Ordenação RA4

Leandro Cardoso Vieira

[1]Escola Politécnica da Pontifícia Universidade Católica do Paraná (PUCPR)

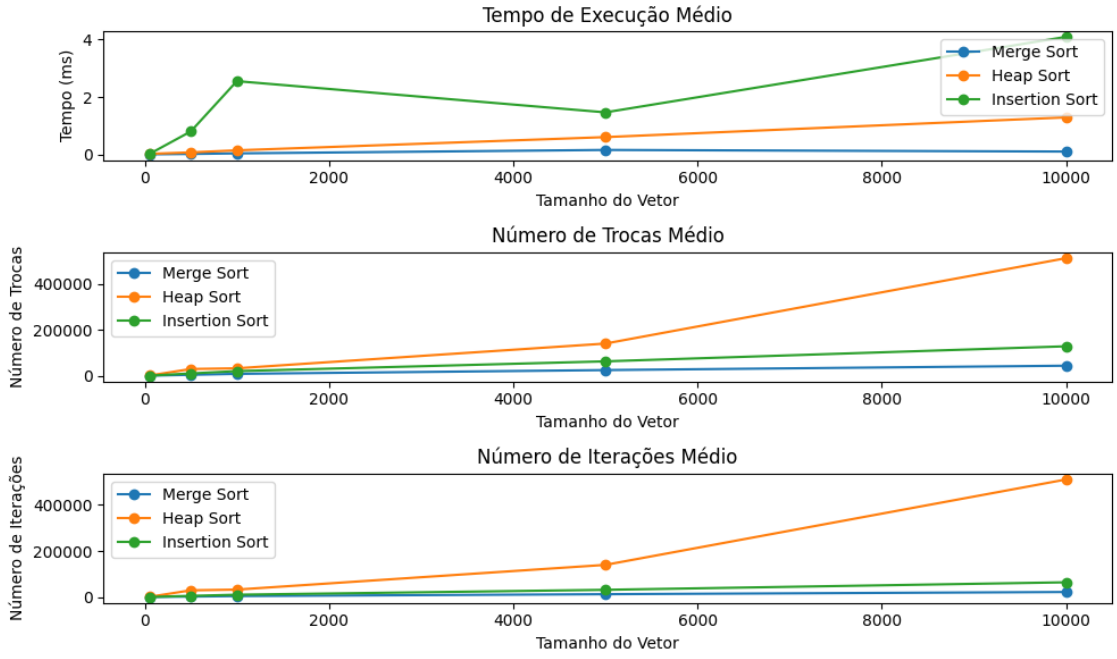
1. Introdução

Este relatório apresenta uma análise sobre a eficiência de algoritmos de ordenação.

2. Métodos

Foram criados cinco arrays de tamanhos [50,500,1000,5000,10000], onde foram inseridos números aleatórios entre 0 e 999. Depois, foram usados os algoritmos de ordenação e guardadas as informações sobre o Tempo de Execução, Número de Trocas e Número de Iterações. Após coletar esses dados foram separados pelo tamanho de array fazendo media.

3. Gráfico



4. Insertion Sort

Foi o algoritmo com maior tempo medio , ou seja o mais lento, em questão de trocas e iterações ele ficou no meio dos dois outros algoritmos.

5. Heap Sort

Foi o algoritmo mediano em tempo, porem ele foi por muito o que mais teve trocas e iterações, o que significa pouca eficiência e alto custo de máquina, apesar de seu tempo ser mediano considero o pior dos algoritmos testado.

6. Merge Sort

Foi o melhor algoritmo em todos os campos.

7. Conclusão

Dados os três algoritmos testados, sempre deve-se usar o Merge Sort, pois ele foi o menor tempo em todos os casos, o menor número de trocas e iterações, o que significa que ele gastou menos recurso de máquina e foi mais efetivo.

8. Repositório

Link para o git:
`https://github.com/Levic13/Ordena-o-RA-04/blob/main/SortingAlgorithms.java`