Universidade Federal de Goiás
Goiânia 04 de Maio de 2018
Edson Júnior Frota Silva
Matrícula: 201515412
Engenharia de Computação
Prof°(a): Renata Dutra
Estrutura de Dados II

Atividade parte 1 proposta no dia 03/05/2018:

ORIENTAÇÕES GERAIS

- Atividade em dupla.
- (Parte 1 – 02/05/2018) **Sintetize** os conceitos sobre:
  - Pesquisa sequencial
  - Pesquisa binária
  - Hashing (tabela de dispersão, hashing aberto e hashing fechado)

Pesquisa Sequencial: A pesquisa sequencial consiste que a partir do primeiro registro de um vetor qualquer “v”, pesquisará sequencialmente até encontrar a chave ou valor procurado; caso seja encontrado o que desejamos, então pare. Melhor dizendo, suponhamos que estamos procurando um valor qualquer “x” que esteja disposto em nosso vetor qualquer “v”, a partir da primeira posição do nosso vetor “v” procuraremos o valor desejado até o final. Caso se chegamos na última posição do nosso vetor “v” e não encontramos o que desejamos concluímos que o valor “x” não está em nosso vetor “v”.Podemos concluir que se trata de um armazenamento de um conjunto de registros por meio de um vetor.

Exemplo 1 pesquisa sequencial :

```
1   /*
2       Retorna -1 caso não encontre ou a posição, caso encontre.
3    */
4   int procura(char vetor[], int tamanho, char elementoProcurado) {
5       int i;
6       for (i = 0; i < tamanho; i++) {
7           if (vetor[i] == elementoProcurado) {
8               return i;
9           }
10      }
11      return -1;
12  }
```

Pesquisa Binária: A pesquisa binária consiste em procurar um elemento em um vetor ordenado de dados. O funcionamento é simples,

primeiramente é recebido o valor que se deseja encontrar no caso “x” e a patir daí ele verifica se o elemento que desejamos é igual a posição central do nosso vetor “v” se a afirmação for verdadeira então a busca para. Caso o elemento desejado “x” seja menor que a posição central do vetor “v”, passamos a procurar o apenas na parte da esquerda da posição central do vetor “v”. Caso o elemento desejado “x” seja maior que a posição central, ai procuramos a sua direita, até encontrar o elemento desejado. Essas busca são feitas recursivamente na pesquisa binária.

Exemplo 2 pesquisa binária :

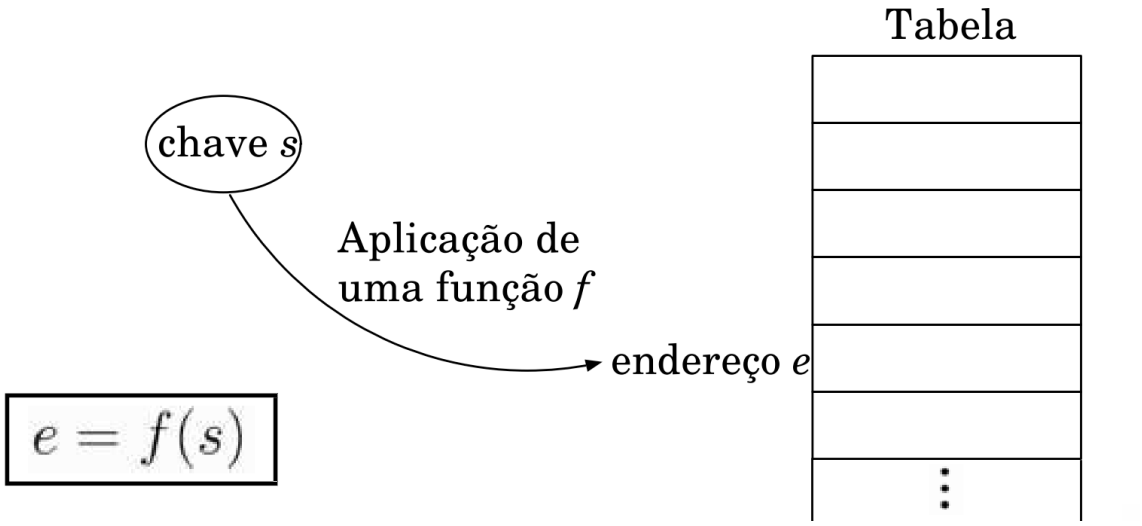
```
 1  // x => chave | v[] => vetor ordenado | e => limite inferior (esquerda) | d
=> limite superior (direita)
 2  int PesquisaBinaria (int x, int vetor[], int e, int d)
 3  {
 4      int meio = (e + d)/2;
 5
 6      if (vetor[meio] == x)
 7      {
 8          return meio;
 9      }
10      if (e >= d)
11      {
12          return -1; // não encontrado}
13      }
14      else if (vetor[meio] < x)
15      {
16          return PesquisaBinaria(x, vetor, meio+1, d);
17      }
18      else
19      {
20          return PesquisaBinaria(x, vetor, e, meio-1);
21      }
22  }
```

Hashing: Podemos dizer que o Hashing é um método de pesquisa uma função que efetua comparações para poder localizar a chave.

Tabelas de Dispersão: As tabelas de dispersão tem a finalidade de armazenar os valores, sendo que cada um desses valores estão associados a uma chave diferente. As chaves associadas aos valores são distintas umas das outras pois são utilizadas para mapear os valores na tabela, a partir dai utilizamos a função hashing para mapear a tabela.

Exemplo 3 tabelas de dispersão :

Transformar as chaves em endereços de uma tabela, como tentativa de fazer a busca de chaves em tempo O(1).



Hashing Aberto: No método de hasing de aberto temos que todos os elementos são armazenados na própria tabela hashing, isto é, não existem listas nem elementos armazenados fora da tabela.. A vantagem de se utilizar hashing aberto é que a quantidade de memória utilizada para armazenar ponteiros é utilizada para aumentar o tamanho da tabela, possibilitando menos colisões e aumentando a velocidade de recuperação das informações.
Para inserir um novo elemento, examinamos sucessivamente a tabela até encontrarmos uma posição vazia onde possamos armazenar o elemento. De fato não percorremos sempre a tabela inteira, ou seja, a busca depende do elemento a ser inserido.

Exemplo 4 hashing aberto:

```
 1  void Hash_Insere(int x)
 2  {
 3      int j,h,i=0;
 4      do
 5      {
 6          j = h(x,i);
 7
 8          if (T[j] == VAZIO)  //T é tabela hash
 9          {
10              T[j] = x;
11              return;
12          }
13          else
14              i++
15          }
16      while(i < tablesize)
17          printf("erro.");              }
```

Hashing Fechado: Na técnica de hashing fechado temos que não necessita de ponteiros na sua implementação, é um forma que foi desenvolvida antigamente, e era adequada para implementação em disco. Quando uma chave for inserida, a função hash é aplicada, e ela é acrescentada à lista adequada.

Edson Júnior Frota Silva
Engenharia de Computação - UFG