Escola
Politécnica

Ordenação

Prof. Marcos Carrard
carrard@univali.br
carrard@gmail.com

UNIVALI Escola Politécnica

Ordenação

- Facilitar e aumentar a eficiência das operações de pesquisa sobre esses dados
- Pode ser crescente ou decrescente
- A sequência de entrada, normalmente, é um vetor com *n* elementos
- Outras possibilidades de estruturas de dados como por exemplo, uma lista encadeada

UNIVALI Escola Politécnica

Ordenação

- Na prática os números a serem ordenados, raramente, são valores isolados
- Normalmente, cada número é componente de um conjunto de dados denominado **registro**
- E o conjunto de registros forma uma tabela
- Cada registro contém uma **chave**, que é o valor a ser ordenado, e demais valores que sempre acompanham a chave
- Numa operação de ordenação, sempre que for preciso trocar a posição de uma chave, será necessário alterar a posição de todos os elementos do registro
- Na prática, quando os registros possuem uma grande quantidade de dados (além da chave), a ordenação é realizada sobre um vetor (ou lista) de ponteiros para os registros, com o objetivo de minimizar as operações de movimentação de dados

UNIVALI Escola Politécnica

Métodos de Ordenação

- Ordenação por Inserção: elementos são inseridos na posição correta.
  - inserção direta
  - incrementos decrescentes (*Shell sort*)
- Ordenação por Troca: comparações sucessivas trocam elementos de posição no vetor
  - método da bolha (*bubble sort*)
  - método da troca e partição (*quicksort*)
- Ordenação por Seleção: é realizada uma seleção do maior ou menor valor e este é realocado no vetor
  - seleção direta
- Ordenação por Intercalação: é realizada uma mescla de dois vetores menores ordenados.
  - Método da intercalação (*mergesort*)

UNIVALI Escola Politécnica

Ordenação - Inserção

- É o método mais simples
- Utilizado para um conjunto pequeno de dados
- Possui baixa eficiência
- Nesse método o vetor a ser ordenado é dividido em dois segmentos: o primeiro segmento contém os elementos já ordenados; o segundo segmento contém os elementos a serem ordenados

UNIVALI Escola Politécnica

Ordenação - Inserção

- Primeiro elemento está no vetor ordenado e os demais no vetor desordenado
- Retirar o primeiro elemento do vetor desordenado e colocá-lo no vetor ordenado, na sua posição correta
- Repetir o processo para todos os elementos do vetor desordenado

UNIVALI Escola Politécnica

Ordenação – Inserção - Algoritmo

Sendo *TAM* o tamanho do vetor e *vet* o vetor com os elementos a serem ordenados

```
...
   para j ← 1 até TAM-1 passo 1 faça
      chave ← vet[j]
      i ← j - 1
      enquanto (i >= 0) e (vet[i] > chave) faça
                  vet[i + 1] ← vet[i]
                  i ← i - 1
      fim enquanto
      vet[i+1] ← chave
   fim para
```

UNIVALI Escola Politécnica

Ordenação – Shell Sort

- Proposto por Ronald L. Shell (1959), é uma extensão do algoritmo de inserção direta
- A diferença com relação à inserção direta é o número de segmentos do vetor
- Na inserção direta é considerado um único segmento do vetor onde os elementos são inseridos ordenadamente
- No método do Shell são considerados diversos segmentos

UNIVALI Escola Politécnica

Ordenação – Shell Sort

- A cada passo o vetor é dividido em *n* segmentos
- A divisão é determinada por um fator i, calculado como 2 elevado ao número do passo.
- Os passos são contados em forma decrescente

UNIVALI Escola Politécnica

Ordenação – Shell Sort - Exemplo

Vetor original (desordenado)

1	2	3	4	5	6	7	8	9	10	11	12
15	27	40	13	19	20	45	35	50	41	25	10

Primeiro passo: I = 2 ^ NP = 4

segmento 1

1	5	9
15	19	50

segmento 2

2	6	10
27	20	41

segmento 3

3	7	11
40	45	25

segmento 4

4	8	12
13	35	10

Aplicando inserção direta em cada segmento:

15	19	50

20	27	41

25	40	45

10	13	35

Obtém-se o vetor:

1	2	3	4	5	6	7	8	9	10	11	12
15	20	25	10	19	27	40	13	50	41	45	35

UNIVALI Escola Politécnica

Ordenação – Shell Sort - Exemplo

Segundo passo: I = 2 ^ NP = 2

segmento 1

1	3	5	7	9	11
15	25	19	40	50	45

segmento 2

2	4	6	8	10	12
20	10	27	13	41	35

Aplicando inserção direta em cada segmento:

15	19	25	40	45	50

10	13	20	27	35	41

Obtém-se o vetor:

1	2	3	4	5	6	7	8	9	10	11	12
15	10	19	13	25	20	40	27	45	35	50	41

Terceiro passo: I = 2 ^ NP = 1
Nesse último passo os elementos estão próximos das suas posições finais, o que leva a um menor número de trocas.

Aplicando inserção direta ao vetor obtido no passo anterior obtém-se o vetor ordenado:

1	2	3	4	5	6	7	8	9	10	11	12
10	13	15	19	20	25	27	35	40	41	45	50

UNIVALI Escola Politécnica

Ordenação – Shell Sort - Algoritmo

Sendo *TAM* o tamanho do vetor, *vet* o vetor com os elementos a serem ordenados e np o número de passos

```
...
   para i ← np até 0 passo – 1 faça
      inc ← 2 ^ i
      para j ← 1 até inc faça
         Shell (vet, inc, j, TAM)
      fim para
   fim para
```

UNIVALI Escola Politécnica

Ordenação – Shell Sort - Algoritmo

```
Shell (vet, r, s, n)
  i, j, k, temp: numérico
  achei: lógico
início
  para i ← (s + r) até n passo r faça
        j ← s
        achei ← FALSO
        enquanto (j < i) e (não achei) faça
                 se (vet[i] < vet[j])
                          achei ← VERDADEIRO
                 senão
                          j ← j + r
                 fim se
         fim enquanto
      se (achei)
         temp ← vet[i]
         k ← i – r
         enquanto k > (j – r) faça
             vet[k + r] ← vet[k]
             k ← k – r
         fim enquanto
         vet[j] ← temp
      fim se
   fim para
fim
```

UNIVALI Escola Politécnica

Ordenação – Bolha

- É um método bastante simples, porém lento
- O nome bolha se deve ao fato de que os valores flutuam até a sua correta posição como bolhas
- A cada passo, cada elemento é comparado com o próximo. Se o elemento estiver fora de ordem, a troca é realizada
- Realizam-se tantos passos quantos necessários até que não ocorram mais trocas

UNIVALI Escola Politécnica

Ordenação – Bolha - Algoritmo

Sendo *TAM* o tamanho do vetor, *vet* o vetor com os elementos

```
troca ← VERDADEIRO
lim ← TAM - 1
 enquanto troca faça
        troca ← FALSO
        para i ← 1 até lim passo 1 faça
                se (vet[i] > vet[i + 1]) então
                        temp ← vet[i]
                        vet[i] ← vet[i + 1]
                        vet[i + 1] ← temp
                        k ← i
                        troca ← verdadeiro
                fim se
        fim para
        lim ← k
fim enquanto
```

UNIVALI Escola Politécnica

Ordenação – Quick Sort

- É o mais rápido entre os métodos apresentados até o momento
- E também o mais utilizado
- Foi proposto por C. A. R. Hoare em 1962
- Parte do princípio que é mais rápido classificar dois vetores com *n*/2 elementos cada um, do que um com *n* elementos (dividir um problema maior em dois menores)

UNIVALI Escola Politécnica

Ordenação – Quick Sort - Algoritmo

- Escolher arbitrariamente um elemento do vetor (normalmente o meio) e colocá-lo em uma variável auxiliar X
- Percorrer o vetor a partir da esquerda até que se encontre um V[I] >= X
- Percorrer o vetor a partir da direita até que se encontre um V[J] <= X
- Trocar os elementos V[I] e V[J]
- Continuar esse processo até que I e J se cruzem em algum ponto do vetor;
- Após obtidos os dois segmentos do vetor através do processo de partição, cada um é ordenado recursivamente

UNIVALI Escola Politécnica

Ordenação – Quick Sort- Algoritmo

```
QuickSort (v[], esq, dir: numérico)
          x, i, j, aux: inteiro
início
   i ← esq
   j ← dir
   x ← v[(i + j) / 2]
   repita
          enquanto x > v[i] faça
                    i ← i + 1
          fim enquanto
          enquanto x < v[j] faça
                    j ← j - 1
          fim enquanto
          se i <= j Então
             aux ← v[i]
             v[i] ← v[j]
             v[j] ← aux
             i ← i + 1
             j ← j - 1
          fim se
   até i > j
   se esq < j então
    quickSort (v, esq, j)
   fim se
   se dir > i então
    quickSort (v, i, dir)
   fim se
fim.
```

UNIVALI Escola Politécnica

Ordenação – Seleção Direta

- A cada passo encontra-se o menor elemento dentro do segmento com os elementos não selecionados
- Troca-se este elemento com o primeiro elemento do segmento
- Atualiza-se o tamanho do segmento (menos um elemento)
- Este processo é repetido até que o segmento fique com apenas um elemento

UNIVALI Escola Politécnica

Ordenação – Seleção Direta - Algoritmo

```
...
   para i ← 1 até TAM – 1 passo 1 faça
        pos_menor ← i
        para j ← i + 1 até TAM faça
                se vet[j] < vet[pos_menor]
                        pos_menor ← j
                fim se
        fim para
        temp ← vet[i]
        vet[i] ← vet[pos_menor]
        vet[pos_menor] ← temp
  fim para
```

UNIVALI Escola Politécnica

Ordenação – Merge Sort - Algoritmo

- Dividir o vetor em subvetores pequenos
- Classificar as duas metades recursivamente aplicando mergesort
- Juntar as duas metades em um único conjunto classificado

UNIVALI Escola Politécnica

Ordenação – Merge Sort- Algoritmo

```
Mergesort(v[], inicio, fim: numérico)
  l, r, m, i: numérico

inicio
  //vtemp = novo vetor do tamanho de v
  if(inicio = fim) retorno
  m ← (inicio+fim)/2

  Mergesort(inicio, m)
        Mergesort(m+1,fim)

    l ← inicio
    r ← m + 1;
```

```
     para i ← inicio até fim passo
1 faça
         se (r > fim || (l <= m &&
v[l] <= v[r])
            vtemp[i] ← A[l];
            l ← l + 1
          senão
            vtemp[i] ← A[r];
            r ← r + 1
           fim se
         fim para
   para i ← inicio até fim passo 1
faça
      v[i] ← vtemp[i]
   fim para

fim
```

UNIVALI Escola Politécnica

Complexidade - Resumo

Algoritmo	Comparações			Movimentações			Espaço	Estável	In situ
	Melhor	Médio	Pior	Melhor	Médio	Pior			
Bubble	$O(n^2)$			$O(n^2)$			$O(1)$	Sim	Sim
Selection	$O(n^2)$			$O(n)$			$O(1)$	Não*	Sim
Insertion	$O(n)$	$O(n^2)$		$O(n)$	$O(n^2)$		$O(1)$	Sim	Sim
Merge	$O(n \log n)$			–			$O(n)$	Sim	Não
Quick	$O(n \log n)$		$O(n^2)$	–			$O(n)$	Não*	Sim
Shell	$O(n^{1.25})$ ou $O(n\ (ln\ n)^2)$			–			$O(1)$	Não	Sim

* Existem versões estáveis.

UNIVALI Escola Politécnica

K-HARD

UNIVALI Escola Politécnica