TDE Resolução de Problemas Estruturados em Computação Algoritmos de Ordenação

Bruno Marques

Pontifícia Universidade Católica do Paraná (PUCPR)
Curitiba – PR – Brazil

Resumo. *Esse artigo apresenta os resultados e análises a respeito do uso, eficiência e diferenças entre diferentes tipos de algoritmos de ordenação. Os dados e códigos referentes a esse relatório se encontram no repositório https://github.com/B0Marques/RA4---Ordena-o*

1. Introdução

Foram escolhidos 3 algoritmos de ordenação: **Bubblesort**, **Mergesort** e **Quicksort**.

Para cada um desses algoritmos, foram preenchidos aleatoriamente vetores com valores inteiros, sendo cada vetor de tamanho: 50, 500, 1000, 5000 e 10000. Após a ordenação desses vetores, foi avaliado o **Tempo de execução**, **Número de trocas** e **Número de iterações**.

2. Bubblesort

O algoritmo Bubblesort é um dos algoritmos de ordenação mais simples. Ele consiste em percorrer várias vezes o vetor, comparando os valores adjacentes uns aos outros e os trocando de lugar, repetindo quantas vezes forem necessárias para ordenar todo o vetor.

Vetores	Tempo Execução	Num Trocas	Num Iterações
Vetor Tam 50	2,4	634,6	1198,8
Vetor Tam 500	8,8	63119,4	124516,2
Vetor Tam 1000	10,2	248307,2	498797,4
Vetor Tam 5000	47	6280896,6	12495230
Vetor Tam 10000	140,4	24943061	49983514

Figure 1. Ordenação usando Bubblesort

3. Mergesort

Esse algoritmo consiste na divisão, dividindo vetores em duas partes, ordenando-os e os fundidos. Esse processo é feito recursivamente, dividindo os vetores em várias partes e os fundindo, até que ele esteja totalmente ordenado.

Vetores	Tempo Execução	Num Trocas	Num Iterações
Vetor Tam 50	2	103,4	49
Vetor Tam 500	7,8	1903,4	499
Vetor Tam 1000	6,8	4308,4	999
Vetor Tam 5000	19,2	27092,4	4999
Vetor Tam 10000	30,2	59137	9999

Figure 2. Ordenação usando Mergesort

4. Quicksort

Assim como o Mergesort, utiliza da divisão do vetor. O algoritmo escolhe um elemento da lista, chamado de pivô e ordena o vetor com base nesse elemento, separando os valores maiores e menores que ele. Esse processo também é repetido recursivamente, até o vetor estar ordenado.

Vetores	Tempo Execuçã	Num Trocas	Num Iterações
Vetor Tam 50	2	162,6	253,6
Vetor Tam 500	5,8	2620,8	5121,4
Vetor Tam 1000	7	6390,6	11014,2
Vetor Tam 5000	24	38434,6	69543
Vetor Tam 10000	37	83234,2	155798,2

Figure 3. Ordenação usando Quicksort

5. Conclusão

Utilizando das informações coletadas das tabelas referente a ordenação dos vetores, é notável a diferença de capacidade entre cada tipo de ordenação de vetor, principalmente entre o Bubblesort e os outros dois tipos de ordenação.

Para os vetores de menor tamanho, o algoritmo Bubblesort consegue acompanhar os outros dois, mas as suas diferenças de eficiência são notáveis nos vetores maiores. Isso se dá ao fato da complexidade do Bubblesort ser unicamente O(n^2), ou seja, possui um crescimento quadrático, enquanto isso não se aplica aos outros.

Apesar de não possuir uma discrepância tão grande quanto em relação ao Bubblesort, ainda há uma diferença significativa entre o Mergesort e o Quicksort. Entre os dois algoritmos, o Mergesort acaba por ser o mais eficaz, devido a sua complexidade de tempo ser unicamente O(n log n) (linearítmico), enquanto o Quicksort pode variar entre linearítmico e quadrático.

Podemos concluir que o algoritmo de ordenação Mergesort, em comparação com os três utilizados para essa análise, é o mais eficaz quando se trata de grandes quantidades de dados.