Computación y Estructuras Discretas I

Andrés A. Aristizábal P.
aaaristizabal@icesi.edu.co

Departamento de Computación y Sistemas Inteligentes

2024-2

Implementación en Java

```java
package hashtables;

public interface IHashTable<K,V> {
  public void insert(K key, V value) throws Exception;
  public V search(K key);
  public void delete(K key);
}
```

Implementación en Java

```java
package hashtables;

public class HNode<K,V> {

  private V value;
  private K key;

  private HNode<K,V> next;
  private HNode<K,V> previous;


  public HNode(K key, V value) {
    this.value = value;
    this.key = key;
    next = null;
    previous = null;
  }
```

Implementación en Java

```java
package hashtables;

public class ChainHashTable<K,V> implements IHashTable<K,V> {

    public static final int ARR_SIZE = 127;
    private HNode<K,V>[] nodes;
```

Agenda del día

- Direccionamiento abierto
- Ejercicios

Direccionamiento abierto

¿Qué pasa en el direccionamiento abierto?

Direccionamiento abierto

¿Qué pasa en el direccionamiento abierto?

- Todos los elementos ocupan la tabla hash.
- Cada espacio de la tabla contiene un elemento o `NIL`.
- Cuando se busca un elemento, sistemáticamente se examinan los espacios de la tabla hasta que
  - Se encuentra el elemento deseado.
  - O nos aseguramos que el elemento no está en la tabla.
- No hay listas o elementos guardados fuera de la tabla (a diferencia del encadenamiento).
  - La tabla se puede llenar.
  - No se pueden hacer más inserciones.
  - El factor de carga α nunca puede exceder 1.

Direccionamiento abierto

¿Entonces que hacer sino se pueden guardar elementos fuera de la tabla?

Direccionamiento abierto

¿Entonces que hacer sino se pueden guardar elementos fuera de la tabla?

- Se pueden guardar las listas adentro de la tabla para hacer encadenamiento en los espacios no utilizados.
- Pero se perdería la ventaja de evitar el uso de punteros.
- En vez de seguir punteros, se computa la secuencia de espacios a examinar.
- La memoria extra liberada por no guardar punteros le provee a la tabla hash:
  - Más espacios por la misma cantidad de memoria.
  - Produciendo, potencialmente, menos colisiones y extracción más rápida.

Direccionamiento abierto

¿Cómo se realiza la inserción con el direccionamiento abierto?

Direccionamiento abierto

¿Cómo se realiza la inserción con el direccionamiento abierto?

- Sucesivamente examinamos (o exploramos) la tabla hash hasta que encontramos un espacio en el que poner la clave.
- No se fija un orden para la exploración de 0 a $m - 1$ (tiempo de búsqueda de $\Theta(n)$).
- La secuencia de exploración dependerá de la clave a insertar.

Direccionamiento abierto

¿Cómo se realiza la inserción con el direccionamiento abierto?

Direccionamiento abierto

¿Cómo se realiza la inserción con el direccionamiento abierto?

- Para determinar que espacios explorar, se extiende la función hash
  - Se incluye el número de exploración (empezando desde 0) como segundo valor de entrada la función.
  - $h : U \times \{0, 1, ..., m-1\} \rightarrow \{0, 1, ..., m-1\}$
  - Se requiere que para cada clave k, la secuencia de exploración
    $$\langle h(k, 0), h(k, 1), ..., h(k, m-1) \rangle$$
    sea una permutación de $\langle 0, 1, ..., m-1 \rangle$.
  - Para que cada posición de la tabla hash sea considerada eventualmente como espacio para la nueva clave, a medida que se llena la tabla.

Direccionamiento abierto

¿Qué hace el método de inserción?

Direccionamiento abierto

¿Qué hace el método de inserción?

- El método toma como entradas T y x.
- Retorna
  - El número del espacio donde ubica a x o
  - Un error que indica que la tabla estaba llena.

Direccionamiento abierto

¿Cómo será la implementación de la inserción?

Direccionamiento abierto

¿Cómo será la implementación de la inserción?

```
HASH_INSERT(T,x)
  i = 0
  repeat
    j = h(x.key,i)
    if T[j] == NIL
      T[j] = x
      return j
    else
      i = i + 1
  until i == m
  error "desbordamiento de tabla hash"
```

Direccionamiento abierto

¿Qué hace el método de búsqueda?

Direccionamiento abierto

¿Qué hace el método de búsqueda?

- Como el algoritmo para la búsqueda para la clave $x.key$ explora la misma secuencia de espacios que la que examina el algoritmo de inserción cuando se inserta x
  - La búsqueda puede terminar sin éxito si llega a un espacio vacío.
  - x debería haberse insertado ahí y no después.
  - Se asume que no se borran claves de la tabla hash.
- El método toma como entradas T y x.
- Retorna

Direccionamiento abierto

¿Qué hace el método de búsqueda?

- Como el algoritmo para la búsqueda para la clave $x.key$ explora la misma secuencia de espacios que la que examina el algoritmo de inserción cuando se inserta x
  - La búsqueda puede terminar sin éxito si llega a un espacio vacío.
  - x debería haberse insertado ahí y no después.
  - Se asume que no se borran claves de la tabla hash.
- El método toma como entradas T y x.
- Retorna
  - j si encuentra que el espacio j contiene x
  - NIL si x no se encuentra en la tabla.

Direccionamiento abierto

¿Cómo será la implementación de la búsqueda?

Direccionamiento abierto

¿Cómo será la implementación de la búsqueda?

```
HASH_SEARCH(T,x)
  i = 0
  repeat
    j = h(x.key,i)
    if T[j] == x
      return j
    i = i + 1
  until T[j] == NIL or i == m
  return NIL
```

Direccionamiento abierto

¿Qué pasa con el eliminar en una tabla hash con direccionamiento abierto?

Direccionamiento abierto

¿Qué pasa con el eliminar en una tabla hash con direccionamiento abierto?

- El eliminar en una tabla hash de este tipo es difícil.
- Cuando se borra un elemento x no se puede puede marcar el espacio como *NIL*
- Si se hace esto, tal vez no se encuentre el elemento x que se insertó y encontró ese espacio lleno.
- Se puede resolver el problema marcando ese espacio con un nuevo valor especial *DELETED*.

Direccionamiento abierto

- Se puede modificar el `HASH_INSERT` para que tome los espacios marcados con *DELETED* como vacíos.
- El `HASH_SEARCH` no se debería modificar, pues se salta esos espacios.
- Los tiempos del eliminar ya no dependen solamente del α.
- Encadenamiento es más utilizado para resolución de colisiones cuando se han de borrar claves.

Direccionamiento abierto

¿Qué se debe asumir en nuestro análisis de las operaciones?

Direccionamiento abierto

¿Qué se debe asumir en nuestro análisis de las operaciones?

- Hashing uniforme
  - Para una clave k es igualmente posible que su secuencia de exploración sea una de las $m!$ permutaciones de $\langle 0, 1, ..., m-1 \rangle$.

Direccionamiento abierto

¿Cómo se computan las secuencias de exploración para el direccionamiento abierto?

Direccionamiento abierto

¿Cómo se computan las secuencias de exploración para el direccionamiento abierto?

- Se utilizan ciertas técnicas
  - Exploración lineal
  - Exploración cuadrática
  - Hashing doble

Direccionamiento abierto

¿Cómo se computan las secuencias de exploración para el direccionamiento abierto?

- Se utilizan ciertas técnicas
  - Exploración lineal
  - Exploración cuadrática
  - Hashing doble
- Garantizan que $\langle h(k,0), h(k,1), ..., h(k,m-1)\rangle$ es una permutación de $\langle 0, 1, ..., m-1\rangle$ para cade clave k.
- No garantizan hashing uniforme.
- Ninguna de éstas puede generar más de m^2 secuencias diferentes de exploración (deberían ser $m!$)

Direccionamiento abierto

¿Cómo es la exploración lineal?

Direccionamiento abierto

¿Cómo es la exploración lineal?

- Se utiliza la función hash $h(k, i) = (h'(k) + i) \, mod \, m$
- Donde $h' : U \rightarrow \{0, 1, ..., m-1\}$ es una función hash auxiliar.
- $i = 0, 1, ..., m-1$
- Hay m secuencias distintas de exploración.

Direccionamiento abierto

¿Cómo es la exploración cuadrática?

Direccionamiento abierto

¿Cómo es la exploración cuadrática?

- Se utiliza la función hash $h(k, i) = (h'(k) + c_1 i + c_2 i^2) \, mod \, m$
- Donde $h' : U \rightarrow \{0, 1, ..., m-1\}$ es una función hash auxiliar.
- $i = 0, 1, ..., m-1$
- c_1 y c_2 son constantes positivas auxiliares.
- Hay m secuencias distintas de exploración.

Direccionamiento abierto

¿Cómo es el hashing doble?

Direccionamiento abierto

¿Cómo es el hashing doble?

- Ofrece uno de los mejores métodos disponibles para el direccionamiento abierto.
- Las permutaciones producidas por esta tienen muchas de las características de permutaciones elegidas aleatoriamente.
- Se utiliza la función hash $h(k, i) = (h_1(k) + ih_2(k)) \; mod \; m$
- Donde h_1 y h_2 son funciones hash auxiliares.

Implementación en Java

```java
package hashtables;

public interface IHashTable<K,V> {
  public void insert(K key, V value) throws Exception;
  public V search(K key);
  public void delete(K key);
}
```

Implementación en Java

```java
package hashtables;

public class HNode<K,V> {

  private V value;
  private K key;

  public HNode(K key, V value) {
    this.value = value;
    this.key = key;
  }
```

Implementación en Java

```java
package hashtables;

import java.util.ArrayList;

public class OpenHashTable<K,V> implements IHashTable<K,V> {

  private ArrayList<HNode<K,V>> table;
  private HNode<K,V> deleted;
```

Agenda del día

- Direccionamiento abierto
- Ejercicios

Ejercicios

- Completar la hoja de trabajo sobre direccionamiento directo y tablas hash con resolución de colisiones por encadenamiento.
- Completar la hoja de trabajo sobre tablas hash con resolución de colisiones por direccionamiento abierto.