

Algoritmo Intercola:

Entrada: Vetor $A[p..r]$

tal que $A[p..q]$ e

$A[q+1..r]$ estão ordenados

Saída: $A[p..r]$ ordenado.

Intercola (A, p, q, r)

$$n_1 \leftarrow q - p + 1$$

$$n_2 \leftarrow r - q$$

crie vetores $L[1..n_1+1]$
e $R[1..n_2+1]$

$$L[n_1+1] \leftarrow \infty$$

$$R[n_2+1] \leftarrow \infty$$

Para $i \leftarrow 1$ até n_1
 $L[i] \leftarrow A[p+i-1]$

Para $i \leftarrow 1$ até n_2
 $R[i] \leftarrow A[q+i]$

$$i \leftarrow 1$$

$$j \leftarrow 1$$

Para $k \leftarrow p$ até r

Se $L[i] \leq R[j]$
 $A[k] \leftarrow L[i]$
 $i \leftarrow i + 1$

Senão

$A[k] \leftarrow R[j]$
 $j \leftarrow j + 1$

Complexidade $\Theta(n)$

Invariante do último loop:

$A[p..k-1]$ contém os $k-p$ menores elementos de L e R , ordenados. Além disso $L[i]$ e $R[j]$ são os menores elementos em seus vetores que não foram copiados de volta em A .

Intercola (A, p, q, r)

$$n_1 \leftarrow q - p + 1$$

$$n_2 \leftarrow r - q$$

Crie vetores $L[1..n_1+1]$
e $R[1..n_2+1]$

$$L[n_1+1] \leftarrow \infty$$

$$R[n_2+1] \leftarrow \infty$$

Para $i \leftarrow 1$ até n_1
 $L[i] \leftarrow A[p+i-1]$

Para $i \leftarrow 1$ até n_2
 $R[i] \leftarrow A[q+i]$

$$i \leftarrow 1$$

$$j \leftarrow 1$$

Para $k \leftarrow p$ até r

Se $L[i] \leq R[j]$
 $A[k] \leftarrow L[i]$
 $i \leftarrow i + 1$

Senão

$A[k] \leftarrow R[j]$
 $j \leftarrow j + 1$

Merge Sort (A, p, r)

1 Se $p < r$

2 $q \leftarrow \left\lfloor \frac{p+r}{2} \right\rfloor$

3 Merge Sort (A, p, q)

4 Merge Sort ($A, q+1, r$)

5 Intercala (A, p, q, r)

Divisão e Conquista:

↳ Dividir em subproblemas

↳ Conquistar resolvendo subproblemas recursivamente

↳ Combinar as soluções

Merge Sort (A, p, r)


1 Se $p < r$

2 $q \leftarrow \lfloor \frac{p+r}{2} \rfloor$

3 Merge Sort (A, p, q)

4 Merge Sort (A, q+1, r)

5 Intercala (A, p, q, r)

Comt.: Na linha 4, o mesmo acontece para a segunda metade do vetor! Na linha 5, a rotina Intercala é chamada. Sua complexidade segue da complexidade de Intercala. 

Teorema: Merge Sort

ordena corretamente um vetor de n elementos.

Prova: Indução em n

Base: $n = 1$

Algoritmo retorna. Vetor de 1 elemento está trivialmente ordenado.

hipótese: Merge Sort

ordena corretamente vetores $n < k$ elementos.

passo: considerar como $(n = k)$.

Supondo $p < r$.

Na linha 3, o algoritmo é chamado recursivamente para um vetor de $k/2$ elementos. Pela hipótese de k menor, a primeira metade do vetor é ordenada corretamente.

Merge Sort (A, p, r)

Se $p < r$

$$q \leftarrow \left\lfloor \frac{p+r}{2} \right\rfloor$$

Merge Sort (A, p, q)

Merge Sort (A, q+1, r)

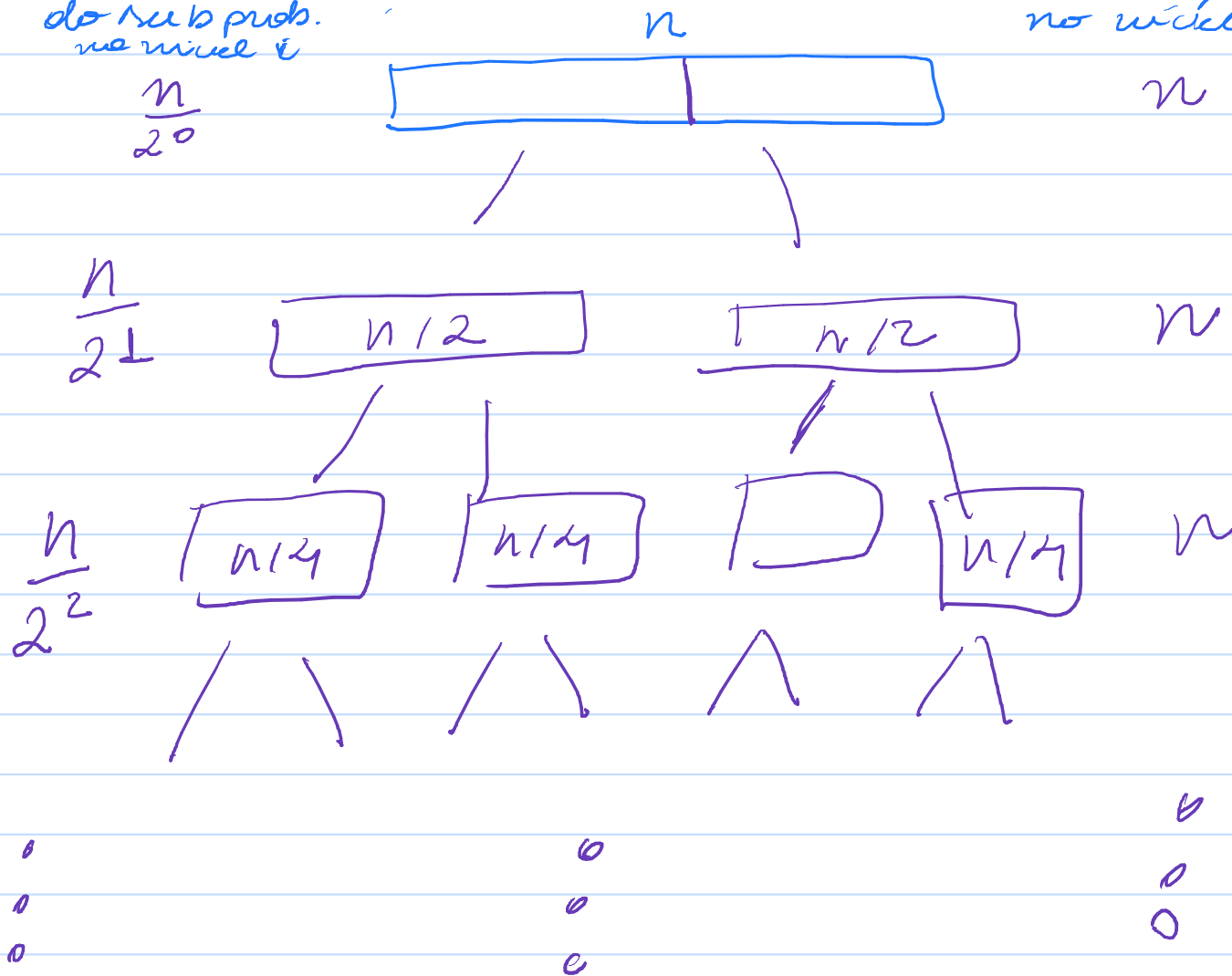
Intercala (A, p, q, r)

$\Theta(n \log n)$

altura da
árvore = $\log n$

↓ a maioria
dos sub-prob.
no nível i
 $\frac{n}{2^i}$

Total gasto
no nível i
 n



$$\frac{n}{2^i} = 1 \Rightarrow n = 2^i$$

$$\log n = \log 2^i \Rightarrow i = \log n$$