso permite uma execução mais eficiente da DCT
  - Os coeficientes gerados pela DCT são quantizados
    - Processo de discretização da imagem
    - Permite eliminar os coeficientes que carregam menos informação
  - Na última etapa a codificação de Huffman é aplicada aos coeficientes quantizados

JPEG: *Joint Photographic Experts Group*

- Passo a passo



JPEG | quantização e seleção

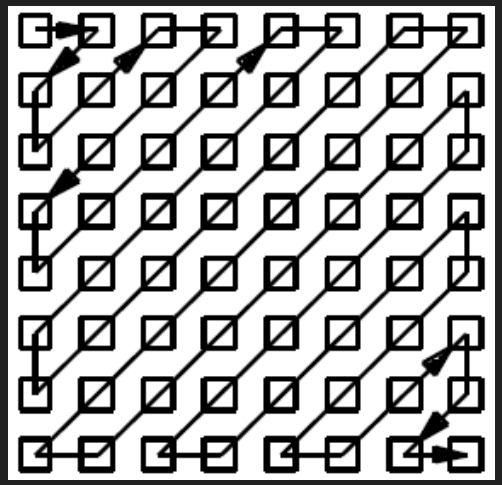
- Após a quantização, a tendência é de que a maior parte dos coeficientes seja zero
  - Concentrados no canto superior esquerdo
- Os coeficientes quantizados são reordenados por um padrão em zigue-zague
  - Agrupa os coeficientes de maior importância no início da sequência
  - 26 coeficientes: [-26 -3 1 -3 -2 -6 2 -4 1 -4 1 1 5 0 2 0 0 -1 2 0 0 0 0 0 -1 -1 EOB]

-415	-29	-62	25	55	-20	-1	3
7	-21	-62	9	11	-7	-6	6
-46	8	77	-25	-30	10	7	-5
-50	13	35	-15	-9	6	0	3
11	-8	-13	-2	-1	1	-4	1
-10	1	3	-3	-1	0	2	-1
-4	-1	2	-1	2	-3	1	-2
-1	-1	-1	-2	-1	-1	0	-1

Coeficientes da DCT

-26	-3	-6	2	2	0	0	0
1	-2	-4	0	0	0	0	0
-3	1	5	-1	-1	0	0	0
-4	1	2	-1	0	0	0	0
1	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

Coeficientes Quantizados



Padrão zigue-zague

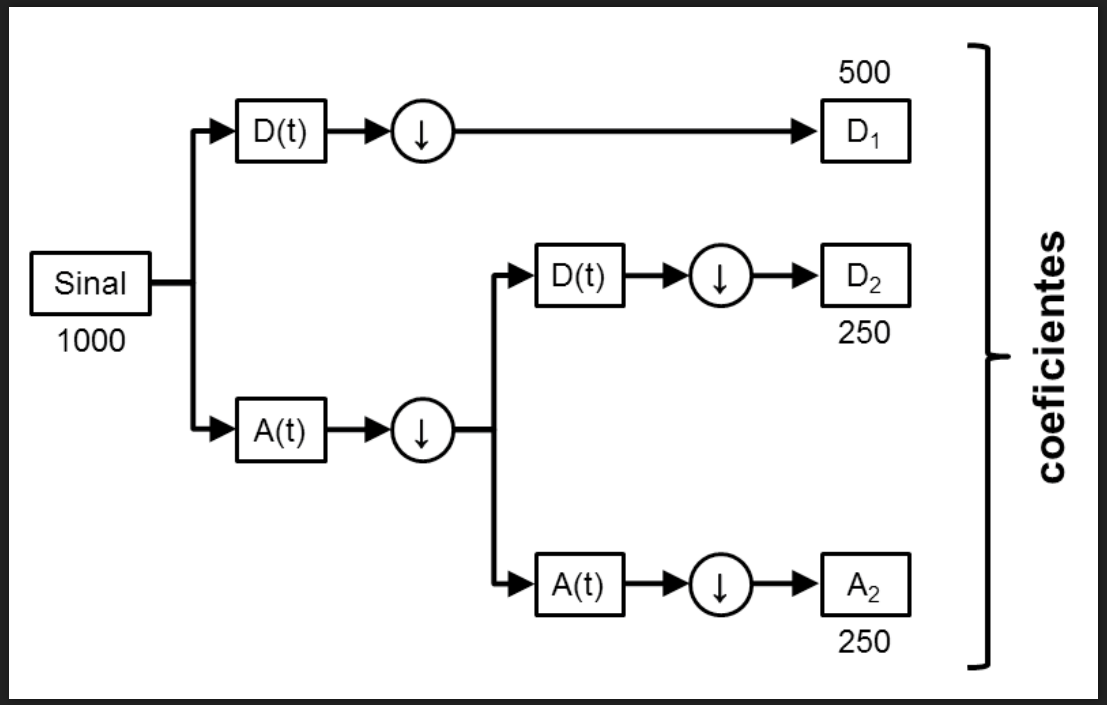
JPEG | blocagem

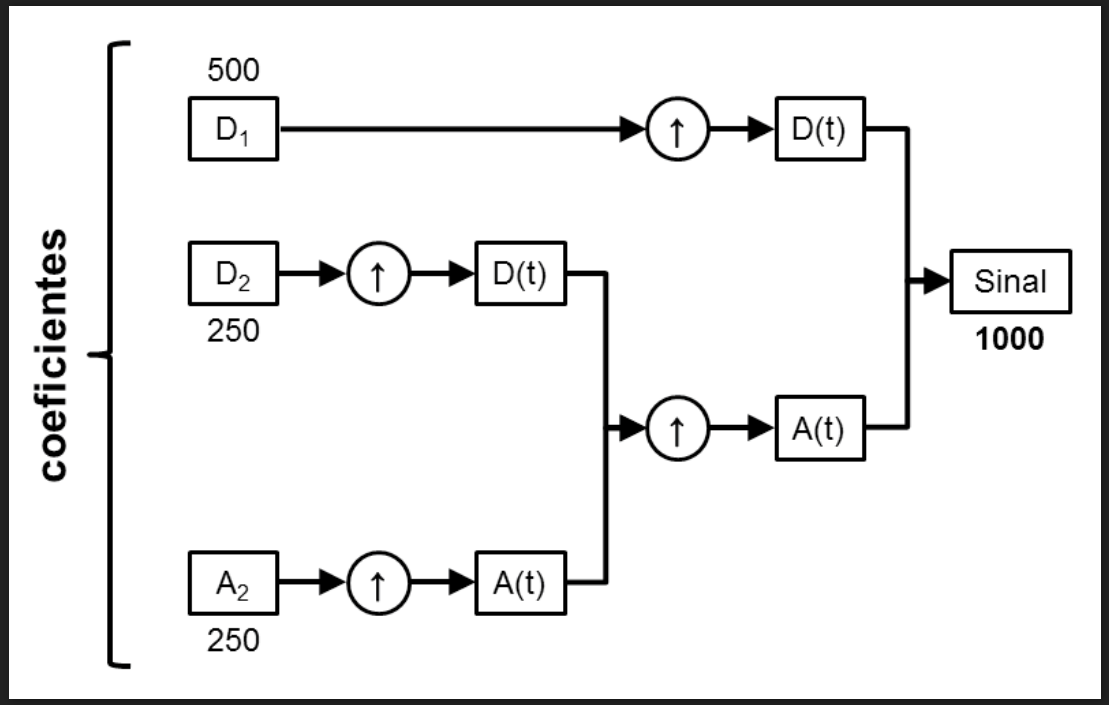
- Ocorre quando o fator de compressão é muito elevado
  - Gera um efeito de descontinuidade nos contornos dos blocos
  - Causado pelo fato de que o tratamento dado a cada bloco é independente dos demais
  - Pode ser reduzida pela aplicação de filtros na imagem reconstruída, ou pela superposição parcial dos blocos durante a codificação
    - Não faz parte do padrão JPEG

JPEG 2000

- Substitui a transformada DCT pela transformada *Wavelet* para efetuar a decomposição espectral da imagem
  - As *Wavelets* são funções matemáticas que descrevem padrões em sinais e imagens, permitindo análises detalhadas de diferentes escalas
  - Utilizam decomposição hierárquica em potência de 2, sempre em conjuntos de aproximações e detalhes
    - Cada nova decomposição diminui o conjunto de dados pela metade
  - Permite análise em diferentes escalas e níveis de detalhes

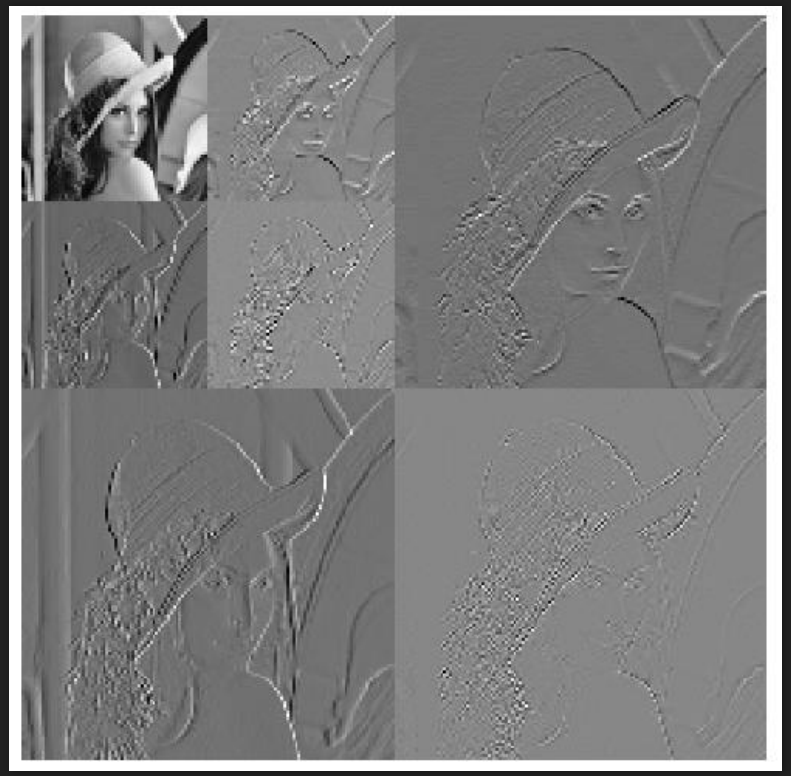
Decomposição *wavelet*





Decomposição *wavelet* de uma imagem

- Aplicada nas linhas e colunas da imagem
- Cada nível de decomposição gera
  - 1 aproximação (aspecto geral)
  - 3 níveis de detalhes (horizontal, vertical, diagonal)



Decomposição *wavelet* de uma imagem

- Podemos descartar parte dos coeficientes de detalhes e ainda assim ter uma boa representação da imagem
  - Na prática, O JPEG 2000 é bem mais complexo. Envolve quantização, etc.

Compressão de vídeo

- Os dados de vídeo (e áudio) não compactados são enormes
  - Em HDTV, a taxa de bits excede facilmente 1 Gbps
    - 1920 x 1080 a 30 quadros por segundo, 8 bits por canal YCbCr (PAL) = 1,5 Gbps
    - Grandes problemas para armazenamento e comunicações de rede
- Compressão é feita com perda
  - A taxa de compactação dos métodos sem perdas não é alta o suficiente para compactação de imagem e vídeo

Compressão de vídeo | Princípio Básico

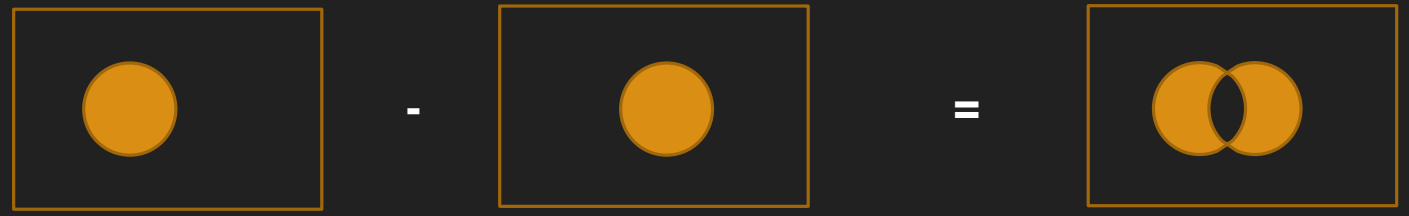
- Explora o fato de que os frames adjacentes são semelhantes
  - Um vídeo é uma "fila" de imagens dispostas pelo tempo
  - Normalmente os elementos em duas imagens sequenciais não mudam rapidamente
  - Elementos da câmera permanecem constantes (distância focal, posição, ângulo de visão, etc.)
  - Quanto maior a redundância temporal, melhor a codificação

Compressão de vídeo | Princípio Básico

- Remoção de redundância espacial
  - Codificação intraframe (JPEG)
- Remoção de redundância temporal
  - Maior compressão usando a coerência temporal ao longo do tempo
  - Essencialmente, considera a diferença entre os frames
- Remoção de redundância espacial e temporal
  - Codificação intraquadro e interquadro (H.261, MPEG)
- Na prática, é muito mais complexos

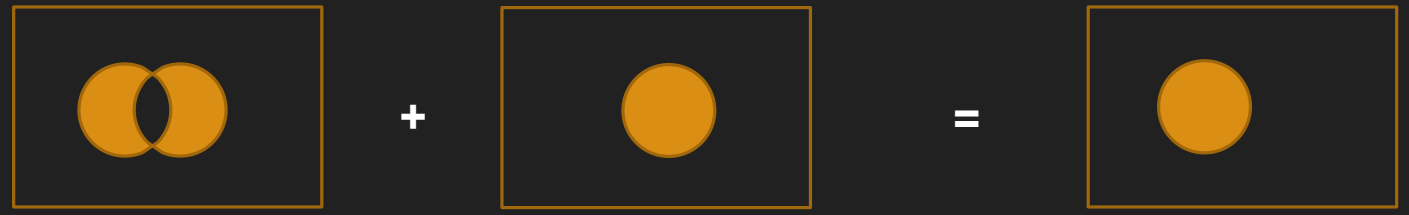
Cálculo da diferença entre frames

Encoder



frame atual - frame anterior = diferença

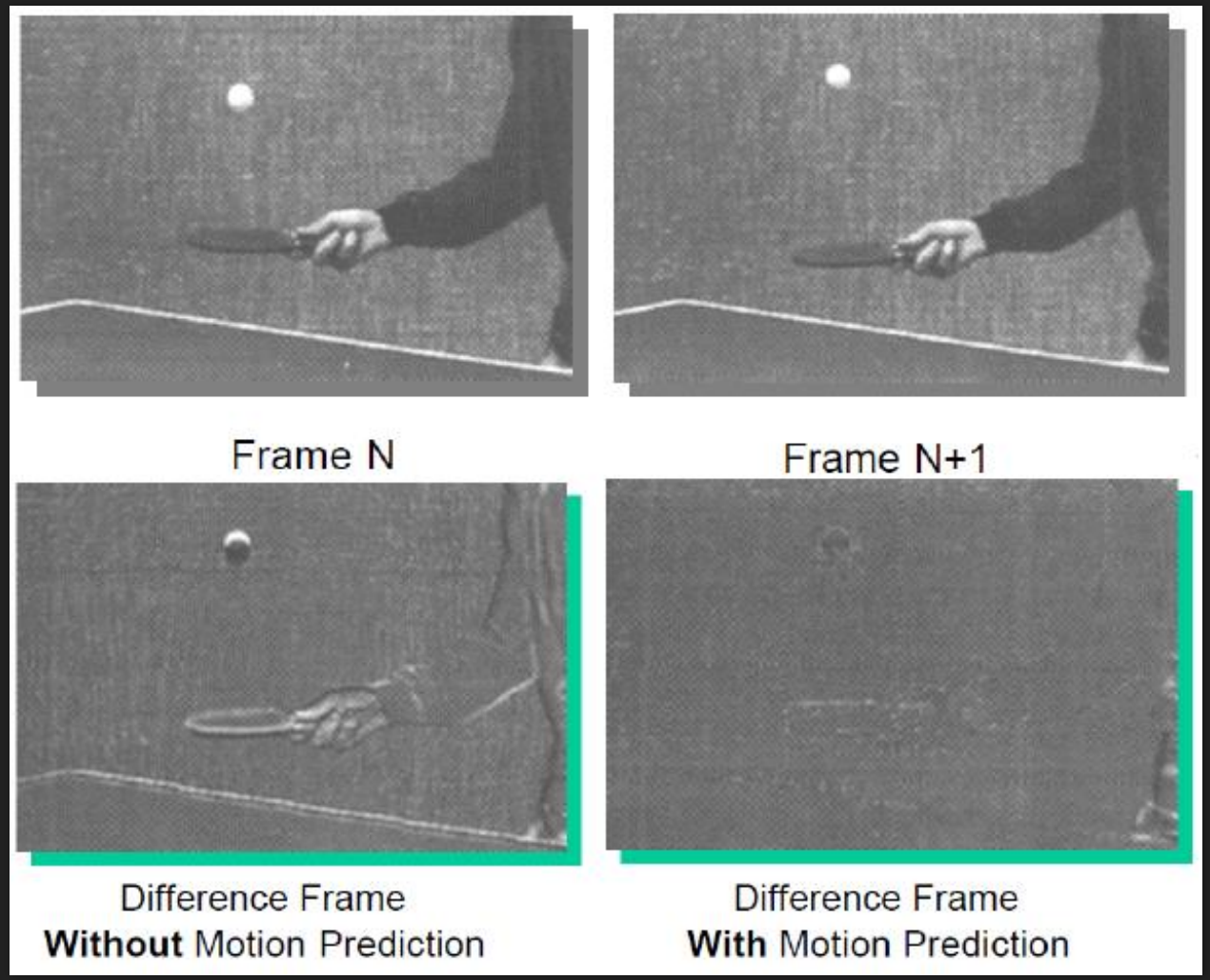
Decoder



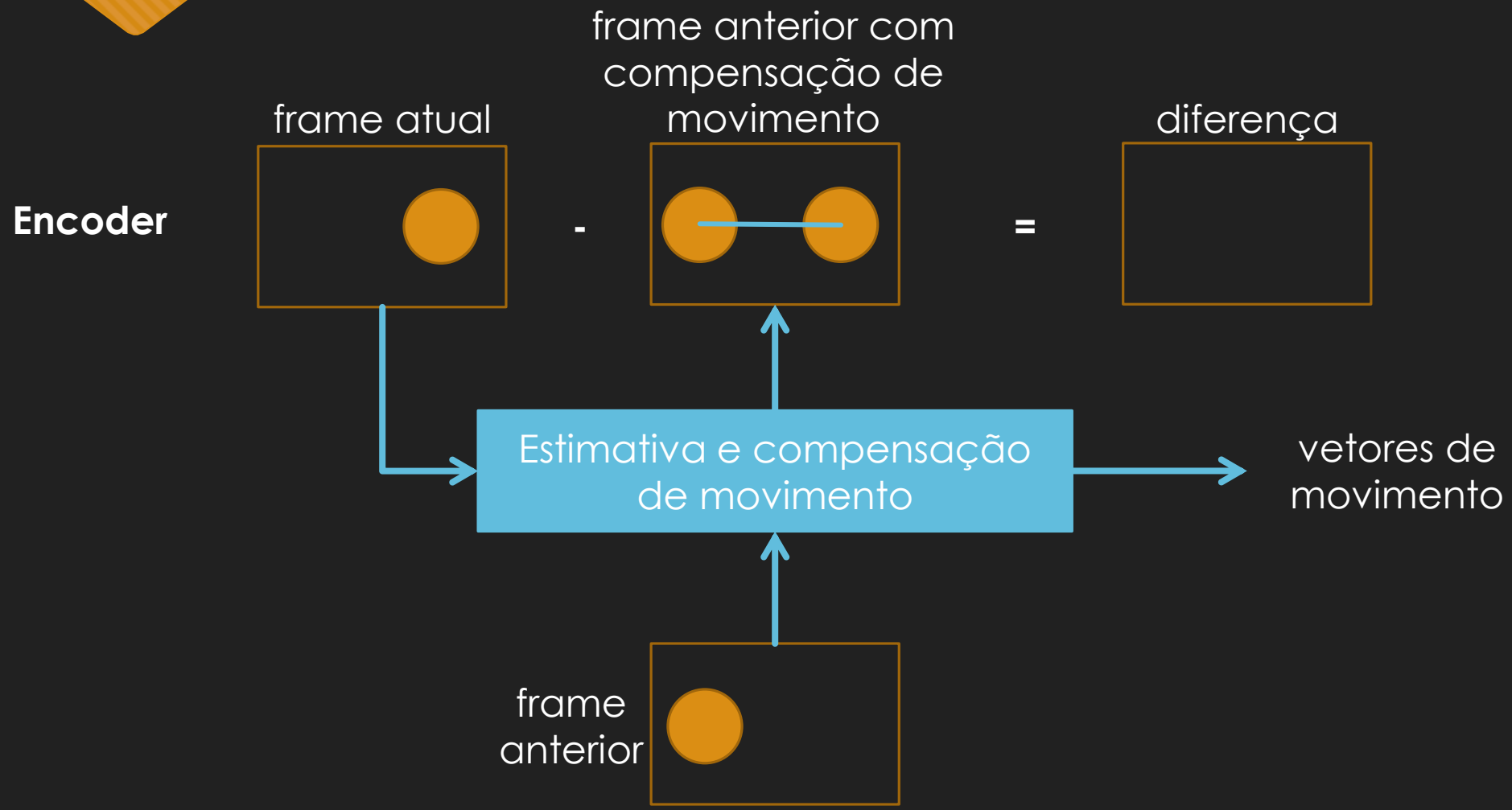
diferença + frame anterior = frame atual

Estimando o movimento entre frames

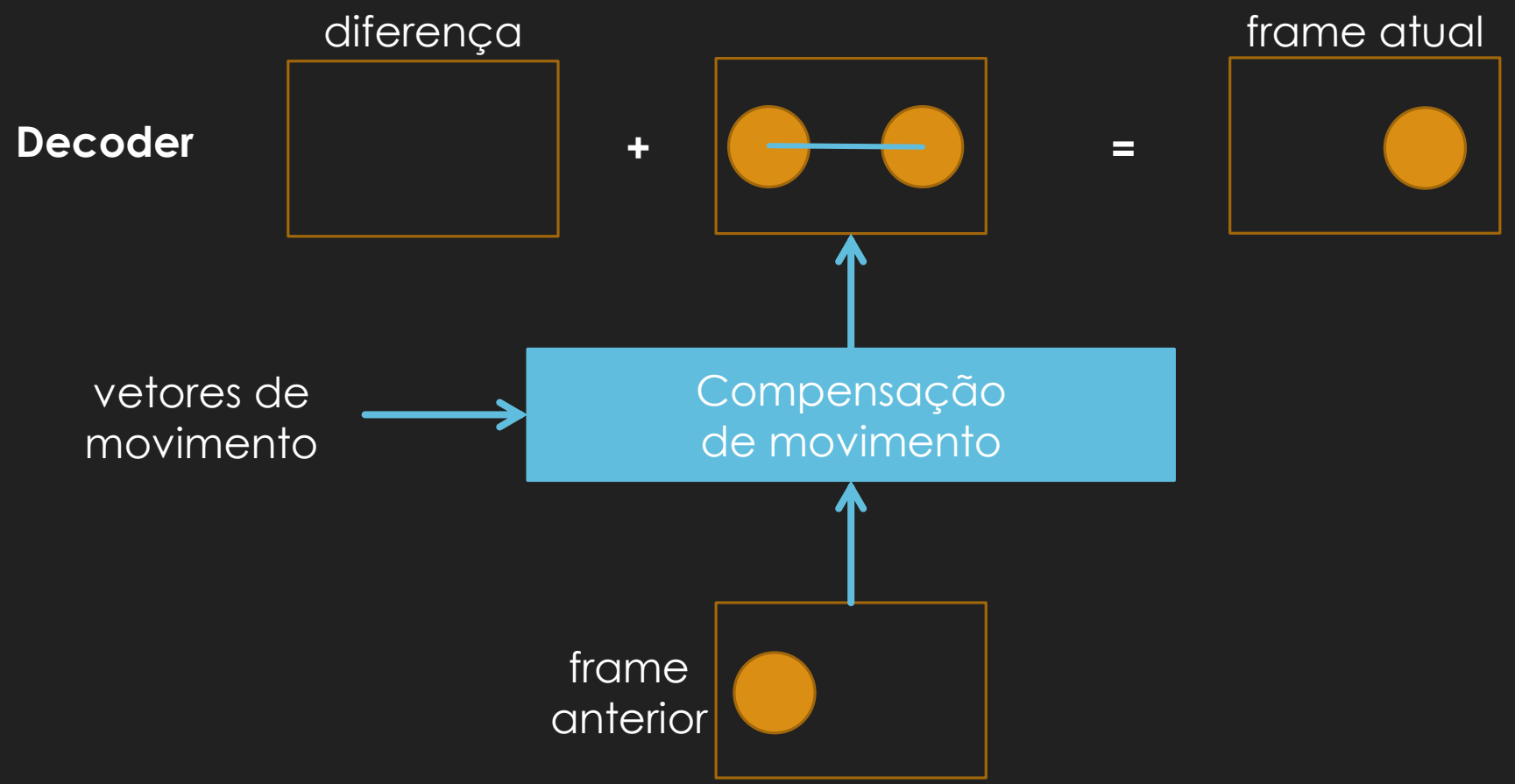
- Movimento é causado principalmente pelo movimento de câmera e/ou algum objeto
- Compensação de movimento
  - As regiões de movimento podem ser estimadas para agilizar a diferença entre imagens consecutivas
  - Partes de uma foto anterior (ou futura) podem ser reutilizadas em uma foto subsequente



Estimando o movimento entre frames



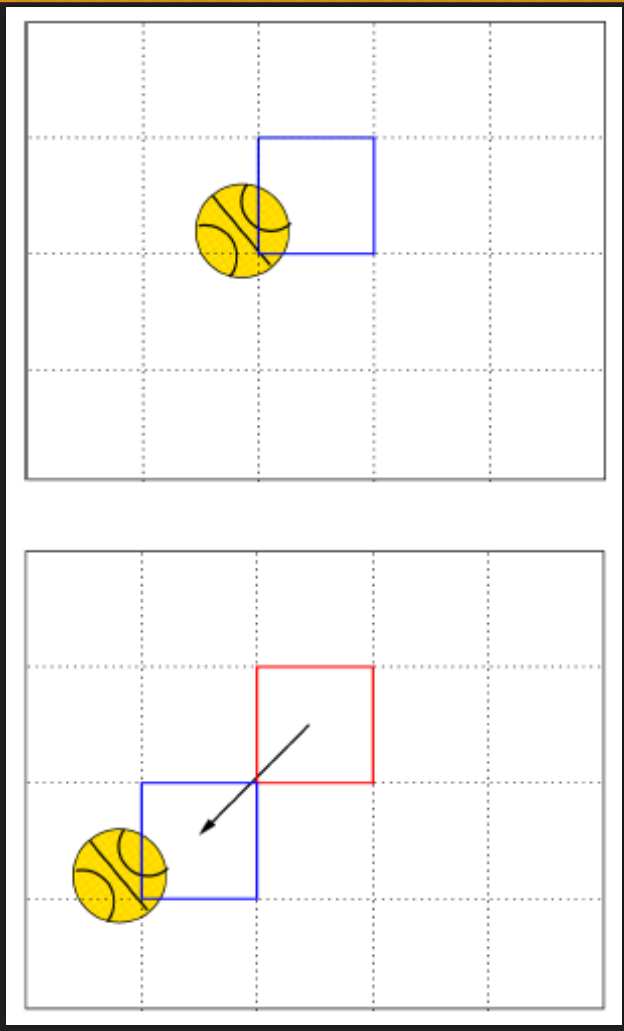
Estimando o movimento entre frames



Estimando o movimento entre frames

Estimando o movimento entre frames

- A compensação do movimento é feita nos macroblocos
  - Imagem atual: frame alvo
  - Imagem passada: frame de referência
- Comparação entre o macrobloco do frame alvo com o mais provável macrobloco do frame de referência
  - Descobrir o vetor de movimento entre os macroblocos
- Após o primeiro frame, somente o vetor de movimento e as diferenças dos macroblocos precisam ser codificadas
  - Informação suficiente para regenerar o vídeo

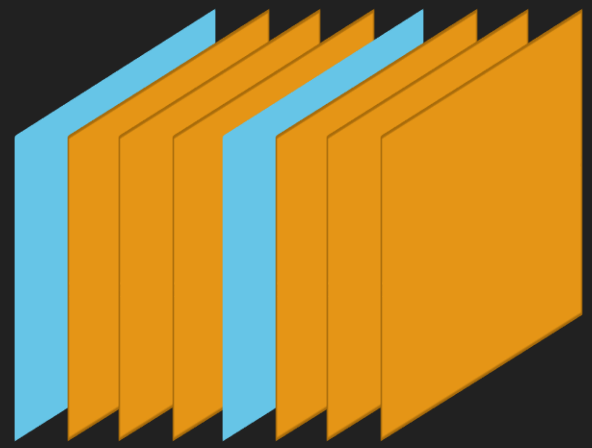


frame de referência

frame alvo

Grupos de frames | H.261 (1993)

- Dois tipos de frames
  - I-frames (Intraframes), em azul
  - P-frames (Interframes), em laranja
  - Normalmente, um frame I seguido por vários frames P
- Quantidade de frames P pode ser fixa ou dinâmica
- I-frames são frames chave
  - Usam basicamente JPEG, mas YUV (YCrCb) e janelas DCT maiores, quantização diferente
- P-frame
  - Usam pseudo-diferenças do frame anterior
  - Frames dependem uns dos outros



Material complementar

- Estrutura de Dados em C | Aula 158 - Compressão de dados
  - https://youtu.be/Fj6kkRPdJ2c
- Estrutura de Dados em C | Aula 159 - Código de Huffman
  - https://youtu.be/U-QA7Gj75T0