Faculdade de Informática

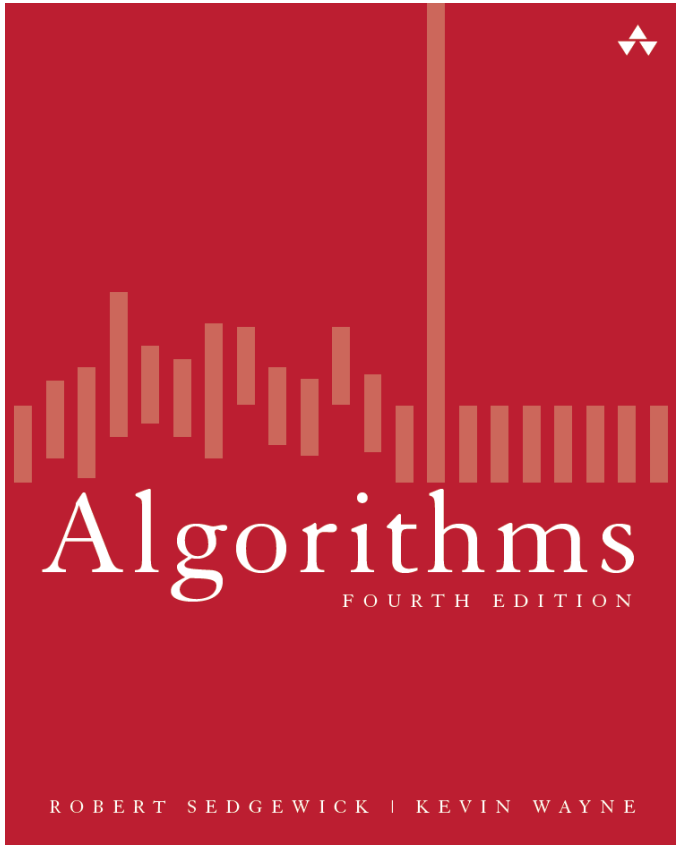
PUCRS

Algoritmos e Estruturas de Dados II
2023-2

Aula #12 – Compressão de Dados e Codificação de Huffman

Prof. Leonardo Heredia

Referências



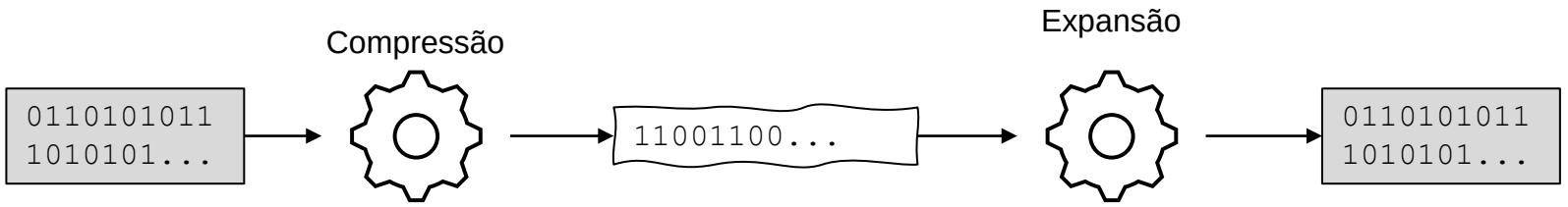
https://algs4.cs.princeton.edu/home/

Compressão de Dados

- Redução do tamanho do arquivo
- Economia de espaço no armazenamento
- Economia de tempo de transmissão
- Arquivos possuem muita redundância (sequencias de bytes repetidos).
- Processo:
  - Compressão (codificação) → Expansão (decodificação)
- Algoritmo de compressão deve garantir a expansão, ou seja, decodificar o dado comprimido para seu conteúdo original, ou parecido. Imagens!

Compressão e Expansão

Modelo básico para compressão de dados



Mensagem com dados binários **B**

Dados comprimidos **C(B)**
Compressed version C(B)

Dados expandidos **B**

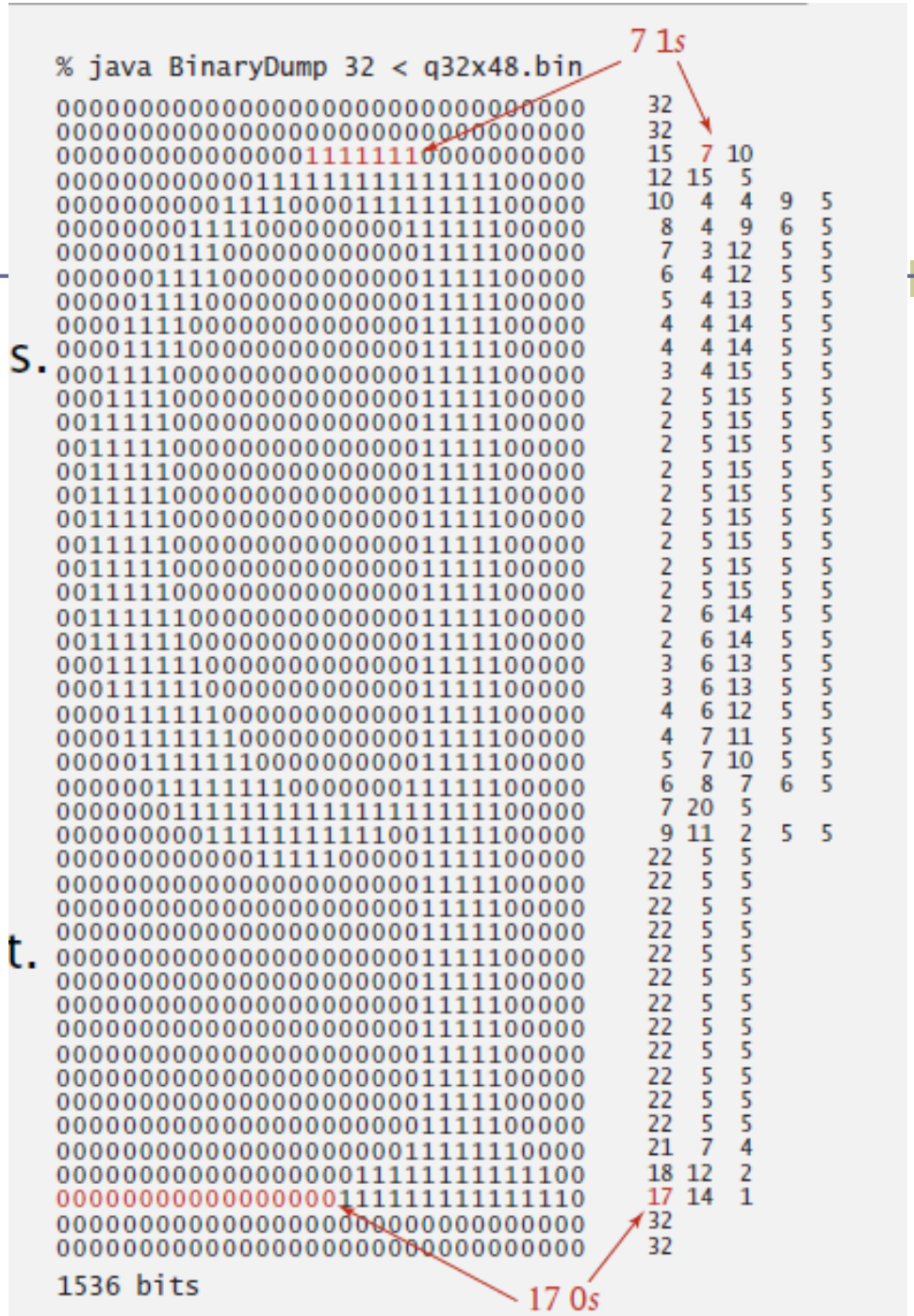
Sem perda → Igual ao original B
Com perda → Similar ao original B

Taxa de compressão: Bits em C(B) / bits em B

100% é teoricamente impossível.
Impossível comprimir uma sequencia até chegar 0 bits

Lossless compression
Conteúdo original completamente restaurado

Lossy compression
Conteúdo restaurado é similar ao original, ocorrendo perda de bits.
Usado em imagens, vídeos e sons, por exemplo.


% java BinaryDump 32 < q32x48.bin
7 1s
17 0s
1536 bits

Compressão e Expansão com perda (lossy)

Exemplos: imagens e vídeos. JPG

https://picwish.com/lossless-and-lossy-compression.html

Compressão e Expansão sem perda (lossless)

Exemplos: imagens, textos, arquivos de programas, etc.
GZIP, ZIP, GIF.

Algoritmos para lossless compression:

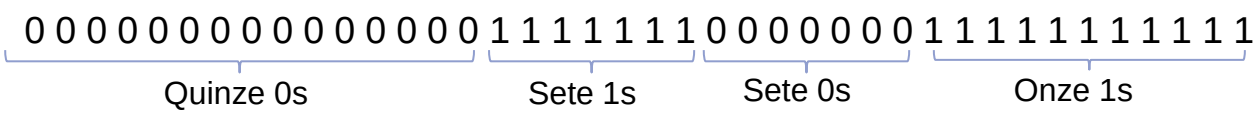
- **Run Lenght Encoding**
- **Huffman Coding**
- LZW
- LZ77
- ...

Compressão RLE

RLE (*run-length encoding*) – codificação de comprimento de carreira

Codifica trechos longos de bits repetidos.

Observe a cadeia abaixo



40 bits

A mesma cadeia pode ser representada pela sequência 15 7 7 11 representando sequencias alternadas de 0 e 1:

16 bits ao invés do original com 40 bits. Sem perda.

Nesse exemplo foi utilizado 4 bits para armazenar as contagens, ou seja, maior valor com 4 bits será justamente o 15. Geralmente se utilizam 8 bits para contagem, nesse caso limite passa para 255.
O que fazer se a sequencia fosse maior que 15? Intercalar com sequencias de tamanho 0.

TIFF, BMP, PCX, JPEG, GIF,...

Compressão RLE - Exercício

Mostre a codificação de compressão RLE para as cadeias de bits abaixo, utilizando 4 bits para representar a quantidade:

a) 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1

b) 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0

c) 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1

Qual foi a taxa de compressão obtida em cada uma das cadeias?

Codificação de Huffman

- Compressão sem perda
- Fluxo de bits é lido como se fosse um fluxo de caracteres, de 8 em 8 bits.
- Cada caractere é adicionado a uma tabela de códigos.
- Ideia do algoritmo de Huffman: usar códigos curtos para os caracteres que ocorrem com frequência e deixar códigos mais longos para caracteres mais raros.
- Códigos de **comprimento variável**.
- Aplicações? PDF, MP3, GZIP

Codificação de Huffman

```
01000001010000100101001001000001010000110100000101000100010000010100001001010010010000010010000 1
```

```
01000001 01000010 01010010 01000001 01000011 01000001 01000100 01000001 01000010 01010010 01000001 00100001
   A        B        R        A        C        A        D        A        B        R        A        !
```

A B R A C A D A B R A !

Qual caractere que mais se repete?

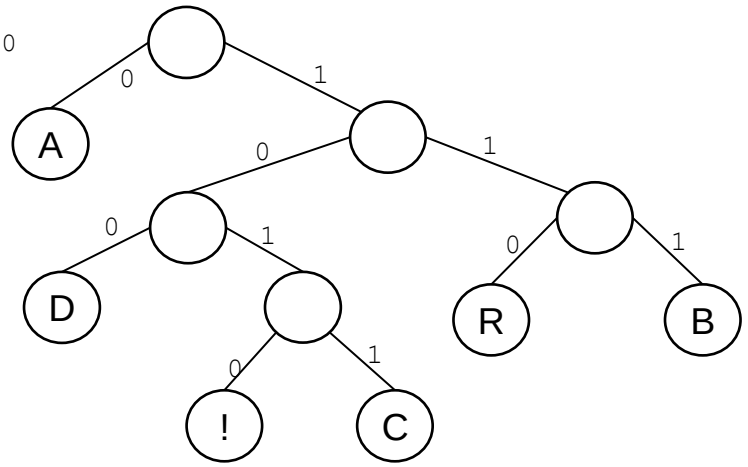
Códigos curtos para os caracteres mais frequentes

Tabela Códigos	
!	1010
A	0
B	111
C	1011
D	100
R	110

Cadeia de bits

```
0 111 110 0 1011 0 100 0 111 110 0 1010
A  B   R  A  C   A  D  A  B   R  A  !
```

Como decodificar?



Códigos gerados não podem ter o mesmo prefixo!
Como fazer? TRIE binária

O segredo está em encontrar códigos livres de prefixo usando o menor código para o caractere mais frequente.

Codificação de Huffman

Compressão:

1. Lê a mensagem
2. Constrói o ***melhor código de livre prefixo para a mensagem***, ou seja, os menores códigos para os caracteres mais frequentes. Isso é a TRIE binária.
3. Salva a trie em um arquivo
4. Comprime a mensagem usando os códigos da TRIE.

Expansão:

1. Lê do arquivo a TRIE
2. Lê a mensagem comprimida e expande ela usando a TRIE

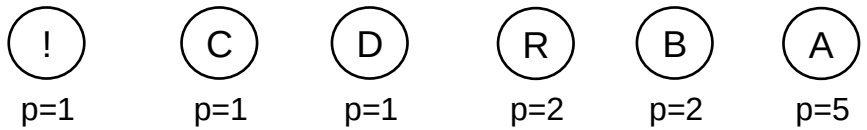
Como encontrar o melhor código livre de prefixo

Como construir a TRIE binária? Trie binária os caracteres ficam nas folhas. Cada código é um caminho da raiz até uma folha.

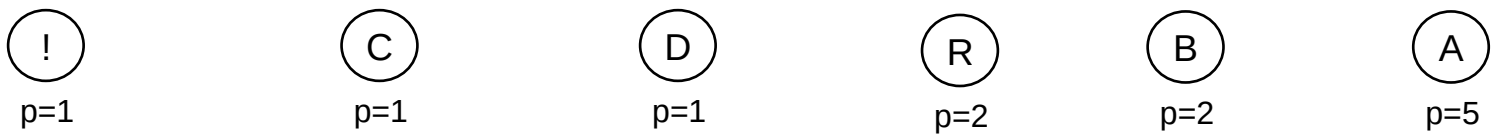
1) Contar a frequência de cada caractere da mensagem e colocar numa tabela.
2) Cada caractere vira uma trie única com um peso associado, que é a quantidade de vezes que ocorreu.
3) A partir desse ponto repetir até que se tenha uma única trie:
   - Selecionar as 2 tries de menor peso
   - Combinar essas 2 tries em uma

A B R A C A D A B R A !

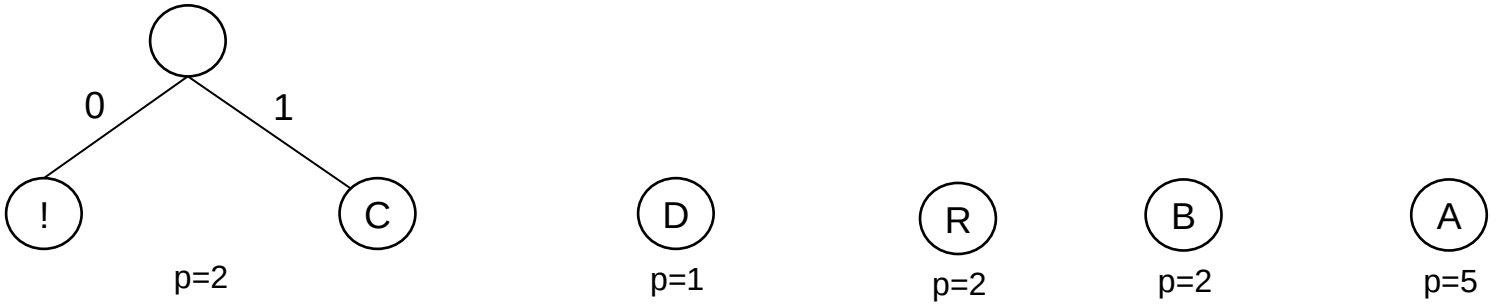
Tabela Códigos		
Char	Frequência	Código
!	1	
A	5	
B	2	
C	1	
D	1	
R	2	



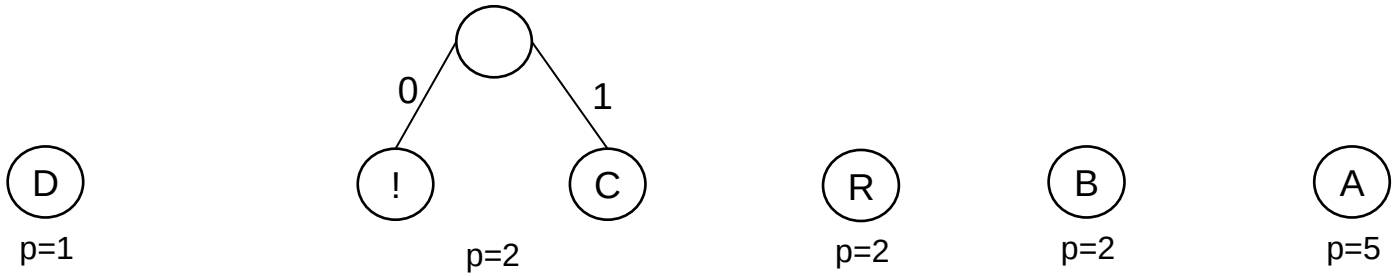
Trie binária



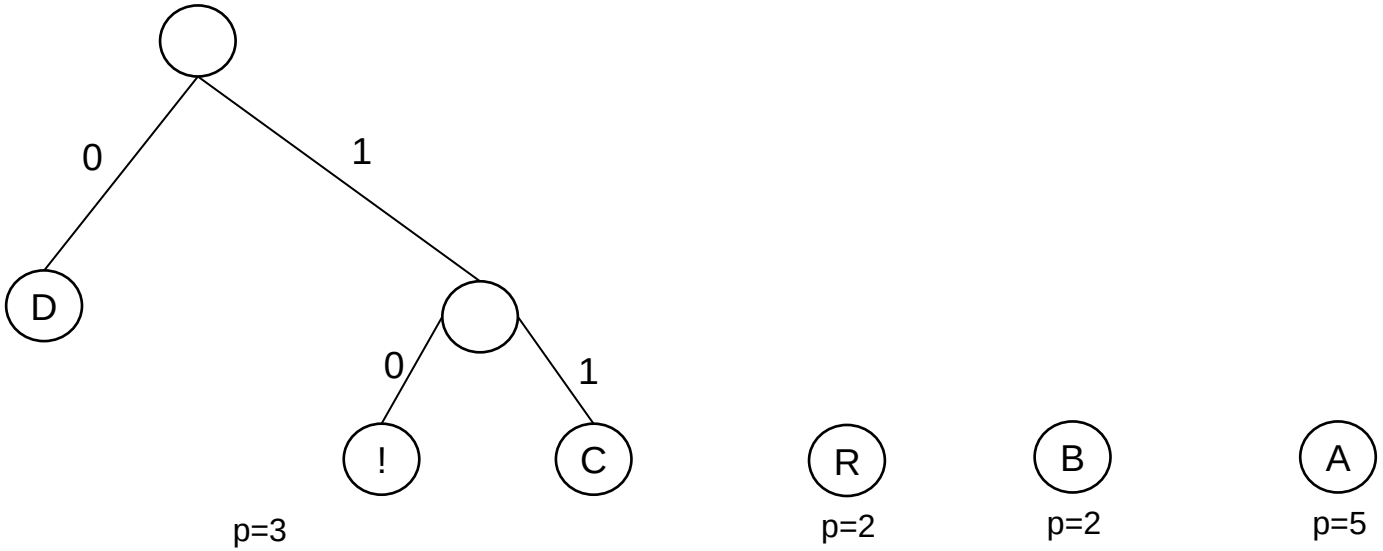
Trie binária



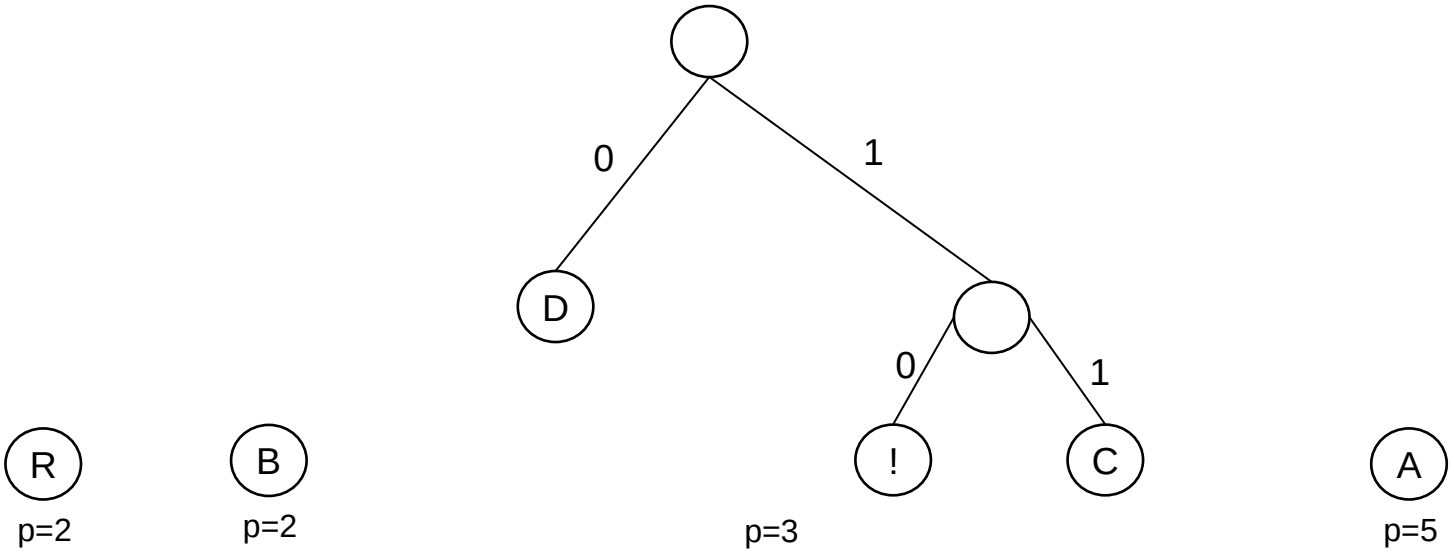
Trie binária



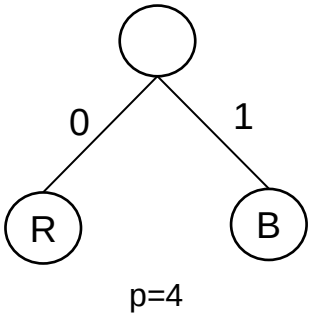
Trie binária



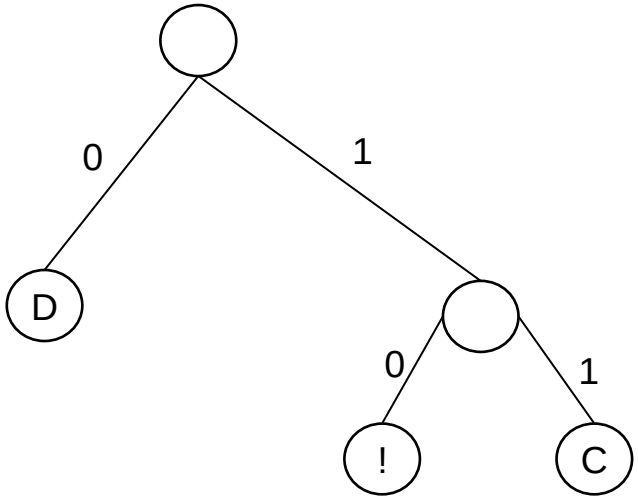
Trie binária



Trie binária



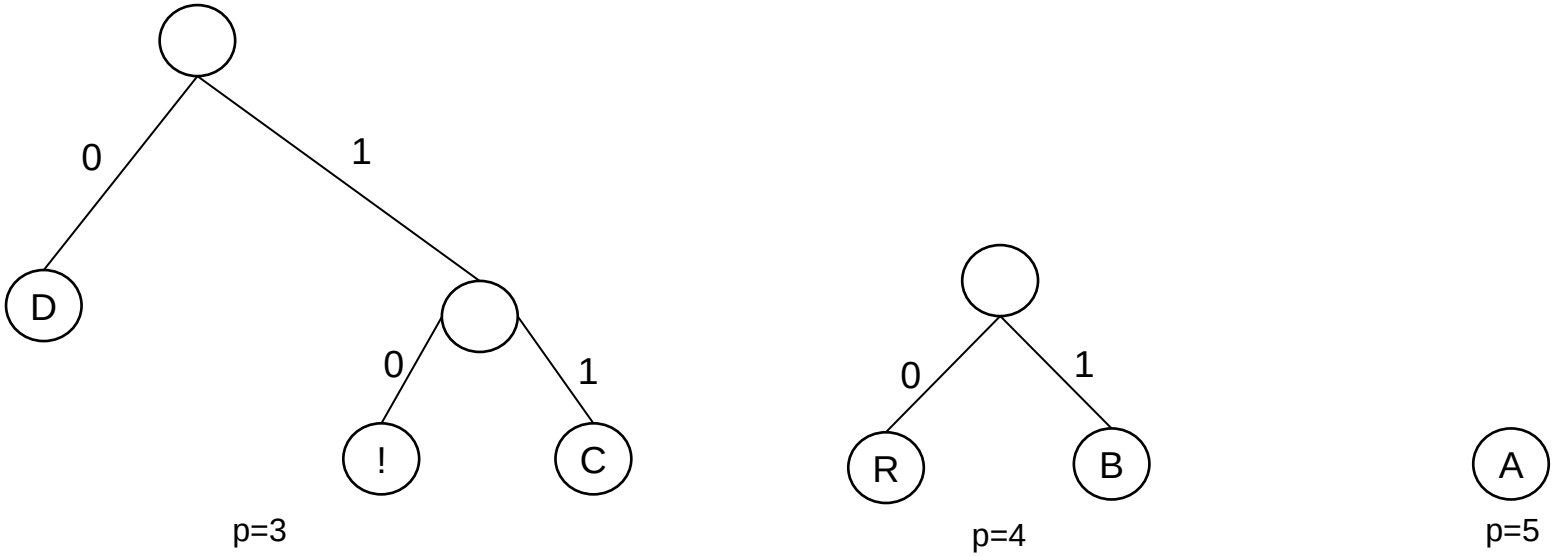
p=4



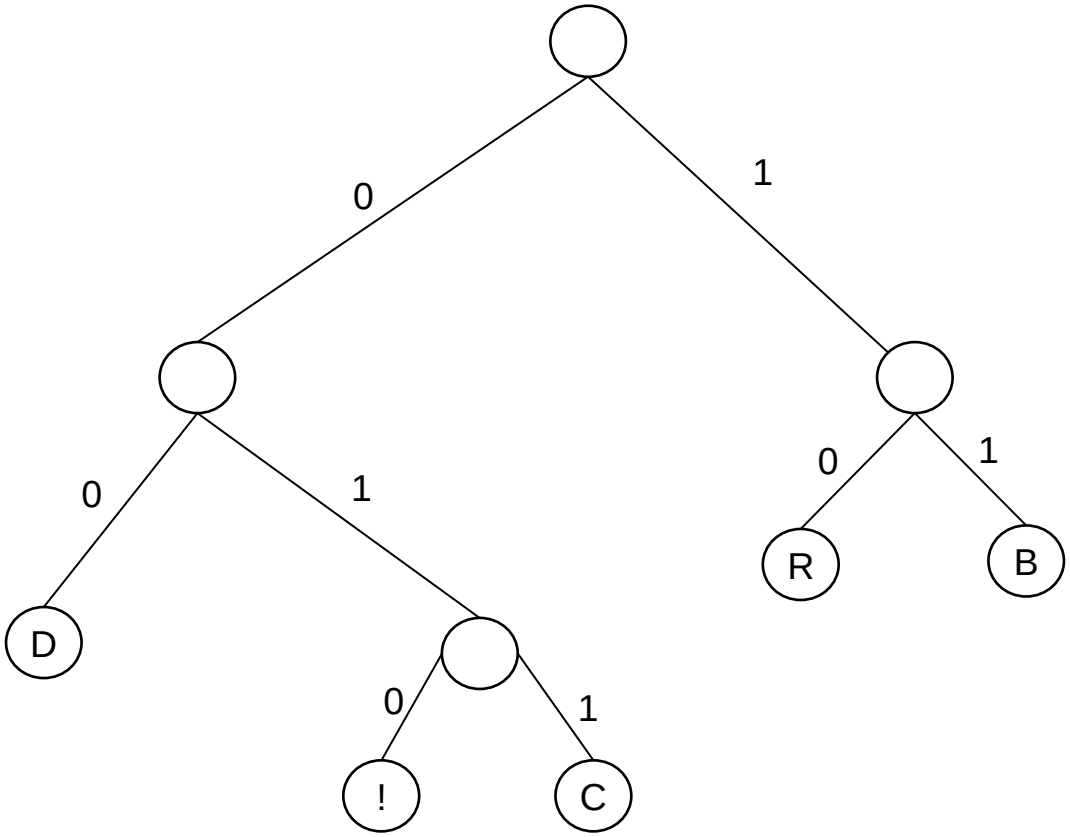
p=3

p=5

Trie binária



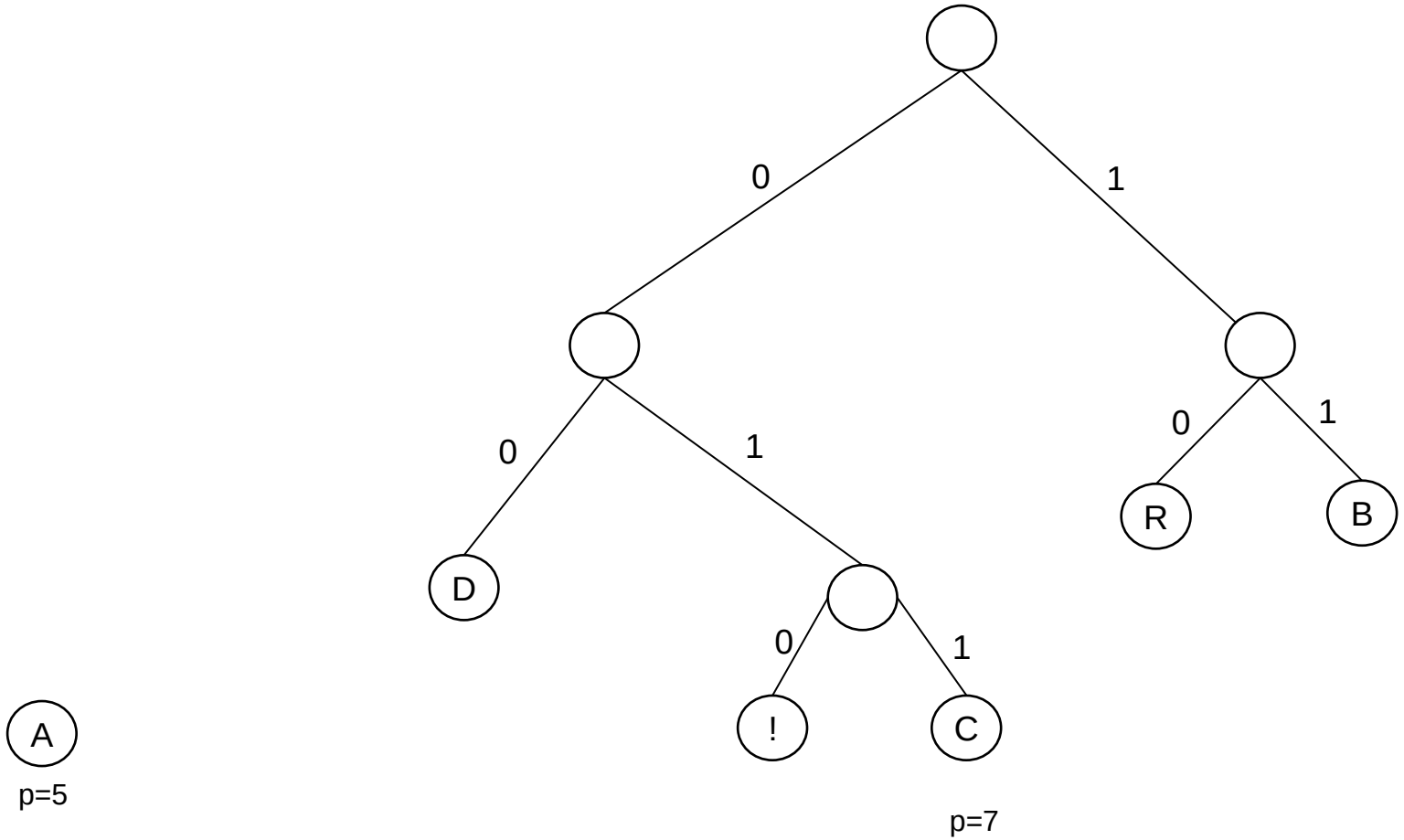
Trie binária



p=7

p=5

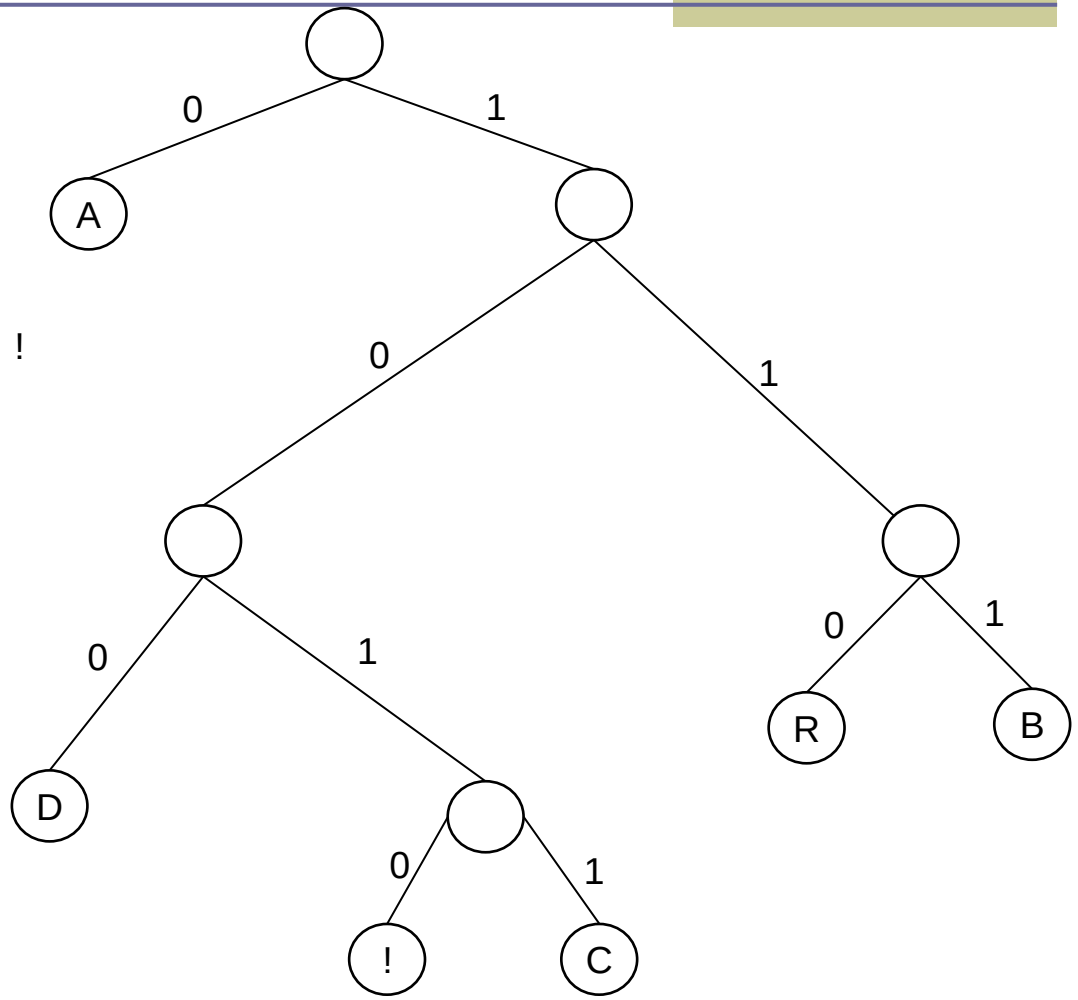
Trie binária



Trie binária

A B R A C A D A B R A !

Tabela Códigos		
Char	Frequência	Código
!	1	1010
A	5	0
B	2	111
C	1	1011
D	1	100
R	2	110



Exercício

- Utilizando o algoritmo de huffman codificar a mensagem abaixo.
- Construir tabela de códigos, a trie e a mensagem comprimida.

B A N A N A

Tabela Códigos		
Char	Frequência	Código
B	1	
A	3	
N	2	