Ordenação por Seleção
Algoritmos e Estrutura de Dados I

Instituto de Engenharia – UFMT

Roteiro

1. Objetivos
2. Introdução
3. SelectionSort

Objetivos

Esta aula tem como objetivos:

1. Apresentar os conceitos básicos sobre ordenação;
2. Explicitar os métodos de ordenação SelectionSort e InsertionSort;
3. Exemplificar a execução dos algoritmos.

Introdução

Ordenar

Ordenar é o processo de rearranjar um conjunto de objetos em uma ordem ascendente ou descendente.

- A ordenação visa facilitar a recuperação posterior de itens do conjunto ordenado;
- As técnicas de ordenação permitem apresentar um amplo de algoritmos distintos para resolver uma mesma tarefa.

Introdução

Existem três razões práticas para estudar os algoritmos de ordenação:

- Analisar os algoritmos de ordenação é uma introdução completa as técnicas de comparação de desempenho de algoritmos;
- Técnicas semelhantes são eficazes no tratamento de outros problemas;
- Muitas vezes usamos algoritmos de ordenação como ponto de partida para resolver outros problemas.

Mais importante do que esses motivos práticos é que os algoritmos são elegantes, clássicos, e eficazes.

Notação

- Em geral, os elementos do vetor a ser ordenado são objetos complexos, com muitos campos.
- Do ponto de vista da ordenação, apenas um desses campos – a chave (= key) – é relevante.
- Vamos supor, para simplificar, que a chave é o único campo do objeto.

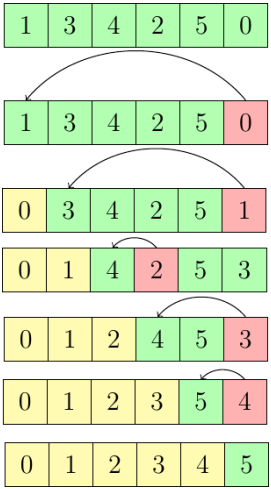
Características

- Qualquer tipo de chave, sobre o qual exista uma regra de ordenação bem definida, pode ser utilizada;
  - As ordens mais usadas são a numérica e a lexicográfica.

SelectionSort

- Um dos algoritmos de ordenação mais simples;
- A cada iteração, seleciona o menor elemento da lista e troque-o com o item na posição correta;
- Passos para o algoritmo, dado um vetor v[1..n]:
  - Procurar o menor elemento no vetor v[1..n] e trocar com o elemento na 1ª posição.
  - Procurar o menor elemento no vetor v[2..n] e trocar com o elemento na 2ª posição.
  - Proceder assim até a ordenação estar completa.

SelectionSort



Pseudo-código SelectionSort

```
void selectionsort(int V[], int n) {
  int i, j, min, aux;
  for (i = 0; i < n-1; i++) {
    min = i;
    for (j = i+1; j < n; j++) {
      if (V[j] < V[min]) {
        min = j;
      }
    }
    aux = V[i];
    V[i] = V[min];
    V[min] = aux;
  }
}
```

Análise SelectionSort

- Ciclo interno apenas faz comparações
  - troca de elementos é feita fora do ciclo interno;
  - cada troca como um elemento na sua posição final;
  - o número de trocas é $n - 1$ (por que não n?).
  - o tempo de execução é dominado pelo número de comparações.

Características

Vantagens

- Custo linear no tamanho da entrada para o número de movimentos de registros;
- É o algoritmo a ser utilizado para arquivos com registros muito grandes (que possuem alto custo de movimentação);
- É muito interessante para arquivos pequenos.

Desvantagens

- O fato do arquivo já estar ordenado não ajuda em nada, pois o custo continua quadrático;

Conclusão

Ordenação

- A tarefa de ordenação é muito importante, ela é uma necessidade básica para a solução de muitos problemas.
- Material para reforçare a aprendizagem:
  - Vídeo-aula selection sort

Dúvidas

Fim