COMPRESSÃO DE DADOS
MÉTODOS BÁSICOS
MÉTODOS ESTATÍSTICOS

Sérgio Mergen

Compressão vs compactação

- Compressão
  - Reduz a quantidade de dados para representar uma informação
    - Ex. Zip
- Compactação
  - Junta dados que estão separados
    - Ex. Desfragmentação de disco

- A compressão também é chamada de codificação
  - Por sua vez, a descompressão é chamada de decodificação

Porque comprimir?

- Benefícios da compressão
  - Redução do espaço de armazenamento
  - Redução do tempo para efetuar a leitura
    - Tanto a partir de um dispositivo local (acesso a disco)
    - Quanto a partir de um dispositivo remoto (via rede)

- Preço da compressão deve ser analisado
  - Custo computacional para codificar e decodificar a informação

Medida de desempenho

- Razão da compressão:
  - Uma das formas usadas para medir a eficiência do algoritmo de compressão
  - É definida pela razão (em percentual) entre:
    - o tamanho do arquivo não comprimido e
    - o tamanho do arquivo comprimido

- Ex.
  - Se o arquivo não comprimido possui 100 bytes
  - E o arquivo comprimido possui 20 bytes
  - A razão de compressão é de 5.

Tipos de compressão

- Quanto a preservação da informação, existem dois tipos de compressão:
  - Sem perda de dados
    - Ex. zip ou rar
  - Com perda de dados
    - Ex. jpg

- Nos deteremos nos algoritmos que não perdem informação

Tipos de compressão

- As técnicas de compressão também podem ser classificadas quanto a natureza dos dados
  - Ex.
    - Compressão de texto
    - Compressão de imagens
    - Compressão de sons
- Observações
  - Algoritmos de compressão de texto não são eficientes para compressão de sons
  - Geralmente, a compressão de imagens e sons aceita perda de dados

- Nos concentraremos mais nos algoritmos de compressão de texto

Tipos de compressão

- As técnicas também podem ser divididas pela estratégia de compressão usada
- Uma possível divisão classifica as técnicas em:
  - Métodos básicos
    - Baseados em heurísticas simples
  - Métodos estatísticos
    - Baseados na probabilidade de ocorrência dos caracteres
  - Métodos de dicionário
    - Baseado na construção de dicionários para conjuntos de caracteres frequentes

COMPRESSÃO DE DADOS

Métodos Básicos

Métodos Básicos

- Tipos estudados
  - Run length Encoding (RLE)
  - Mapas de bits
  - Compressão de meio byte
  - Técnica dos 7 bits
  - Representação não ASCII

Run Length Encoding (RLE)

- **Estratégia 1:**
- Determina-se a quantidade de caracteres idênticos consecutivos na cadeia
- Cada uma dessas cadeias é substituída por
  - um número decimal indicando o número de repetições
  - Uma ocorrência do caractere repetido

B-*B*-*B*-*A*-*A*-*A*-*A*-*9*-*9*-*9*-*9*-*9*

3-*B*-4-*A*-5-*9*

Run Length Encoding (RLE)

- **Problemas da estratégia 1**
  - Não vale a pena codificar todas cadeias
    - As cadeias mais curtas geram códigos mais longos que a cadeia original

B-A-B-B-B-A-9-A-B

1-B-1-A-3-B-1-A-1-9-1-A-1-B

Run Length Encoding (RLE)

- **Estratégia 2:**
- Codificar somente as cadeias maiores
- Usar como código
  - um caractere especial como escape
  - O número de repetições
  - O caractere repetido

- Normalmente é um dos caracteres não imprimíveis da tabela ASCII
  - Para simplificar, usaremos o símbolo #

Amostra da Tabela ASCII

Código binário	Caractere
0000 1110	Shift-out (SO)
0000 1111	Shift-in (SI)
0001 1000	Cancel line (CAN)

B-B-B-B-B-A-A-9-9-9-9

#-5-B-A-A-#-4-9

Run Length Encoding (RLE)

- **Problema da Estratégia 2:**
  - Nem sempre é possível usar um caractere de escape

B-B-B-B-B-A-A-9-9-9-9

#-5-B-A-A-#-4-9

B-B-B-B-B-#-9-A

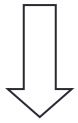
#-5-B-#-9-A

???

Run Length Encoding (RLE)

- **Estratégia 3:**
  - Usar como código
    - O número de repetições
    - 2 x o caractere repetido
  - Ou seja, a aparição do caractere duas vezes consecutivas denota a presença de uma cadeia repetível

B-B-B-B-B-A-A-9-9-9-9



5-B-B-2-A-A-4-9-9

Run Length Encoding (RLE)

- **Problema da Estratégia 3:**
  - Cadeias com duas repetições também precisam ser marcadas

B-B-B-B-B-B-A-A-3-3-5-9-9-9-9-9

6-B-B-2-A-A-2-3-3-5-5-9-9

Run Length Encoding (RLE)

- **Estratégia 4:**
  - Usar como código
    - um caractere especial como escape (**o de menor frequência**)
    - O número de repetições
    - O caractere repetido
  - Quando o caractere especial aparece como parte do texto
    - Usar como código o caractere especial 2 vezes

  - Ex. suponha que o caractere especial seja #

B-B-B-B-B-#-9-A

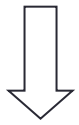
#-5-B-#-#-9-A

Run Length Encoding (RLE)

- **Problema da Estratégia 4:**
  - Necessário pre-processar arquivo para descobrir caractere menos frequente
  - Ocorrências do caractere especial no texto dobram de tamanho
    - Por isso é melhor usar o caractere menos frequente
    - Preferencialmente um que nunca ocorra

- Ex.

B-B-B-B-B-B-A-A-3-3-5-9-9-9-9-9



#-6-B-A-A-3-3-5-#-9-9

Run Length Encoding (RLE)

- Fisicamente, os caracteres são representados por bytes
  - Isso traz a tona uma limitação das abordagens anteriores
- Ex. No texto comprimido abaixo
  - Os números são representados pelo seu código binário
  - Os caracteres pela sua representação em ASC2

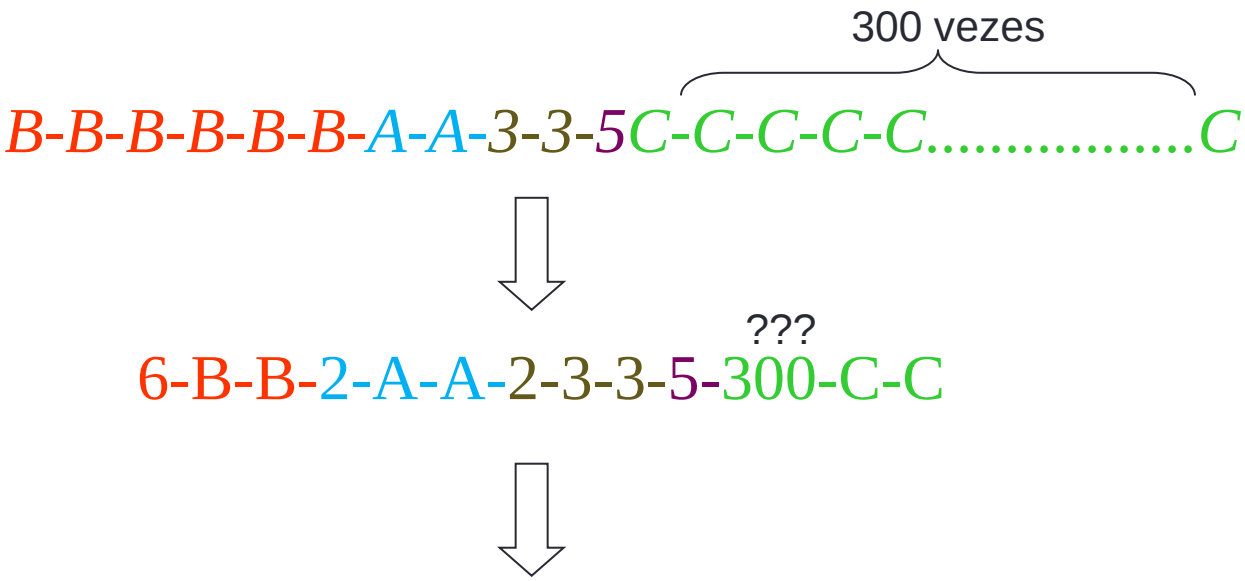
B-B-B-B-B-B-A-A

6-B-2-A

00000110 01000010 00000010 01000001

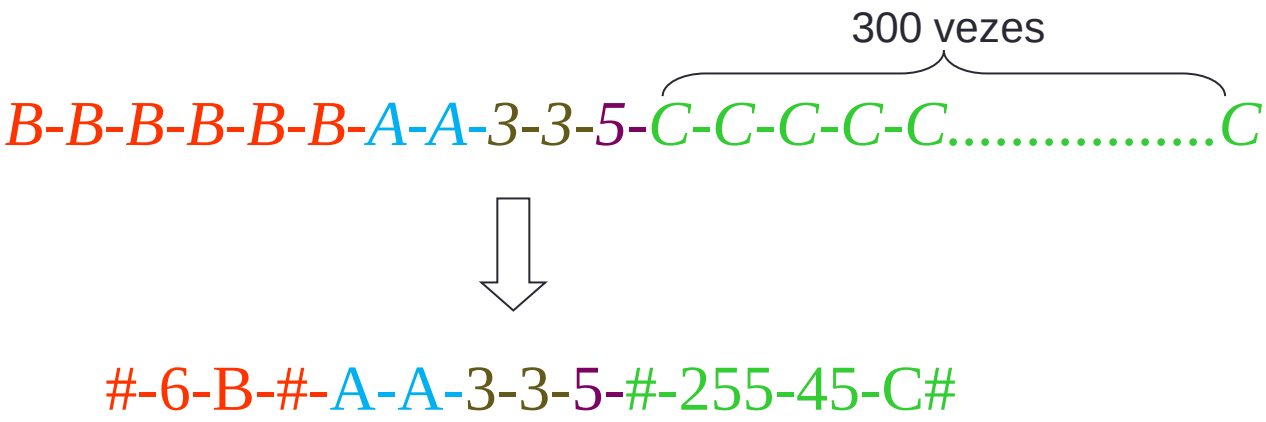
Run Length Encoding (RLE)

- **Problema das estratégias 1,2,3 e 4:**
  - Em código decimal, uma cadeia de 8 bits consegue representar no máximo 255 números
  - Como comprimir cadeias que se repetem mais do que 255 vezes?



Run Length Encoding (RLE)

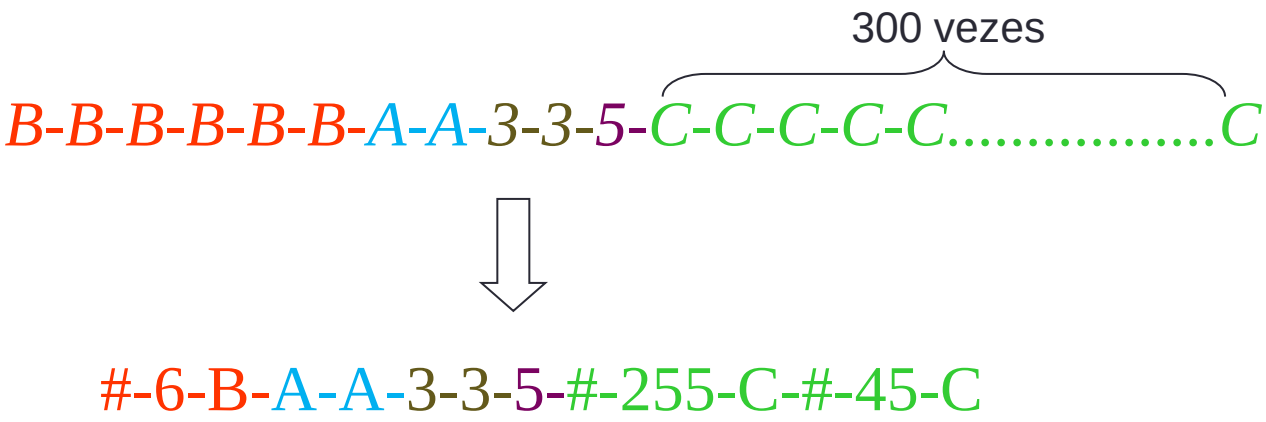
- **Estratégia 5:**
  - Usar um caractere de início e término para indicar quando um código é usado



  - Problema
    - Necessário usar o caractere especial duas vezes em casa código

Run Length Encoding (RLE)

- **Estratégia 6:**
  - Cria mais do que um código para mapear toda a contagem



  - Problema
    - Necessário usar um caractere especial
    - Desempenho pior quando a maioria dos blocos de caracteres repetidos são grandes

Run Length Encoding (RLE)

- Existem outras estratégias:
  - Contudo, todas elas possuem uma limitação em comum
    - Desempenho pobre quando há pouca repetição
      - Seja usando caractere especial ou outra forma de marcação

Mapa de bits

- Usado para comprimir caracteres predefinidos, cuja ocorrência seja bastante frequente
- Um mapa de 8 bits determina se o caractere frequente existe em alguma posição no trecho mapeado
  - 1 significa que existe
  - 0 significa que não existe

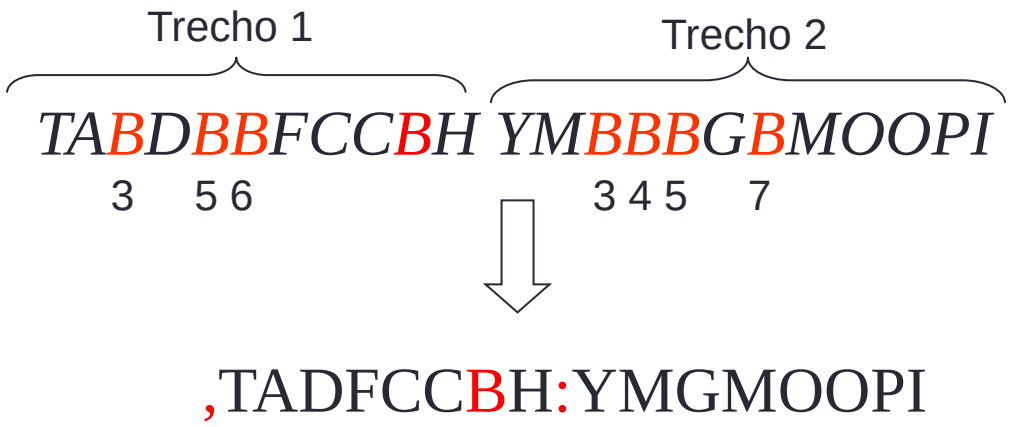
- Ex. caractere frequente = B

TABDBBFC

00101100

Mapa de bits

- O mapa é adicionado antes do trecho que ele codifica
  - Na forma de caractere, para fins de ilustração
- O trecho compreende o espaço contíguo dos primeiros 8 caracteres
  - Com exceção dos caracteres mapeados



Amostra da Tabela ASCII

Byte	caractere
00101100	,
00111010	:
...	...

Mapa de bits

- Problemas
  - Só codifica um caractere (o mais frequente)
  - Só faz sentido se a frequência desse caractere for realmente elevada
  - Requer pré-processamento para descobrir a frequência

Compressão de Meio Byte

- Parte da constatação que alguns caracteres utilizam os mesmos bits para a primeira metade do byte
  - Ex. Os dígitos de 0 a 9

Amostra da Tabela ASCII

Caractere	byte
0	0011 0000
1	0011 0001
2	0011 0010
...	

- Ideia: Usar um código para representar sequências de caracteres que possuam essa característica

Compressão de Meio Byte

- Notação:
  - Ce NM C1C2 C3C4 ...
  - Onde:
    - Ce = caractere especial
    - N = número de caracteres a comprimir
    - M = metade do caractere a comprimir
    - C1 em diante: caracteres comprimidos

785496

00110111 ***0011***1001 ***0011***0101 ***0011***0100 ***0011***1001 ***0011***0110

00011000 0110 0011 0111 1001 0101 0100 1001 0110

Ce N M C1 C2 C3 C4 C5 C6

Compressão de Meio Byte

- Problemas
  - Só consegue comprimir $2^4 = 16$ caracteres consecutivos
  - Só faz sentido se pelo menos 4 caracteres puderem ser comprimidos
    - No exemplo abaixo, o código gerado é maior que a cadeia inicial

785

⇩

00110111 ***0011***1001 ***0011***0101

⇩

00011000 0011 0011 0111 1001 0101 0000

Ce N M C1 C2 C3

Técnica dos 7 bits

- Elimina o bit mais significativo do byte
- Baseia-se no fato de que nenhum caractere de texto utiliza o oitavo bit
  - Se usada a tabela ASCII

Amostra da Tabela ASCII

Caractere	Byte
A	01000001
B	01000010
C	01000011
...	
Z	01011010

Técnica dos 7 bits

- Exemplo

ABA

01000001 01000010 01000001

1000001 1000010 1000001

Amostra da Tabela ASCII

Caractere	Byte
A	01000001
B	01000010
C	01000011
...	
Z	01011010

- Problemas:
  - Aplicável somente a arquivos texto codificados em ASCII
  - Alguns caracteres acentuados não compatíveis usam o oitavo bit

Representação não ASCII

- Usa uma representação binária mais curta para caracteres
- Quanto menor o alfabeto, melhor a compressão

TCAC

01010100010000110100000101000011

11010001

Tabela de códigos

Código	Caractere
A	00
C	01
G	10
T	11

- Problema
  - Só faz sentido se mapear menos do que 128 caracteres

COMPRESSÃO DE DADOS

Métodos Estatísticos

Compressão estatística

- Realiza uma compressão otimizada dos caracteres que mais se repetem
  - A compressão leva em consideração a probabilidade de ocorrência de cada caractere.
- Tipos a estudar:
  - Codificação de Huffman
  - Codificação de Shannon-fano

Codificação de Huffman - Etapas

1. Monta uma tabela com linhas compostas por
   - O caractere e o número de ocorrências do caractere
2. Transformar cada linha (caractere) em um nó
3. Escolher os dois nós com o menor número de ocorrências, ainda sem relacionamentos
4. Criar um novo nó
   - que será o pai dos dois nós escolhidos
   - O número de ocorrências será a soma das ocorrências dos filhos
   - O filho da esquerda recebe o bit 0
   - O filho da direita recebe o bit 1
5. Repetir a partir do passo 3

Exemplo

Texto a ser comprimido

AAAAABBBBCCCDDEe

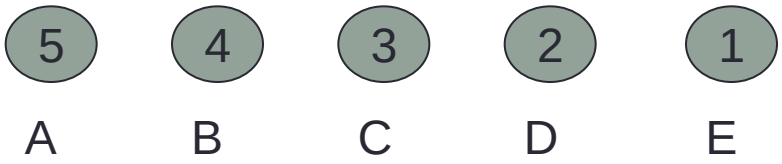
caractere	# ocorrências	Código
A	5	
B	4	
C	3	
D	2	
E	1	

Texto comprimido (em binário)

?

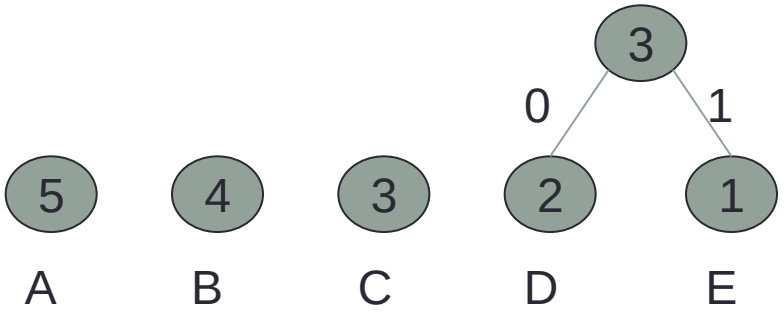
Exemplo

Caractere	# ocorrências	Código
A	5	
B	4	
C	3	
D	2	
E	1	



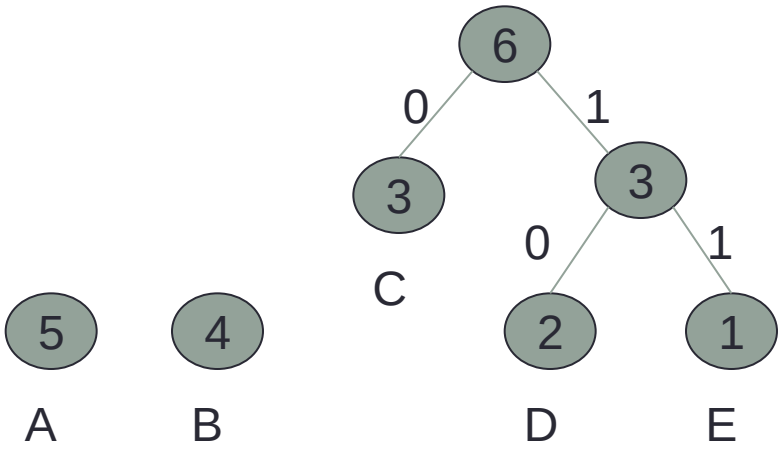
Exemplo

Caractere	# ocorrências	Código
A	5	
B	4	
C	3	
D	2	
E	1	



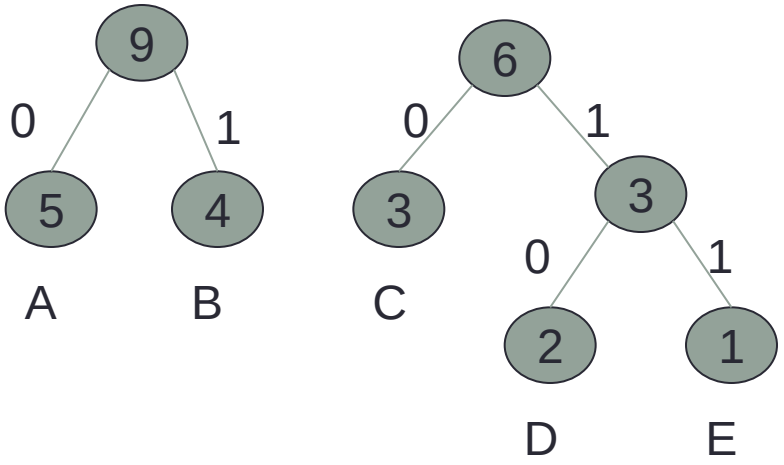
Exemplo

Caractere	# ocorrências	Código
A	5	
B	4	
C	3	
D	2	
E	1	



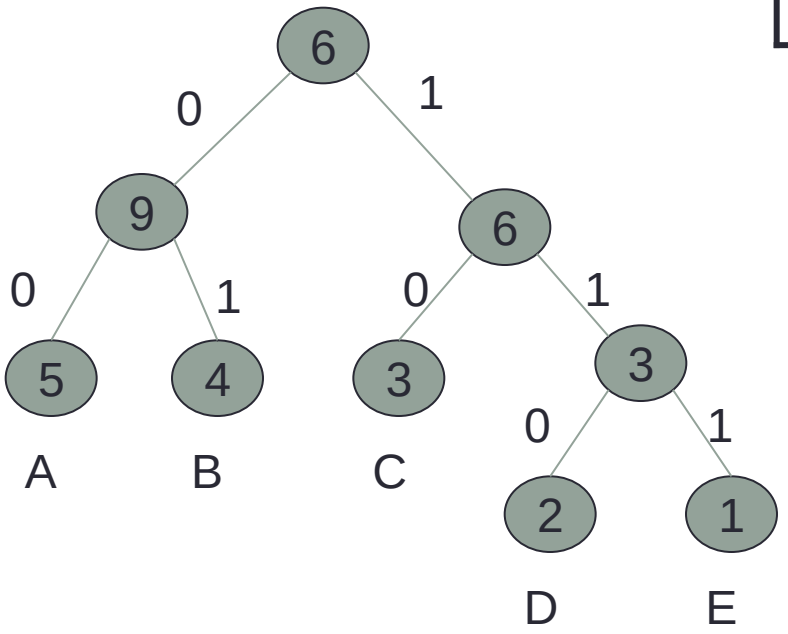
Exemplo

Caractere	# ocorrências	Código
A	5	
B	4	
C	3	
D	2	
E	1	



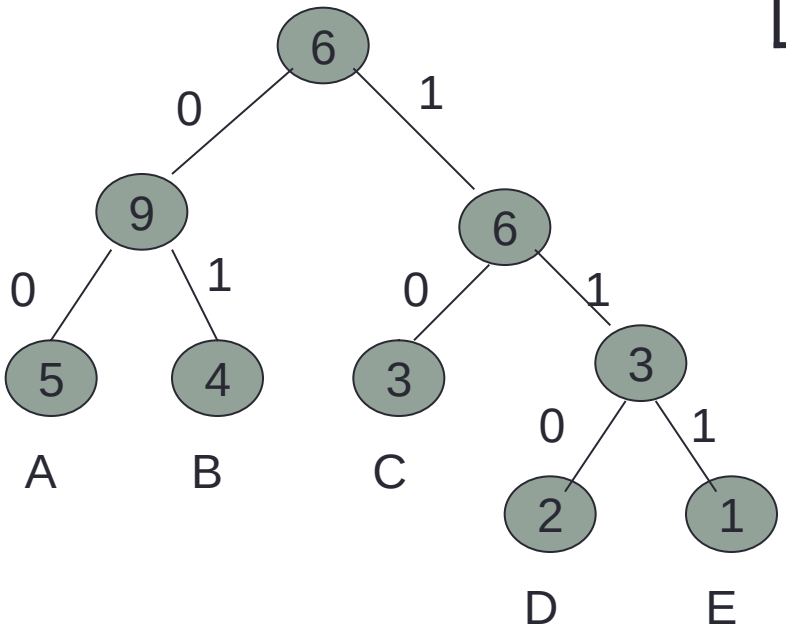
Exemplo

Caractere	# ocorrências	Código
A	5	
B	4	
C	3	
D	2	
E	1	



Exemplo

É uma Trie Binária!



Caractere	# ocorrências	Código
A	5	00
B	4	01
C	3	10
D	2	110
E	1	111

Exemplo

Texto a ser comprimido

AAAAABBBBCCCDDE

120 bitse

caractere	# ocorrências	Código
A	5	00
B	4	01
C	3	10
D	2	110
E	1	111

Texto comprimido (em binário)

000000000001010101101010110110111

33 bits

Codificação de Shannon Fano

1. Monta uma tabela com linhas compostas por
   - O caractere e o número de ocorrências do caractere
2. Adicionar os caracteres em um vetor
   - Ordenado pelo número de ocorrências
3. Encontrar o ponto do vetor que melhor divide os números de ocorrências
4. Dividir o vetor nesse ponto
   - O sub-vetor à esquerda recebe o bit 0
   - O sub-vetor a direita recebe o bit 1
5. Repetir a partir do passo 3, para cada sub-vetor

Exemplo

Texto a ser comprimido

AAAAABBBBCCCDDEe

caractere	# ocorrências	Código
A	5	
b	4	
C	3	
D	2	
E	1	

Texto comprimido (em binário)

?

Exemplo

Caractere	# ocorrências	Código
A	5	
B	4	
C	3	
D	2	
E	1	

5	A
4	B
3	C
2	D
1	E

Exemplo

Caractere	# ocorrências	Código
A	5	
B	4	
C	3	
D	2	
E	1	

5	A	9	0
4	B		
3	C	6	1
2	D		
1	E		

Exemplo

Caractere	# ocorrências	Código
A	5	
B	4	
C	3	
D	2	
E	1	

<table>
<tr><td>5</td><td>A</td><td rowspan="2">9 0</td><td>5 0</td></tr>
<tr><td>4</td><td>B</td><td>4 1</td></tr>
<tr><td>3</td><td>C</td><td rowspan="3">6 1</td><td>3 0</td></tr>
<tr><td>2</td><td>D</td><td rowspan="2">3 1</td></tr>
<tr><td>1</td><td>E</td></tr>
</table>

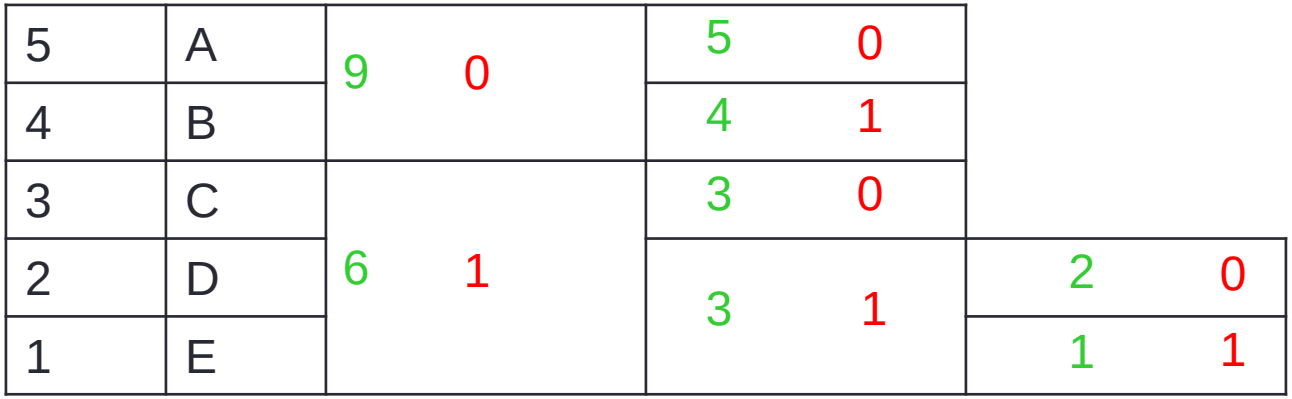
Exemplo

Caractere	# ocorrências	Código
A	5	
B	4	
C	3	
D	2	
E	1	

<table>
<tr><td>5</td><td>A</td><td rowspan="2">9</td><td rowspan="2">0</td><td>5</td><td>0</td></tr>
<tr><td>4</td><td>B</td><td>4</td><td>1</td></tr>
<tr><td>3</td><td>C</td><td rowspan="3">6</td><td rowspan="3">1</td><td>3</td><td>0</td></tr>
<tr><td>2</td><td>D</td><td rowspan="2">3</td><td rowspan="2">1</td><td>2</td><td>0</td></tr>
<tr><td>1</td><td>E</td><td>1</td><td>1</td></tr>
</table>

Exemplo

Caractere	# ocorrências	Código
A	5	00
B	4	01
C	3	10
D	2	110
E	1	111



Exemplo

Texto a ser comprimido

AAAAABBBBCCCDDE

120 bitse

caractere	# ocorrências	Código
A	5	00
B	4	01
C	3	10
D	2	110
E	1	111

Texto comprimido (em binário)

000000000001010101101010110110111

33 bits

Análise

- No exemplo, as duas técnicas geraram o mesmo código
  - Isso nem sempre acontece
- A codificação de Huffman produz códigos melhores (ótimos)
  - Por essa razão ela é usado pelas ferramentas de compressão
    - Em abordagens híbridas
- Os métodos estatísticos produzem resultados bem superiores aos métodos básicos
  - Com o custo de ter que pré-processar o arquivo a compactar
  - Normalmente, o custo compensa em razão da taxa de compressão obtida.

Exercício

- Use um método de RLE que consiga comprimir a cadeia abaixo
  - 12777222225

- Use os algoritmos de Huffman e o de Shannon Fano para codificar a seguinte cadeia
  - A_BAITA_BATATA
- O que exibir
  - Algoritmo de Huffman
    - A Trie binária gerada
    - A cadeia comprimida
  - Algoritmo de Shannon
    - O vetor dividido
    - A cadeia comprimida