

Tablas de hash basadas en direccionamiento abierto

Guillermo Palma

Universidad Simón Bolívar
Departamento de Computación y T.I.

CI-2612: Algoritmos y Estructuras II



Plan

- 1 Direccionamiento abierto
- 2 Funciones de hash para el Direccionamiento abierto



Características de las tablas de hash basadas en direccionamiento abierto

- La idea es guardar todas las claves (o el par clave, valor) en la misma tabla
- Se tiene que el tamaño de la tabla es mayor que el numero de elementos a agregar, esto es $m > n$
- No hay necesidad de usar listas enlazadas
- La inserción consiste en recorrer la tabla en busca de una casilla disponible
- Para la búsqueda se recorre la tabla de la misma manera que la inserción para encontrar un elemento
- Borrar un elemento es complicado, porque hay que hacer una marca especial en la casilla en donde se encontraba el elemento eliminado
- El tiempo de la búsqueda de la inserción depende de la longitud de la secuencia de elementos probados hasta alcanzar una casilla libre (probe sequence)



Ejemplo de Direccionamiento abierto

0	
1	79
2	
3	
4	69
5	98
6	
7	72
8	
9	14
10	
11	50
12	

Figura: Ejemplo de la inserción del elemento 14, en una tabla conteniendo a los elementos 79, 69, 98, 72 y 50. En la inserción la probe sequence es $\{1, 5, 7\}$



Función de hash del Direccionamiento abierto

- La función de hash tiene dos argumentos, primero la clave k del objeto a insertar y segundo el número de prueba (probe number) i

$$h(k, i), \quad \text{donde } i = 0, 1, \dots, m - 1 \quad (1)$$

- Secuencia de prueba (Probe sequence)

$$\langle h(k, 0), h(k, 1), \dots, h(k, m - 1) \rangle \quad (2)$$

- La secuencia de prueba deber ser una permutación de la secuencia de las m casillas, es decir, permutación de $\{0, 1, \dots, m - 1\}$
- Existen $m!$ secuencias de prueba
- Una buena función de hash debe ser capaz de producir todas las $m!$ posibles secuencias de prueba



Implementación de la inserción en Direccionamiento abierto

```

HASH-INSERT( $T, k$ )
1   $i = 0$ 
2  repeat
3       $j = h(k, i)$ 
4      if  $T[j] == \text{NIL}$ 
5           $T[j] = k$ 
6          return  $j$ 
7      else  $i = i + 1$ 
8  until  $i == m$ 
9  error "hash table overflow"

```



Implementación de la búsqueda en Direccionamiento abierto

```
HASH-SEARCH( $T, k$ )
1   $i = 0$ 
2  repeat
3       $j = h(k, i)$ 
4      if  $T[j] == k$ 
5          return  $j$ 
6       $i = i + 1$ 
7  until  $T[j] == \text{NIL}$  or  $i == m$ 
8  return NIL
```



Principales funciones de hash Direccionamiento abierto

- Linear probing
- Quadratic probing
- Double hashing

Ninguna de estas funciones es capaz de generar más de m^2 secuencias de prueba



Linear probing

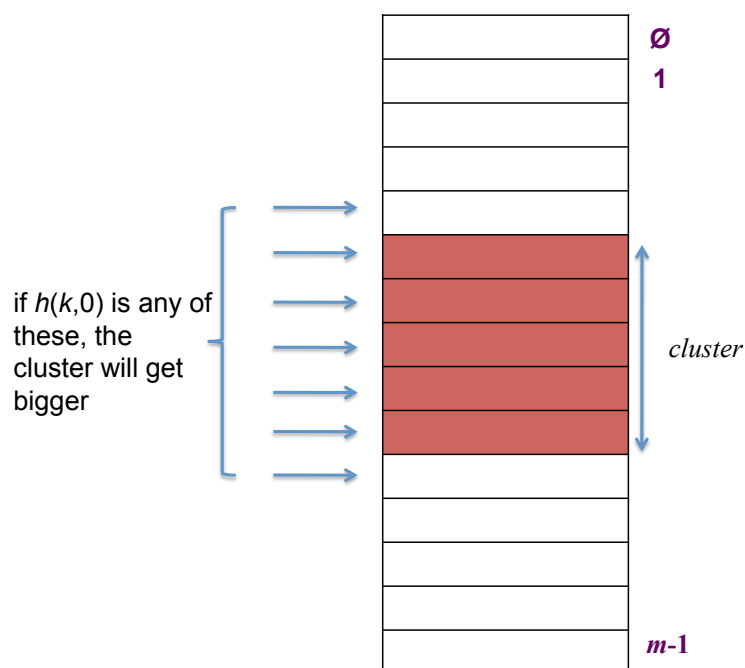
- Consiste en que cuando hay una colisión, se prueba la próxima posición en la tabla
- Viene dada por la función:

$$h(k, i) = (h_1(k) + i) \text{ mód } m, \quad \text{donde } i = 0, 1, \dots, m - 1$$

- La secuencia de prueba es $\{h_1(k), h_1(k) + 1, h_1(k) + 2, \dots\}$
- Solo puede generar m secuencias de prueba



Ejemplo de *clustering* durante la inserción



Quadratic probing

- Se usa una función de hash auxiliar h_1 y dos constantes c_1 y c_2
- Viene dada por la función:

$$h(k, i) = (h_1(k) + c_1 i + c_2 i^2) \text{ mód } m, \quad \text{donde } i = 0, 1, \dots, m-1$$

- Solo puede generar m secuencias de prueba



Double Hashing

- Se usa una primera función de hash h_1 para determinar la primera casilla
- Se usa una segunda función de hash h_2 para determinar el incremento de la secuencia de prueba
- Viene dada por la función:


$$h(k, i) = (h_1(k) + h_2(k)i) \text{ mód } m, \quad \text{donde } i = 0, 1, \dots, m-1$$

- Primer elemento de la secuencia de prueba es $h_1(k)$
- Tiene la ventaja que evita la aglomeración de elementos en una parte de la tabla
- Hace que la operación de eliminar un elemento de la tabla se más difícil
- Es la función más usada de las tres
- Puede generar un máximo de m^2 secuencias de prueba



Análisis del caso en que la clave no se encuentre en la tabla

- El factor de carga a (load factor) es el número de claves en la tabla dividido entre el número de casillas de la tabla
- La probabilidad de que la casilla este ocupada a
- La probabilidad de que la casilla este vacía $1 - a$
- Probabilidad de finalizar las pruebas de búsqueda (Probe) en dos pasos $a(1 - a)$
- Probabilidad de finalizar las pruebas de búsqueda (Probe) en k pasos $a^{k-1}(1 - a)$
- El número de intentos promedio que se hacen al realizar una búsqueda de una clave que no está en la tabla es

$$E(\#intentos) = \sum_{k=1}^m ka^{k-1}(1 - a) = \sum_{k=1}^{\infty} ka^{k-1}(1 - a) = \frac{1}{1 - a}$$


Análisis del caso en que la clave se encuentre en la tabla

$$E(\#intentos) = \frac{1}{a} \ln\left(\frac{1}{1 - a}\right)$$

