Universidade Estadual da Paraíba - UEPB

Centro de Ciências e Tecnologia - CCT

Curso de Ciências da Computação

Relatório comparativo dos algoritmos elementares de ordenação

Laboratório de Estrutura de Dados

Mateus Sobrinho Gomes de Almeida

Campina Grande

Março - 2024

InsertionSort

O Insertion Sort é um algoritmo de ordenação simples e eficiente. Ele funciona construindo uma matriz ordenada um elemento de cada vez, movendo elementos não ordenados para suas posições corretas na matriz ordenada.

A ideia básica por trás do Insertion Sort é que ele percorre o array, começando do segundo elemento (índice 1) até o último, e a cada passo, ele insere o elemento atual na parte correta do array já classificado. Ou seja, ele "insere" o elemento na posição correta do subarray classificado.

TESTES	COMPARAÇÕES	TROCAS	TEMPO
100.000 números	2.496.773.447	2.496.773.447	341 ms
500.000 números	62.534.914.315	62.534.914.315	8956 ms
1.000.000 números	250.097.755.563	250.097.755.563	37566 ms

SelectionSort

O Selection funciona dividindo o array em duas partes: uma parte ordenada e uma parte não ordenada. O algoritmo encontra repetidamente o menor elemento na parte não ordenada e o move para o final da parte ordenada. Esse processo continua até que toda a matriz esteja ordenada.

É fácil de implementar e entender, mas não é tão eficiente quanto outros algoritmos de ordenação, especialmente para conjuntos de dados maiores, porque ele realiza muitas trocas. Sua complexidade de tempo é O(n^2), onde 'n' é o número de elementos no array. Isso significa que seu desempenho piora rapidamente à medida que o tamanho do conjunto de dados aumenta. No entanto, para conjuntos de dados pequenos ou quase classificados, o Selection Sort pode ser uma opção viável.

TESTES	**COMPARAÇÕES**	**TROCAS**	**TEMPO**
100.000 números	4.999.950.000	99.999	2769 ms
500.000 números	124.999.750.000	499.999	66773 ms
1.000.000 números	499.999.500.000	999.999	268028 ms

BubbleSort

O Bubble Sort é um algoritmo de ordenação que percorre repetidamente a lista, compara elementos adjacentes e os troca se estiverem na ordem errada. O processo continua até que nenhum swap seja necessário, o que indica que a lista está ordenada.

Bubble Sort não é muito eficiente, especialmente para conjuntos de dados grandes. Sua complexidade de tempo é O(n^2), o que significa que seu desempenho piora rapidamente com o aumento do tamanho do conjunto de dados. No entanto, assim como o Selection Sort, o Bubble Sort pode ser útil para conjuntos de dados pequenos ou quase classificados.

TESTES	COMPARAÇÕES	TROCAS	TEMPO
100.000 números	4.999.941.354	2.512.107.000	15554 ms
500.000 números	124.999.658.194	62.601.203.311	355513 ms
1.000.000 números	499.999.390.254	250.240.981.728	1568693 ms

Comparações:

- Insertion Sort:
  100.000 números: 2.496.773.447
  500.000 números: 62.534.914.315
  1.000.000 números: 250.097.755.563

- Selection Sort:
  100.000 números: 4.999.950.000
  500.000 números: 124.999.750.000
  1.000.000 números: 499.999.500.000

- Bubble Sort:
  100.000 números: 4.999.941.354
  500.000 números: 124.999.658.194
  1.000.000 números: 499.999.390.254

Trocas:

- Insertion Sort:
  100.000 números: 2.496.773.447
  500.000 números: 62.534.914.315
  1.000.000 números: 250.097.755.563

- Selection Sort:
  100.000 números: 99.999
  500.000 números: 499.999
  1.000.000 números: 999.999

- Bubble Sort:
  100.000 números: 2.512.107.000
  500.000 números: 62.601.203.311
  1.000.000 números: 250.240.981.728

Tempo:

- Insertion Sort:
  100.000 números: 341 ms
  500.000 números: 8956 ms
  1.000.000 números: 37566 ms

- Selection Sort:
  100.000 números: 2769 ms
  500.000 números: 66773 ms
  1.000.000 números: 268028 ms

- Bubble Sort:
  100.000 números: 15554 ms
  500.000 números: 355513 ms
  1.000.000 números: 1568693 ms

Avaliando os três critérios - comparações, trocas e tempo de execução - podemos determinar a eficiência relativa dos algoritmos da seguinte forma:

- **Melhor em comparações:** O algoritmo que teve o menor número de comparações foi o Insertion Sort, seguido pelo Selection Sort e, por último, o Bubble Sort.

- **Melhor em trocas:** O algoritmo que teve o menor número de trocas foi o Selection Sort, seguido pelo Insertion Sort e, em seguida, o Bubble Sort.

- **Melhor em tempo de execução:** O algoritmo que teve o menor tempo de execução foi o Insertion Sort, seguido pelo Selection Sort e, por fim, o Bubble Sort.

Com base nesses critérios, podemos classificar os algoritmos da seguinte forma:

- **Melhor desempenho geral:** Insertion Sort
- **Segundo melhor desempenho geral:** Selection Sort
- **Último em desempenho geral:** Bubble Sort

Portanto, em termos de eficiência geral, o Insertion Sort foi o melhor, seguido pelo Selection Sort, e o Bubble Sort ficou em último lugar.