Grafos e Árvores

- Grafos e Suas Representações
- Árvores e suas Representações
- Árvores de Decisão (Parte B)
- Códigos de Huffman

Algorítmos de Ordenação

- Usam somente comparacões entre dois elementos do conjunto a ser ordenado para definir a posicão relativa desses elementos.
- No Selection sort a menor cota superior é dada por Θ(n^2)
- A menor cota superior é dada pelos algoritmos Merge-Sort e o Heap-Sort, que efetuam Θ(nlogn) comparacões no pior caso.

SELECTION SORT

ESTE ALGORITMO É BASEADO EM SE PASSAR SEMPRE O MENOR VALOR DO VETOR PARA A PRIMEIRA POSIÇÃO (OU O MAIOR DEPENDENDO DA ORDEM REQUERIDA), DEPOIS O SEGUNDO MENOR VALOR PARA A SEGUNDA POSIÇÃO E ASSIM SUCESSIVAMENTE, ATÉ OS ÚLTIMOS DOIS ELEMENTOS.

SELECTION SORT - EXEMPLO

Exemplo: Dada a sequência [70,90,1,3,0,100,2] , aplique o algoritmo do Selection sort para ordenar a lista

Execuções:

[70, 90, 1, 3, 0, 100, 2] – 1ª execução
[0, 90, 1, 3, 70, 100, 2] – 2ª execução
[0, 90, 1, 3, 70, 100, 2] – 3ª execução
[0, 1, 90, 3, 70, 100, 2] – 4ª execução
[0, 1, 90, 3, 70, 100, 2] – 5ª execução
[0, 1, 2, 3, 70, 100, 90] – 6ª execução
[0, 1, 2, 3, 70, 100, 90] – 7ª execução
[0, 1, 2, 3, 70, 90, 100] – 8ª execução

Grafos e Suas Representações

Algoritmo de Ordenação

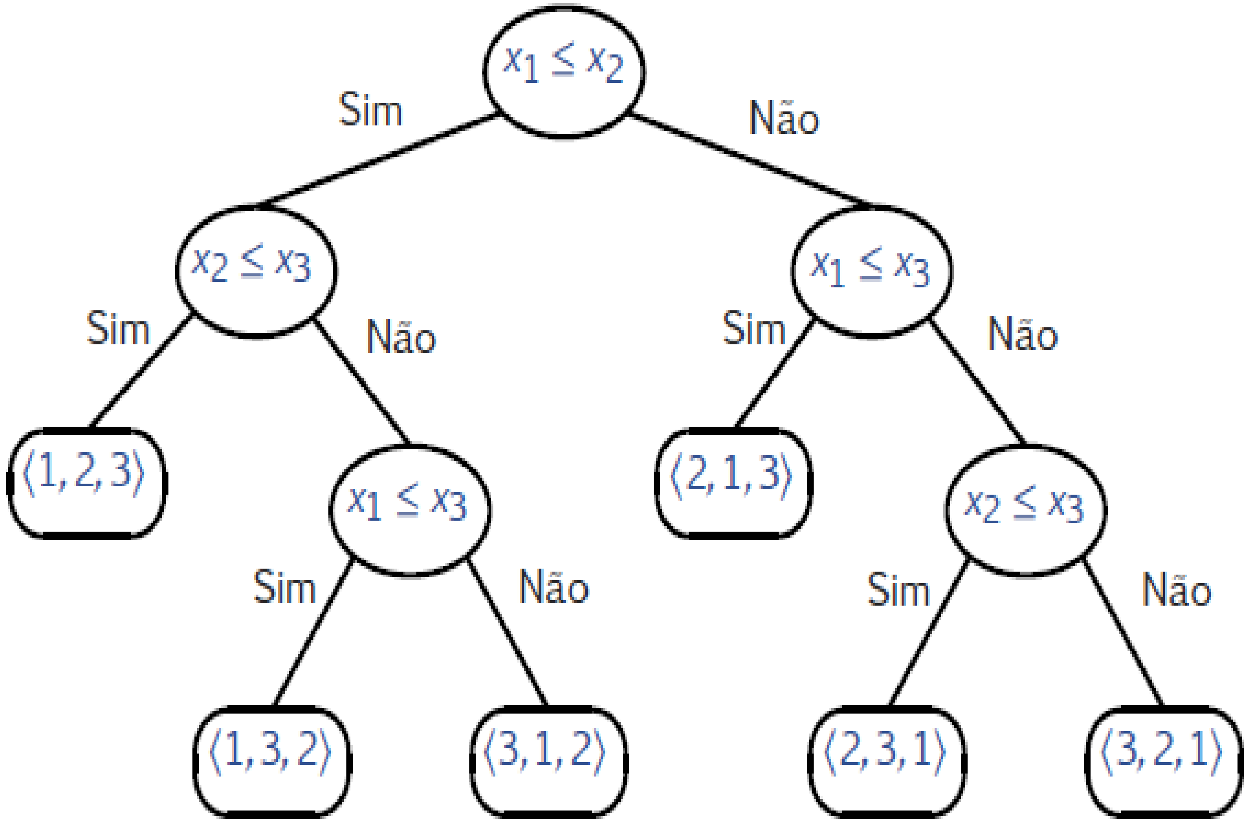
Ex: Desenhe uma árvore de decisão para um algoritmo de ordenação (Insertion sort) , para 3 elementos. Considere um algoritmo pouco esperto (ignorando a transitividade de <).

OBS: Note que uma folha representa o resultado final, isto é, as folhas representam as possibilidades de arranjos ordenados dos *n* elementos ($n!$).

Seja p o nº de folha da árvore de altura d, então $p \leq 2^d$ (exercício)

→ $\log p \leq d$, *se d é inteiro então* $d = \lceil \log p \rceil$

Árvore de decisão que representa o comportamento do Insertion Sort para um conjunto de 3 elementos:



INSTITUTO FEDERAL
ESPÍRITO SANTO
Campus Cariacica

Grafos e Suas Representações

Algoritmo de Ordenação

Seja p o nº de folhas da árvore de altura d,
então $p \le 2^d$ (exercício)

→ log p ≤ d , se d é inteiro, então
$d = \lceil \log p \rceil \geq \lceil \log n! \rceil$ (pois $p \geq n!$)

Teorema sobre Cotas Inferiores para Algoritmos de Ordenação: Qualquer algoritmo que ordena uma lista de n elementos comparando pares de elementos na lista tem que fazer, pelo menos, $\lceil \log n! \rceil$ comparações no pior caso.

Teorema sobre Cotas Inferiores para Algoritmos de Ordenação

$$log_2\ n! = \sum_{i=1}^{n} log_2 i \quad \geq \sum_{i=n/2}^{n} log_2 i \quad \geq \sum_{i=n/2}^{n} log_2 \frac{n}{2}$$

Mostre que o teorema anterior é da ordem $\boldsymbol{\Theta(n.logn)}$

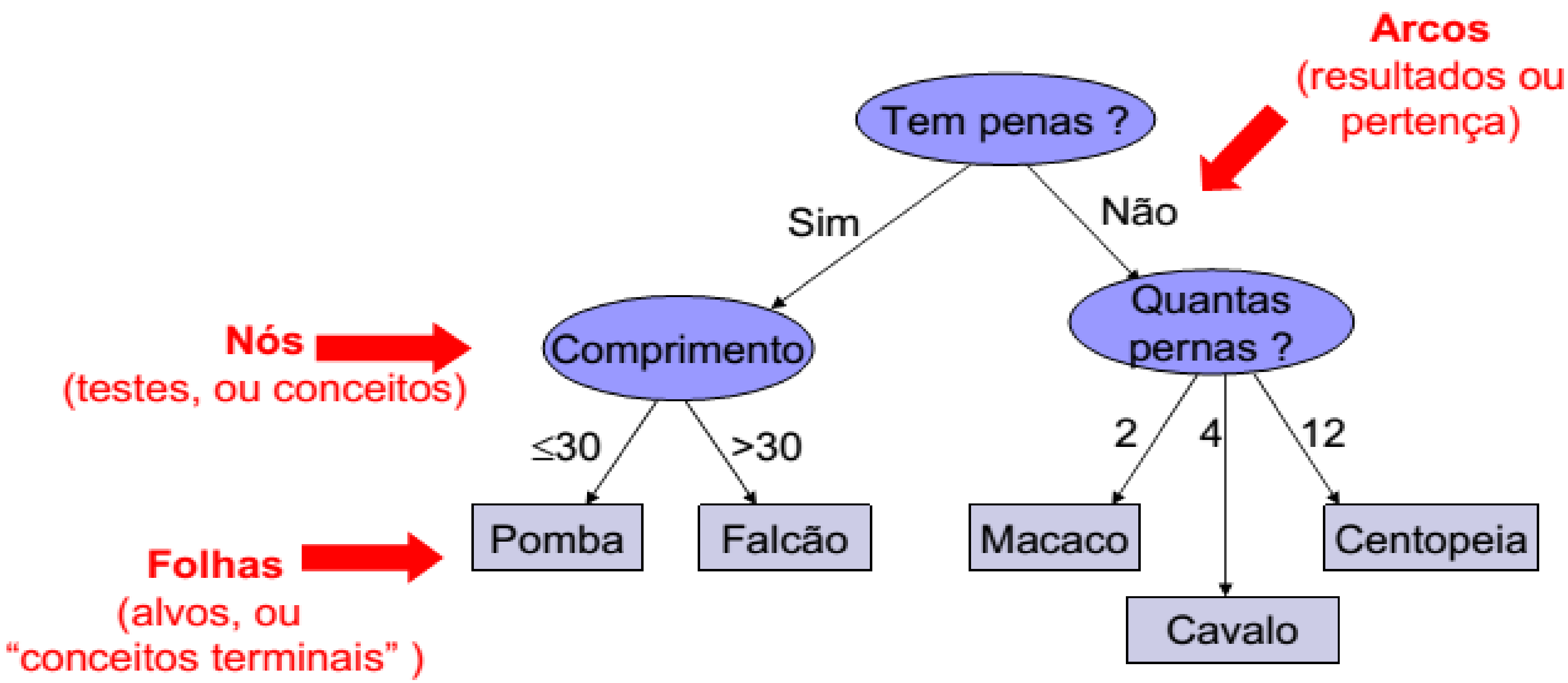
$$\geq \left(\frac{n}{2} - 1\right) \sum_{i=\frac{n}{2}}^{n} log_2 \frac{n}{2}$$

$$= \frac{n}{2} log_2 n - \frac{n}{2} - log_2 n + log_2 2$$

$$\geq \frac{n}{4} \log n, \ \ para\ \ n \geq 16$$

Árvore de Decisão

Exemplo: Ferramenta para tomada de decisão (ou Classificar)



Lista Mínima de Exercícios

Seção 5.3: 7, 10, 11, 13, 14, 15, 16, 18, 19, 20, 21, 23.