Instituto Federal do Espírito Santo

Campus Serra, Coordenadoria de Informática

Curso Superior Bacharelado em Sistemas de Informação – Programação II

Tópicos de Técnicas de Programação

(VELOSO, Paulo. “Estruturas de Dados”. Editora Campus, 1995.)

Métodos de Classificação e Busca

I. Classificação

Tipos de Métodos de classificação

Métodos de classificação interna: dados se encontram todos em memória.

Métodos de classificação externa: dados se encontram em dispositivos de armazenamento (memória secundária).

Métodos de classificação estudados neste tópico:

- Classificação por inserção (método de inserção direta)
- Classificação por seleção
- Classificação tipo Bolha

I.1. Classificação por Inserção

A característica comum a todos os métodos de classificação por inserção é que eles efetivam a ordenação do conjunto de dados pela inserção de cada um dos elementos em sua posição correta dentro de um subconjunto (do conjunto original) não ordenado.

Existem dois métodos muito conhecidos de classificação por inserção, os métodos de inserção direta e o método de inserção por incrementos decrescentes (método de Shell). Neste tópico, estudaremos apenas o primeiro método.

Classificação por Inserção Direta

O mais simples dos métodos de classificação por inserção. Utilizado em pequenos conjuntos de dados. Considerado como tendo baixa eficiência computacional.

Descrição: o conjunto de dados é dividido em dois segmentos. Inicialmente, o primeiro segmento contém apenas o primeiro elemento, estando, portando, classificado. O sendo segmento, ainda não classificado, contém os elementos restantes do conjunto. Por meio de iterações sucessivas cada elemento do segundo segmento é inserido ordenadamente no primeiro, até que todos os sejam.

O exemplo a seguir ilustra um trecho da sequencia de configurações do conjunto de dados (lista de inteiros), do seu arranjo inicial até o arranjo final, completamente ordenado (de forma crescente).

```
380 260 340 220 200 420 280 320 240 300
260 380 340 220 200 420 280 320 240 300
260 340 380 220 200 420 280 320 240 300
220 260 340 380 200 420 280 320 240 300
200 220 260 340 380 420 280 320 240 300
200 220 260 340 380 420 280 320 240 300
200 220 260 280 340 380 420 320 240 300
200 220 260 280 320 340 380 420 240 300
200 220 240 260 280 320 340 380 420 300
200 220 240 260 280 300 320 340 380 420
```

A representação do processo como um algoritmo geral:

Algoritmo fazInsercao(lst)
para *i* do segundo até o último elemento de *lst* **faça**
k = 0;
enquanto (elemento da posição corrente k menor que o elemento da posição corrente *i*) e (*k* < *i*) **faça**
vá para a próxima posição *k;*
fim enquanto
se o valor do elemento na posição *k* for maior que o do elemento na posição *i* **então**
INSERIR elemento da posição *i* na posição *k* ;
fim se
fim para
retorna *lst*
fim fazInsercao

I.2. Classificação por Seleção

A característica comum a todos os métodos de classificação por seleção é que eles efetivam a ordenação por meio da seleção sucessiva do menor valor contido no conjunto de dados. A cada passo o elemento de menor valor é colocado em sua posição definitiva no conjunto ordenado e o processo é repetido para o segmento que contém os elementos ainda não selecionados.

Os métodos dessa família se diferenciam pelo abordagem utilizada para a seleção do menor elemento nos passos sucessivos. Existem dois métodos muito conhecidos de classificação por seleção, os métodos de seleção direta e o método de seleção utilizando árvore binária, conhecido como heapsort. Neste tópico, estudaremos apenas o primeiro método.

Classificação por Seleção Direta

Nesse método, no passo inicial, o primeiro elemento é tomado como o primeiro menor valor do conjunto, com os elementos restantes formando o segmento dos ainda não selecionados. Então, a cada passo é feita uma varredura do segmento contendo os elementos ainda não selecionados (o segmento não ordenado) e determinado aquele elemento de menor valor, o qual é colocado, por troca, na primeira posição, ficando então, o segmento, com 1 elemento a menos. O processo é repetido até que o segmento (daqueles ainda não selecionados) fique com apenas 1 elemento, que é o de maior valor entre todos, estando assim classificado todo o conjunto.

O exemplo a seguir ilustra um trecho da sequencia de configurações do conjunto de dados (lista de inteiros), do seu arranjo inicial até o arranjo final, completamente ordenado (de forma crescente).

```
19 25 10 18 35 17 15 13
10 25 19 18 35 17 15 13
10 13 19 18 35 17 15 25
10 13 15 18 35 17 19 25
10 13 15 17 35 18 19 25
10 13 15 17 18 35 19 25
10 13 15 17 18 19 35 25
10 13 15 17 18 19 25 35
```

A representação do processo como um algoritmo:

Algoritmo fazSelecao(lst)
    para *i* do primeiro ao penúltimo elemento de *lst* **faça**
        indiceMenor = *i*
        k = *i* +1
        enquanto *k* da posição corrente até o último elemento da lista **faça**
            se o elemento na posição *k* corrente for menor que o elemento na posição *indiceMenor* **então**
                indiceMenor = k **<= o novo menor elemento.**
            fim se
            k = *k* + 1
        fim enquanto
        trocar o elemento da posição *i* com o da posição *indiceMenor*
    fim para
    retorna *lst*
fim fazSelecao

I.3. Classificação por Troca

A característica comum a essa família de métodos é a de efetuarem a classificação por comparação sucessiva de pares de elementos do conjunto de dados, trocando-os de posição caso estejam fora da ordem desejada.

A estratégia de escolha dos pares de elementos é o que estabelece a diferença entre os diversos métodos de classificação baseado em troca.

Existem dois métodos muito conhecidos de classificação por troca, a saber, o método da bolha (bubble sort) e o método de troca e partição (quicksort). Neste tópico, estudaremos apenas o primeiro método.

Classificação por Método da Bolha

Este é um dos métodos mais simples de classificação por troca.

Descrição: A cada passo, cada elemento do conjunto de dados é comparado com o seu sucessor, sendo os dois trocados de posição caso estejam fora da ordem desejada. São executados passos sucessivos, até que em cada um deles não ocorram mais trocas, estando, pois, o conjunto de dados totalmente ordenado.

O exemplo a seguir ilustra um trecho da sequencia de configurações do conjunto de dados (lista de inteiros), do seu arranjo inicial até o arranjo final, completamente ordenado (de forma crescente).

```
280 260 240 220 200 420 380 320 340 300
260 240 220 200 280 380 320 340 300 420
240 220 200 260 280 320 340 300 380 420
220 200 240 260 280 320 300 340 380 420
200 220 240 260 280 300 320 340 380 420
```

A representação do processo como um algoritmo:

Algoritmo fazBolha(lst)
    trocou = **Verdadeiro**
    enquanto *trocou* **faça**
        trocou = Falso
        para *i* do primeiro ao penúltimo elemento de *lst* **faça**
            se o elemento da posição corrente *i* for maior que o seu sucessor **então**
                trocar elemento da posição corrente com seu sucessor
                trocou = **Verdadeiro**
            fim se
        fim para
    fim enquanto
    retorna *lst*
fim fazBolha

I.4. Classificação de Dados Estruturados

Um problema frequente na computação é a classificação de dados estruturados, dados formados por vários campos de informação. A tabela a seguir é um exemplo desse tipo de dado. A tabela em si pode ser vista como uma lista de linhas. Mantendo a coesão das linhas, como classificar esse conjunto de dados ?

Para classificar essa estrutura de dados é necessário escolher qual dos campos de informação será aquele usado como elemento a ser comparado, seja para troca, inserção ou seleção. Esse campo de dado é chamado de **chave de classificação**. A chave de classificação é **simples** quando apenas um dos **campos** é utilizado. É uma **chave composta** quando mais de um campo é utilizado. É comum mais de um campo ser utilizado como chave de classificação, especialmente quando existe a possibilidade de valores repetidos entre campos de diferentes linhas da tabela.

A tabela abaixo mostra um fragmento de uma base de dados de CEP por regiões do Brasil.

Analise e responda:

1) Qual estrutura de programação Python seria mais adequada para representar esse tipo de tabela ?

2) Como classificar a tabela utilizando as seguintes chaves de classificação: a) id_cidade e id_bairro; b) id_bairro, tipo_logradouro, logradouro ?

3) Como tornar os algoritmos de classificação menos sensíveis à estrutura da chave de classificação ?

cep	logradouro	tipo_logradouro	local	id_cidade	id_bairro
01029901	Rua Florêncio de Abreu 157	Rua	Edifício São Bento	9668	25279
01227200	Angélica	Avenida		9668	25422
01251000	Cardoso de Almeida	Rua		9775	26525
01255010	Antonina	Rua		9025	26525
01308040	São Miguel	Rua		9182	25243
01311914	Avenida Paulista 777	Avenida	Edifício Viking	9668	25243
01332906	Alameda Rio Claro 190	Avenida	Edifício Humberto	9025	25243
01416902	Rua da Consolação 3064	Rua	Edifício Jardim Am	9668	25280
01426040	das Estrelas Novas	Travessa		9775	25280
01501001	Doutor João Mendes	Praça		9668	25279
01543000	Heitor Peixoto	Rua		9025	25258
01546120	São João Del Rei	Rua		9025	25633
02072001	Conceição	Avenida		9668	25270
02084150	Articura	Rua		9668	26807
02119020	Prefeito Milton Improta	Rua		9182	26853
02123005	Manuel Lainez	Travessa		9775	25820
02131040	Hiroshima	Rua		9025	26851
02138000	Lourenço Billis	Praça		9668	27006
02140030	Antionha	Rua		9025	27006
02147015	Cláudio Leonardi	Praça		9775	26379

II. Busca/Pesquisa

Métodos de busca/pesquisa: procedimentos que testam se um determinado elemento pertence a um conjunto de elementos.

Tipos de Métodos de busca

Métodos de pesquisa interna: dados se encontram todos em memória.

Métodos de pesquisa externa: dados se encontram em dispositivos de armazenamento (memória secundária).

Métodos de pesquisa estudados neste tópico:

- Pesquisa Sequencial;
- Pesquisa Binária
- Pesquisa por Mapeamento ou Hashing.

II.1. Pesquisa Binária

Um dos métodos de busca, e conceito, mais populares da ciência da computação. Utilizada para encontrar um elemento entre os vários componentes de um conjunto de dados ordenado. Este, por sua vez, é uma das marcas do método, a necessidade de que o conjunto de dados esteja ordenado.

Descrição: O cenário consiste de um conjunto de dados ordenado, e um elemento o qual se deseja verificar se faz parte desse conjunto (se o elemento pertence ou não, e se pertence, em qual posição do conjunto ele se encontra armazenado). Primeiramente o procedimento encontra o elemento de posição central do conjunto de dados. Em seguida, compara o elemento procurado com esse elemento central. Se os elementos forem idênticos, então a busca terminou. Se o elemento procurado for maior que o elemento central, então o procedimento é repetido para o segmento compreendido entre o elemento central e o último elemento do conjunto. Se o elemento procurado for menor que o elemento central, então o procedimento é repetido para o segmento compreendido entre o o primeiro elemento do conjunto e o elemento central (não incluído).

A representação do processo como um algoritmo:

```
Algoritmo fazBuscaBin(lista, elem)
        inicio = 0
        fim = tamanho da lista - 1
        enquanto inicio <= fim faça
                meio = (inicio + fim) / 2 <= divisão inteira.
                se elemento da posição meio for igual a elem então
                        inicio = fim + 1 # Provoca o fim do enquanto.
                senão
                        se elemento da posição meio for maior que elem então
                                fim = meio - 1
                        senão
                                inicio = meio + 1
                        fim se
                fim se
        fim enquanto

  se elemento da posição meio for igual a elem então
        retorna meio
  senão
        retorna  -1
fim  fazBuscaBin
```

Exercício para revisão:

1) Construa a implementação em Python para os algoritmos de classificação vistos anteriormente (Inserção, Seleção e Troca-Bolha).

2) Construa a versão estendida do algoritmo de busca binária delegando a avaliação da condição de identidade para uma função externa via parâmetro funcional.

III. Bibliografia

VELOSO, Paulo. “Estruturas de Dados”. Editora Campus, 1995.

HOROWITZ, E. “Fundamentos de Estruturas de Dados”. Editora Campus, 1987.