UNIVERSIDADE FEDERAL DE OURO PRETO
Faculdade de Ciência da Computação

Algoritmos de ordenação

Marcus Vinícius Souza Fernandes

Ouro Preto - MG
Outubro de 2019

Algoritmos de ordenação

Marcus Vinícius Souza Fernandes

Trabalho Prático do Curso de Ciência da Computação da Universidade Federal de Ouro Preto.

INTRODUÇÃO

Como comando do trabalho prático foi pedido para desenvolver um código que possuísse 4 algoritmos de ordenação sendo eles BubleSort, SelectionSort, MergeSort e InsertionSort. Eles deveriam ordenar 3 tipos de vetores, com valores aleatórios, crescentes e decrescentes. Por fim deveria ser contabilizado o tempo gasto de cada um deles e realizar uma comparação dos casos obtidos.

DISCUSSÃO

Após rodar cada um dos algoritmos de ordenação para cada um dos seus respectivos casos, obtivemos estes valores de tempo para execução.

Testes realizados em minha máquina pessoal (Acer Aspire VX, i5 7th, 8gb ram).

Algoritmo	Complex tempo	Complex espaço	Tempo de execução		
			Vetor aleatório	Vetor crescente	Vetor decrescente
BubleSort	O (n ^ 2)	O(1)	00:51:38	00:00:00	00:38:36
SelectionSort	O (n ^ 2)	O(1)	00:30:46	00:21:24	00:21:59
MergeSort	O (n log n)	O(n)	00:00:00	00:00:00	00:00:00
InsertionSort	O (n ^ 2)	O(1)	00:19:29	00:00:00	00:17:48

Os algoritmos utilizados foram todos implementados de maneira melhorada, dessa forma o tempo de execução geral não foi tão longo.

O BubleSort, MergeSort e o InsertionSort foram executados em milésimos de segundos quando o vetor estava ordenado de forma crescente. Isso é devido a estrutura dos algoritmos que apenas verificaram condições e não entraram em nenhum caso, diferente do SelectionSort que percorre todo o vetor várias vezes para marcar o maior dos termos a cada repetição, assim obtendo um tempo de execução elevado.

O MergeSort realizou a ordenação para cada um dos tipos de vetores de forma instantânea, diferentemente dos demais algoritmos.

Quando se tratou do vetor com conteúdo aleatório notamos que foi o que demandou o maior tempo de execução para todos os algoritmos com exceção do MergeSort que foi analisado anteriormente.

Como os algoritmos estavam em suas formas melhoradas os vetores com conteúdos de forma decrescente obtiveram uma ligeira vantagem de tempo em relação aos aleatórios.

Em suma podemos dizer que o pior caso dos algoritmos foram quando relacionados com o vetor de conteúdo aleatório e o melhor caso com o vetor de forma crescente, com exceção do MergeSort que obteve tempo igual em ambos.

REFERENCIAS

http://www.decom.ufop.br/anascimento/ensino/bcc202/aulas-praticas/

https://www.geeksforgeeks.org/binary-insertion-sort/