Relatório de

Estrutura de Dados

Ordenação

Aluno: Gabriel Araújo Teixeira.

Matricula: 511476.

Introdução

Este relatório tem como objetivo analisar o desempenho dos algoritmos BubbleSort, InsertionSort, QuickSort, MergeSort e HeapSort. Para isso, foi construído um programa no qual cada algoritmo recebeu a tarefa de ordenar um vetor de números inteiros de tamanho variando desde 100 até 1000000 casas. Os tempos de execução estão dispostos nas tabelas abaixo.

Resultados

Tabela 1

BubbleSort	
Tamanho do vetor	Tempo de execução (ms)
100	0.000000
1000	6.000000
10000	704.000000
100000	49478.000000
1000000	2656025.000000

Tabela 2

InsertionSort	
Tamanho do vetor	Tempo de execução (ms)
100	0.000000
1000	0.000000
10000	84.000000
100000	8411.000000
1000000	848010.000000

Tabela 3

QuickSort	
Tamanho do vetor	Tempo de execução (ms)
100	0.000000
1000	0.000000
10000	0.000000
100000	10.000000
1000000	122.000000

Tabela 4

MergeSort	
Tamanho do vetor	Tempo de execução (ms)
100	0.000000
1000	0.000000
10000	2.000000
100000	27.000000
1000000	294.000000

Tabela 5

HeapSort	
Tamanho do vetor	Tempo de execução (ms)
100	0.000000
1000	0.000000
10000	2.000000
100000	16.000000
1000000	208.000000

Análise

De posse dos tempos de cada algoritmo, é valido analisar o comportamento de cada um deles à medida que o tamanho do vetor aumenta:

- **BubbleSort:** O BubbleSort apresentou um comportamento quadrático, ou seja, a medida que o problema aumentava o tempo de execução se elevava ao quadrado, o que, na prática, é bastante ineficiente.
- **InsertionSort:** Embora tenha apresentado tempo melhores que o BubbleSort, ainda se mostrou ineficiente tendo em vista que também apresentou comportamento quadrático.
- **QuickSort:** Dentre os cinco algoritmos testados o QuickSort foi o que se mostrou mais eficiente. Demonstrou um comportamento quase linear a medida que o tamanho do vetor aumentava.
- **MergeSort:** Assim como o QuickSort, também demostrou um comportamento próximo do linear. Logo, também é bastante eficiente.
- **HeapSort:** Este é outro algoritmo que possui comportamento quase linear e, portanto, se mostra muito eficaz.

Conclusão

Diante dos resultados obtidos, verifica-se que o QuickSort foi o algoritmo que demonstrou maior eficiência para ordenar vetores. Logo atrás dele vem os algoritmos HeapSort e MergeSort que também possuem alta eficiência. Por outro lado, os algoritmos BubbleSort e InsertionSort se mostraram se mostraram bastante ineficazes tendo em vista o grande aumento dos tempos a medida que o problema cresce.