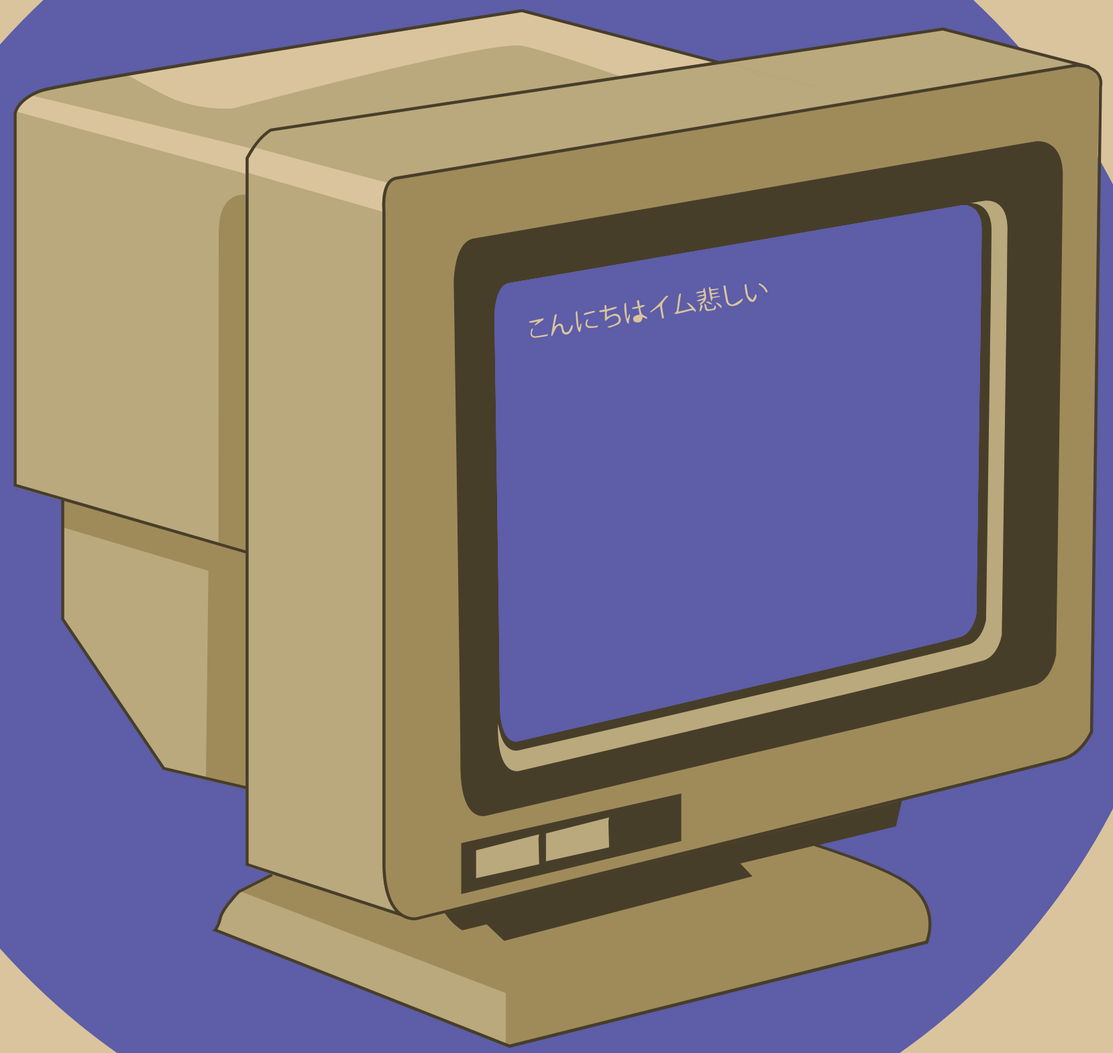


IFMA - IMPERATRIZ

ANÁLISE DE ALGORITMOS

ANÁLISE TEÓRICA E EMPÍRICA DOS ALGORITMOS MERGE SORT E INSERTION SORT E HYBRID SORT

INTEGRANTES: ALOÍSIO, CAIO, DIEGO, JONATHAN, NATHAN

OBjetivo

- **Principal: Demonstrar as limitações dos algoritmos clássicos O(N^2) e O(N log N) em cenários extremos.**
- **Proposta: Desenvolver e validar uma solução Híbrida que utiliza um limiar N(0) para alternar dinamicamente entre estratégias, visando superar o desempenho dos algoritmos puros.**

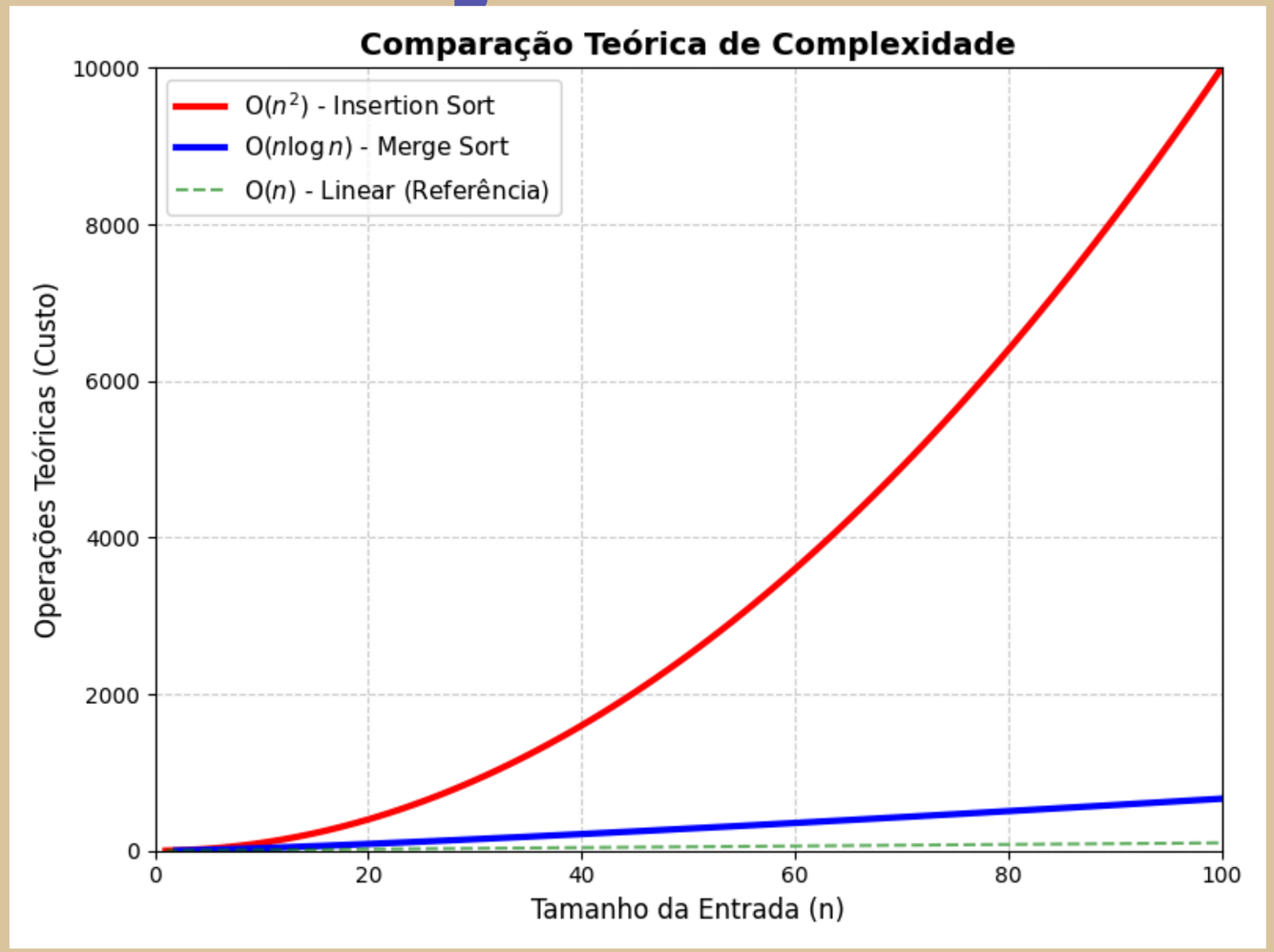
Algoritmo Merge Sort

O Merge Sort é um algoritmo de ordenação estável que utiliza a estratégia de "Dividir para Conquistar", funcionando através da divisão recursiva de uma lista em duas metades até que cada sublista contenha apenas um elemento isolado. Após esse desmembramento total, o algoritmo inicia o processo inverso de reconstrução, onde ele combina (faz o merge) essas sublistas menores novamente, mas comparando os elementos passo a passo para inseri-los na ordem correta na lista unificada. Basicamente, ele resolve a ordenação quebrando o problema em partes triviais e remontando-as já organizadas, garantindo uma eficiência de tempo consistente de O(N log N) independentemente da disposição inicial dos dados.

Algoritmo Insertion Sort

O Insertion Sort (Ordenação por Inserção) é um algoritmo simples e intuitivo que constrói a lista ordenada um item de cada vez, funcionando de maneira muito similar a como organizamos cartas de baralho na mão. Ele percorre o vetor da esquerda para a direita e, a cada novo elemento (chamado de chave), olha para trás comparando-o com os itens anteriores que já estão ordenados; se os itens anteriores forem maiores que a chave, ele os desloca para a direita para abrir espaço. Quando encontra a posição correta ou chega ao início da lista, ele insere a chave naquele local, repetindo esse processo até o fim, sendo muito eficiente para listas pequenas ou quase ordenadas (complexidade O(n) no melhor caso), mas lento para grandes volumes de dados desordenados (complexidade O(N^2).

Comparação matemática



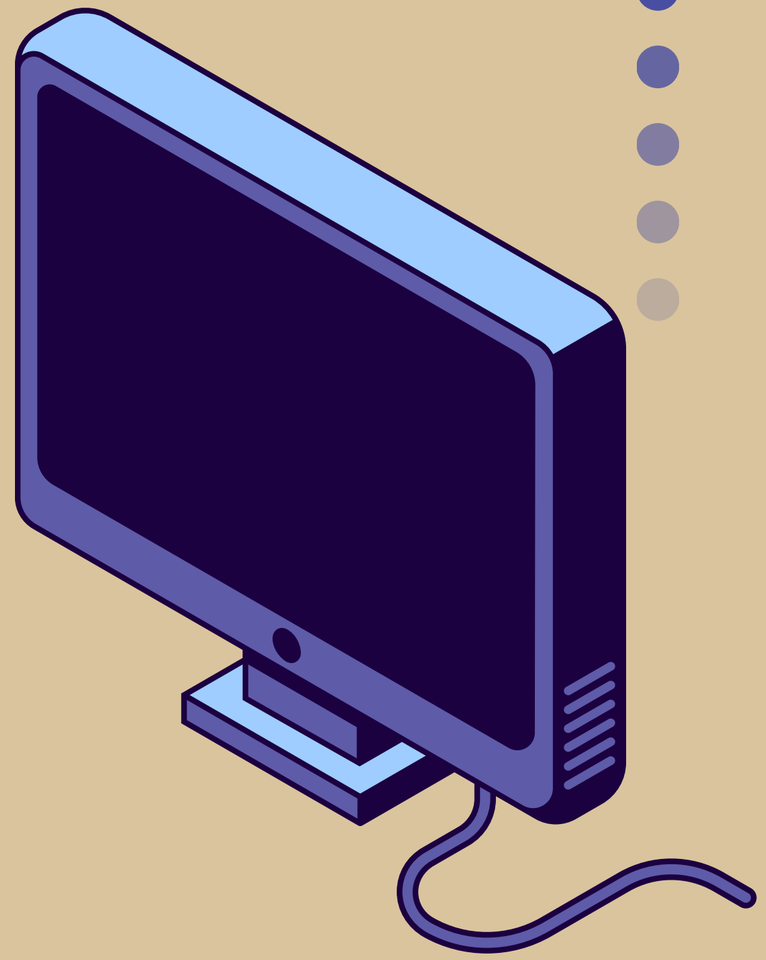
Metodologia e Ambiente Experimental

- Cálculo do N(0): Determinado empiricamente em 158 elementos através de testes incrementais.
- Volume de Dados: Vetores de 100.000 inteiros.
- Precisão: Média de 200 repetições por cenário (com reset de memória).
- Ambiente (Laboratório):
  - SO: Linux Ubuntu 24.04 LTS (Ambiente estável e de baixo ruído).
  - CPU: Intel® Core™ i5-6200U (4 threads).
  - RAM: 16 GB.
  - Compilador: GCC via terminal.

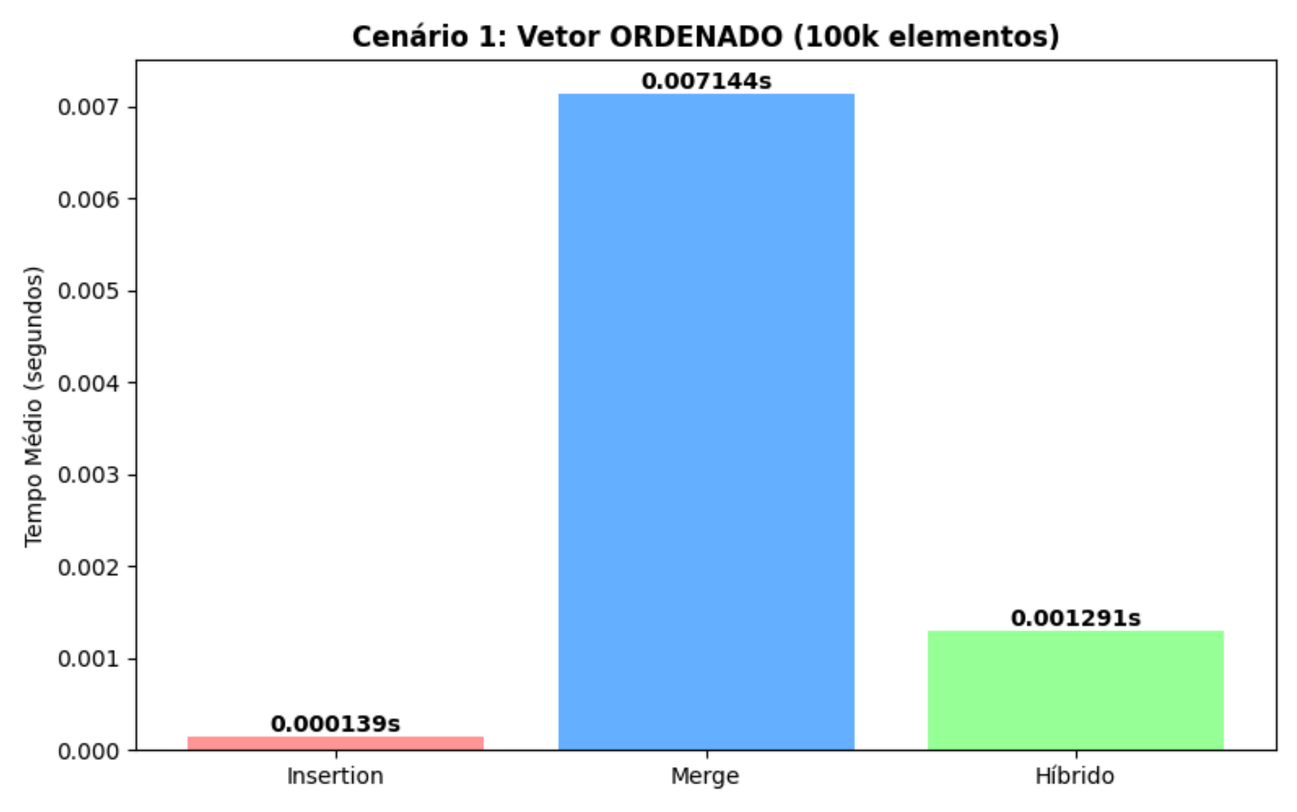
ANÁLISE DE ALGORITMOS

Cenário 1: Vetor Ordenado

ALGORITMO	MÍN (S)	MÁX (S)	MÉDIA (S)	DESVIO P.
Insertion	0,000109	0,000305	**0,000139**	0,000034
Merge	0,006052	0,017014	0,007144	0,001205
Híbrido	0,001120	0,001963	0,001291	0,000201



Resultados - Melhor Caso (Ordenado)



ANÁLISE TEÓRICA

Melhor caso:

- **Complexidade: O(N^2).**

Pior caso:

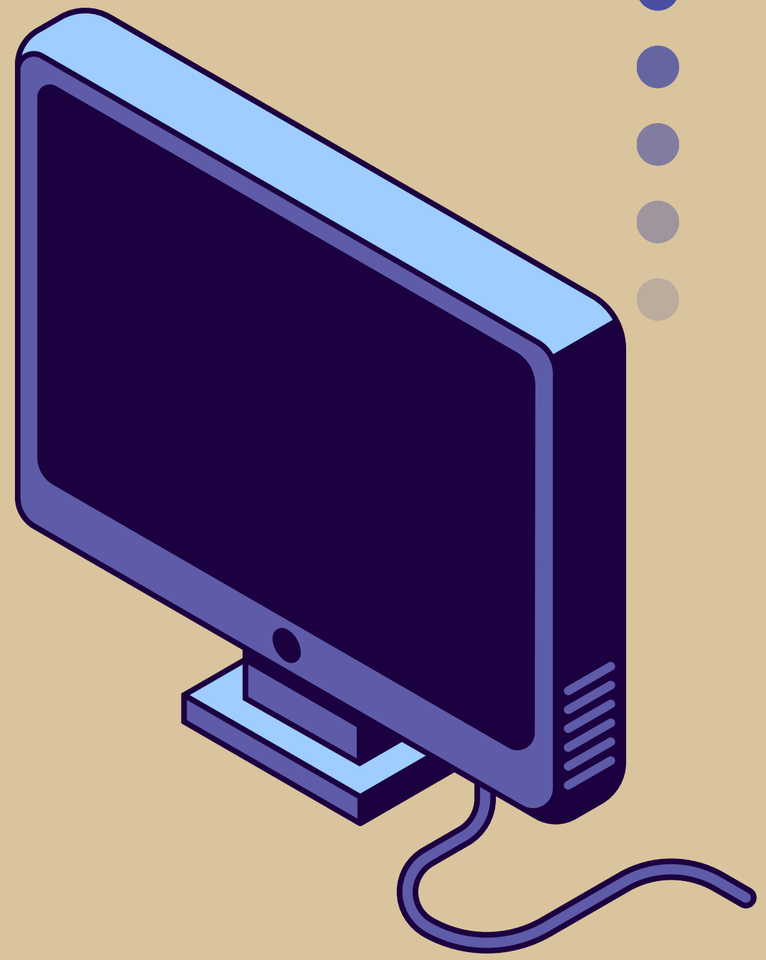
- **Complexidade: O(N log N)**

Análise: O Insertion Sort é imbatível (quase instantâneo) pois apenas verifica os dados O(n). O Híbrido aproveita isso e é muito superior ao Merge puro, que perde tempo dividindo o que já está pronto.

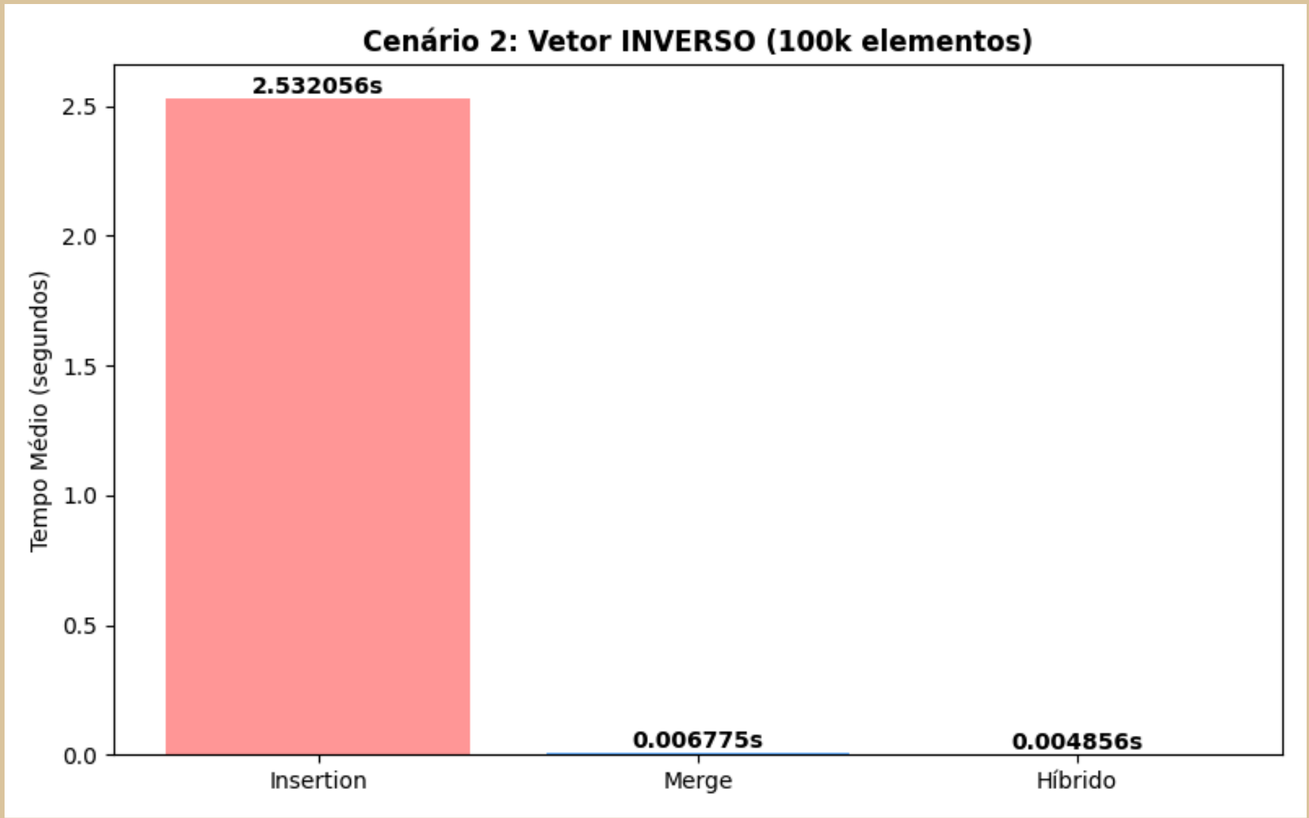
ANÁLISE DE ALGORITMOS

Cenário 2: Vetor Inverso

ALGORITMO	MÍN (S)	MÁX (S)	MÉDIA (S)	DESVIO P.
Insertion	2,386518	3,079592	**2,532056**	0,167599
Merge	0,005585	0,016663	0,006775	0,001475
Híbrido	0,004417	0,005813	**0,004856**	0,000341



Resultados - Pior Caso (Inverso)



ANÁLISE TEÓRICA

Melhor caso:

- **Complexidade: O(N log N).**

Pior caso:

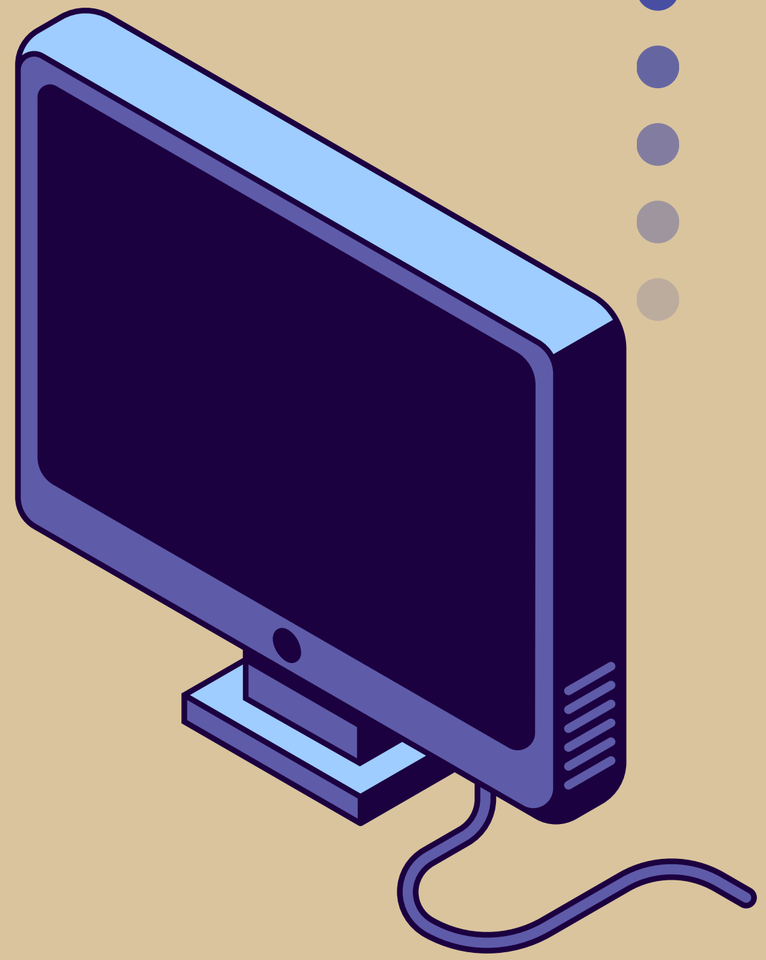
- **Complexidade: O(N^2)**

Análise: O Insertion falha catastroficamente (2.53 segundos). O Merge Sort se mantém estável. O Híbrido consegue ser competitivo (até ligeiramente melhor que o Merge), provando que usar Insertion apenas em blocos pequenos (158) não prejudica a performance geral mesmo no pior cenário.

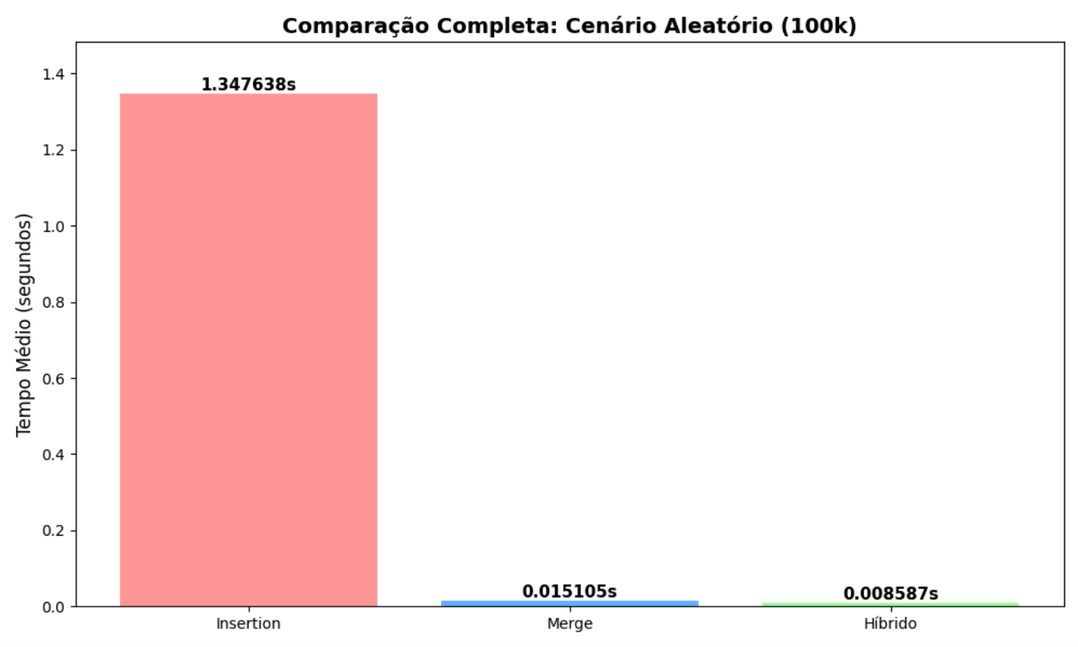
ANÁLISE DE ALGORITMOS

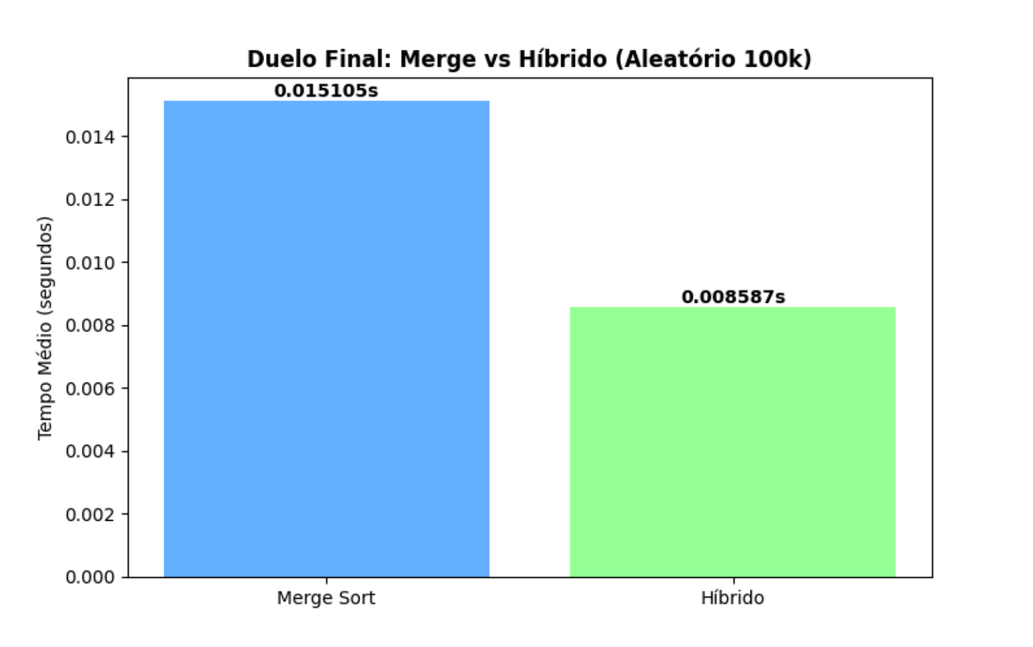
Cenário 3: Vetor Aleatório

ALGORITMO	MÍN (S)	MÁX (S)	MÉDIA (S)	DESVIO P.
Insertion	1,189278	2,143418	1,347638	0,197023
Merge	0,014029	0,026192	0,015105	0,001444
Híbrido	0,007962	0,013260	**0,008587**	0,000528



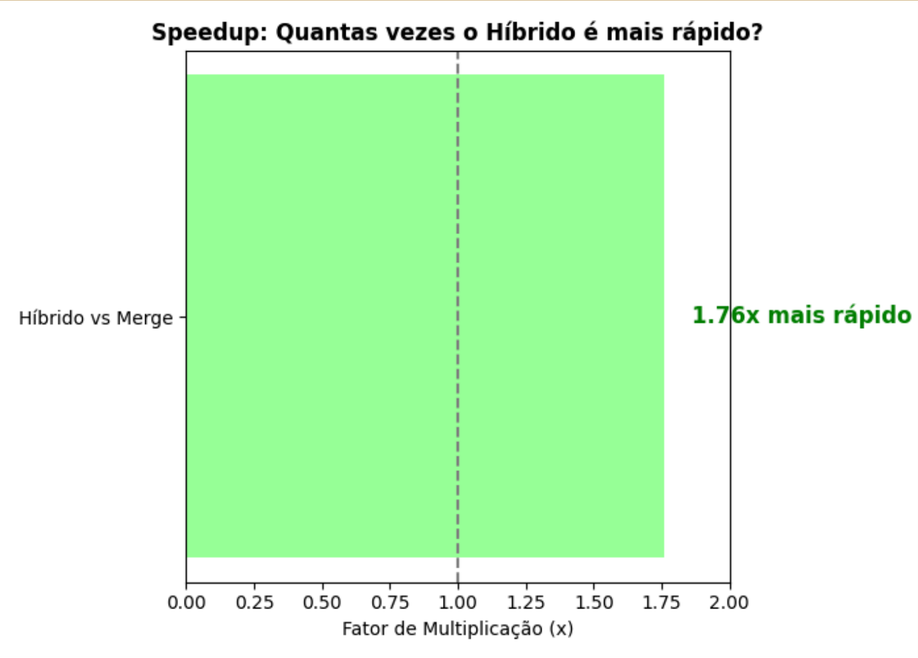
Resultados - Cenário Real (Aleatório)

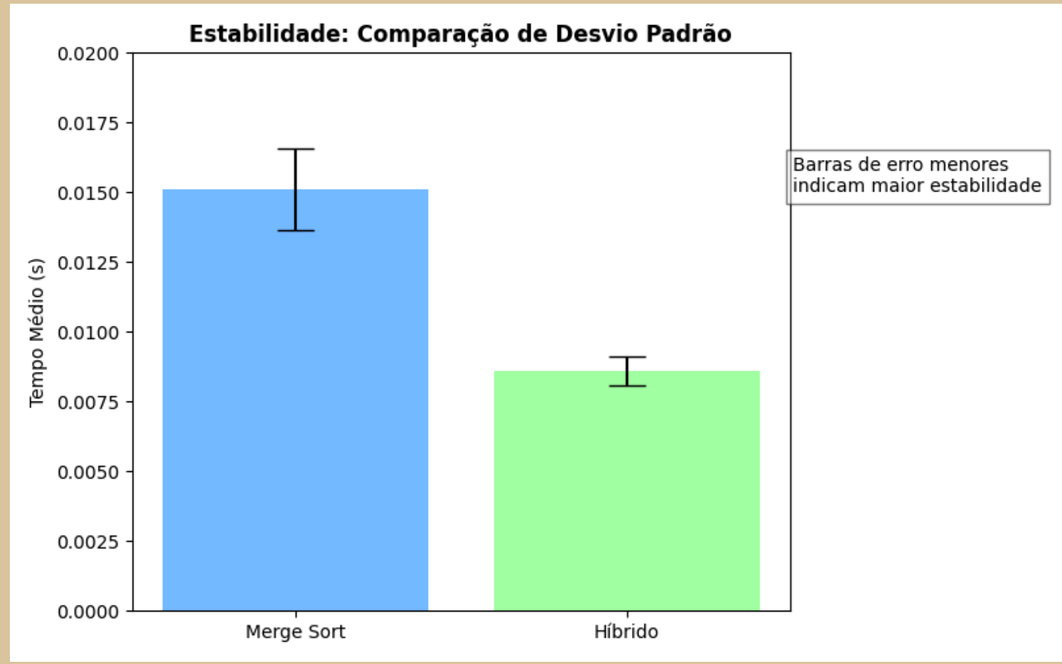




- **O Grande Vencedor: No cenário mais comum do dia a dia (dados bagunçados), o Híbrido venceu.**
- **Tempos: Merge (0.015s) vs Híbrido (0.008s).**

Análise de velocidade e estabilidade





- **Speedup: Otimizamos o processo cortando as chamadas recursivas finais.**

- **Estabilidade: As barras de erro mostram que o Híbrido não só é rápido, como é consistente (baixo desvio padrão).**

Conclusão

- O cálculo empírico do N(0) foi eficaz.
- O Algoritmo Híbrido uniu a baixa sobrecarga do Insertion com a eficiência assintótica do Merge.
- Resultado Final: Obtivemos um algoritmo 1.76x mais rápido (redução de 43,15% no tempo) em comparação ao Merge Sort padrão no cenário aleatório.

ANÁLISE DE ALGORITIMOS

OBRIGADO PELA ATENÇÃO