RELATÓRIO DE BENCHMARK DE ALGORITMOS DE ORDENAÇÃO

Alunos: Kaio Guilherme e Angelo Ferro

1. Introdução

Este projeto tem como objetivo analisar e comparar o desempenho de diferentes algoritmos de ordenação ao serem executados em dois processadores distintos: Apple M1 (ARM) e Intel Core i7-14700KF (x86). A avaliação leva em conta não apenas o tempo de execução, mas também o uso de CPU, consumo de memória e número médio de comparações realizadas.

2. Objetivos

- Comparar o desempenho de algoritmos de ordenação clássicos em diferentes arquiteturas.

- Avaliar como o comportamento dos algoritmos varia de acordo com o tipo de entrada (ordenada, reversa, aleatória).

- Entender como diferentes métricas de desempenho (tempo, CPU, memória, comparações) são impactadas pelo tipo de entrada e arquitetura.

3. Algoritmos Testados

Os algoritmos implementados e testados foram:
- Bubble Sort
- Insertion Sort
- Merge Sort
- Quick Sort

4. Ambiente de Testes

- Processador 1 (ARM): Apple M1
- Processador 2 (x86): Intel Core i7-14700KF
- Sistema Operacional: macOS e Windows 11
- Medições realizadas:
  - Tempo médio de execução (segundos)
  - Uso médio de CPU (%)
  - Uso médio de memória (MB)
  - Número médio de comparações

5. Metodologia

Cada algoritmo foi executado 13 vezes por entrada. Foram utilizados arquivos de entrada com vetores em diferentes tamanhos (de 50 até 1.000.000 elementos), e os dados foram categorizados em três tipos:
- Random (valores aleatórios)

- Decrescente
- Ordenado

As médias das métricas foram salvas em arquivos .csv e os gráficos foram gerados com Dash + Plotly e exportados em PDF.

6. Análise dos Resultados

Entradas Aleatórias (Random)

- Quick Sort foi o mais eficiente em ambas as arquiteturas.
- Algoritmos quadráticos (Bubble e Insertion) apresentaram desempenho significativamente pior.
- O i7 apresentou tempos levemente menores com maior uso de CPU.

Entradas Decrescentes

- Merge Sort e Quick Sort mantiveram bons tempos.
- Insertion Sort e Bubble Sort tiveram desempenho ruim, como esperado.
- A diferença entre as arquiteturas foi pequena.

Entradas Ordenadas

- Insertion Sort foi muito eficiente.
- Bubble Sort teve desempenho fraco.
- Quick Sort se manteve estável. Diferenças entre ARM e x86 foram mínimas.

7. Conclusão

- A arquitetura do processador influencia, mas a escolha do algoritmo é o fator determinante.
- Quick Sort foi o mais robusto.
- Insertion Sort foi ideal para dados já ordenados.
- ARM apresentou menor uso de CPU.
- Intel i7 teve vantagem em tempo bruto com entradas grandes.

8. Visualização dos Resultados

Os dashboards abaixo apresentam os gráficos de comparação para as diferentes métricas e tipos de entrada:
- Dashboard Entradas Aleatórias: ver “Dashboard Random.pdf”
- Dashboard Entradas Decrescentes: ver “Dashboard Decrescente.pdf”
- Dashboard Entradas Ordenadas: ver “Dashboard Ordenado.pdf”

Cada dashboard contém gráficos de:
- Tempo médio de execução
- Uso médio de CPU
- Consumo médio de memória
- Número médio de comparações

9. Reprodutibilidade

- test.py → controla execuções e salva os resultados.
- benchmark.py → executa os testes e coleta as métricas.
- grafico.py → gera os dashboards interativos.