

Estruturas de Dados II

Tabela Hash

Prof^a. Juliana de Santi

Prof. Rodrigo Minetto

Universidade Tecnológica Federal do Paraná

Material compilado de: Cormen, Notas de aula Minetto

- UTFPR, IC-UNICAMP e IME-USP

Sumário

1 Introdução

2 Funções Hash

- Método da divisão
- Método de multiplicação

3 Tratamento de colisões

- Encadeamento
- Endereçamento aberto
- End. aberto: Sondagem linear
- End. aberto: Sondagem quadrática
- End. aberto: Hash duplo

4 Aplicações

Motivação

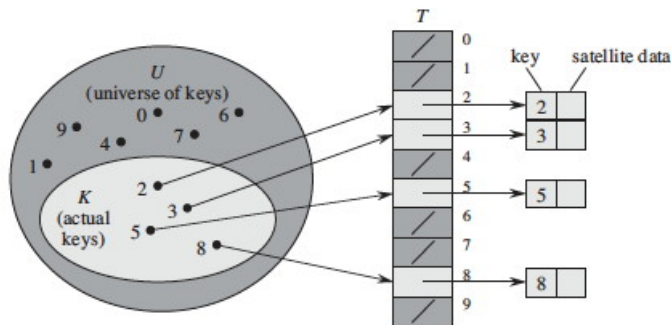
Vimos que uma árvore AVL permite realizar as operações de busca, inserção e remoção em $\mathcal{O}(\log n)$. Seria possível realizar estas operações em $\mathcal{O}(1)$? Uma **tabela hash** é uma estrutura de dados projetada para ser eficiente neste tipo de operação — também chamadas de *operações de dicionário*.

Tabela hash (hash table), ou *tabela de dispersão*, é uma estrutura de dados que permite armazenar informações de maneira associativa. O uso eficiente de **arrays** é um conceito chave em uma tabela hash.

Introdução

Alguns parâmetros da **tabela hash** são:

- M : tamanho da tabela hash.
- N : número de chaves armazenadas.
- $\alpha = N/M$: fator de carga da tabela.



$$M = 10 \quad N = 4 \quad \alpha = 0.4$$

Introdução

Um hash tem dois ingredientes fundamentais: uma **função de hashing** e um mecanismo para **re-solução de colisões**. A função de hashing mapeia uma chave (número inteiro, número real, string, ...) para um índice inteiro. O índice é então utilizado para acessar a posição no array. Se a posição já estiver ocupada é necessário um mecanismo para tratamento de colisões.

Sumário

1 Introdução

2 Funções Hash

- Método da divisão
- Método de multiplicação

3 Tratamento de colisões

- Encadeamento
- Endereçamento aberto
- End. aberto: Sondagem linear
- End. aberto: Sondagem quadrática
- End. aberto: Hash duplo

4 Aplicações

Funções Hash

O desempenho médio de uma tabela hash depende em como a função h distribui o conjunto de chaves através das M posições. A **hipótese de hash uniforme simples** supõe que uma chave k tem igual probabilidade de efetuar o hash para qualquer uma das M posições, independentemente de onde qualquer outro elemento tenha efetuado o hash

$$Pr(h(k_i) = h(k_j)) = \frac{1}{M}$$

Funções Hash

Idealmente uma **função hash** deve satisfazer as seguintes condições:

- Espalhar as chaves de maneira razoavelmente uniforme: $P_k = 1/M$, para toda chave k e endereços $h(k) \in [0, \dots, M - 1]$.
- Produzir um número baixo de colisões
- Ser computacionalmente eficiente $\mathcal{O}(1)$

Sumário

1 Introdução

2 Funções Hash

- Método da divisão
- Método de multiplicação

3 Tratamento de colisões

- Encadeamento
- Endereçamento aberto
- End. aberto: Sondagem linear
- End. aberto: Sondagem quadrática
- End. aberto: Hash duplo

4 Aplicações

Funções Hash - Método de divisão

No **método da divisão** mapeamos uma chave k para um intervalo $[0, \dots, M - 1]$ através do resto da divisão de k por M .

Método da divisão

A função hash h é definida como

$$h(k) = k \bmod M$$

Este método é muito rápido pelo fato de exigir uma única operação de divisão.

Funções Hash - Método da divisão

No **método da divisão** evita-se certos valores para M . Suponha que todas as chaves são números pares (todas as chaves são divisíveis por 2). Suponha também que M é par, neste caso, todos os índices produzidos pelo método da divisão são pares. Assim, as posições $1, 3, 5, 7, \dots, M - 2$ jamais serão utilizadas. O fenômeno existe mesmo em situações mais sutis (um conjunto com muito mais chaves pares que ímpares).

Funções Hash - Método da divisão

O ideal é escolher como valor de M um número primo. Por exemplo, para armazenar 2000 elementos em uma tabela hash, supondo que uma busca com sucesso possa visitar até 3 elementos, então M deve ser um número primo perto de $2000/3$, ou seja, M poderia ser 701.

Sumário

1 Introdução

2 Funções Hash

- Método da divisão
- Método de multiplicação

3 Tratamento de colisões

- Encadeamento
- Endereçamento aberto
- End. aberto: Sondagem linear
- End. aberto: Sondagem quadrática
- End. aberto: Hash duplo

4 Aplicações

Funções Hash - Método de multiplicação

Método da multiplicação

A função hash h é definida como

$$h(k) = \lfloor M(kA - \lfloor kA \rfloor) \rfloor$$

para uma constante $0 < A < 1$. A escolha de A depende dos dados, Knuth sugere $A = \frac{\sqrt{5}-1}{2} = 0.618033\dots$. A vantagem deste método é que a escolha de M não é crítica, em geral $M = 2^p$ para algum inteiro p , o que implica em vantagem computacional.

Sumário

1 Introdução

2 Funções Hash

- Método da divisão
- Método de multiplicação

3 Tratamento de colisões

- Encadeamento
- Endereçamento aberto
- End. aberto: Sondagem linear
- End. aberto: Sondagem quadrática
- End. aberto: Hash duplo

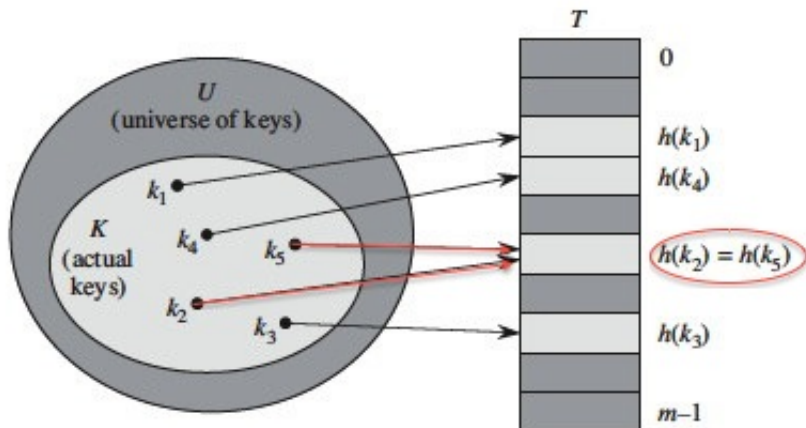
4 Aplicações

Tratamento de colisões

É possível que existam duas (ou mais) chaves x e y , onde $x \neq y$, tal que $h(x) = h(y)$. Note que nesse caso a posição $h(y)$ pode estar ocupada pela chave x , essa situação é conhecida como **colisão**. Assim, para armazenar y , utiliza-se um procedimento especial, conhecido como **tratamento de colisões**.

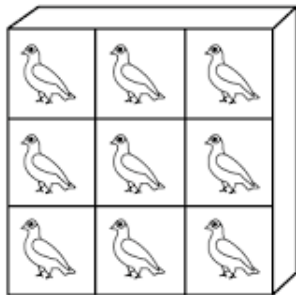
Tratamento de colisões

Colisão entre as chaves k_2 e k_5 .



Tratamento de colisões

Princípio da casa dos pombos (teorema de Dirichlet): se N pombos devem ser postos em M casas, e se $N > M$, então pelo menos uma casa irá conter mais de um pombo.



Introdução

Devido ao princípio da casa dos pombos, sabemos que colisões sempre existem. **Questão:**

qual a probabilidade de uma colisão? **Para-**

doxo do aniversário: quantas pessoas são necessárias para que a probabilidade de que duas delas façam aniversário no mesmo dia (colisão) seja maior do que 50%?

Prob. de colisão de aniversários para 2 pessoas:

$$p(\text{colisão}) = 1 - \left(\frac{365}{365} \times \frac{364}{365} \right)$$

Introdução

Devido ao princípio da casa dos pombos, sabemos que colisões sempre existem. **Questão:** qual a probabilidade de uma colisão? **Paradoxo do aniversário:** quantas pessoas são necessárias para que a probabilidade de que duas delas façam aniversário no mesmo dia (colisão) seja maior do que 50%? Prob. de colisão de aniversários para 2 pessoas:

$$p(\text{colisão}) = 0.002$$

Introdução

Devido ao princípio da casa dos pombos, sabemos que colisões sempre existem. **Questão:**

qual a probabilidade de uma colisão? **Para-**

doxo do aniversário: quantas pessoas são necessárias para que a probabilidade de que duas delas façam aniversário no mesmo dia (colisão) seja maior do que 50%?

Prob. de colisão de aniversários para 3 pessoas:

$$p(\text{colisão}) = 1 - \left(\frac{365}{365} \times \frac{364}{365} \times \frac{363}{365} \right)$$

Introdução

Devido ao princípio da casa dos pombos, sabemos que colisões sempre existem. **Questão:** qual a probabilidade de uma colisão? **Paradoxo do aniversário:** quantas pessoas são necessárias para que a probabilidade de que duas delas façam aniversário no mesmo dia (colisão) seja maior do que 50%? Prob. de colisão de aniversários para 3 pessoas:

$$p(\text{colisão}) = 0.008$$

Introdução

Devido ao princípio da casa dos pombos, sabemos que colisões sempre existem. **Questão:**

qual a probabilidade de uma colisão? **Paradoxo do aniversário:**

quantas pessoas são necessárias para que a probabilidade de que duas delas façam aniversário no mesmo dia (colisão) seja maior do que 50%?

Prob. de colisão de aniversários para n pessoas:

$$p(\text{colisão}) = 1 - \left(\frac{365}{365} \times \frac{364}{365} \times \cdots \times \frac{365 - (n - 1)}{365} \right)$$

Introdução

Devido ao princípio da casa dos pombos, sabemos que colisões sempre existem. **Questão:** qual a probabilidade de uma colisão? **Paradoxo do aniversário:** quantas pessoas são necessárias para que a probabilidade de que duas delas façam aniversário no mesmo dia (colisão) seja maior do que 50%? Prob. de colisão de aniversários para 23 pessoas:

$$p(\text{colisão}) = .538$$

Introdução

Devido ao princípio da casa dos pombos, sabemos que colisões sempre existem. **Questão:** qual a probabilidade de uma colisão? **Paradoxo do aniversário:** quantas pessoas são necessárias para que a probabilidade de que duas delas façam aniversário no mesmo dia (colisão) seja maior do que 50%? Prob. de colisão de aniversários para 366 pessoas:

$$p(\text{colisão}) = 1$$

Sumário

1 Introdução

2 Funções Hash

- Método da divisão
- Método de multiplicação

3 Tratamento de colisões

- Encadeamento
- Endereçamento aberto
- End. aberto: Sondagem linear
- End. aberto: Sondagem quadrática
- End. aberto: Hash duplo

4 Aplicações

Tratamento de colisões por encadeamento

No **encadeamento**, colocamos todos os elementos que efetuam o hash para a mesma posição em uma lista ligada. A posição **i** contém um ponteiro para o início da lista encadeada de todos os elementos armazenados que efetuam hash para **i**. Se a lista for vazia a posição contém NIL.

Tratamento de colisões por encadeamento

Função hash $h(k) = k \bmod M$

0	
1	
2	
3	
4	
5	
6	
7	

Tratamento de colisões por encadeamento

Função hash $h(k) = k \bmod 8$

0	
1	
2	
3	
4	
5	
6	
7	

Tratamento de colisões por encadeamento

Função hash $h(k) = k \bmod 8$

0	
1	
2	
3	
4	
5	
6	
7	

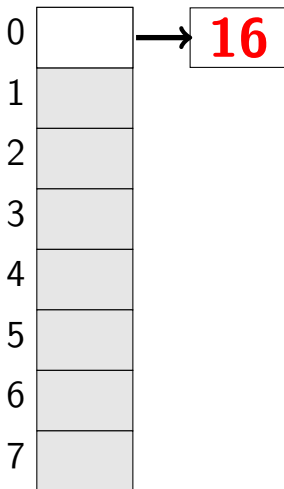
Tratamento de colisões por encadeamento

Função hash $h(\mathbf{16}) = \mathbf{16} \bmod 8 = \mathbf{0}$

0	
1	
2	
3	
4	
5	
6	
7	

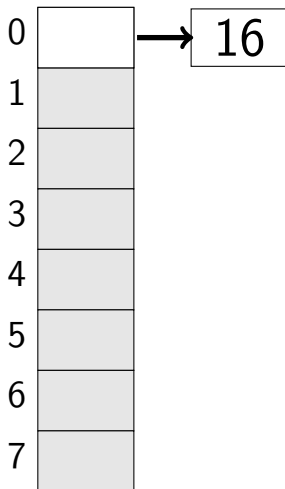
Tratamento de colisões por encadeamento

Função hash $h(\mathbf{16}) = \mathbf{16} \bmod 8 = \mathbf{0}$



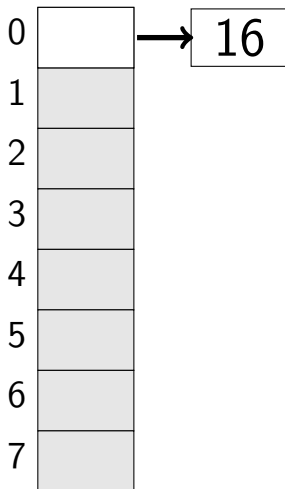
Tratamento de colisões por encadeamento

Função hash $h(16) = 16 \bmod 8 = 0$



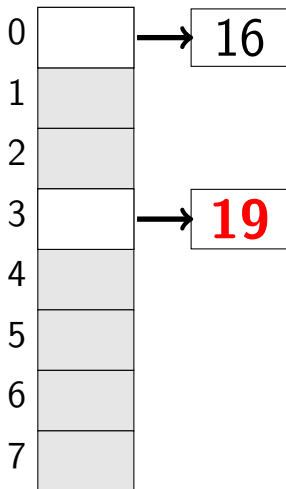
Tratamento de colisões por encadeamento

Função hash $h(\mathbf{19}) = \mathbf{19} \bmod 8 = \mathbf{3}$



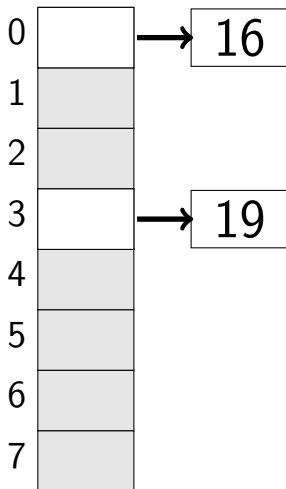
Tratamento de colisões por encadeamento

Função hash $h(\mathbf{19}) = \mathbf{19} \bmod 8 = \mathbf{3}$



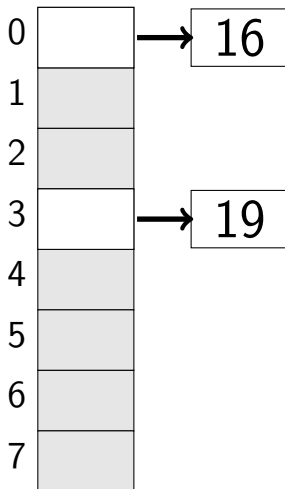
Tratamento de colisões por encadeamento

Função hash $h(19) = 19 \bmod 8 = 3$



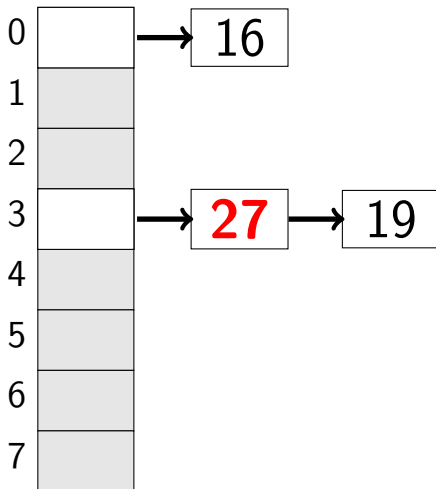
Tratamento de colisões por encadeamento

Função hash $h(27) = 27 \bmod 8 = 3$



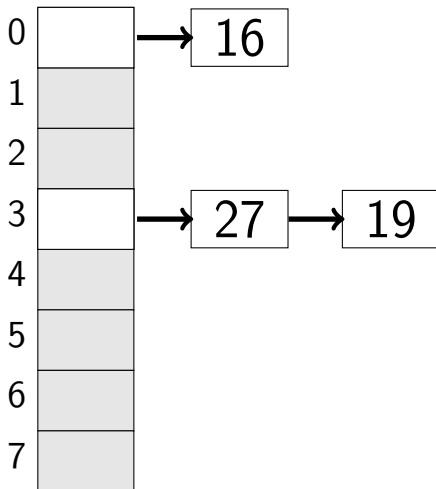
Tratamento de colisões por encadeamento

Função hash $h(\mathbf{27}) = \mathbf{27} \bmod 8 = \mathbf{3}$



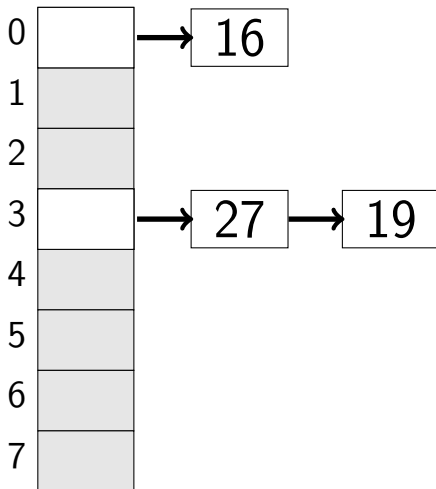
Tratamento de colisões por encadeamento

Função hash $h(27) = 27 \bmod 8 = 3$



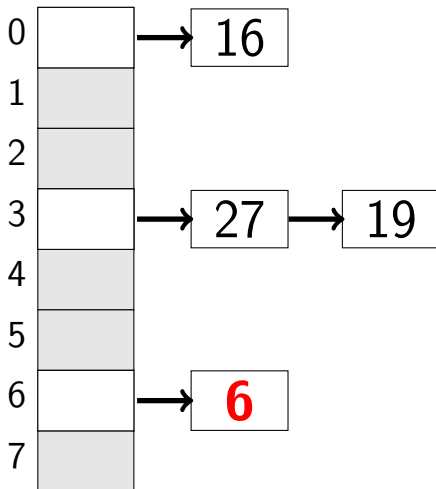
Tratamento de colisões por encadeamento

Função hash $h(\mathbf{6}) = \mathbf{6} \bmod 8 = \mathbf{6}$



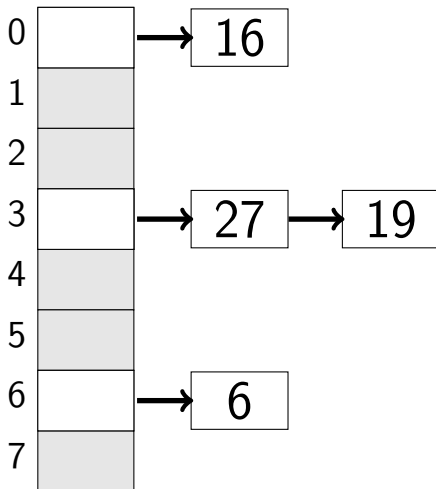
Tratamento de colisões por encadeamento

Função hash $h(\mathbf{6}) = \mathbf{6} \bmod 8 = \mathbf{6}$



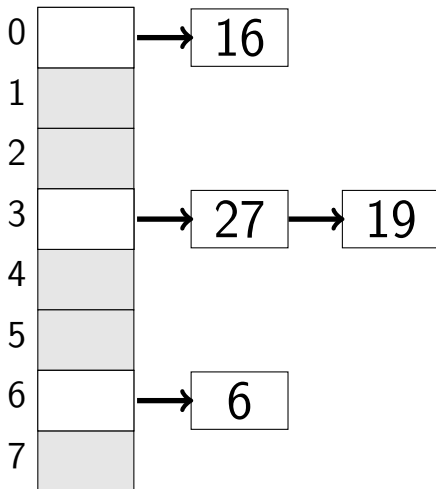
Tratamento de colisões por encadeamento

Função hash $h(6) = 6 \bmod 8 = 6$



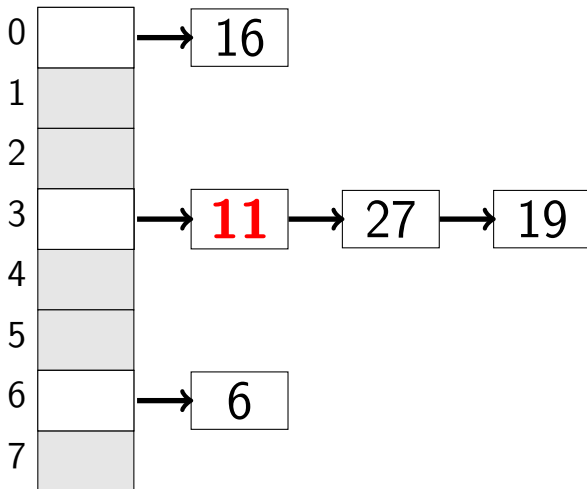
Tratamento de colisões por encadeamento

Função hash $h(\mathbf{11}) = \mathbf{11} \bmod 8 = \mathbf{3}$



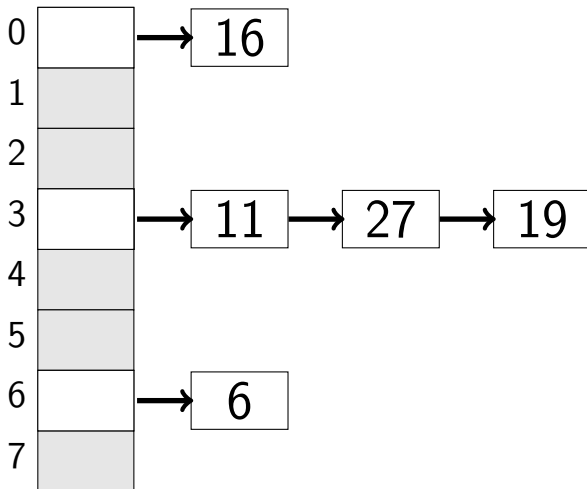
Tratamento de colisões por encadeamento

Função hash $h(\mathbf{11}) = \mathbf{11} \bmod 8 = \mathbf{3}$



Tratamento de colisões por encadeamento

Função hash $h(11) = 11 \bmod 8 = 3$



Tratamento de colisões por encadeamento

Em uma tabela hash com tratamento de colisões por encadeamento, uma busca depende do comprimento da lista encadeada. No pior caso todos os elementos são mapeados para a mesma posição e a busca tem **complexidade** $\mathcal{O}(n)$ mais o cálculo da função hash h .

Tratamento de colisões por encadeamento

Em uma busca **sem sucesso** por uma chave k examina-se toda a lista $H[k]$. Sob a hipótese de hash uniforme simples o tempo esperado dessa busca é $\Theta(1 + \alpha)$, tal que $\alpha = N/M$ e supondo que calcular $h(k)$ custa $\mathcal{O}(1)$.

Se o número de posições é proporcional ao números de elementos, ou $N = \mathcal{O}(M)$, o fator de carga é $\alpha = \mathcal{O}(1)$ e a busca tem tempo constante no caso médio.

Tratamento de colisões por encadeamento

Uma possível estrutura para um hash com tratamento de colisões por encadeamento é:

```
typedef struct node {  
    int k;                /*Chave de pesquisa.*/  
    struct node *next; /*Encadeamento.*/  
} Node;  
  
typedef struct hash {  
    int M; /*Número de entradas no hash. */  
    int N; /*Número de chaves armazenadas.*/  
    Node **list; /*Vetor de listas.*/  
} Hash;
```

Tratamento de colisões por encadeamento

A função criar inicializa as listas como NIL.

```
Hash* create_hash (int M) {  
    Hash *H = (Hash *)malloc(sizeof(Hash));  
    H->M = M;  
    H->N = 0;  
    H->list = (Node **)malloc(M*sizeof(Node*));  
    int h;  
    for (h = 0; h < H->M; h++) {  
        H->list[h] = NULL;  
    }  
    return H;  
}
```

Tratamento de colisões por encadeamento

A operação de inserção tem complexidade $\mathcal{O}(1)$ no pior caso. Por exemplo, inserção sempre no início da lista.

```
void insert_chained (Hash *H, int k) {  
    int h = hash_function (k, H->M);  
    Node *n = (Node *)malloc(sizeof(Node));  
    n->k = k;  
    n->next = H->list[h];  
    H->list[h] = n; /*n é a nova cabeça!*/  
    H->N++; /*incremento do num. de chaves!*/  
}
```

Sumário

1 Introdução

2 Funções Hash

- Método da divisão
- Método de multiplicação

3 Tratamento de colisões

- Encadeamento
- Endereçamento aberto
 - End. aberto: Sondagem linear
 - End. aberto: Sondagem quadrática
 - End. aberto: Hash duplo

4 Aplicações

Resolução de colisões por endereçamento aberto

No **endereçamento aberto**, todos os elementos estão armazenados na própria tabela hash. Não existe encadeamento e nenhum elemento é armazenado fora da tabela. Em uma colisão examinamos sucessivamente, ou **sondamos**, a tabela hash até encontrar uma posição vazia. Note que a tabela hash com esta técnica pode ficar “**cheia**”. Naturalmente o fator de carga α nunca excede 1.

Resolução de colisões por endereçamento aberto

Em uma operação de inserção, uma sequência de posições do hash é examinada até que uma posição livre seja encontrada.

INSERIR-HASH (H, k)

$i \leftarrow 0;$

do

$j \leftarrow h(k, i);$

if $H[j] = \text{NIL}$ **then**

$H[j] \leftarrow k;$

return $j;$

$i \leftarrow i + 1;$

while ($i \neq M$);

Resolução de colisões por endereçamento aberto

A função hash h nesse caso recebe um parâmetro inteiro i que indica o número da tentativa.

INSERIR-HASH (H, k)

$i \leftarrow 0;$

do

$j \leftarrow h(k, i);$

if $H[j] = \text{NIL}$ **then**

$H[j] \leftarrow k;$

return $j;$

$i \leftarrow i + 1;$

while ($i \neq M$);

Resolução de colisões por endereçamento aberto

A operação de busca percorre a mesma sequência examinada pela operação de inserção ao inserir **k**.

BUSCAR-HASH (**H**, **k**)

i \leftarrow 0;

do

j \leftarrow *h* (**k**, **i**);

if *H*[**j**] = **k** **then**

return j;

i \leftarrow **i** + 1;

while (**i** \neq *M*) **ou** (*H*[**j**] = NIL);

Sumário

1 Introdução

2 Funções Hash

- Método da divisão
- Método de multiplicação

3 Tratamento de colisões

- Encadeamento
- Endereçamento aberto
- **End. aberto: Sondagem linear**
- End. aberto: Sondagem quadrática
- End. aberto: Hash duplo

4 Aplicações

Resolução de colisões por sondagem linear

A função hash h para endereçamento aberto com **sondagem linear** é definida como

Sondagem linear

$$h(\mathbf{k}, \mathbf{i}) = (h'(\mathbf{k}) + \mathbf{i}) \bmod M$$

tal que $\mathbf{i} = 0, 1, 2, \dots, M - 1$ e h' é uma função hash auxiliar, por exemplo $h'(\mathbf{k}) = \mathbf{k} \bmod M$.

O deslocamento é sequencial (i), levando à M sequências de sondagens distintas.

Resolução de colisões por sondagem linear

Hash $h(k) = k \bmod 7$

0	
1	
2	
3	
4	
5	
6	

Resolução de colisões por sondagem linear

$$\text{Hash } h(\mathbf{76}) = \mathbf{76} \bmod 7 = 6$$

0	
1	
2	
3	
4	
5	
6	76

Resolução de colisões por sondagem linear

$$\text{Hash } h(76) = 76 \bmod 7 = 6$$

0	
1	
2	
3	
4	
5	
6	76

Resolução de colisões por sondagem linear

Hash $h(40) = 40 \bmod 7 = 5$

0	
1	
2	
3	
4	
5	40
6	76

Resolução de colisões por sondagem linear

$$\text{Hash } h(40) = 40 \bmod 7 = 5$$

0	
1	
2	
3	
4	
5	40
6	76

Resolução de colisões por sondagem linear

Hash $h(47) = 47 \bmod 7 = 5$


Colisão!

0	
1	
2	
3	
4	
5	40
6	76

Resolução de colisões por sondagem linear

$$\text{Hash } h(\textcolor{red}{47}) = (\textcolor{red}{47} \bmod 7 + 1) \bmod 7 = 6$$

0	
1	
2	
3	
4	
5	40
6	76

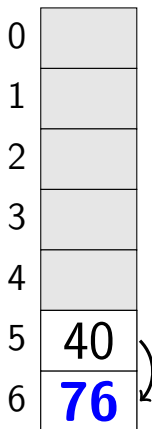


Resolução de colisões por sondagem linear

Hash $h(47) = (47 \bmod 7 + 1) \bmod 7 = 6$

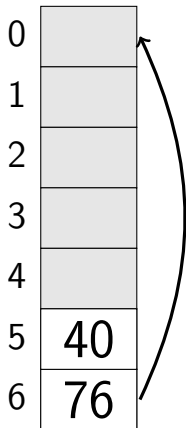
Colisão!

0	
1	
2	
3	
4	
5	40
6	76



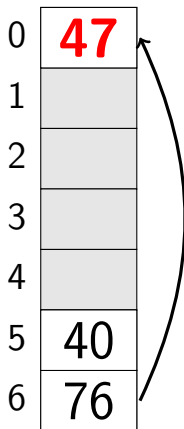
Resolução de colisões por sondagem linear

$$\text{Hash } h(\textcolor{red}{47}) = (\textcolor{red}{47} \bmod 7 + 2) \bmod 7 = 0$$



Resolução de colisões por sondagem linear

$$\text{Hash } h(47) = (47 \bmod 7 + 2) \bmod 7 = 0$$



0	47
1	
2	
3	
4	
5	40
6	76

Resolução de colisões por sondagem linear

$$\text{Hash } h(47) = (47 \bmod 7 + 2) \bmod 7 = 0$$

0	47
1	
2	
3	
4	
5	40
6	76

Resolução de colisões por sondagem linear

$$\text{Hash } h(32) = 32 \bmod 7 = 4$$

0	47
1	
2	
3	
4	32
5	40
6	76

Resolução de colisões por sondagem linear

$$\text{Hash } h(32) = 32 \bmod 7 = 4$$

0	47
1	
2	
3	
4	32
5	40
6	76

Resolução de colisões por sondagem linear

Hash $h(\mathbf{18}) = \mathbf{18} \bmod 7 = 4$


Colisão!

0	47
1	
2	
3	
4	32
5	40
6	76

Resolução de colisões por sondagem linear

$$\text{Hash } h(\mathbf{18}) = (\mathbf{18} \bmod 7 + 1) \bmod 7 = 5$$

0	47
1	
2	
3	
4	32
5	40
6	76




Resolução de colisões por sondagem linear

Hash $h(\mathbf{18}) = (\mathbf{18} \bmod 7 + 1) \bmod 7 = 5$

Colisão!


0	47
1	
2	
3	
4	32
5	40
6	76



Resolução de colisões por sondagem linear

$$\text{Hash } h(\mathbf{18}) = (\mathbf{18} \bmod 7 + 2) \bmod 7 = 6$$

0	47
1	
2	
3	
4	32
5	40
6	76




Resolução de colisões por sondagem linear

Hash $h(\mathbf{18}) = (\mathbf{18} \bmod 7 + 2) \bmod 7 = 6$

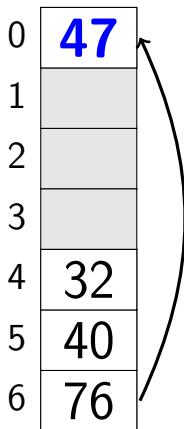
Colisão!

0	47
1	
2	
3	
4	32
5	40
6	76



Resolução de colisões por sondagem linear

$$\text{Hash } h(\mathbf{18}) = (\mathbf{18} \bmod 7 + 3) \bmod 7 = 0$$

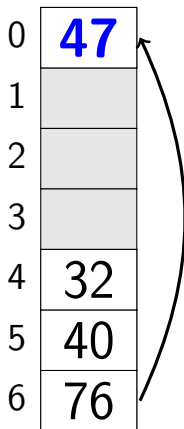


0	47
1	
2	
3	
4	32
5	40
6	76

Resolução de colisões por sondagem linear

Hash $h(\mathbf{18}) = (\mathbf{18} \bmod 7 + 3) \bmod 7 = 0$

Colisão!



0	47
1	
2	
3	
4	32
5	40
6	76

Resolução de colisões por sondagem linear

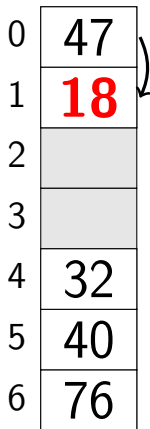
$$\text{Hash } h(\mathbf{18}) = (\mathbf{18} \bmod 7 + 4) \bmod 7 = 1$$

0	47
1	
2	
3	
4	32
5	40
6	76

Resolução de colisões por sondagem linear

$$\text{Hash } h(\mathbf{18}) = (\mathbf{18} \bmod 7 + 4) \bmod 7 = 1$$

0	47
1	18
2	
3	
4	32
5	40
6	76



Resolução de colisões por sondagem linear

A técnica de **sondagem linear** tem a desvantagem de sofrer com um problema conhecido como **agrupamento primário**. Isto é, são construídas sequências longas de posições ocupadas, aumentando o tempo de busca.

Sumário

1 Introdução

2 Funções Hash

- Método da divisão
- Método de multiplicação

3 Tratamento de colisões

- Encadeamento
- Endereçamento aberto
- End. aberto: Sondagem linear
- End. aberto: Sondagem quadrática
- End. aberto: Hash duplo

4 Aplicações

Resolução de colisões por sondagem quadrática

A função hash h para endereçamento aberto com **sondagem quadrática** é definida como

Sondagem quadrática

$$h(\mathbf{k}, i) = (h'(\mathbf{k}) + c_1 i + c_2 i^2) \bmod M$$

tal que c_1 e c_2 são duas constantes não nulas, $i = 0, 1, 2, \dots, M-1$ e h' é uma função hash auxiliar, por exemplo $h'(\mathbf{k}) = \mathbf{k} \bmod M$. Deslocamento depende quadraticamente de i .

Resolução de colisões por sondagem quadrática

Hash $h(k, i) = (h'(k) + c_1i + c_2i^2) \bmod 7$; com $c_1 = 1$ e $c_2 = 2$

0	
1	
2	
3	
4	
5	
6	

Resolução de colisões por sondagem quadrática

$$\text{Hash } h(\mathbf{76}) = (\mathbf{76} \bmod 7 + 0 + 0) \bmod 7 = 6$$

0	
1	
2	
3	
4	
5	
6	76

Resolução de colisões por sondagem quadrática

$$\text{Hash } h(\mathbf{76}) = (\mathbf{76} \bmod 7 + 0 + 0) \bmod 7 = 6$$

0	
1	
2	
3	
4	
5	
6	76

Resolução de colisões por sondagem quadrática

$$\text{Hash } h(\mathbf{40}) = (\mathbf{40} \bmod 7 + 0 + 0) \bmod 7 = 5$$

0	
1	
2	
3	
4	
5	40
6	76

Resolução de colisões por sondagem quadrática

$$\text{Hash } h(\mathbf{40}) = (\mathbf{40} \bmod 7 + 0 + 0) \bmod 7 = 5$$

0	
1	
2	
3	
4	
5	40
6	76

Resolução de colisões por sondagem quadrática

Hash $h(47) = (47 \bmod 7 + 0 + 0) \bmod 7 = 5$

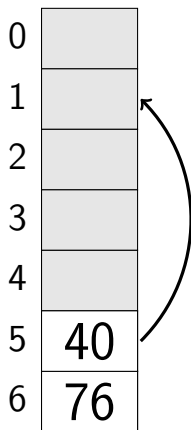
Colisão!

0	
1	
2	
3	
4	
5	40
6	76

Resolução de colisões por sondagem quadrática

$$\text{Hash } h(47) = (47 \bmod 7 + (1 * 1) + (2 * 1^2)) \bmod 7 = 1$$


0	
1	
2	
3	
4	
5	40
6	76



Resolução de colisões por sondagem quadrática

$$\text{Hash } h(47) = (47 \bmod 7 + (1 * 1) + (2 * 1^2)) \bmod 7 = 1$$

0	
1	47
2	
3	
4	
5	40
6	76



Resolução de colisões por sondagem quadrática

$$\text{Hash } h(\mathbf{47}) = (\mathbf{47} \bmod 7 + (1 * 1) + (2 * 1^2)) \bmod 7 = 1$$

0	
1	47
2	
3	
4	
5	40
6	76

Resolução de colisões por sondagem quadrática

$$\text{Hash } h(\mathbf{32}) = (\mathbf{32} \bmod 7 + 0 + 0) \bmod 7 = 4$$

0	
1	47
2	
3	
4	32
5	40
6	76

Resolução de colisões por sondagem quadrática

$$\text{Hash } h(32) = (32 \bmod 7 + 0 + 0) \bmod 7 = 4$$

0	
1	47
2	
3	
4	32
5	40
6	76

Resolução de colisões por sondagem quadrática

Hash $h(\mathbf{18}) = (\mathbf{18} \bmod 7 + 0 + 0) \bmod 7 = 4$


Colisão!

0	
1	47
2	
3	
4	32
5	40
6	76

Resolução de colisões por sondagem quadrática

$$\text{Hash } h(\mathbf{18}) = (\mathbf{18} \bmod 7 + (1 * 1) + (2 * 1^2)) \bmod 7 = 0$$

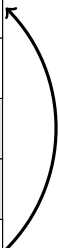
0	
1	47
2	
3	
4	32
5	40
6	76



Resolução de colisões por sondagem quadrática

$$\text{Hash } h(\mathbf{18}) = (\mathbf{18} \bmod 7 + (1 * 1) + (2 * 1^2)) \bmod 7 = 0$$

0	18
1	47
2	
3	
4	32
5	40
6	76



Resolução de colisões por sondagem quadrática

A técnica de **sondagem quadrática** sofre com o problema de **agrupamento secundário**: se duas chaves k_1 e k_2 têm a mesma posição inicial, então a sequência de sondagem é a mesma.

É melhor do que a sondagem linear, mas a sequência da sondagem também depende da sondagem inicial. Logo, existem apenas M sequências de sondagens distintas.

Sumário

1 Introdução

2 Funções Hash

- Método da divisão
- Método de multiplicação

3 Tratamento de colisões

- Encadeamento
- Endereçamento aberto
- End. aberto: Sondagem linear
- End. aberto: Sondagem quadrática
- End. aberto: Hash duplo

4 Aplicações

Resolução de colisões por hash duplo

A função hash h para endereçamento aberto com **hash duplo** é definida como

Hash duplo

$$h(\mathbf{k}, i) = (h_1(\mathbf{k}) + i h_2(\mathbf{k})) \bmod M$$

h_1 e h_2 são funções hash auxiliares. $h_1(k)$ é a posição inicial sondada; cada sondagem posterior é deslocada por $h_2(k)$ em relação à posição anterior, módulo de M .

Resolução de colisões por hash duplo

Hash duplo

$$h(\mathbf{k}, i) = (h_1(\mathbf{k}) + i h_2(\mathbf{k})) \bmod M$$

Ex: $h_1(\mathbf{k}) = \mathbf{k} \bmod M$

Pode-se considerar um M primo e projetar h_2 de modo que ele sempre retorne um inteiro positivo menor do que M :

$h_2(\mathbf{k}) = 1 + (\mathbf{k} \bmod M')$, e $M' < M$.

Resolução de colisões por hash duplo

$$\text{Hash } h(k, i) = (k \bmod 7 + i * (1 + k \bmod 5)) \bmod 7$$

0	
1	
2	
3	
4	
5	
6	

Resolução de colisões por hash duplo

$$\text{Hash } h(\mathbf{76}) = (\mathbf{76} \bmod 7 + 0 * (1 + \mathbf{76} \bmod 5)) \bmod 7 = 6$$

0	
1	
2	
3	
4	
5	
6	76

Resolução de colisões por hash duplo

$$\text{Hash } h(76) = (76 \bmod 7 + 0 * (1 + 76 \bmod 5)) \bmod 7 = 6$$

0	
1	
2	
3	
4	
5	
6	76

Resolução de colisões por hash duplo

$$\text{Hash } h(\mathbf{40}) = (\mathbf{40} \bmod 7 + 0 * (1 + \mathbf{40} \bmod 5)) \bmod 7 = 5$$

0	
1	
2	
3	
4	
5	40
6	76

Resolução de colisões por hash duplo

$$\text{Hash } h(40) = (40 \bmod 7 + 0 * (1 + 40 \bmod 5)) \bmod 7 = 5$$

0	
1	
2	
3	
4	
5	40
6	76

Resolução de colisões por hash duplo

$$\text{Hash } h(47) = (47 \bmod 7 + 0 * (1 + 47 \bmod 5)) \bmod 7 = 5$$


Colisão!

0	
1	
2	
3	
4	
5	40
6	76

Resolução de colisões por hash duplo

$$\text{Hash } h(47) = (47 \bmod 7 + 1 * (1 + 47 \bmod 5)) \bmod 7 = 1$$


0	
1	
2	
3	
4	
5	40
6	76



Resolução de colisões por hash duplo

$$\text{Hash } h(47) = (47 \bmod 7 + 1 * (1 + 47 \bmod 5)) \bmod 7 = 1$$


0	
1	47
2	
3	
4	
5	40
6	76



Resolução de colisões por hash duplo

$$\text{Hash } h(47) = (47 \bmod 7 + 1 * (1 + 47 \bmod 5)) \bmod 7 = 1$$

0	
1	47
2	
3	
4	
5	40
6	76



Resolução de colisões por hash duplo

$$\text{Hash } h(\mathbf{32}) = (\mathbf{32} \bmod 7 + 0 * (1 + \mathbf{32} \bmod 5)) \bmod 7 = 4$$

0	
1	47
2	
3	
4	32
5	40
6	76

Resolução de colisões por hash duplo

$$\text{Hash } h(32) = (32 \bmod 7 + 0 * (1 + 32 \bmod 5)) \bmod 7 = 4$$

0	
1	47
2	
3	
4	32
5	40
6	76

Resolução de colisões por hash duplo

$$\text{Hash } h(\mathbf{18}) = (\mathbf{18} \bmod 7 + 0 * (1 + \mathbf{18} \bmod 5)) \bmod 7 = 4$$

Colisão!

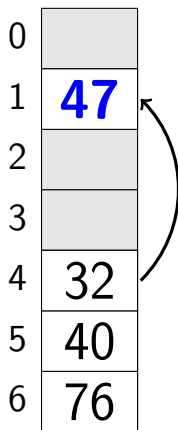
0	
1	47
2	
3	
4	32
5	40
6	76

Resolução de colisões por hash duplo

$$\text{Hash } h(\mathbf{18}) = (\mathbf{18} \bmod 7 + 1 * (1 + \mathbf{18} \bmod 5)) \bmod 7 = 1$$

Colisão!

0	
1	47
2	
3	
4	32
5	40
6	76




Resolução de colisões por hash duplo

$$\text{Hash } h(\mathbf{18}) = (\mathbf{18} \bmod 7 + 2 * (1 + \mathbf{18} \bmod 5)) \bmod 7 = 5$$

Colisão!


0	
1	47
2	
3	
4	32
5	40
6	76



Resolução de colisões por hash duplo

$$\text{Hash } h(\mathbf{18}) = (\mathbf{18} \bmod 7 + 3 * (1 + \mathbf{18} \bmod 5)) \bmod 7 = 2$$


0	
1	47
2	
3	
4	32
5	40
6	76



Resolução de colisões por hash duplo

$$\text{Hash } h(\mathbf{18}) = (\mathbf{18} \bmod 7 + 3 * (1 + \mathbf{18} \bmod 5)) \bmod 7 = 2$$

0	
1	47
2	18
3	
4	32
5	40
6	76



Resolução de colisões por hash duplo

No hash duplo, cada par $(h_1(k), h_2(k))$ gera uma sequência de sondagem distinta, levando à $\Theta(m^2)$ sequências. Essa abordagem é um aperfeiçoamento em relação à sondagem linear e quadrática, que apresentam $\Theta(m)$ sequências de sondagens.

Resolução de colisões por hash duplo

A inserção de um elemento em uma tabela hash de endereço aberto com fator de carga α exige no máximo $1/(1 - \alpha)$ sondagens em média, supondo-se o hash uniforme. Por exemplo, se a tabela tiver metade dos elementos, o número de sondagens em uma pesquisa malsucedida será $1/(1 - 0.5) = 2$. Com 90% das posições ocupadas é necessário $1/(1 - 0.9) = 10$ sondagens.

Sumário

1 Introdução

2 Funções Hash

- Método da divisão
- Método de multiplicação

3 Tratamento de colisões

- Encadeamento
- Endereçamento aberto
- End. aberto: Sondagem linear
- End. aberto: Sondagem quadrática
- End. aberto: Hash duplo

4 Aplicações

Aplicações

Consulta DNS, roteamento, manipulação de strings, projeto de compiladores, autenticação de senha, criptografia (SHA-1, MD5), ...

https://shraddhapandhe.files.wordpress.com/2012/09/project_report.pdf

<http://linux-ip.net/html/routing-selection.html#tb-routing-selection-adv>

https://en.wikipedia.org/wiki/Rabin%E2%80%93Karp_algorithm

http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0101-74382015000200423

<http://www.cs.uregina.ca/Links/class-info/210/Hash/#IMPLEMENTATION>

<http://algs4.cs.princeton.edu/34hash/>