Análise de Desempenho de Algoritmos Clássicos de Ordenação

Lucca Sander Frisso

5 de maio de 2025

Este trabalho apresenta uma análise comparativa do desempenho de quatro algoritmos de ordenação clássicos: Selection Sort, Insertion Sort, Bubble Sort e Quicksort. Foram analisados os tempos de execução, número de comparações e movimentações para entradas de diferentes tamanhos. Os resultados demonstram diferenças significativas entre os algoritmos, especialmente para grandes volumes de dados.

1 Introdução

A ordenação de dados é uma tarefa fundamental em computação. Diversos algoritmos foram propostos com diferentes eficiências. Este trabalho visa comparar o desempenho dos algoritmos Selection Sort, Insertion Sort, Bubble Sort e Quicksort, utilizando medidas de tempo, comparações e movimentações.

2 Metodologia

Cada algoritmo foi implementado em Python, com contadores de comparações e movimentações inseridos no código. Foram utilizados vetores de tamanhos 100, 1000 e 10000 com números inteiros aleatórios. Para cada execução, foram registrados:

- Tempo de execução (em milissegundos);
- Número de comparações realizadas;
- Número de movimentações (trocas ou inserções).

3 Resultados

Os resultados foram organizados em três gráficos, apresentados a seguir.

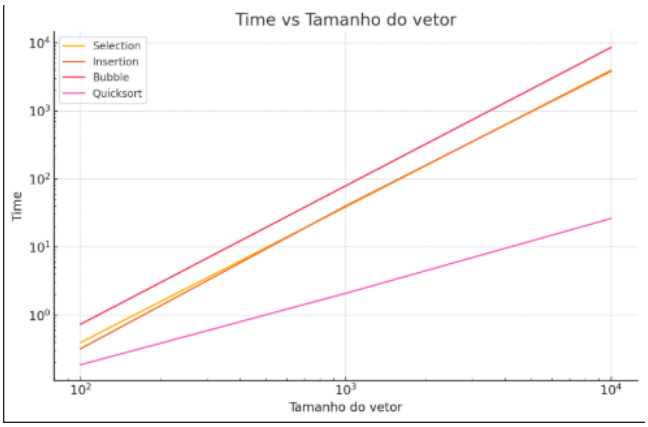

Figure 1: Tempo de execução dos algoritmos por tamanho de vetor.

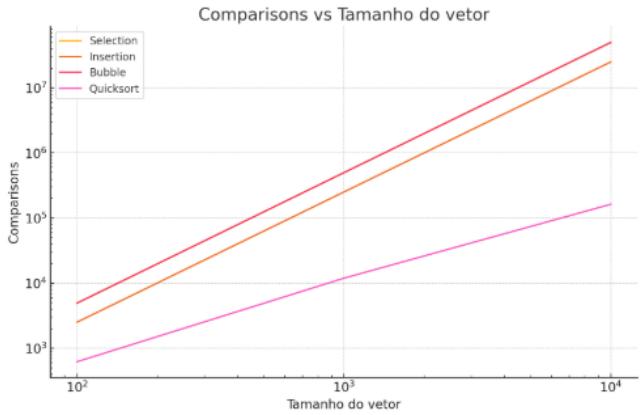

Figure 2: Número de comparações realizadas por algoritmo.

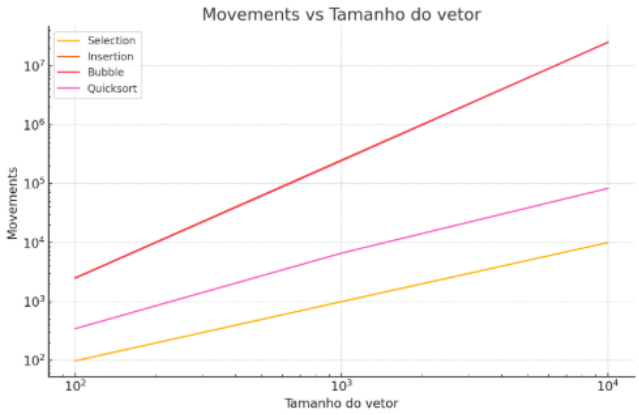

Figure 3: Número de movimentações realizadas por algoritmo.

4 Análise Crítica dos Resultados

Os algoritmos quadráticos (Selection, Insertion e Bubble Sort) apresentaram desempenho significativamente inferior ao Quicksort conforme o tamanho da entrada aumentou. Isso era esperado, dado que sua complexidade é $O(n^2)$, o que os torna inadequados para entradas maiores. Para vetores com 100 e 1000 elementos, os algoritmos mais simples ainda conseguem concluir a ordenação em tempo razoável, embora o número de comparações e movimentações já comece a crescer de forma expressiva. Já em vetores de 10.000 elementos, o tempo de execução dos algoritmos quadráticos torna-se muito alto, refletindo a ineficiência para grandes entradas. O Quicksort, por outro lado, se mostrou eficiente em todos os tamanhos de entrada, com crescimento sublinear no número de comparações e movimentações, justificando sua popularidade em aplicações práticas. Sua complexidade média é $O(n \log n)$, e ele se beneficia de boas estratégias de particionamento.

5 Conclusão

Conclui-se que algoritmos como o Bubble Sort, Insertion Sort e Selection Sort são adequados apenas para conjuntos de dados pequenos ou com propósitos didáticos. Para aplicações reais com grandes volumes de dados, o Quicksort é claramente superior em termos de desempenho geral, tanto em tempo de execução quanto em eficiência computacional.