Tema 10: Organización Hashing

Contenidos extraídos del libro:

Ficheros. Organizaciones clásicas para el almacenamiento del a información

Autores: *Irene Luque Ruiz, Juan Antonio Romero del Castillo y Miguel Ángel Gómez-Nieto*
Editorial: Servicio de Publicaciones de la Universidad de Córdoba, 1998.
ISBN 84-7801-468-3

10.1 Introducción

Inconvenientes de la organización indexada:

- La organización indexada vista en el tema anterior ofrece un buen desempeño en accesos directos.
- Pare encontrar un registro se leen ***h*** bloques de índice siendo ***h*** la altura del árbol:

$$h = \log_{\bar{F}_{B_I}+1} r$$

$$O\left(\log_{\bar{F}_{B_I}+1} r\right)$$

Esto se repite en operaciones de actualización, inserción, etc.

10.1 Introducción

Objetivos de la organización *hashing*:

- Conseguir accesos mediante una única operación:

$$O(1)$$

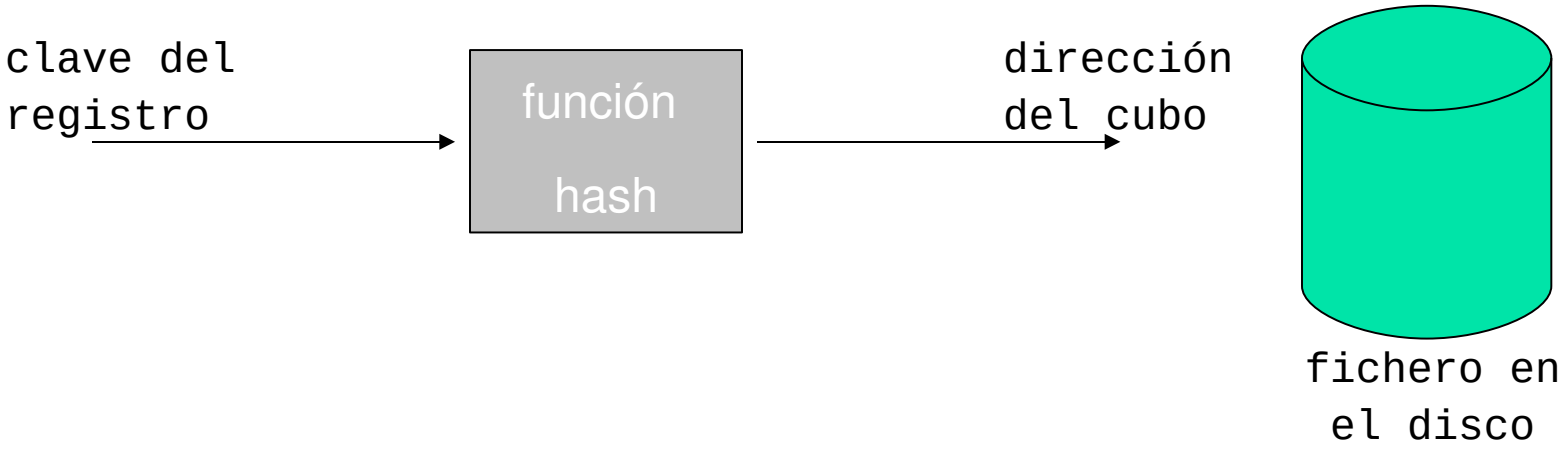
- La organización *hashing* data de finales de los 60, pero no es hasta mucho después cuando se utiliza en accesos rápidos a la información.
- Se utiliza tanto en memoria principal como en secundaria en organizaciones de ficheros.

También conocido como: dispersión, asociación; y en sus comienzos: *aleatorización*, *randomizing*

10.2 características de la organización hashing

Fundamento:

- Aplicar a la clave un algoritmo, denominado *función hash* o de dispersión.
- La *función hash* proporciona como salida una dirección en la que se almacenará el registro (denominada *cubo*).
- Idealmente:
  - O bien no existirán dos registros que se direccionen al mismo cubo.
  - O bien siempre habrá espacio suficiente en el cubo direccionado.



10.2 características de la organización hashing

Tipos:

Dependiendo de las características de la función *hash* y del número de cubos disponibles:

- **Hashing estático**: requiere que la función *hash* conozca a priori el número de cubos disponibles. Se divide en:
  a. *Hashing* cerrado: solo existe un único cubo. La función *hash* obtiene el desplazamiento interior. Se usa cuando se almacena en la memoria del ordenador.
  b. *Hasing* abierto: el número de cubos disponible es mayor que 1.
- **Hasing dinámico**: la función *hash* no requiere conocer el número de cubos disponibles. El número de cubos varía dinámicamente.

10.2 características de la organización hashing

Factores de influencia del método *hash*:

- El número y el tamaño de los cubos
- La densidad de empaquetamiento
- La función *hash*
- La existencia de colisiones o sobredireccionamiento

10.2 características de la organización hashing

Los cubos (o *home buckets*)

- Son unidades de espacio de direccionamiento a disposición de la función *hash*
- El cubo tiene una capacidad
- Colisión: cuando una función *hash* direcciona un cubo lleno
- Un cubo pequeño provoca muchas colisiones
- Un cubo grande hace que la recuperación de los registros dentro de él sea costosa

es necesario un compromiso

10.2 características de la organización hashing

Los cubos

- Número máximo de registros por cubo: $F_C = \lfloor \frac{C}{R} \rfloor$

El número de cubos necesarios: $c = \lceil \frac{r}{F_C} \rceil$

La densidad de empaquetamiento

A mayor densidad de empaquetamiento, mayor probabilidad de colisiones:

$$\frac{r}{Fc \times c}$$

r → número de registros a almacenar

$Fc \times c$ → espacio disponible

10.3 La función Hash

Es un algoritmo que realiza tres operaciones básicas:

1. Si la clave de los registros está definida en un dominio no numérico, transforma el dominio de la clave a numérico sin pérdida de información. Este proceso debe ser reversible.

2. A partir de la clave definida en el dominio numérico se obtiene un valor dentro del orden de magnitud del número de cubos reservados en el espacio de direccionamiento.

3. Mediante un proceso de traslación se convierte el número obtenido en el paso anterior, en un valor de cubo c_i , donde $0 \leq c_i < c$

10.3 La función Hash

Características:

- La función hash siempre dará como resultado un valor de cubo en el intervalo de cubos reservados [0, c-1].
- La bondad del algoritmo hash vendrá determinada por la capacidad de éste de distribuir lo más uniformemente posible el conjunto de los registros en los cubos disponibles para direccionar esta información.
- *Ejemplo:* Una mala función hash sería si en un fichero de personas cuya clave es *apellidos*, usaramos como función hash la primera letra de la clave.
- *Ejemplo:* Una función peor es otra que diera como resultado siempre el mismo cubo.

10.3 Algoritmos Hash

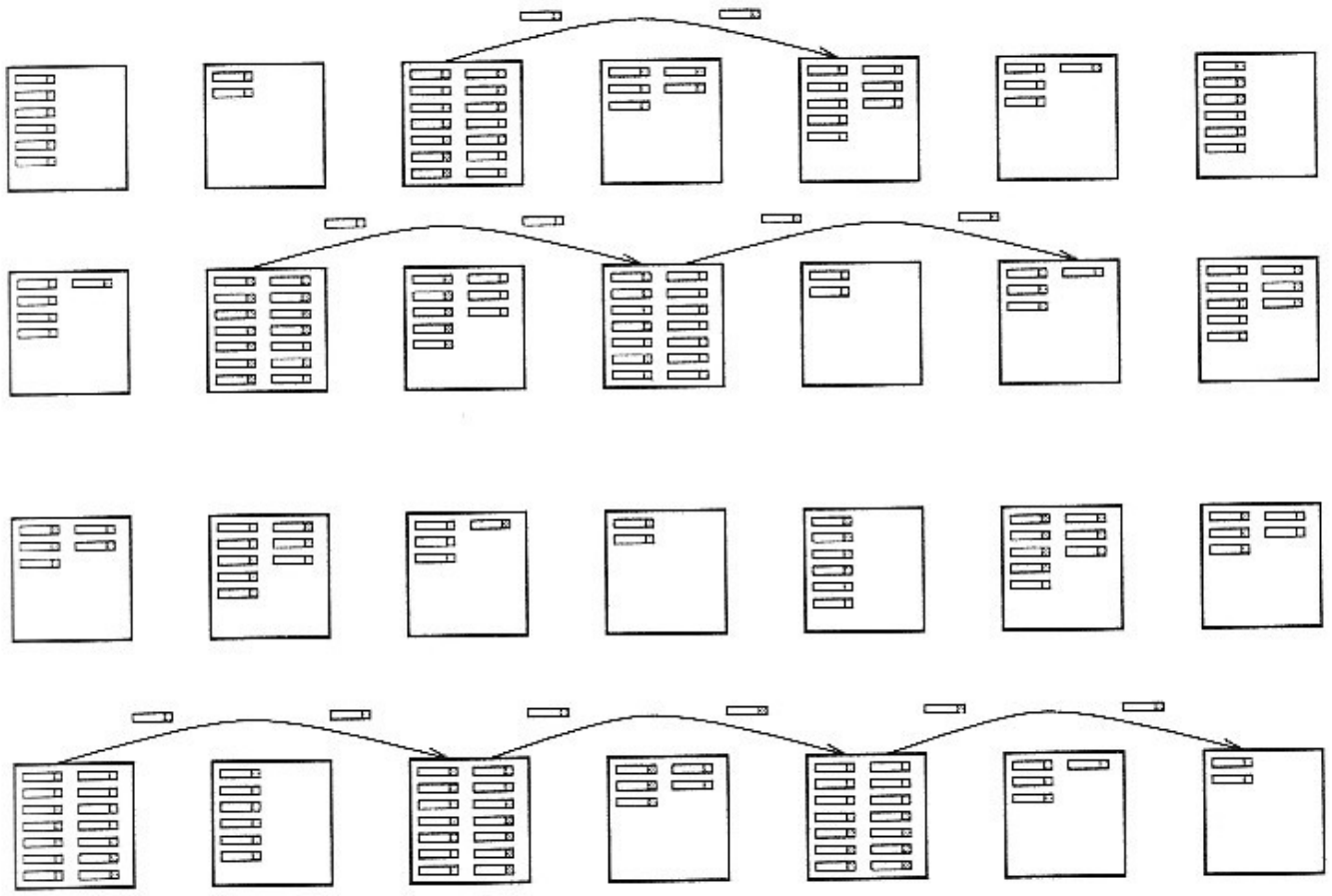
- Algoritmo del resto de la división.
- Algoritmo del centro de los cuadrados.
- Algoritmo del desplazamiento.
- Algoritmo del plegado.
- Algoritmo de la conversión de la raíz, etc.

10.4 Tratamiento de desbordes

- Las colisiones provocan desbordes de los cubos.
- Un desborde se produce cuando la función ***hash*** direcciona una clave de un registro a un cubo que se encuentra completamente ocupado.
- Es necesario buscar otra zona de almacenamiento para el registro: área de **derrama** o área de desborde.
- Disposición del área de derrama.
  1. El área de derrama es una zona independiente al área maestra
  2. El área de derrama se encuentra distribuida dentro del área maestra.

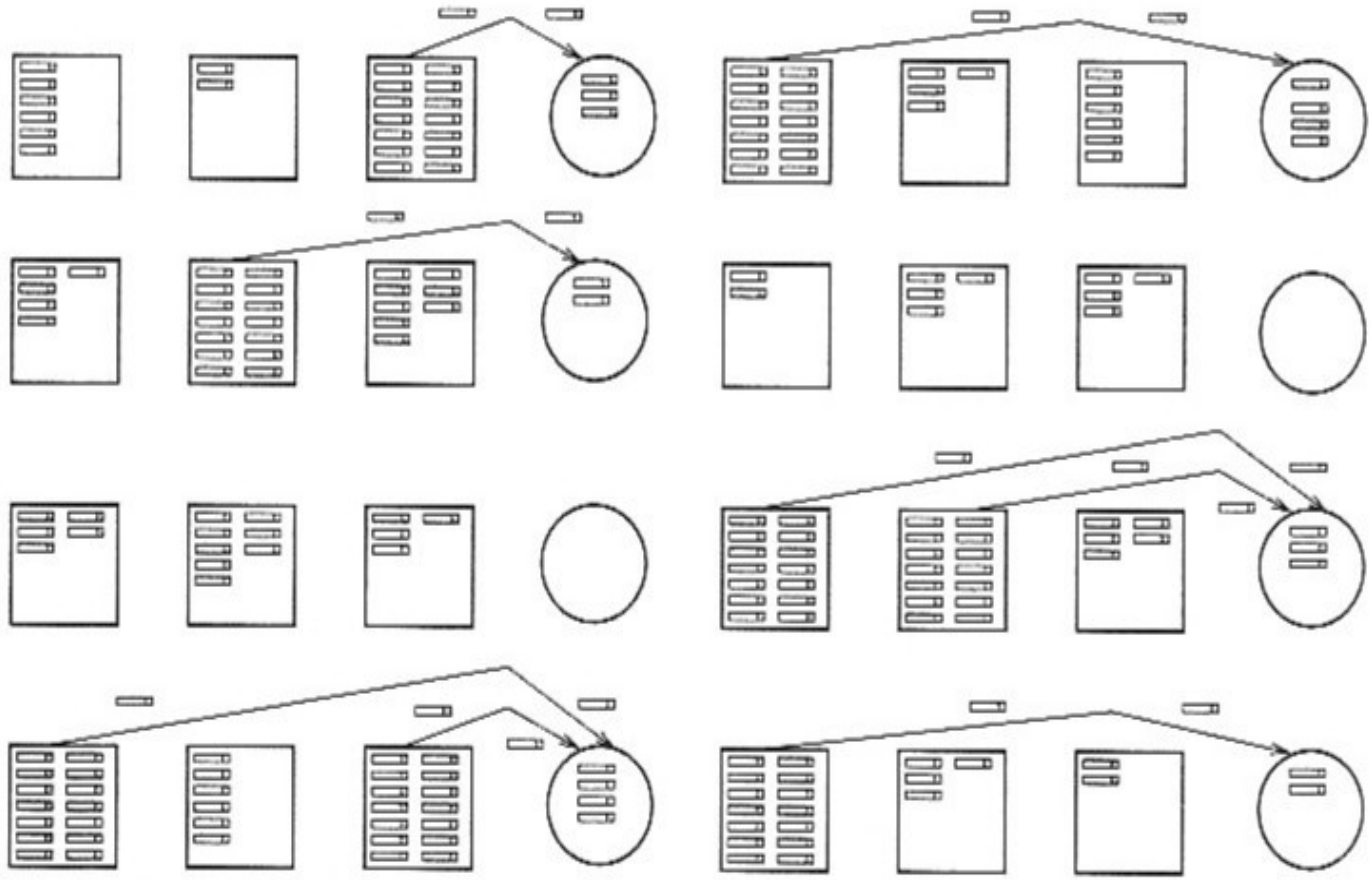
10.4 Tratamiento de desbordes

Desbordes en cubos maestros



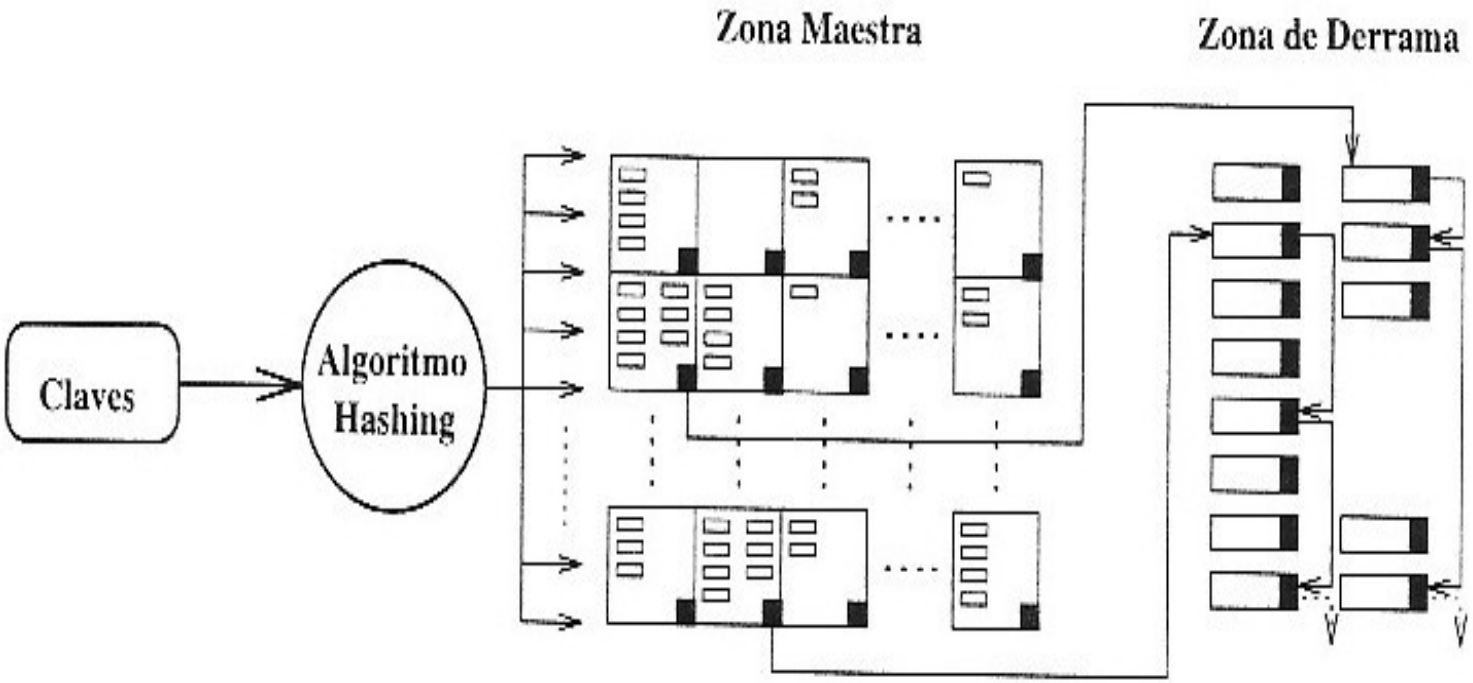
10.4 Tratamiento de desbordes

Desbordes en cubos de desbordes en el área maestra



10.4 Tratamiento de desbordes

Desbordes en zona de derrama independiente



Fin