Teorema: Qualquer algoritmo de ordenação baseado em comparações executa em $\Omega(n \log n)$ no pior caso.

Ideia: Mostrar que qualquer algoritmo deste tipo realiza $\Omega(n \log n)$ comparações

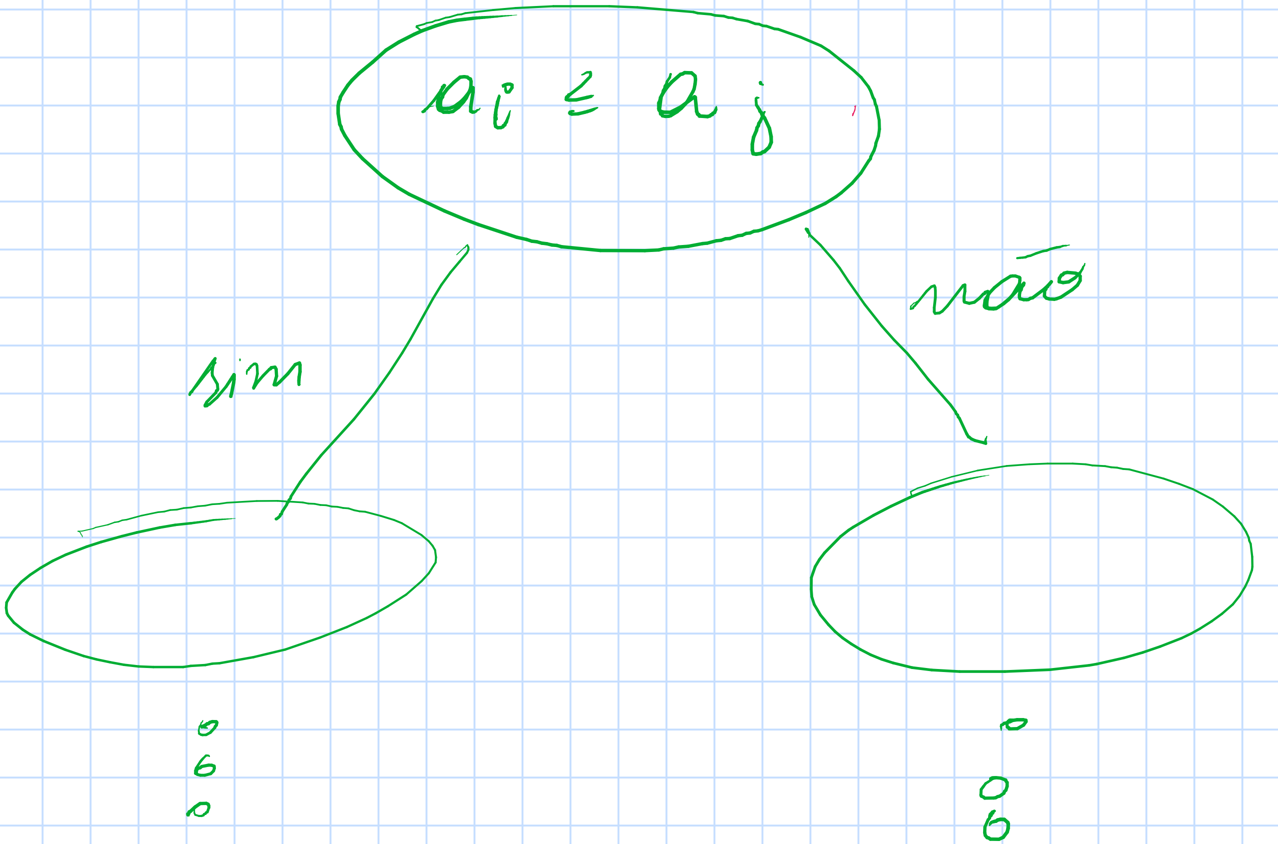
Prova:

- Representar um algoritmo como uma árvore de decisão, baseada nas comparações que o algoritmo executa, tal que as folhas representam todas as possíveis permutações de elementos.

→ árvore binária

- A árvore representa o fluxo do algoritmo:

entrada: $A = [a_1, a_2, \dots, a_n]$



todos os possíveis saídas

folhas: $[a_1, a_2, \dots, a_n] \dots [a_5, a_3, \dots a_1] \dots [a_n, a_{n-1}, \dots, a_1]$

OBS: - cada algoritmo tem sua árvore.

2 - caminho da raiz até folhas corresponde a uma execução.

- quantas folhas?

↳ Entrada de n elementos

n! folhas

- Como a análise da complexidade de pior caso do algoritmo está representada na árvore?

no maior caminho

raiz - folha = altura da árvore.

- Vamos calcular o limite inferior para a altura h da árvore.

↳ Quantas folhas tem uma árvore binária de altura h? 2^h no máximo (completa)

- Na nossa árvore, temos $n!$ folhas.

Logo:

$$n! \leq 2^h$$

$$\log n! \leq h$$

$$h = \Omega(n \log n)$$ → Aproximação Stirling

[redacted]