Análise Comparativa de Algoritmos de Ordenação

Matheus Filipe Rocha Viana

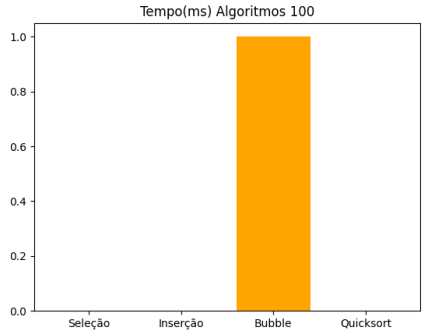
Resumo

Este estudo avalia o desempenho de quatro algoritmos clássicos de ordenação (Selection Sort, Insertion Sort, Bubble Sort e QuickSort) utilizando vetores com 100, 1.000, 10.000 e 100.000 elementos aleatórios. Foram medidos tempo de execução, número de comparações e movimentações. Os resultados mostram diferenças significativas entre os algoritmos, com o QuickSort se destacando como o mais eficiente para grandes volumes de dados, enquanto o Insertion Sort apresenta melhor desempenho em conjuntos pequenos.

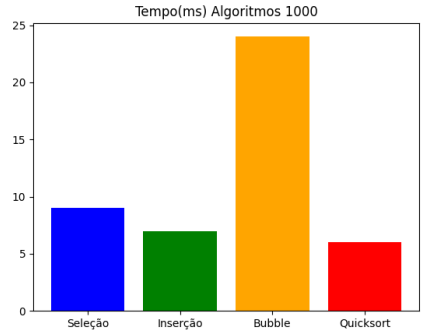
Resultados

Os testes realizados revelaram:

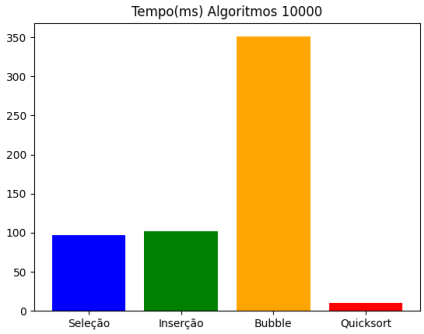
- **QuickSort**: Demonstrou complexidade $O(n \log n)$, sendo 100x mais rápido que os outros algoritmos para 100.000 elementos
- **Bubble Sort**: Apresentou o pior desempenho, com complexidade $O(n^2)$ e mais de 5 bilhões de comparações para 100.000 elementos
- **Insertion Sort**: Eficiente para pequenos conjuntos (até 1.000 elementos), com tempo menor que o QuickSort nesses casos
- **Selection Sort**: Menor número de movimentações ($O(n)$), mas alto custo em comparações



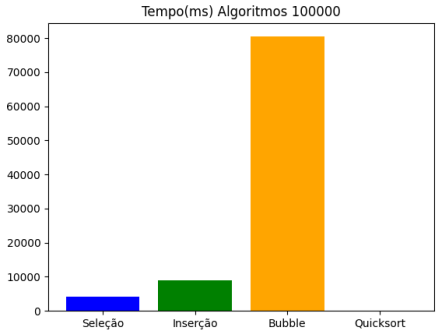
(a) 100 elementos



(b) 1.000 elementos

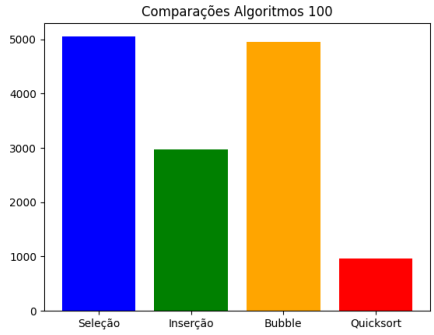


(c) 10.000 elementos

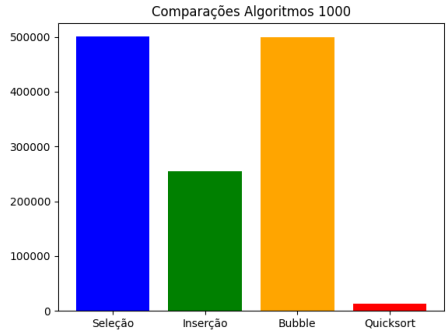


(d) 100.000 elementos

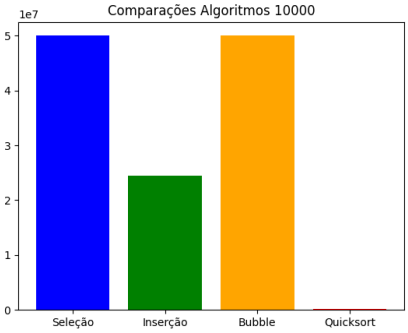
Figura 1: Tempos de execução comparativos para diferentes tamanhos de entrada



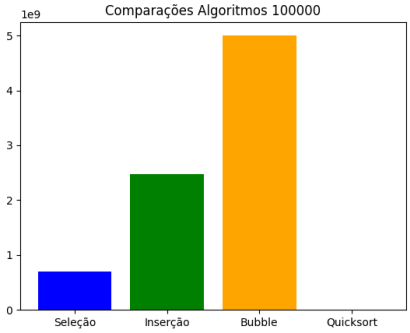
(a) 100 elementos



(b) 1.000 elementos

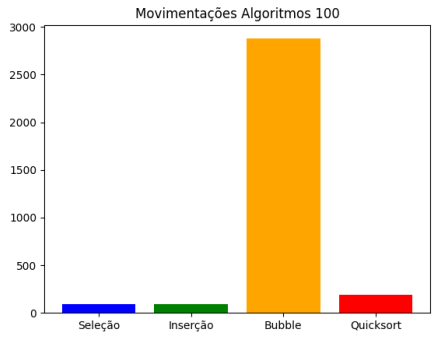


(c) 10.000 elementos

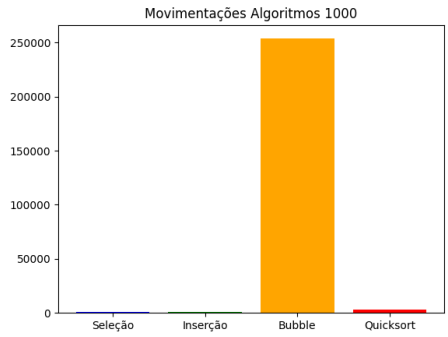


(d) 100.000 elementos

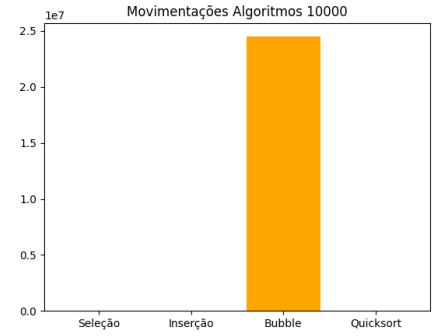
Figura 2: Número de comparações realizadas para diferentes tamanhos de entrada



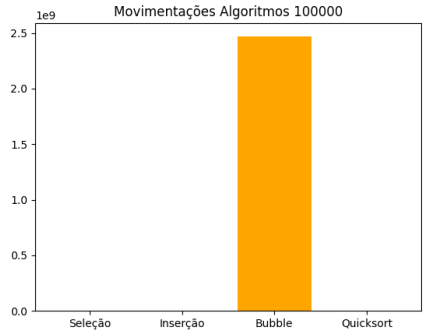
(a) 100 elementos



(b) 1.000 elementos



(c) 10.000 elementos



(d) 100.000 elementos

Figura 3: Número de movimentações realizadas para diferentes tamanhos de entrada

Conclusão

A análise permite concluir que:

- Para **grandes conjuntos de dados**, o QuickSort é a escolha ideal devido à sua eficiência comprovada
- O **Insertion Sort** é recomendado para aplicações com pequenos volumes de dados ou dados parcialmente ordenados
- O **Bubble Sort** mostrou-se ineficiente para uso prático em qualquer cenário
- O **Selection Sort** pode ser útil em situações onde movimentações são extremamente custosas

Em aplicações reais, recomenda-se priorizar o QuickSort na maioria dos casos, considerando alternativas apenas quando características específicas dos dados ou requisitos do sistema justifiquem.