Tabelas de hashing (espalhamento)

Espalhar – uma bagunça organizada

Sei exatamente onde está cada folha de papel na minha mesa!!!

É uma questão de saber onde procurar !!!

Conjuntos dinâmicos/ dicionários

- Inserir
- Buscar
- Remover
-
- Endereçamento direto
  - Função de hashing
  - Quantidade de memória
  - Colisões

Calculando chaves

- No computador qualquer coisa é codificada e representada como conteúdo de memória. Este conteúdo por sua vez, pode ser visto como números.

```
int ascii_of_A = (int)'A';  // retorna 65

int calculaChave (String txt) {
      int h=0;
      for (int i = 0; i < txt.length(); i++) {
           int chr = txt.charAt(i);
           h+= chr;
       }
       return (h%103)
```

Hashing com solução de colisões por encadeamento

- Inserção
- Busca
- Remoção
-

Complexidade de tempo da busca (CLR)

- fator de carga (ou taxa de ocupação) $\alpha = \frac{n}{m}$
- Inserção no final da lista.
- Complexidade (média de pior caso) de uma busca com sucesso

$$T(n) = \frac{1}{n}\sum_{i=1}^{n}\left(1+\frac{i-1}{m}\right) = 1+\frac{1}{nm}\sum_{i=1}^{n}\left(i-1\right) =$$

$$= 1+\frac{(n-1)n}{2nm} = 1+\frac{\alpha}{2}-\frac{1}{2m} = \Theta(1+\alpha)$$

Uma outra argumentação

- fator de carga (ou taxa de ocupação) $\alpha = \frac{n}{m}$
- Complexidade (média de pior caso) de uma busca com sucesso

$$T(n) = \frac{1}{n} \sum_{i=1}^{n} \left(1 + \frac{n}{m} \right) = 1 + \frac{1}{nm} \sum_{i=1}^{n} (n) =$$

$$1 + \frac{n^2}{nm} = 1 + \alpha = \Theta(1 + \alpha)$$

...

- O que acontece se todas as chaves levam ao mesmo “endereço” em uma tabela de hashing?
-
- Qual a complexidade de tempo das operações nesse caso?
-
- Como resolver esse “problema”??

...

- Qual função de hash ideal?
-
- O que pode ser feito na prática para se aproximar do ideal?

- “Método da divisão”
- $$h(k) = k \bmod m$$
- “Método da multiplicação”
- $$h(k) = m * ((k * A) \bmod 1)$$
- Onde A é, de preferência um irracional (ex. Pi, número de ouro,...)
-

Hashing Universal

- Há casos em que