EXPERIMENTO COM MÉTODOS DE ORDENAÇÃO

Giovana Piazza e Rodrigo Peraça

INTRODUÇÃO

- Ordenação de dados é essencial em computação.
- Afeta o desempenho de sistemas e algoritmos.
- Objetivo: comparar algoritmos clássicos (Bubble, Insertion, Selection, Shell, Heap, Merge, Quick) e lineares (Counting, Radix, Bucket).
- Avaliação em vetores de diferentes tamanhos e tipos: crescente, decrescente e aleatório.
- Análise considera complexidade assintótica e comportamento prático.

METODOLOGIA

- Cada algoritmo executado 20 vezes por tamanho de vetor.
- Vetores testados: 100, 1.000, 10.000, 100.000 e 1.000.000 elementos.
- Tipos de vetores: crescente, decrescente e aleatório.
- Vetores com 10.000 elementos em ordem crescente/decrescente testados 20 vezes.
- Tempos medidos em nanossegundos, convertidos para milissegundos.

1. CÓDIGO

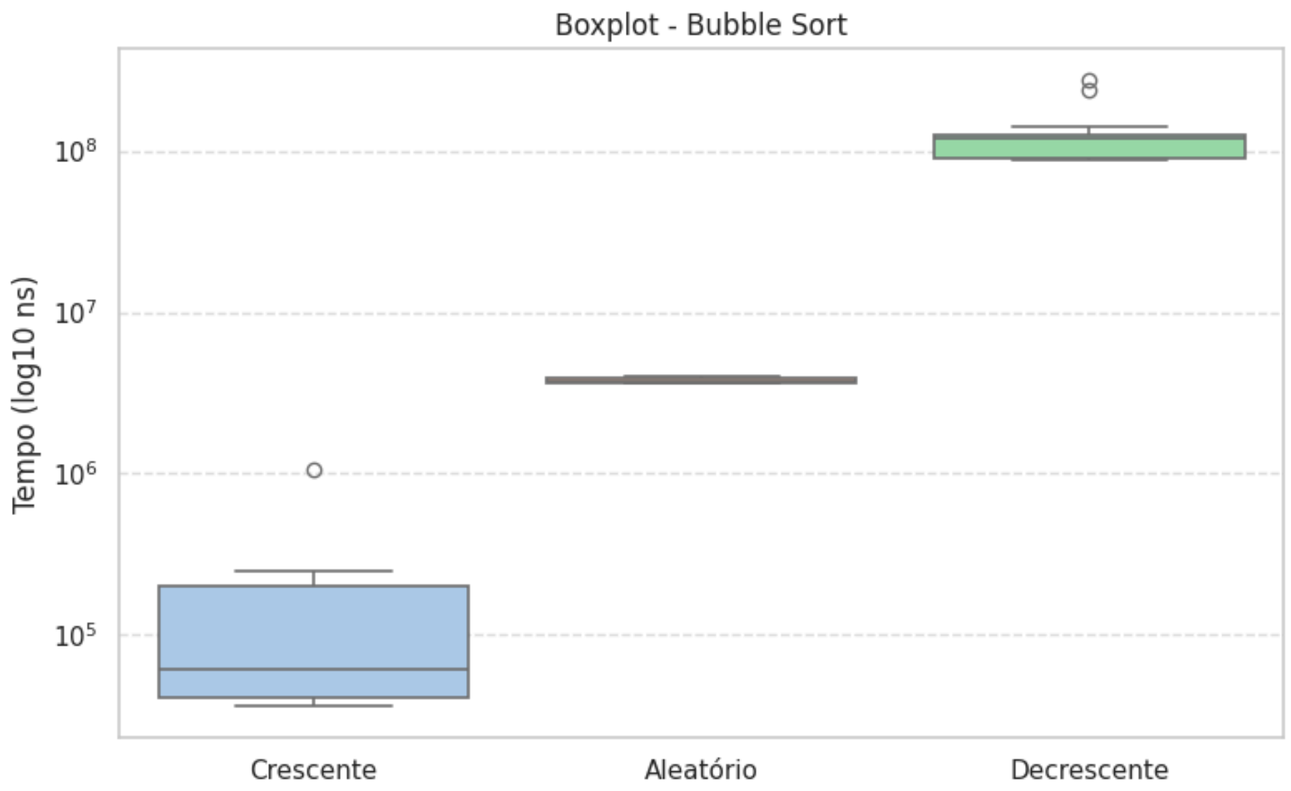
- Programa principal em Java: Main.java.
- Cada algoritmo encapsulado em métodos da classe Ordenadores.
- Modularidade facilita comparação entre algoritmos.
- Vetores gerados pela biblioteca java.util.Random utilizando random.nextInt(100000) e random.nextDouble() para o Bucket.
- Mede empiricamente o tempo de execução para diferentes tamanhos e tipos de vetores.
- Ferramenta de experimentação e análise prática.

2. ALGORITMOS

- Algoritmos tirados do moodle da cadeira Algoritmos e Classificação de dados.
- Para tamanho de vetor foi testado com valores randômicos.
- Todos os tempos foram marcados pela função System.nanoTime().
- Vetores gerados no tipo Int (Bubble, Insertion, Selection, Shell, Heap, Merge e Quick Sort), Primitivo (Counting e Radix) e double (Bucket).

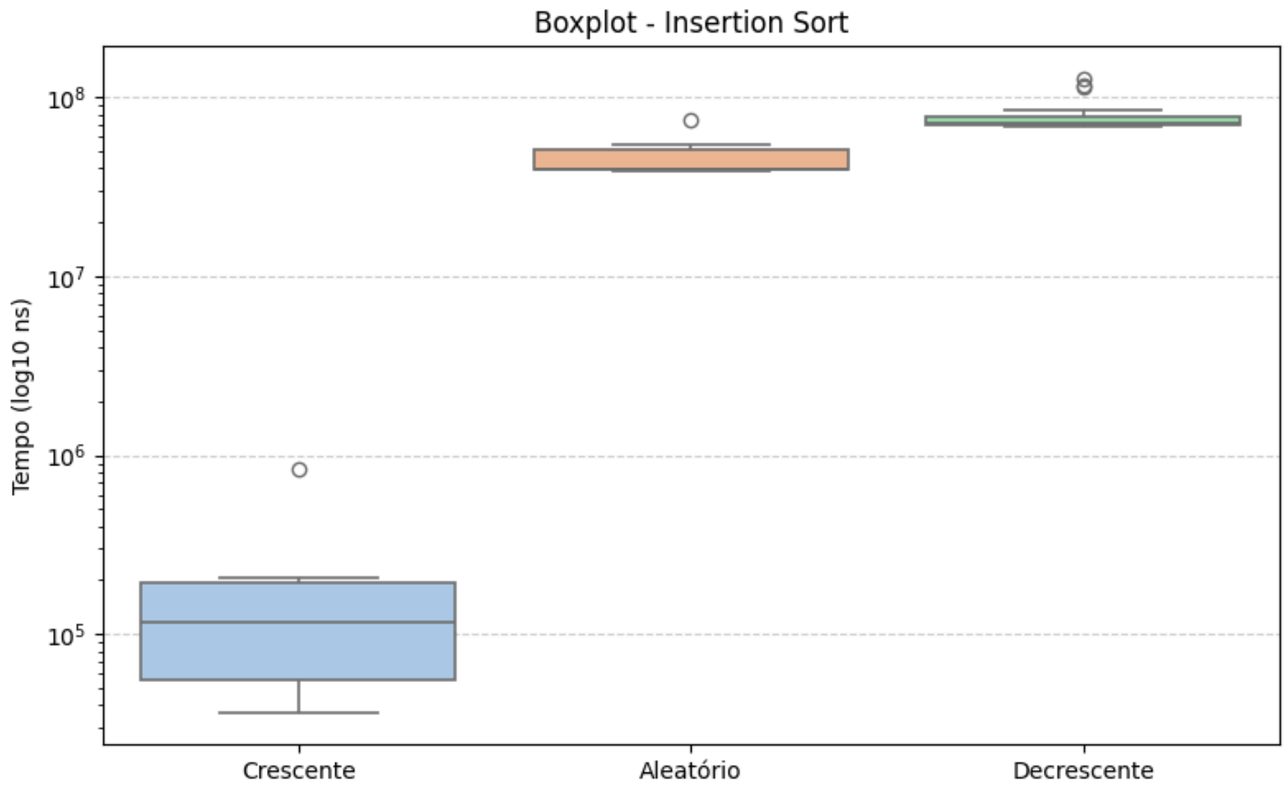
2. QUADRATICOS

2. BUBBLE SORT



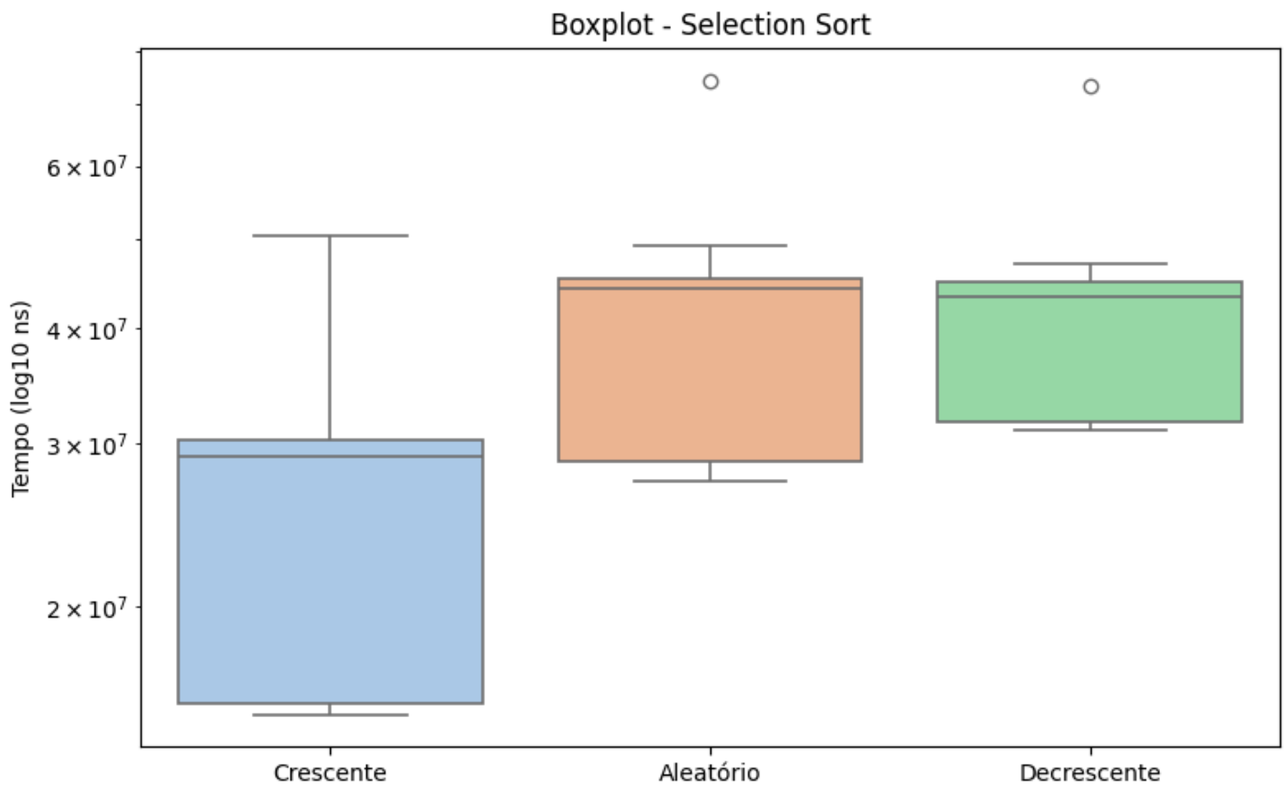
- Crescente:
  - Média ≈ 0.26 ms;
  - Mediana ≈ 0.20 ms.
- Aleatória:
  - Média ≈ 3.81 ms;
  - Mediana ≈ 3.88 ms.
- Decrescente:
  - Média ≈ 136.72 ms;
  - Mediana ≈ 130.22 ms.

2. INSERTION SORT



- Crescente:
  - Média ≈ 0.16 ms;
  - Mediana ≈ 0.14 ms.
- Aleatória:
  - Média ≈ 44.48 ms;
  - Mediana ≈ 39.71 ms.
- Decrescente:
  - Média ≈ 81.26 ms;
  - Mediana ≈ 70.75 ms.

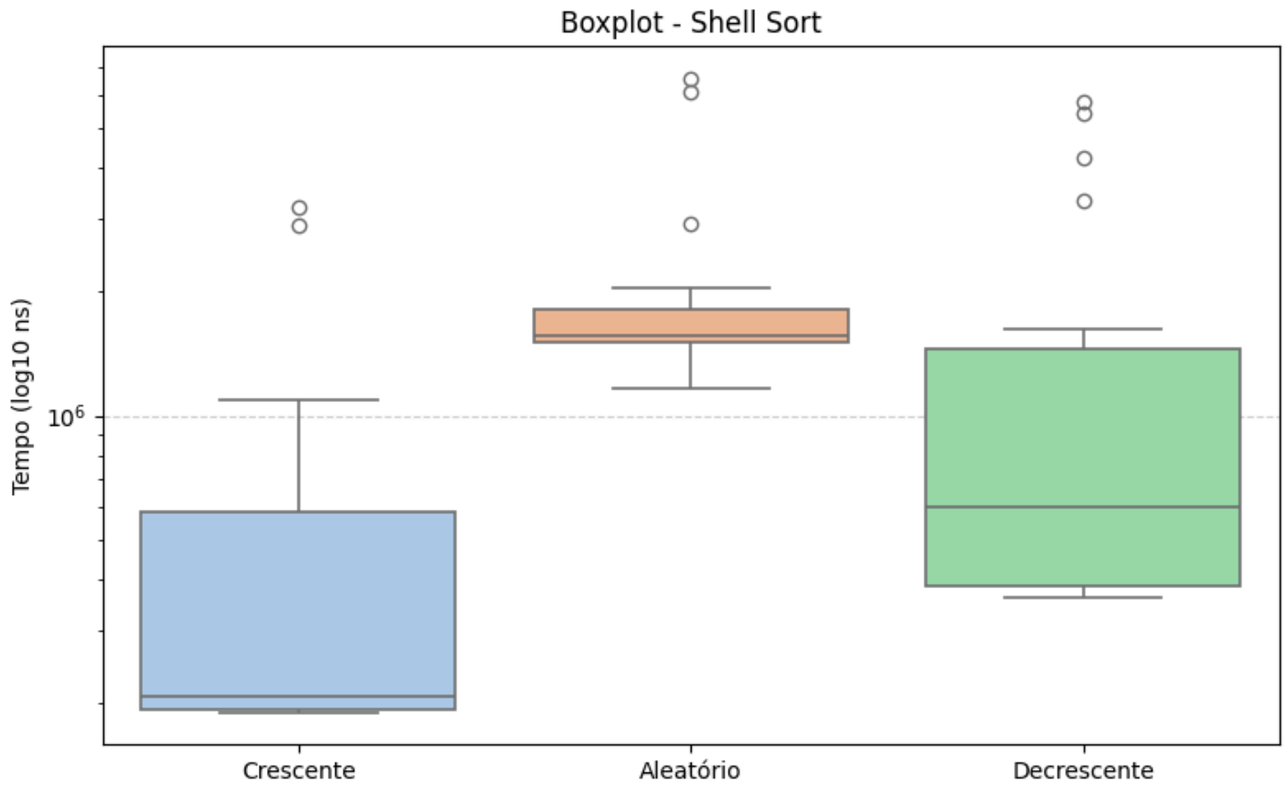
2. SELECTION SORT



- Crescente:
  - Média ≈ 27,073.45 ms;
  - Mediana ≈ 29,149.60 ms.
- Aleatória:
  - Média ≈ 39,089.70 ms;
  - Mediana ≈ 44,313.30 ms.
- Decrescente:
  - Média ≈ 40,367.40 ms;
  - Mediana ≈ 43,458.40 ms.

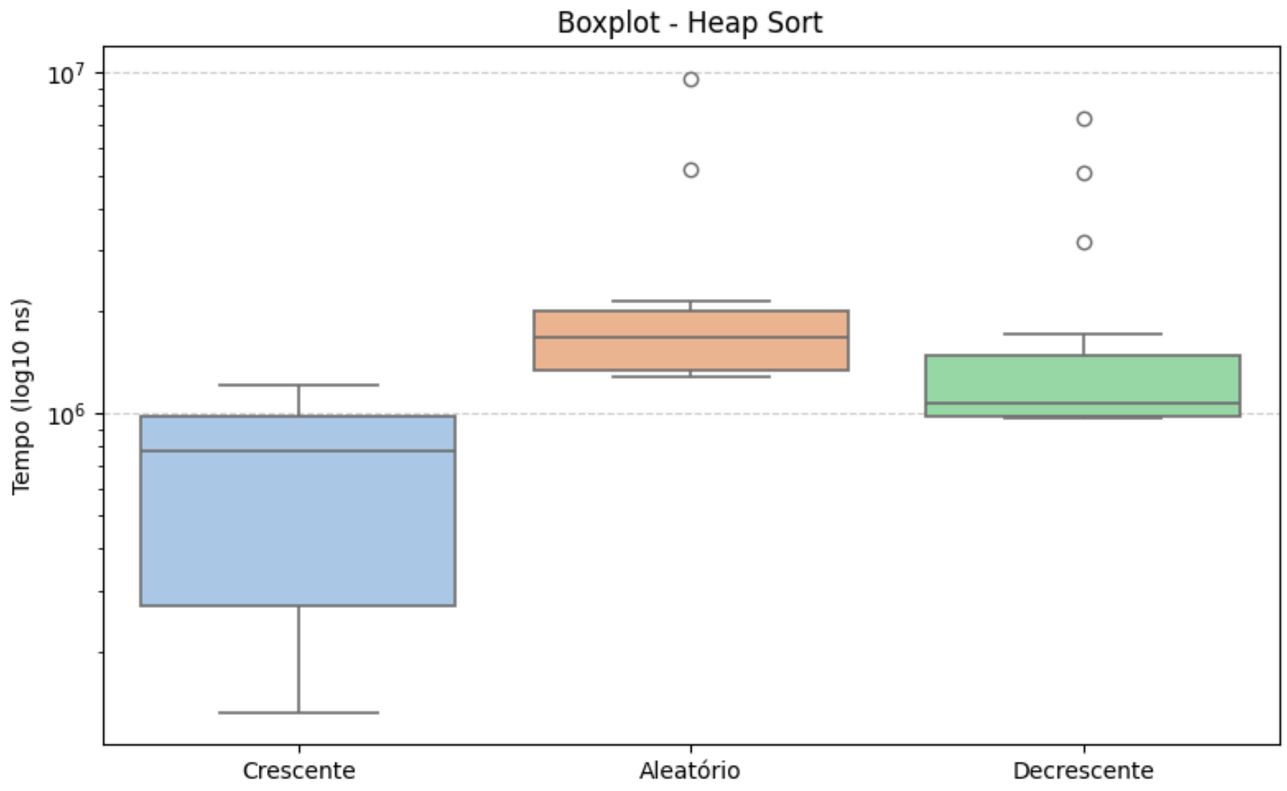
2. LOGARITMOS

2. SHELL SORT



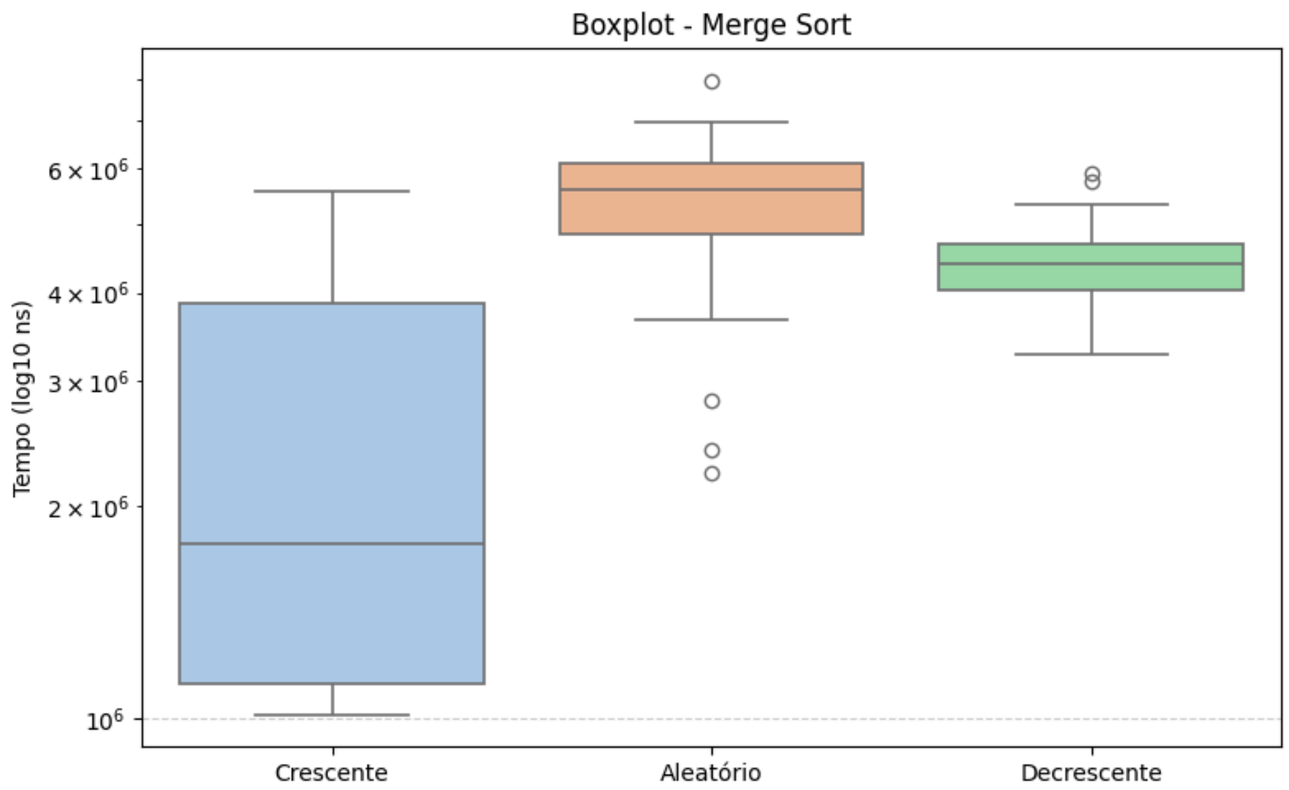
- Crescente:
  - Média ≈ 1.08 ms;
  - Mediana ≈ 0.20 ms.
- Aleatória:
  - Média ≈ 2.05 ms;
  - Mediana ≈ 1.55 ms.
- Decrescente:
  - Média ≈ 1.78 ms;
  - Mediana ≈ 0.63 ms.

2. HEAP SORT



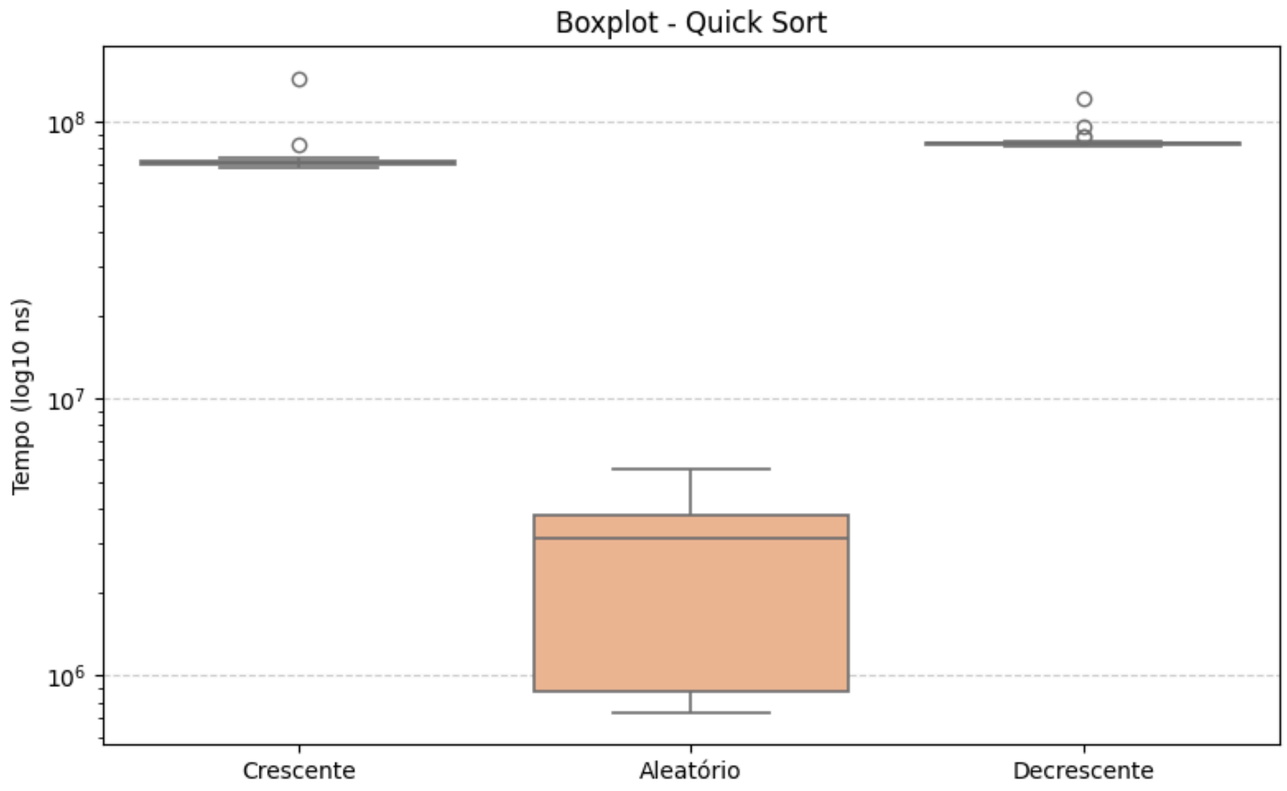
- Crescente:
  - Média ≈ 0.63 ms;
  - Mediana ≈ 0.26 ms.
- Aleatória:
  - Média ≈ 2.38 ms;
  - Mediana ≈ 1.56 ms.
- Decrescente:
  - Média ≈ 2.01 ms;
  - Mediana ≈ 1.26 ms.

2. MERGE SORT



- Crescente:
  - Média ≈ 2.42 ms;
  - Mediana ≈ 1.18 ms.
- Aleatória:
  - Média ≈ 5.38 ms;
  - Mediana ≈ 5.61 ms.
- Decrescente:
  - Média ≈ 4.47 ms;
  - Mediana ≈ 4.50 ms.

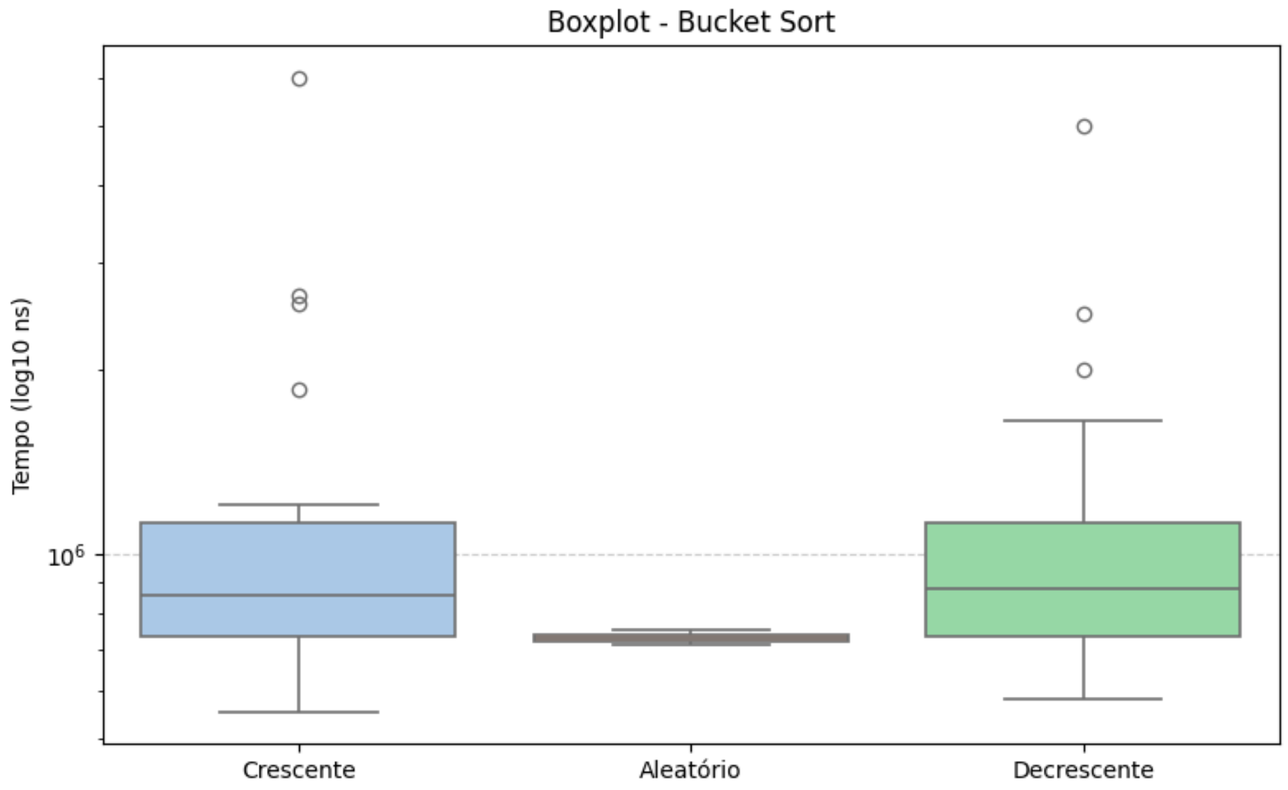
2. QUICK SORT



- Crescente:
  - Média ≈ 70.50 ms;
  - Mediana ≈ 70.64 ms.
- Aleatória:
  - Média ≈ 2.71 ms;
  - Mediana ≈ 3.56 ms.
- Decrescente:
  - Média ≈ 84.43 ms;
  - Mediana ≈ 83.26 ms.

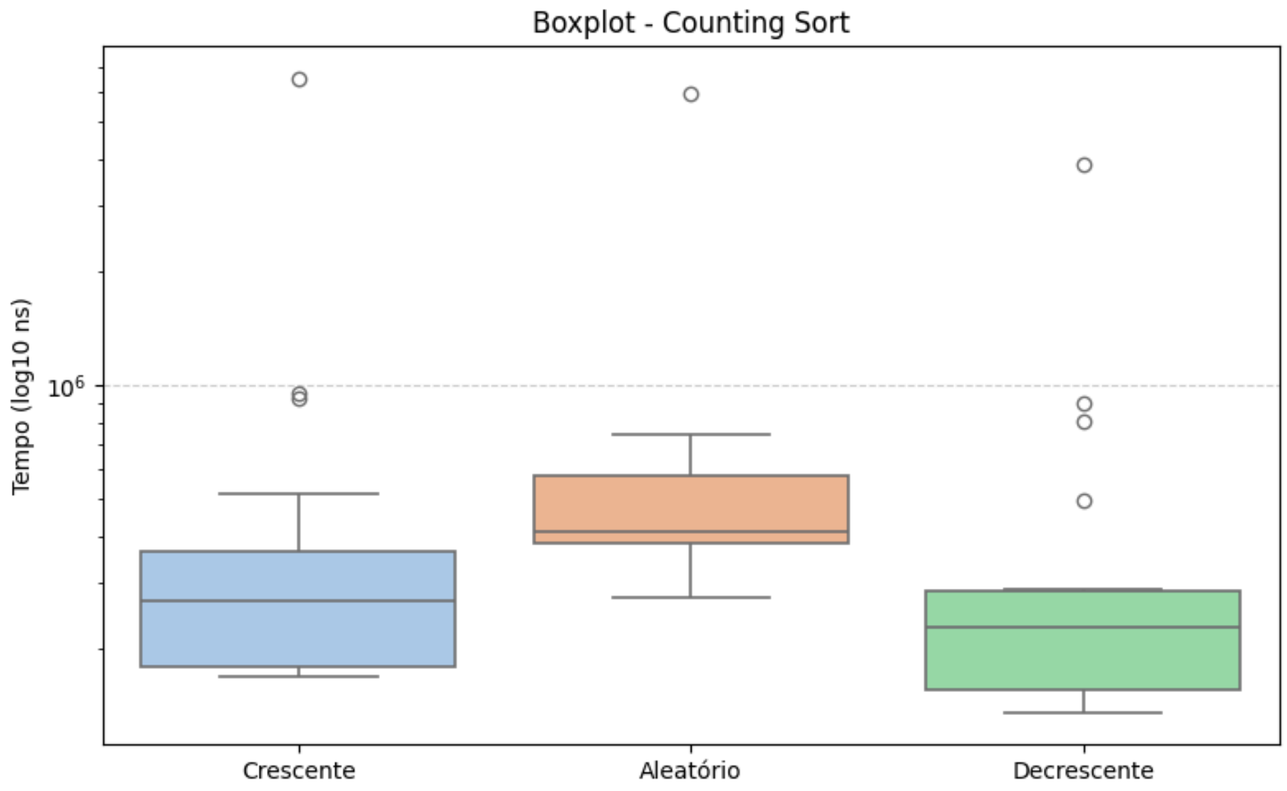
2. LINEARES

2. BUCKET SORT



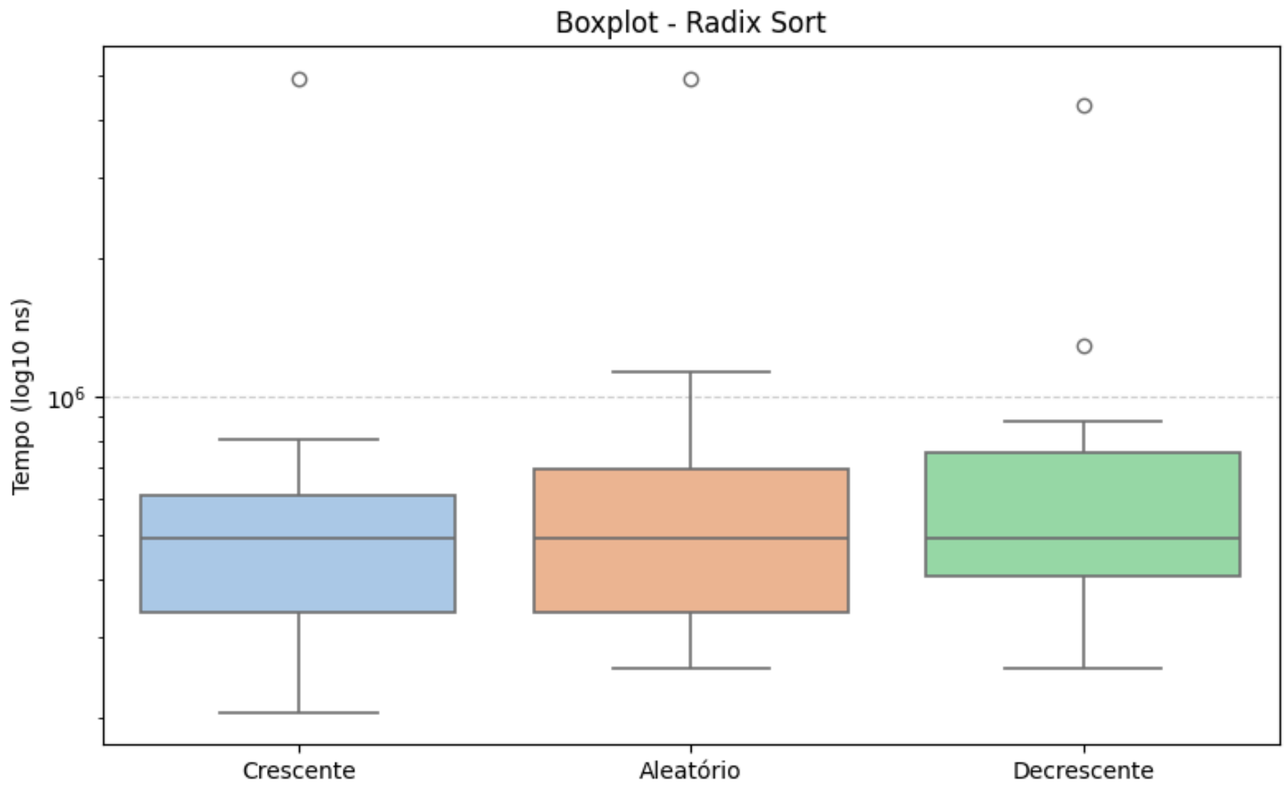
- Crescente:
  - Média ≈ 1.27 ms;
  - Mediana ≈ 0.77 ms.
- Aleatória:
  - Média ≈ 0.73 ms;
  - Mediana ≈ 0.73 ms.
- Decrescente:
  - Média ≈ 1.05 ms;
  - Mediana ≈ 0.86 ms.

2. COUNTING SORT



- Crescente:
  - Média ≈ 1.65 ms;
  - Mediana ≈ 0.28 ms.
- Aleatória:
  - Média ≈ 1.63 ms;
  - Mediana ≈ 0.41 ms.
- Decrescente:
  - Média ≈ 0.66 ms;
  - Mediana ≈ 0.23 ms.

2. RADIX SORT

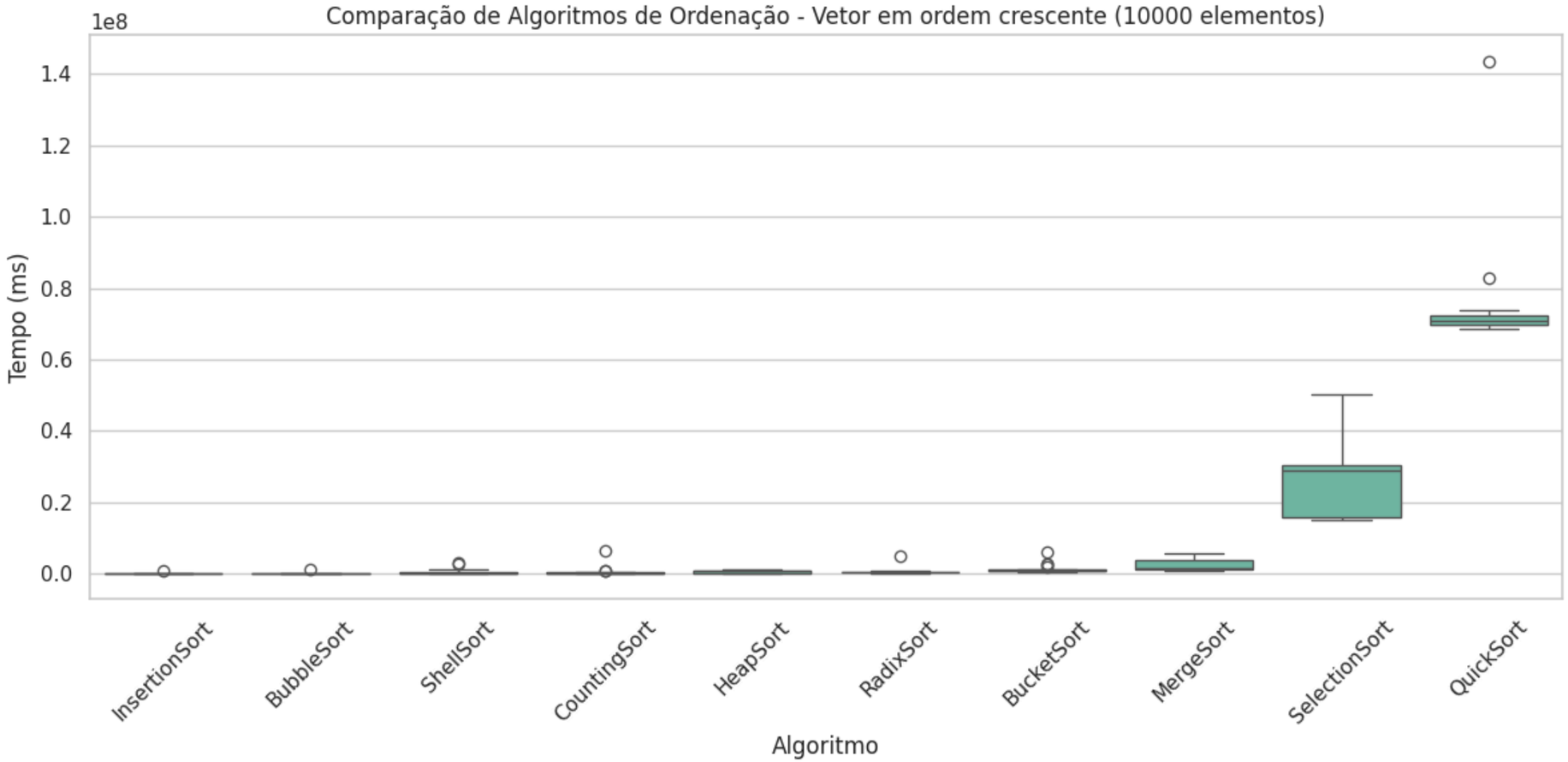


- Crescente:
  - Média ≈ 1.15 ms;
  - Mediana ≈ 0.34 ms.
- Aleatória:
  - Média ≈ 0.99 ms;
  - Mediana ≈ 0.48 ms.
- Decrescente:
  - Média ≈ 1.11 ms;
  - Mediana ≈ 0.45 ms.

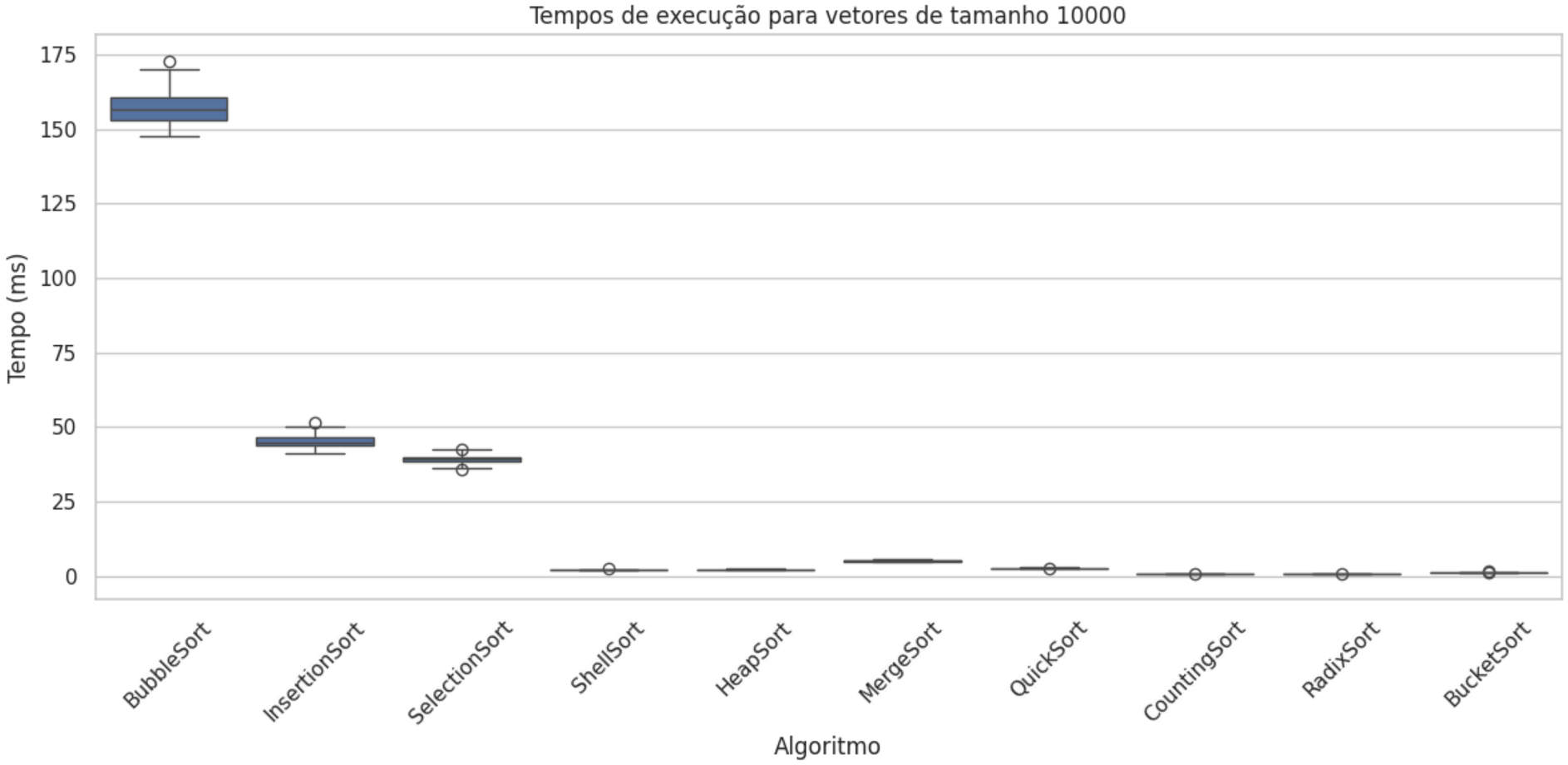
3. RESULTADOS

Análise geral dos tempos de execução dos algoritmos com vetor de 10 mil elementos em diferentes ordens (crescente, aleatória e decrescente).

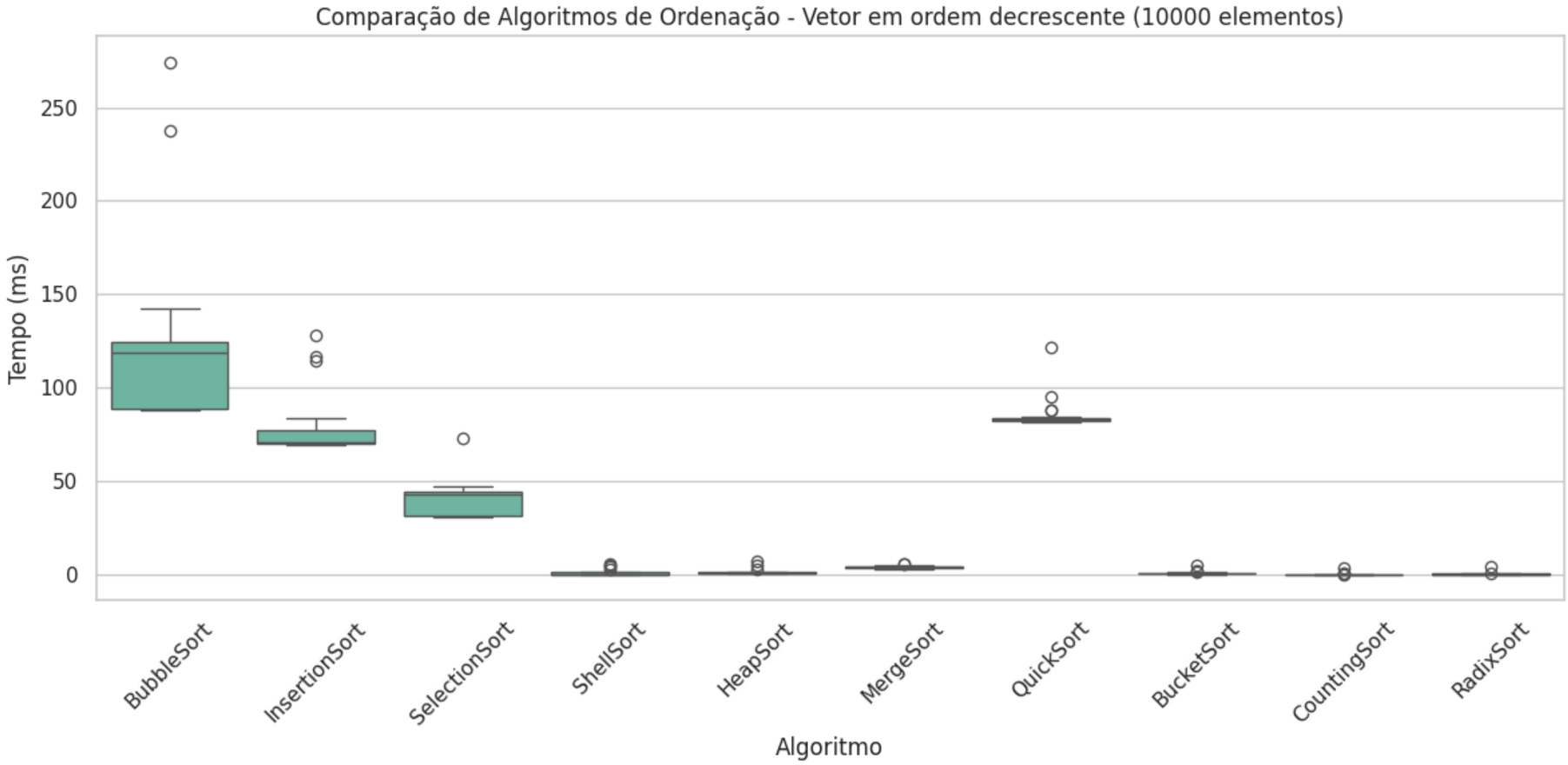
3. CRESCENTE


Comparação de Algoritmos de Ordenação - Vetor em ordem crescente (10000 elementos)
1e8
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
Tempo (ms)
InsertionSort
BubbleSort
ShellSort
CountingSort
HeapSort
RadixSort
BucketSort
MergeSort
SelectionSort
QuickSort
Algoritmo

2 ALEATORIO


Tempos de execução para vetores de tamanho 10000
Tempo (ms)
175
150
125
100
75
50
25
0
BubbleSort
InsertionSort
SelectionSort
ShellSort
HeapSort
MergeSort
QuickSort
CountingSort
RadixSort
BucketSort
Algoritmo

3. DECRESCENTE



4. ANÁLISE DOS RESULTADOS

- Algoritmos quadráticos: Bubble, Insertion e Selection;
- Algoritmos logaritmos: Shell, Heap, Merge e Quick Sort;
- Algoritmos lineares ou quase lineares: Counting, Radix e Bucket Sort.

4. ANÁLISE DOS RESULTADOS

- Algoritmos quadráticos: Bubble, Insertion, Selection
- Realizam muitas comparações e trocas.
- Bubble Sort é o mais lento;
- Selection Sort realiza menos trocas;
- Insertion Sort se beneficia de vetores quase ordenados;
- Os boxplots mostram alta dispersão e sensibilidade à ordem inicial dos dados.

4. ANÁLISE DOS RESULTADOS

- Algoritmos logaritmos: Shell, Heap, Merge e Quick Sort

- Têm desempenho O(n log n) ou próximo disso.
- Merge Sort é estável e consistente;
- Quick Sort é rápido na média;
- Heap Sort eficiente e determinístico;
- Shell Sort se aproxima de O(n log n);
- Boxplots indicam tempos menores e mais estáveis que os quadráticos.

4. ANÁLISE DOS RESULTADOS

- Algoritmos lineares ou quase lineares: Counting, Radix e Bucket Sort

  - Tempos muito baixos, com pouca dispersão.
  - São altamente eficientes, especialmente para dados com valores limitados ou distribuídos uniformemente.
  - Os boxplots revelam tempos muito baixos e com pouca dispersão, evidenciando a estabilidade e a alta eficiência desses métodos

5. CONCLUSÃO

- Quadráticos: bons apenas para pequenos conjuntos.
- Divisão e Conquista: melhor equilíbrio entre velocidade e estabilidade.
- Lineares: eficientes somente em condições ideais.
- Quick Sort foi o mais rápido, enquanto Bubble Sort teve o pior desempenho.
- O estudo reforça que a escolha do algoritmo depende do contexto e deve considerar natureza dos dados e custo computacional.

REFERÊNCIAS

- https://www.devmedia.com.br/algoritmos-de-ordenacao-em-java/32693
- https://www.youtube.com/watch?v=dESIzDq4kIw&list=PLldrzuO43P_zNudi9VTp3RDPDBRbaWW6b
- https://moodle.unipampa.edu.br/moodle/course/view.php?id=25710

OBRIGADO!