

# 13 – Hashing Interno (parte 1)

SCC501 - Introdução à Ciência de Computação II

Paulo H. R. Gabriel  
phrg@icmc.usp.br  
Prof. Moacir Ponti Jr.  
[www.icmc.usp.br/~moacir](http://www.icmc.usp.br/~moacir)

Instituto de Ciências Matemáticas e de Computação – USP

2010/2



- 1 Contextualização
- 2 Motivação
- 3 Tabela *Hash*
- 4 Tratamento de Colisões

- Métodos de busca estudados até agora: Comparação de chaves
- Valor relativo de cada chave
- Tiramos proveito da ordenação

## Custo

Ordenação:  $\mathcal{O}(n \cdot \log(n))$  ou  $\mathcal{O}(n)$

Busca:  $\mathcal{O}(\log(n))$

Inserção?

Remoção?



# Tabela *Hash* ou de Espalhamento ou de Dispersão

- Valor absoluto das chaves
- Não compara, **define** valor (função)
- Atingir diretamente a posição

Em outras palavras...

$\mathcal{O}(1)$

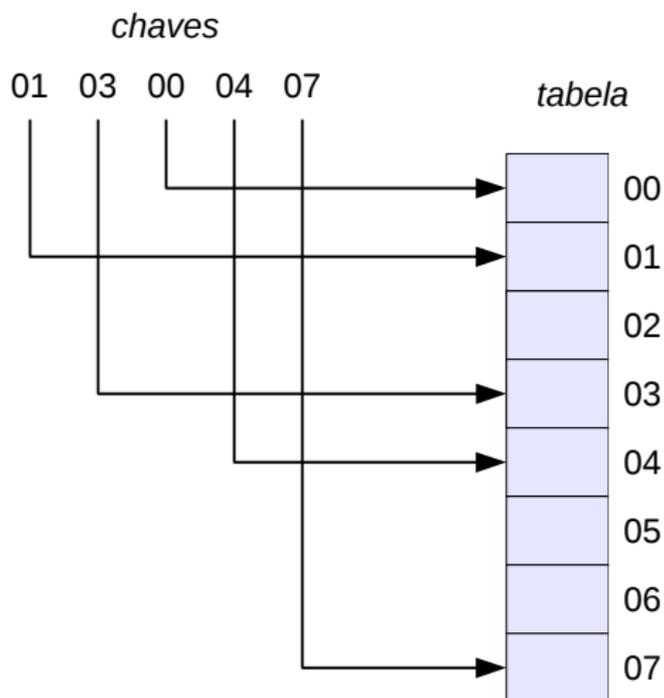
Intuitivo

Fazemos com frequência...



# Acesso direto (1/2)

- Indexar elementos em vetores



## Limitações

- Elementos não numéricos
- Quantidade de espaço vazio
- Frequência de atualizações

## Exemplo: Dicionários

- Conjunto de palavras e significados
- Diversas aplicações práticas (e.g., pré-processamento)
- Necessidade de acesso direto (**eficiência**)
- “Volátil”

## Ideia Central

Utilizar uma **função**, aplicada a [parte] da informação (*i.e.*, à **chave**), para devolver o **índice** onde a informação deve [deveria] ser armazenada.

- A função é chamada **função de espalhamento** (ou função *hash*)
- A estrutura de dados é a **tabela de espalhamento** (ou tabela *hash*)
- A posição onde a chave é alocada recebe o nome de **compartimento** (ou *bucket*)



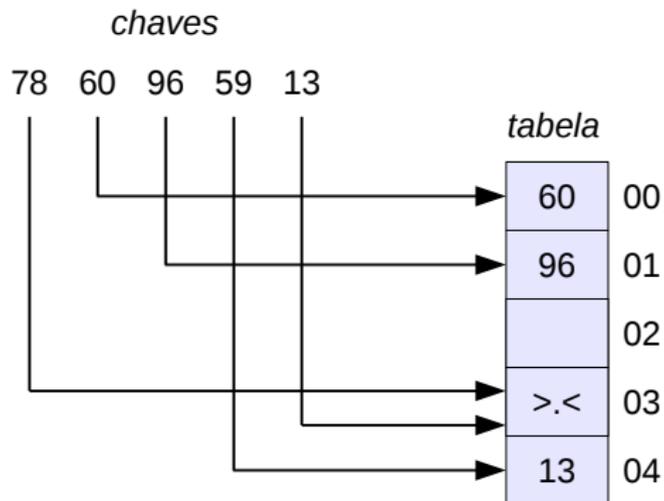
## Função de Espalhamento

Uma função de espalhamento  $h()$  transforma uma chave  $x$  em um endereço-base  $h(x)$  da tabela hash.

- Função hash: Função **injetora**
- Na prática:
  - E se  $h(x)$  estiver ocupado?
  - Não há garantias de injetividade: pode existir  $y \neq x$  tal que  $h(y) = h(x)$



# Conceitos (3/4)



*chave mod 5*



- Idealmente:
  - Número baixo de colisões
  - Facilmente computável
  - Uniforme



- Fator de carga (número esperado de espaços ocupados):

$$\alpha = \frac{n}{m}$$

- Menor fator de carga  $\rightarrow$  Menor número de colisões
- Sempre pode haver uma colisão
- Previsão de tratamento de colisões
  - Endereçamento aberto
  - Encadeamento



## Ideia básica

Calcular novo endereço para chaves sinônimas

- Em caso de colisão, calcula-se o novo índice
- Processo iterativo, enquanto a chave não for armazenada
- Pode-se utilizar vetor circular



# Tratamento de Colisões por Endereçamento Aberto (2/3)

- *Overflow* progressivo
  - Sondagem linear
  - Sondagem quadrática
- *Rehash* – Aplicar segunda função *hash*



- Sondagem linear
  - Todas posições da tabela são checadas
  - $rh(h(x)) = (h(x) + k) \bmod m$   
 $0 \leq k \leq m - 1$
  - Longos trechos consecutivos podem ser ocupados
- Sondagem quadrática
  - Diversifica sequência de endereços
  - $rh(h(x)) = (h(x) + c_1 \cdot k + c_2 \cdot k^2) \bmod m$   
 $0 \leq k \leq m - 1$   
 $c_1$  e  $c_2$  são constantes,  $c_2 \neq 0$
  - Agrupamento ocorre com menos frequência



## Mais natural

Armazena chaves sinônimas em listas encadeadas

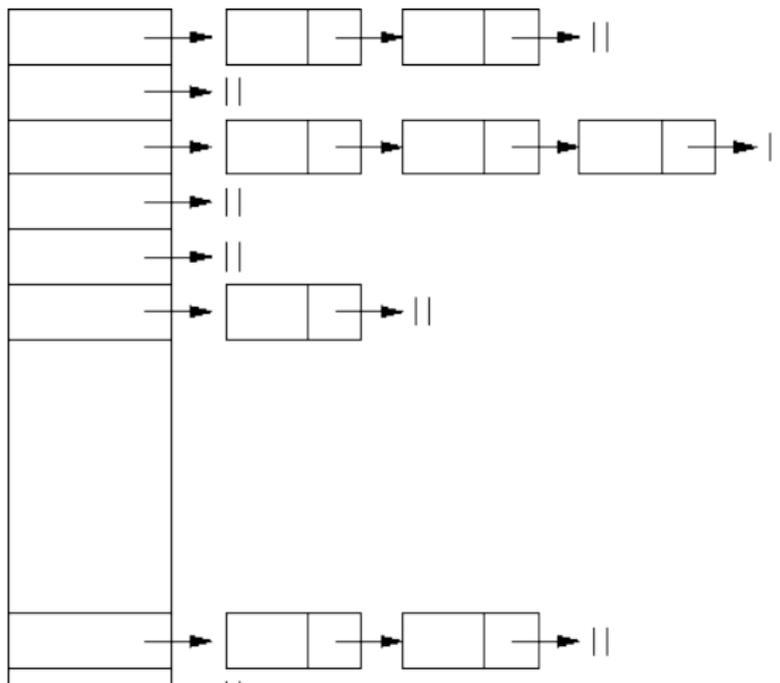
## Duas implementações

- “Externo” à tabela
- “Interno” à tabela



# Encadeamento Externo (1/2)

- $m$  listas encadeadas
- Endereço base: cabeça da lista
- Busca, inserção e remoção em listas encadeadas



## Encadeamento Externo (2/2)

- Melhor caso (inserir):  $\mathcal{O}(1)$
- Pior caso (buscar):  $\mathcal{O}(n)$
- Pode-se inserir ordenado para reduzir a busca
- **Deve-se questionar:** Qual operação será realizada mais vezes?

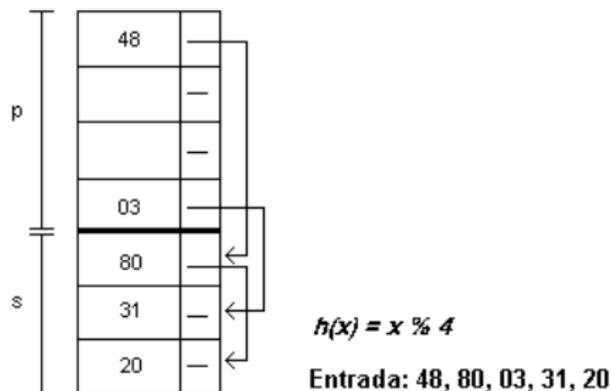


# Encadeamento Interno (1/3)

- Divisão da tabela:  $p + s = m$
- Função de espalhamento busca endereços em  $[0, p - 1]$
- Cada posição tem dois campos
  - Armazenamento da chave
  - Ponteiro para próximo sinônimo



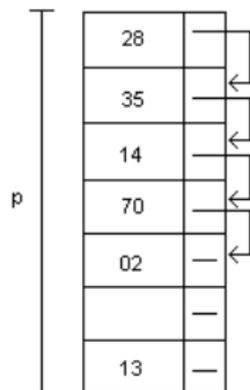
# Encadeamento Externo (2/3)



- Pode degenerar em uma lista encadeada

# Encadeamento Externo (3/3)

- Pode-se não diferenciar as duas regiões



$$h(x) = x \% 7$$

Entrada: 28, 35, 13, 14, 70, 02

- Problema: colisões secundárias





ROSA, J. L. G.

*Métodos de Busca.*

Slides de aula SCC-201, ICMC-USP.



SHEWCHUK, J.

*Hash Table – Data Structures University of California, Berkeley.*

Disponível em: <http://www.youtube.com/watch?v=UPo-M8bzRrc>



ZIVIANI, N.

*Projeto de Algoritmos (Capítulo 5). 3.ed..*

Cengage Learning



CORMEN, T. H., LEISERSON, C. E., RIVEST, R. L., STEIN, C.

*Introduction to Algorithms.*

MIT Press, 2001



-  LEVITIN, A. V.  
*Introduction to the Design and Analysis of Algorithms (Capítulo 7), 1.ed..*  
Addison-Wesley Longman Publishing Co., Inc., 2002.
-  Szwarcfiter, J. L., Markenzon, L.  
*Estruturas de Dados e seus Algoritmos (Capítulo 8), 2.ed.*  
LTC Editora, 1994.
-  NONATO, L. G.  
*Material da disciplina SCE5763 – Tipos e Estrutura de Dados.*  
[www.lcad.usp.br/~nonato/ED/EDpos.html](http://www.lcad.usp.br/~nonato/ED/EDpos.html), 2000.