Unidade III:

Ordenação Interna - Comparação entre Métodos

Instituto de Ciências Exatas e Informática
Departamento de Ciência da Computação

Qual é o Melhor Algoritmo de Ordenação?

- A resposta não é simples, pois os resultados obtido não levam em conta outras operações como controle dos laços e cálculos dos índices. Além disso, eles desconsideram arquitetura, SO, compilador e hardware

- Uma forma de ajudar na resposta é a avaliação experimental dos métodos de ordenação

Avaliação para Registros na Ordem Aleatória

Algoritmo	500	5000	10000	30000
Inserção	11,3	87	161	-
Seleção	16,2	124	228	-
Shellsort	1,2	1,6	1,7	2
Quicksort	1	1	1	1
Heapsort	1,5	1,6	1,6	1,6

Avaliação para Registros na Ordem Crescente

Algoritmo	500	5000	10000	30000
Inserção	1	1	1	1
Seleção	128	1524	3066	-
Shellsort	3,9	6,8	7,3	8,1
Quicksort	4,1	6,3	6,8	7,1
Heapsort	12,2	20,8	22,4	24,6

Avaliação para Registros na Ordem Decrescente

Algoritmo	500	5000	10000	30000
Inserção	40,3	305	575	-
Seleção	29,3	221	417	-
Shellsort	1,5	1,5	1,6	1,6
Quicksort	1	1	1	1
Heapsort	2,5	2,7	2,7	2,9

Exercício

- Confirme experimentalmente as três tabelas anteriores e adicione o Mergesort e Countingsort. Em seus experimentos, capture o tempo de execução e os números de comparações e movimentações envolvendo elementos do *array* para cada algoritmo. Além disso, considere as ordens iniciais crescente, decrescente e aleatória com *arrays* contendo 100, 1000, 10000 e 100000 elementos. Como resultado faça nove gráficos variando as métricas avaliadas (tempo, comparação e movimentação) e ordem inicial dos elementos (crescente, decrescente e aleatória) . Em seguida Em seguida, construa e discuta três gráficos comparativos entre os algoritmos. Um gráfico para cada uma das três ordens iniciais dos elementos e fazendo com que o eixo x sempre tenha a quantidade de elementos e no eixo o y o

Considerações Finais

- A vantagem do algoritmo de seleção é seu número de movimentos de registros que é $\Theta(n)$

- O algoritmos de Inserção é interessante para *arrays* ordenados (ou praticamente)

- Os métodos de Inserção e Countingsort são estáveis

Considerações Finais

- O Quicksort é o mais eficiente para uma grande variedade de situações
- O pior caso do Quicksort é $\Theta(n^2)$
- O pior caso do Mergesort e do Heapsort é $\Theta(n*lg(n))$
- Uma desvantagem do Mergesort é o fato dele “duplicar” sistematicamente os vetores

Considerações Finais

- Uma vantagem do Quicksort é sua pilha de recursividade reduzida

- O Coutingsort é uma opção que sempre deve ser considerada para a ordenação de inteiros ou similares (e.g., números reais com um número fixo de casas decimais)