Computação 2

Aula 12

Algoritmos de Pesquisa

Profª. Fabiany
fabianyl@utfpr.edu.br

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Pesquisa

- A pesquisa de dados na memória envolve a busca por conjunto de informações armazenados em estruturas de dados;
- Este tipo de busca pode ser considerado uma pesquisa simples, onde sabe-se exatamente qual o dado a ser procurado.
- Este dado conhecido pode ser chamado de chave da pesquisa.

Pesquisa

- A idéia básica da pesquisa portanto é dado um conjunto de dados e uma chave, localizar um elemento neste conjunto cujo valor da chave de pesquisa seja o mesmo da chave usada.

- O conjunto de dados pode ser armazenado e organizado usando diferentes estruturas e tipos abstratos de dados: vetores, matrizes, listas encadeadas, árvores e etc.

- A partir dessas estruturas, pode-se realizar diferentes tipos de pesquisa.

Pesquisa

- Em uma pesquisa, o conceito de eficiência é muito importante. A eficiência pode ser medida pelo número de inspeções ou teste realizados para encontrar o dado procurado.
- Quanto menos testes forem feitos, mas eficiente será a busca.
- Os métodos de pesquisa a serem utilizados dependem da forma como os dados estão armazenados e da quantidade de dados armazenados. Os dados podem estar armazenados de forma desordenada e desconhecida ou de forma ordenada.

Pesquisa

- Encontrar os dados em um vetor (ou matriz) desordenada requer uma busca sequencial, começando no primeiro elemento e parando quando o elemento procurado ou o final do vetor é encontrado.
- A pesquisa sequencial geralmente é aplicada quando os dados estão desordenados, mas também pode ser aplicada a dados ordenados.
- Se os dados forem ordenados, usa-se a pesquisa binária, o que ajuda a localizar o dado mais rapidamente.

Pesquisa Sequencial

- Método de pesquisa mais simples;
- Este método consiste em realizar a busca do elemento procurando através da comparação com cada um dos elementos no conjunto de dados, na ordem em que estão inseridos.
- Fácil de ser implementada;
- Análise:
  - Pesquisa com sucesso:

    melhor caso : C(n) = 1

    pior caso : C(n) = n

    caso médio: C(n) = (n + 1) / 2
  - Pesquisa sem sucesso:

    C (n) = n + 1.

Pesquisa Sequencial

```
int busca_sequencial (char *item, int count, char key)
{
    int t;
    for(t=0; t<count; t++)
        if(key == item[t]) return t;
    return -1;
 }

main(){
  char chave, vogais[6] = {'a','e','i','o','u','\0'};
   int indice;

  printf("Digite a vogal a ser procurada\n");
  scanf("%c", &chave);

 indice = busca_sequencial(vogais,6,chave);
  if (indice == -1)  printf("Não encontrou\n");
 else  printf("A vogal está posição %i\n", indice);
}
```

Pesquisa Binária

- Este tipo de pesquisa pode ser mais eficiente se os dados estiverem ordenados.
- Este método utiliza a abordagem "dividir e conquistar";
- Primeiramente compara a chave de pesquisa com o elemento central, se a chave é menor que esse elemento, procura na primeira metade, se a chave é maior que esse elemento procura na segunda metade e assim sucessivamente.
- Análise:

  melhor caso : $C(n) = 1$

  pior caso : $C(n) = \log n$

  caso médio: $C(n) < \log (n)$

Pesquisa Binária

```
int busca_binaria (int *item, int count, char key)
{
    int menor, maior, meio;
    menor = 0; maior = count – 1;
    while( menor <= maior)
        meio = (menor + maior)/2;
        if (key < item[meio]) maior = meio -1;
        else if (key > item[meio]) menor = meio + 1;
        else return meio;  //encontrou
    }
    return -1;
  }
main(){
   int chave, numeros[6] = {1,2,3,4,5,6};
   int indice;

  printf("Digite o numero a ser procurado\n");
  scanf("%d", &chave);

 indice = busca_binaria (numeros, 6, chave);
  if (indice == -1)  printf("Não encontrou\n");
 else  printf("A vogal está posição %i\n", indice);
}
```

Referências Bibliográficas

- **SHILDT**, H. C, Completo e Total, 3a edição, rev. e atual. Ed. Makron. São Paulo, c1997.