Estruturas de Dados

Aula 4.2 – Algoritmos de Ordenação - Selection Sort e Merge Sort

Prof. Ana Carolina Sokolonski

Bacharelado em Sistemas de Informação
Instituto Federal da Bahia – Campus Feira de Santana

2026

Agenda

Selection Sort

Merge Sort

Comparação Geral

Referências

Selection Sort

Ordenação por Seleção

Selection Sort — Conceito

▪ Como funciona

Seleciona o **menor elemento** da sequência não ordenada e o coloca na primeira posição disponível. Repete o processo com o restante do vetor.

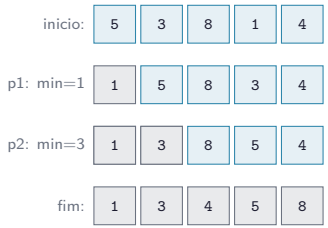
É intuitivo: se pedirmos para alguém ordenar números, há uma boa chance de que aplique exatamente esse algoritmo. [IME 2023]

▪ Complexidade

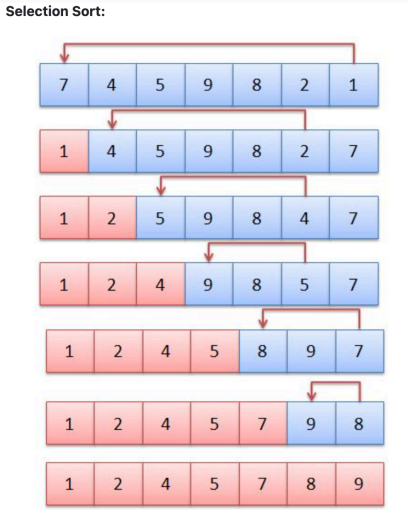
- Pior e melhor caso: $O(N^2)$
- Espaço: $O(1)$ — apenas uma variável auxiliar

Vídeo: `https://youtu.be/lyZQPjUT5B4`

Passo a passo



Selection Sort — Visualização



Selection Sort — Pseudocódigo e Código

Algorithm 1 `void selectionSort(int tam, int v[])`

```
1:  for i = 0 to N do
2:      menor = i
3:      for j = i to N do
4:          if v[j] < v[menor] then
5:              menor = j
6:          end if
7:      end for
8:      aux = v[menor]
9:      v[menor] = v[i]
10:     v[i] = aux
11: end for
```

```c
void selectionSort(int tam, int v[]){
    int aux, menor;
    for (int i = 0; i < tam; i++){
        menor = i;
        for(int j = i; j < tam; j++)
            if (v[j] < v[menor])
                menor = j;

        aux = v[menor];
        v[menor] = v[i];
        v[i] = aux;
    }
}
```

Merge Sort

Divisão e Conquista

Merge Sort — Conceito

Como funciona

Algoritmo de **divisão e conquista**: divide o vetor ao meio recursivamente até ter subvetores de tamanho 1, depois os **mescla** (*merge*) em ordem.

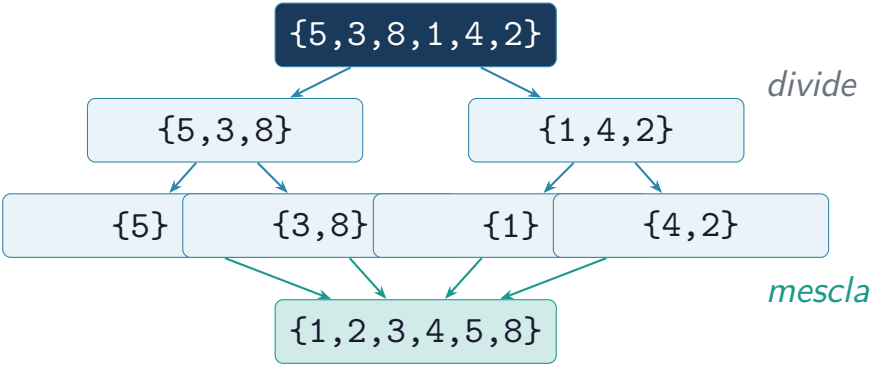
A rotina **Merge** é o coração do algoritmo — combina dois subvetores já ordenados em um único vetor. [D.E. Knuth 1973]

Complexidade

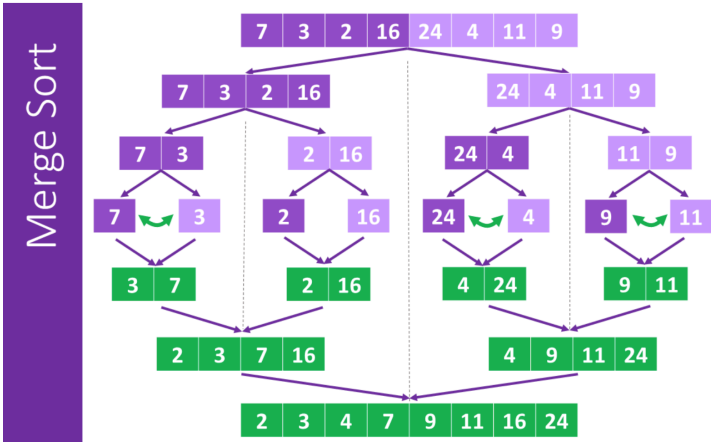
- Melhor, médio e pior caso: $O(N \log N)$
- Espaço extra: $O(N)$ — vetores auxiliares

Vídeo: `https://youtu.be/XaqR3G_NVoo`
Simulação: `https://www.101computing.net/Merge-Sort-algorithm/`

Merge Sort



Merge Sort — Visualização



Merge Sort — Pseudocódigo (parte 1)

Algorithm 2 `void mergeSort(int V[], int l, int r)`

1: **if** $l < r$ **then**
2: $m = (l + r)/2$
3: mergeSort(V, l, m)
4: mergeSort$(V, m + 1, r)$
5: merge(V, l, m, r)
6: **end if**

▪ Estrutura recursiva

Divide ao meio, ordena cada metade e mescla. Caso base: $l \geq r$ (subvetor de 1 elemento).

Algorithm 3 `void merge(int V[], int l, int m, int r) -- parte 1`

1: $n1 = m - l + 1; \quad n2 = r - m$
2: **for** $i = 0$ **to** $n1$ **do**
3: $L[i] = V[l + i]$
4: **end for**
5: **for** $j = 0$ **to** $n2$ **do**
6: $R[j] = V[m + 1 + j]$
7: **end for**
8: $i = 0;\ j = 0;\ k = l$
9: *(continua no próximo slide)*

Merge Sort — Pseudocódigo (parte 2)

Algorithm 4 `void merge(int V[], int l, int m, int r) -- parte 2/2`

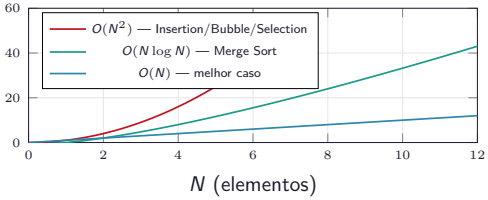
```
1: ··· (continuação de merge)
2: while (i < n1) and (j < n2) do
3:     if L[i] ≤ R[j] then
4:         V[k] = L[i];  i++
5:     else
6:         V[k] = R[j];  j++
7:     end if
8:     k++
9: end while
10: while i < n1 do
11:     V[k++] = L[i++]
12: end while
13: while j < n2 do
14:     V[k++] = R[j++]
15: end while
```

Comparação Geral

Os 4 algoritmos de ordenação

Comparando os Algoritmos de Ordenação

Algoritmo	**Pior caso**	**Melhor caso**	**Espaço**
Insertion Sort	$O(N^2)$	$O(N)$	$O(1)$
Bubble Sort	$O(N^2)$	$O(N)$	$O(1)$
Selection Sort	$O(N^2)$	$O(N^2)$	$O(1)$
Merge Sort	$\mathbf{O(N \log N)}$	$\mathbf{O(N \log N)}$	$O(N)$



Referências I

D.E. Knuth. *The Art of Computer Programming*. [S.l.]: Addison-Wesley, 1973. v. 1 - 3.

IME. *Aulas de Estruturas de Dados*. 2023. `https://www.ime.usp.br/~pf/estruturas-de-dados/aulas/index.html`. [Online; accessed 13-February-2023].