Relatório PBL-4: Algorítmos de Ordenação

-

Eduardo Nicolosi | Francielly Pazello | João Pedro Brum

1. Insertion Sort.

1.1 Tabela de tempos de execução de acordo com ordem e número de elementos.

Ordem/Número de elementos	100	1000	10.000	Total
Aleatória	0,383	1,415	21,808	23,606
Crescente	0,041	0,044	0,049	0,134
Decrescente	0,005	0,125	11,946	12,046
Total Geral	0,429	1,584	33,803	35,786

1.2 Análise dos resultados.

O Insertion Sort leva vantagem em cenários listas já ordenadas pela facilidade de adaptação quando percebe que os elementos já estão ordenados, evitando trocas desnecessárias. A desvantagem se encontra nos cenários com listas aleatórias e decrescentes pois o número de trocas aumenta exponencialmente junto com a complexidade de acordo com o número de elementos, gerando tempos maiores.

2. Quick Sort.

2.1 Tabela de tempos de execução (ms) de acordo com ordem e número de elementos.

Ordem/Número de elementos	100	1000	10.000	Total
Aleatória	0,029	0,314	0,785	1,128
Crescente	0,025	1,519	48,705	50,249
Decrescente	0,013	0,405	59,884	60,302
Total Geral	0,067	2,238	109,374	111,679

2.2 Análise dos resultados

O Quick Sort leva vantagem em cenários com listas de dados aleatórios graças à recursividade que gera partições já balanceadas com base no número pivô. A desvantagem se encontra nos cenários com listas ordenadas (crescente ou decrescente) porque a escolha inadequada do pivô gera partições desbalanceadas, aumentando o tempo de execução exponencialmente junto com a complexidade de acordo com o número de elementos, gerando tempos maiores.

3. Bubble Sort.

3.1 Tabela de tempos de execução (ms) de acordo com ordem e número de elementos.

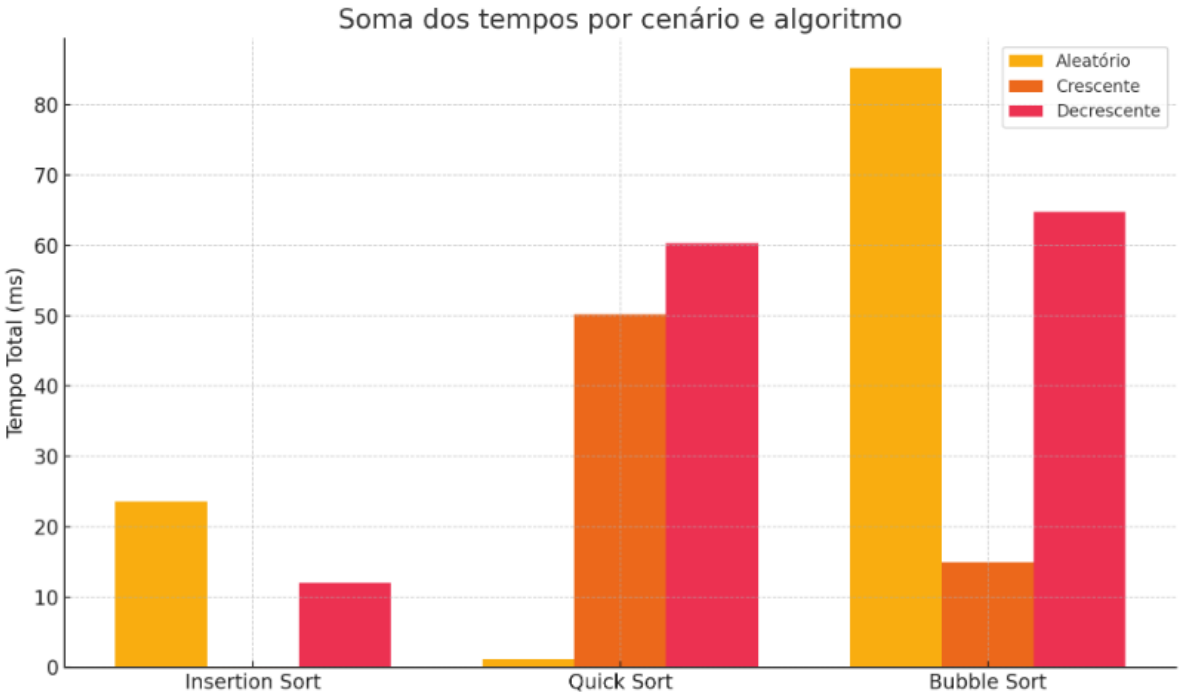
Ordem/Número de elementos	100	1000	10.000	Total
Aleatória	0,136	3,442	81,637	85,215
Crescente	0,004	0,185	14,746	14,935
Decrescente	0,006	0,624	64,191	64,821
Total Geral	0,146	4,251	160,574	164,971

3.2 Análise dos resultados

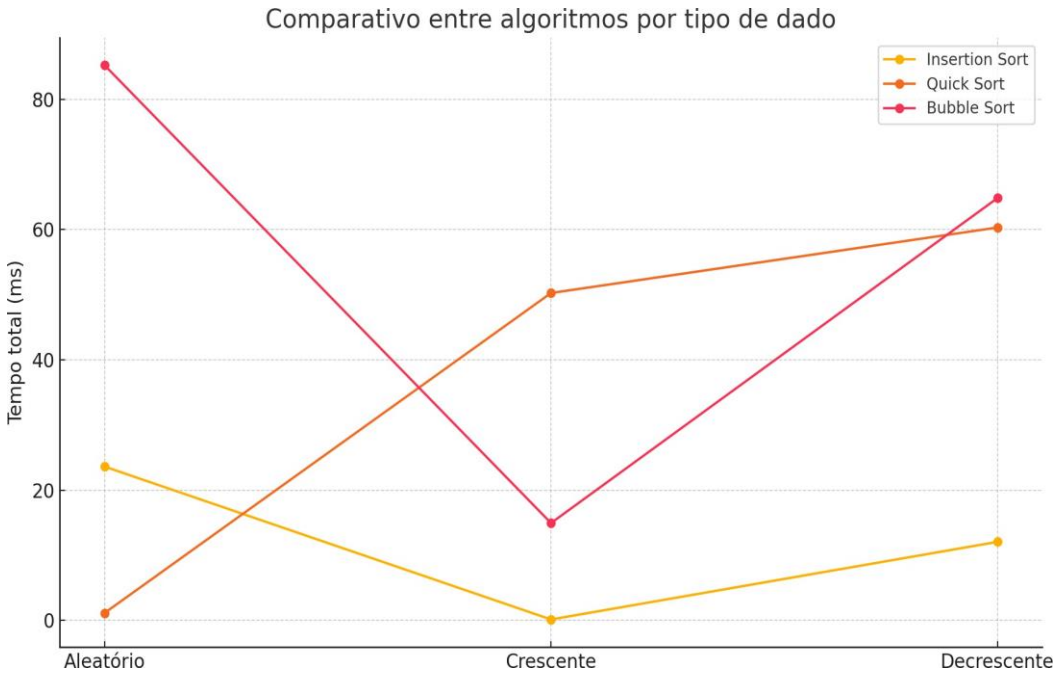
O Bubble Sort leva vantagem em listas crescentes por uma verificação simples ser suficiente, sendo mais simples e rápido. A desvantagem se encontra em praticamente qualquer outro cenário, como listas desordenadas ou decrescentes, por fazer muitas comparações e trocas mesmo quando se a lista já estiver quase totalmente ordenada, aumentando o tempo de execução exponencialmente junto com a complexidade de acordo com o número de elementos, gerando tempos maiores.

4. Comparação entre os algorítmos por meio de representação gráfica

4.1 Comparativo de tempo total dos algoritmos por tipo de dado



4.2 Comparativo entre variação de tempo total dos algoritmos por tipo de dado.



4.3 Tempo médio por algoritmo