Organización y manejo de archivos

www.inf.ucv.cl/ wpalma/oma

Dr. Wenceslao Palma
wenceslao.palma@ucv.cl

Organización de Archivos

- Secuencial desordenado
- Secuencia ordenado (ordenamiento externo)
- Basada en árboles
- Directa (hashing)

Organización de Archivos

Lógicamente, un archivo es una colección de registros estructurados, generalmente, en torno a una clave. Esta organización de datos considera la visión del programador de aplicaciones sobre el archivo, la cual es independiente del dispositivo; interesa más su uso que sus atributos físicos. Dicho programador de aplicaciones es la persona que visualiza un archivo como una colección de registros lógicos, aplicándoles las siguientes operaciones:

- Recorrer el archivo
- Agregar registros al archivo
- Eliminar registros del archivo
- Modificar valores del archivo
- Ordenar el archivo

Organización de Archivos

En la organización lógica de una archivo, se distinguen dos conceptos importantes: registro campo/atributo.

- **Registro**: corresponde a la definición de cada uno de los registros de datos que serán usados dentro de una aplicación, y cuya extensión determina el tipo de archivo que lo contiene, el cual puede ser:
  - Archivo de Largo Fijo: el archivo se compone de registros del mismo tipo.
  - Archivo de Largo Variable: el archivo se compone de registros del mismo tipo, pero al menos uno de los campos tiene un número de ocurrencias variable.
    El archivo se compone de registros del mismo tipo, pero el largo de al menos uno de sus campos es variable.
    El archivo se compone de registros del mismo tipo, pero uno de los campos es opcional en cuanto a su tipo.
    El archivo se compone de más de un tipo de registro.

- **Campo/Atributo**: ítem de dato que describe una característica de la entidad representada por un registro lógico. Este campo puede tener nombres como:
  - Clave Candidata: atributo(s) que no soporta(n) valores repetidos, y que por lo mismo, puede(n) identificar en forma única a un registro dentro del archivo.
  - Identificador o Clave Primaria:clave candidata escogida como clave de acceso principal al archivo, dado su constante uso en los accesos al mismo.
  - Clave Alternativa: clave candidata que no es escogida como identificador.
  - Clave Secundaria: atributo que permite valores repetidos.
  - Clave Inteligente: atributo cuyo contenido permite derivar información adicional de la entidad a la cual pertenece.

Los registros, compuestos de diversos atributos, se van a organizar entre sí, para formar lo que se entiende por un archivo. En general, las tres estructuras de archivos más comunes son la **secuencial** (ordenado/desordenado), la **directa** (hashing), y el esquema **jerárquico** (árbol).

Archivos secuenciales

Los registros son almacenados en secuencia, uno tras otro. Caracteristicas:

- Simple de usar.
- Conviene cuando se utilizan todos o la mayoría de los registros.
- El espacio de almacenamiento es mínimo, sólo se almacenan datos.
- El orden físico y lógico es el mismo.

Existen dos tipos de archivos secuenciales: desordenado y ordenado.

Secuencial desordenado

Los registros se almacenan uno tras otro según orden de llegada.

- Búsqueda: se realiza en forma lineal.
- Inserción: al final del archivo, lo que hace que la operación sea rápida.
- Eliminación: se puede realizar de dos formas; física y lógica. Si la eliminación es física, será necesario un corrimiento de los datos, lo cual perjudica el rendimiento. Por el contrario si la operación es lógica se utilizan marcas de borrado. Sin embargo, en ambos casos, es necesaria una reorganización de los registros.

Secuencial ordenado

Los registros se almacenan en forma ordenada de acuerdo al valor de un campo (en la mayoria de los casos clave) de ordenamiento. Favorece la bússqueda y generación de listados ordenados.

- Búsqueda: se realiza en forma lineal y binaria .
- Inserción: requiere un corrimiento de los registros para mantener el orden. Para evitar esto, existen varias técnicas:
  - Incluir a cada bloque de datos un puntero que direccione a una lista enlazada de bloques de overflow, donde se almacenan los registros en dicha situación.
  - Utilizar un archivo de overflow, el cual en cierta medida se transforma en un archivo de transacciones, para almacenar registros. Posteriormente se actualiza el archivo de datos mediante un ordenamiento y mezcla con el archivo de overflow.
- Eliminación: al igual que en el caso de los archivos secuenciales desordenados la eliminación puede ser física o lógica.

Ordenamiento externo

Los métodos de ordenamiento externo se aplican sobre archivos que no pueden ser almacenados completamente en memoria principal para su ordenamiento.

Ordenamiento externo

Se habla de ordenamiento externo cuando se utiliza un proceso de dos fases: una de ordenamiento y otra de mezcla. La fase de ordenamiento toma datos del archivo (particiones) los deja en memoria principal, los ordena y los retorna al disco. Los registros quedan ordenados en forma relativa a la particion en donde se encuentran. En la fase de mezcla se mezclan las particiones generando particiones mas grandes hasta llegar a una sola: el archivo ordenado.

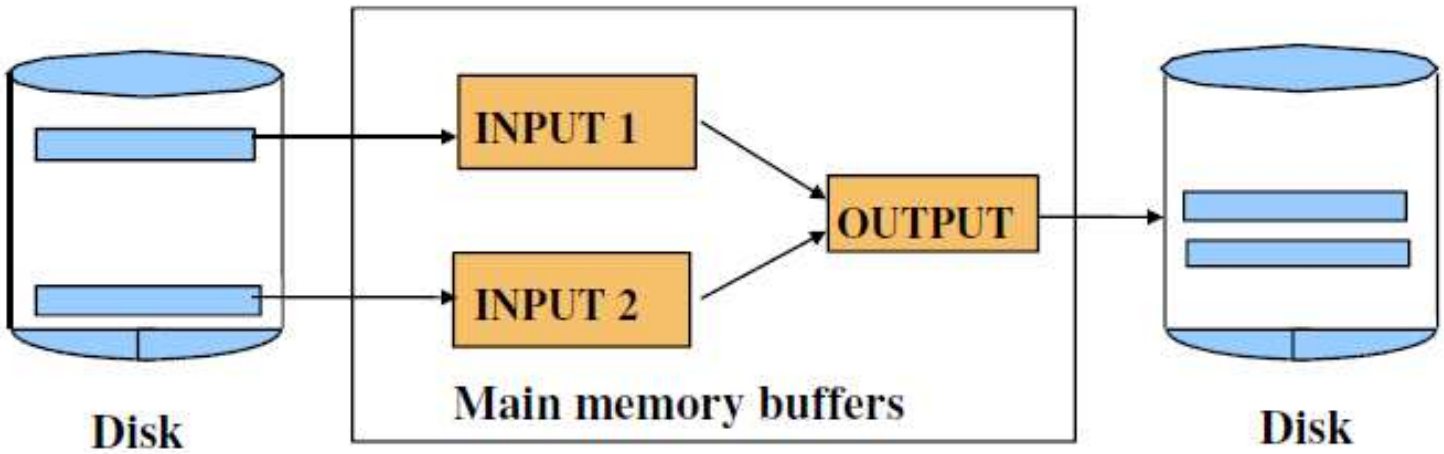
Ordenamiento externo

- Una variedad de operaciones se benefician de un conjunto de registros ordenados.
- Por ejemplo, una consulta SQL podría requerir una salida ordenada de los registros que cumplen con alguna condición.

Select Rut, Nombre, Direccion, Region
From alumnos
Order BY Region

Debido a que no resulta razonable limitar el tamaño de un archivo. Cómo es posible ordenar los registros de un archivo cuyo tamaño es superior al espacio disponible en memoria principal?

Para tal efecto se considera un proceso de ordenamiento y mezcla, mediante el cual el archivo a ordenar se divide y cada una de sus partes se ordena y mezcla con alguna otra hasta obtener el resultado esperado.
Es posible realizar lo anterior en una modalidad conocida como dos vías (two-way) siempre que se cuente con 3 buffers en memoria principal.

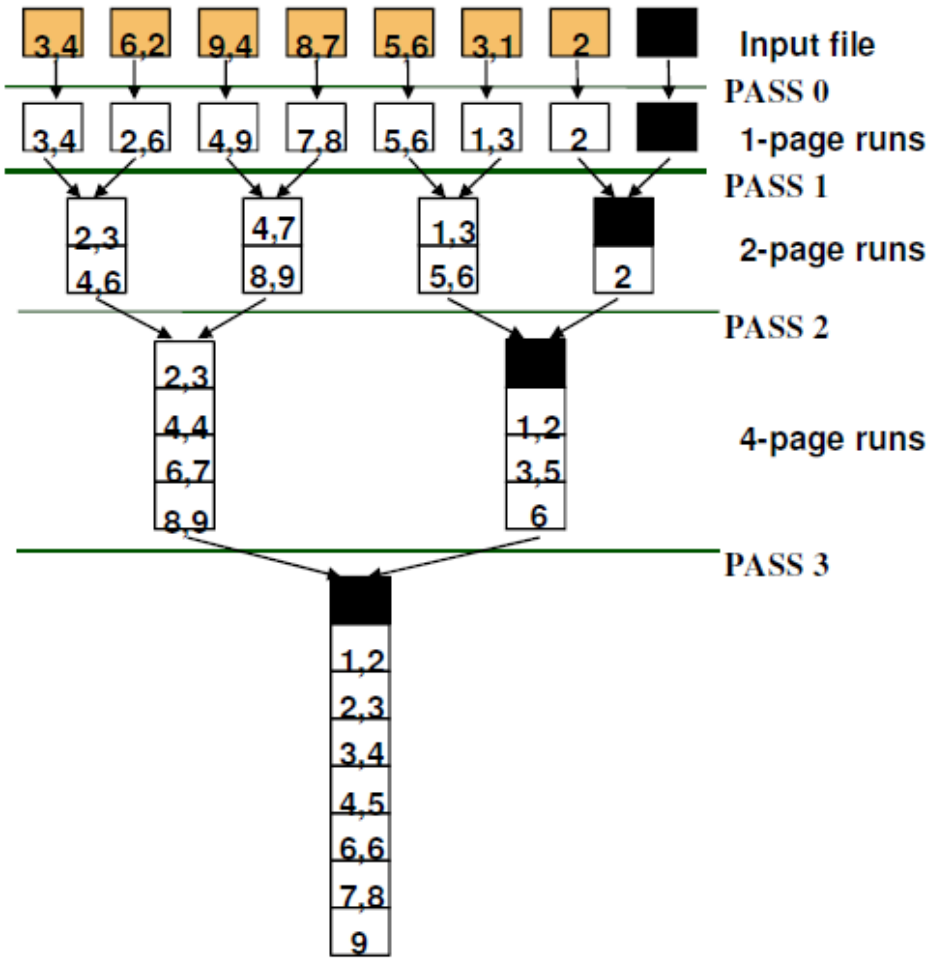


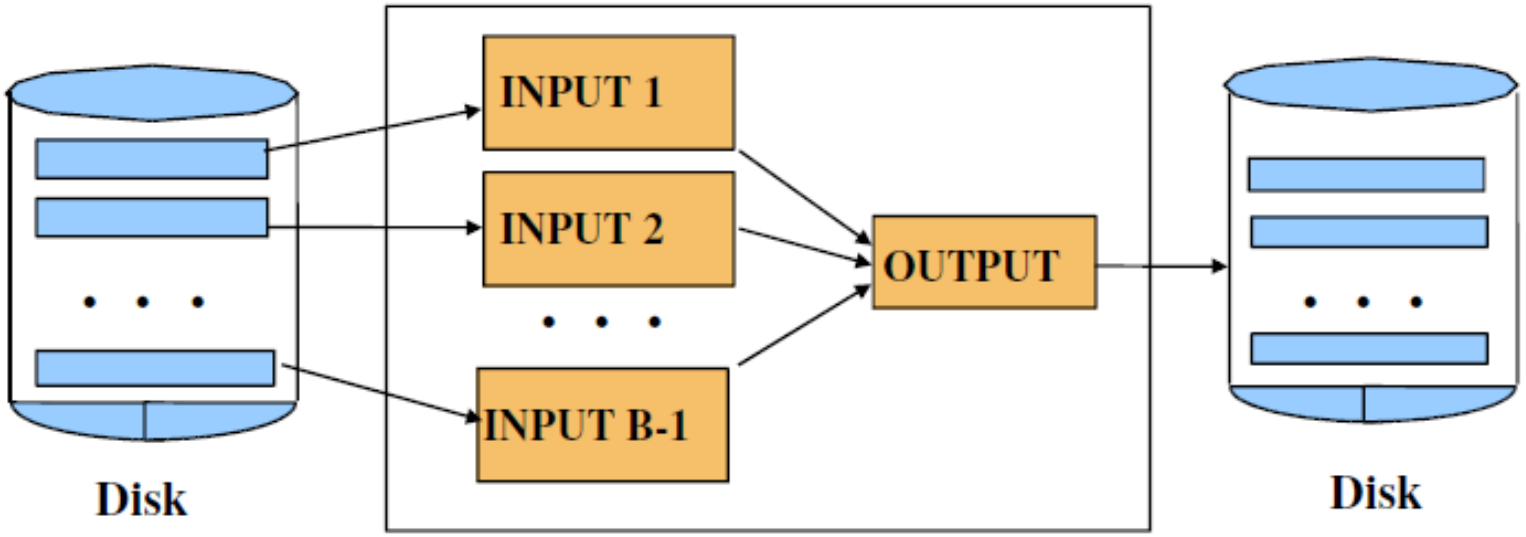
La estrategia de solución es la siguiente:

- **Paso 0** Se lee cada página del archivo.
  Para cada página se realiza un ordenamiento de sus registros.
  Se escribe cada página (run) ordenada hacia disco.
- **Paso 1** Seleccionar 2 runs y llevarlas a memoria.
  Mezclar sus registros con respecto al criterio de ordenamiento.
  Escribir el resultado (run de 2 páginas) a disco.
- .....
- **Paso n** Seleccionar y leer 2 runs escritas en el paso n-1
  Mezclar sus registros con respecto al criterio de ordenamiento.
  Escribir el resultado (run de 2^n páginas) a disco.

Ordenamiento externo

- En cada paso se escriben/leen las N páginas del archivo.
- El número de pasos es $1 + \lceil log_2 N \rceil$.
- Cada paso involucra operciones de E/S. La cantidad de operaciones de E/S: $2N(1 + \lceil log_2 N \rceil)$, el orden de magnitud es $O(Nlog_2 N)$.
- Qué sucede si se tienen B buffers?





- **Paso 0** Se generan N/B runs de B páginas cada uno (el último podría tener menos).
- **Paso 1.....** Seleccionar $B-1$ runs del paso anterior.
  Realiar una mezcla de $B-1$ vías usando la B-ésima página como buffer de salida.

Al igual que en el caso de 2 vías se escriben/leen las N páginas del archivo.
En el paso 0 se escriben N/B runs.
El número de pasos adicionales es $log_{B-1}\lceil N/B \rceil$. Luego el total de operaciones de E/S: $2N(1 + \lceil log_{B-1}\lceil N/B \rceil \rceil)$, el orden de magnitud es $O(Nlog_{B-1}N)$

N	B=3	B=5	B=9	B=17	B=129	B=257
100	7	4	3	2	1	1
1,000	10	5	4	3	2	2
10,000	13	7	5	4	2	2
100,000	17	9	6	5	3	3
1,000,000	20	10	7	5	3	3
10,000,000	23	12	8	6	4	3
100,000,000	26	14	9	7	4	4
1,000,000,000	30	15	10	8	5	4

Ejercicio

Considere un archivo de 10000 páginas y espacio de almacenamiento en memoria principal de 64 páginas. El disco posee una latencia de 5ms, seek de 10ms y puede transferir una página en 1ms. Determine el tiempo requerido para ordenar el archivo considerando:

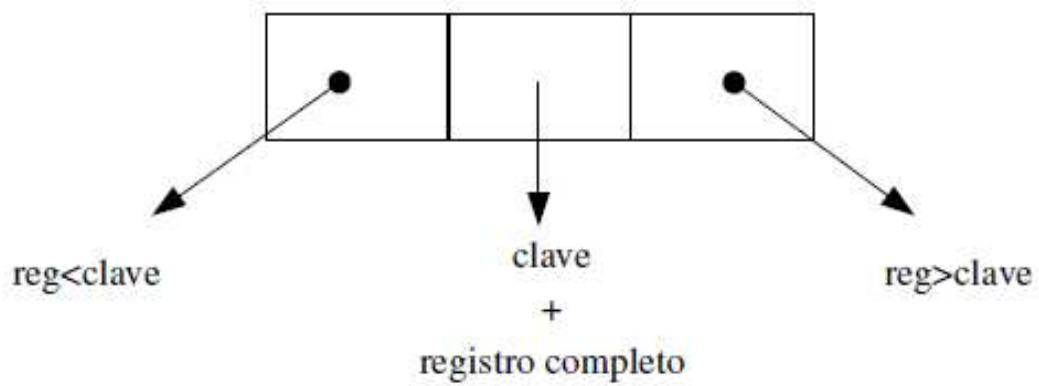
- Caso 1: 63 buffers de entrada de una página cada uno y un buffer de salida de una página.
- Caso 2: 3 buffers de entrada de 16 páginas cada uno y un buffer de salida de 16 páginas.
- Caso 3: 13 buffers de entrada de 4 páginas cada uno y un buffer de salida de 12 páginas.

Archivos organizados como árboles

Los registros se organizan en bloques los cuales se relacionan en forma jerárquica de acuerdo a un cierto orden entre los valores que contiene cada registro.

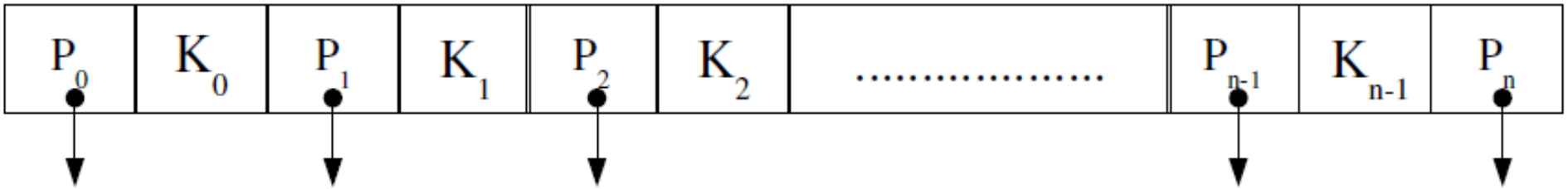
Arboles binarios

La gran ventaja es que permiten búsqueda binaria, pero los tiempos de recuperación se ven afectados por un eventual desbalanceo.



Arboles multiway

Corresponden a una generalización de un árbol binario. En lugar de un registro y dos punteros, un nodo contiene R registros y $R+1$ punteros. Los tiempos de recuperación mejoran, pero se requieren operaciones de balanceo al insertar y eliminar registros.



P_0.....P_n son apuntadores a nodos de subárboles.
K_0.....K_n son los valores de las claves.
Los valores de las claves de un nodo se encuentran en orden ascendente.

Arboles Multiway

Todos los valores de las claves que estan en los nodos del subárbol apuntado por P_i son menores al valor de la clave K_i.
Todos los valores de las claves que estan en los nodos del subárbol apuntado por P_n son mayores al valor de la clave $K - n - 1$.
Los subárboles apuntados por P_i también son árboles de búsqueda de m-vías.

Arboles B

Corresponden a árboles multiway con operaciones de balanceo, es decir, son multiway balanceados.
Se define un árbol B de orden m a aquél árbol que cumple con las siguientes condiciones:

- Ningún nodo tiene más de m hijos.
- Un nodo interno con k hijos contiene $k-1$ registros.
- Todos los nodos terminales se encuentran en el mismo nivel.

Ejercicio

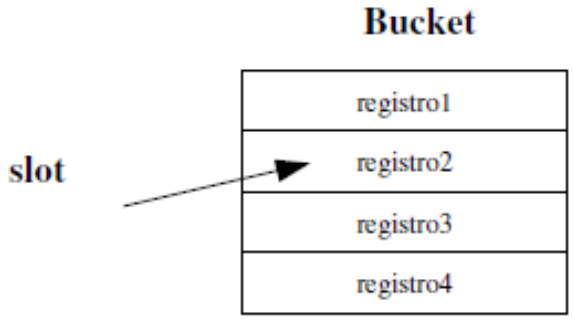
Suponga un archivo de 40000 registros organizado como árbol B. El tamaño de cada registro es 70 bytes. Si el disco donde se almacena utiliza bloques físicos de 1KB y punteros a bloque de 6 bytes.

- Cuál es el orden del árbol?
- Cuantos accesos a disco se necesitan como máximo para recuperar un registro?
- Cuanto tiempo de E/S se necesita para la operación anterior? Considere un disco con los siguientes parámetros; seek:18ms , latencia: 8.3ms, transferencia: 1229 bytes/ms

Direccionamiento directo

La idea general es alcanzar un registro usando una función de transformación (hashing) sobre una clave (que identifica de manera única a cada registro), lo cual llevará a la posición del registro en el medio de almacenamiento (direccionamiento directo).

- El espacio de almacenamiento es dividido en secciones llamadas buckets.
- Un bucket almacena uno o más registros en casilleros de tamaño fijo llamados slots.
- La cantidad de registros en un Bucket define el factor de bloqueo (fb).
- La función hashing transforma la clave en una dirección de bucket relativa.



Desventajas

Se almacena una tabla en el encabezado del archivo para mantener una correpondencia con los bloques físicos. Este enfoque posee 2 desventajas:

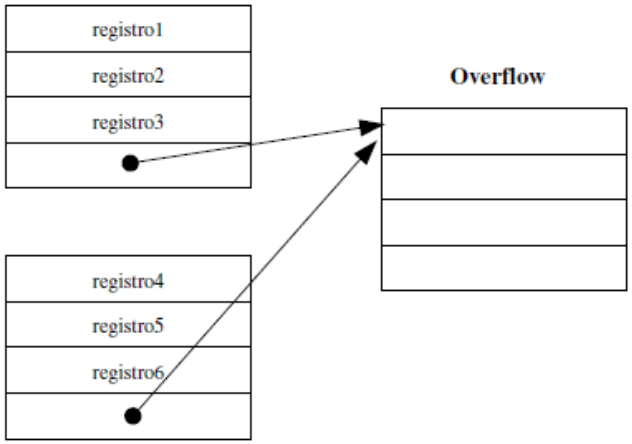
- (a) Colisiones: para registros con distinta clave es posible que la función de transformación entregue un mismo valor. Es necesario resolver las colisiones ya que pueden generar overflow.
- (b) Zona de áreas muertas: la función distribuye de forma no equitativa los datos entre los distintos buckets.

Resolución de colisiones

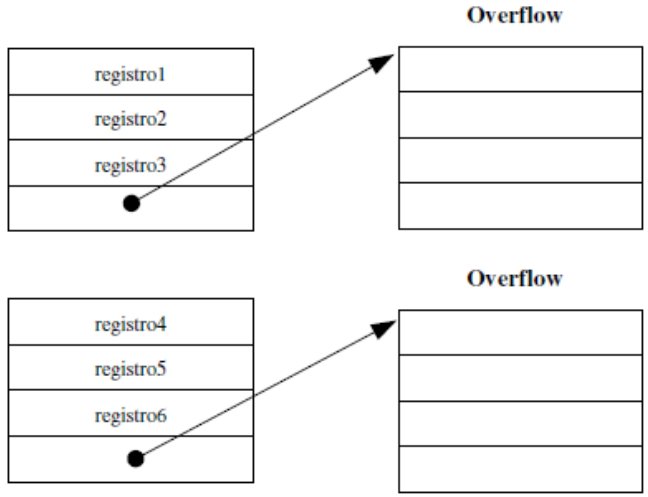
- (a) Overflow Abierto: en este caso la idea es almacenar el registro que está en colisión en el siguiente slot disponible.
  - Ventaja: Minimización de áreas muertas.
  - Desventaja: Búsqueda lineal de registros en colisión.

Resolución de colisiones

- (b) Encadenamiento: en este caso un bucket sirve como nexo hacia un bucket de overflow cuando sea necesario. Existen dos tipos de encadenamiento:
  - Unificado: bucket de overflow es compartido por varios buckets que tienen overflow.
  - Exclusivo: cada bucket tiene potencialmente un bucket de overflow exclusivo.



Overflow Unificado



Overflow Exclusivo

Archivos directos

Todo lo anterior implica:

- Un espacio de direcciones fijo.
- Lo normal es que los archivos cambien.

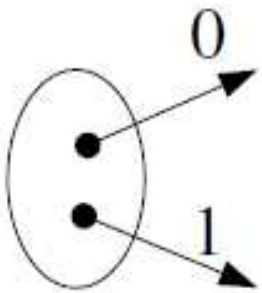
Idea

Proponer un mecanismo que permita la expansión dinámica del archivo.

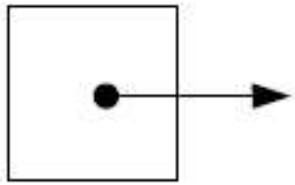
Hashing dinámico

Idea: llegar a los datos por medio de estructura de acceso jerárquica.
Características:

- El número de buckets no es fijo.
- Se puede partir con un solo bucket.
- Cuando un bucket se encuentra en overflow se divide en dos buckets considerando como criterio de división el valor del bit más significativo de lo entregado por la función de transformación. En este caso se crea una estructura de árbol o directorio.
- Existen 2 tipos de nodo
  - nodo interno: guían la búsqueda.
  - nodo hoja: apuntan hacia un bucket.



nodo interno

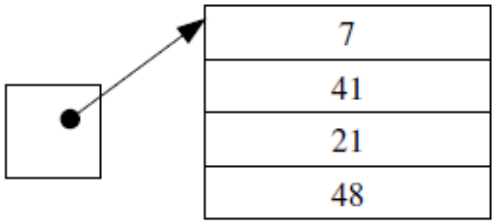


nodo hoja

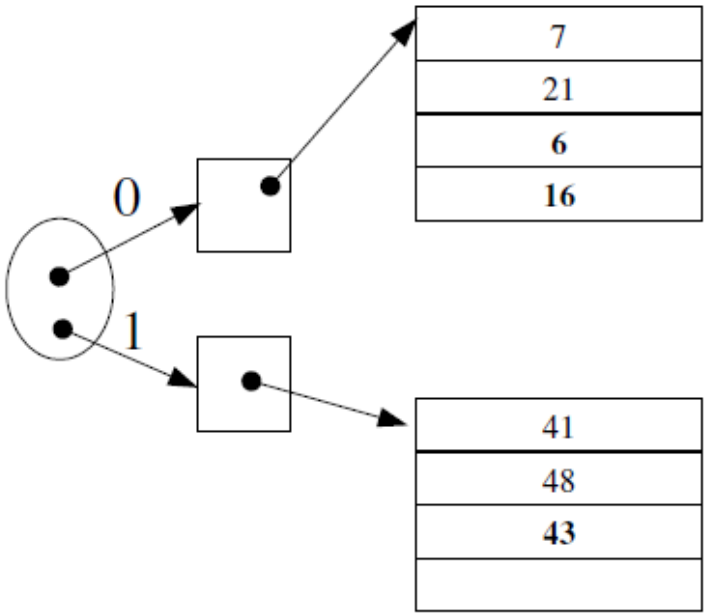
Ejemplo

Suponer buckets con fb=4. Crear un archivo directo en base a:

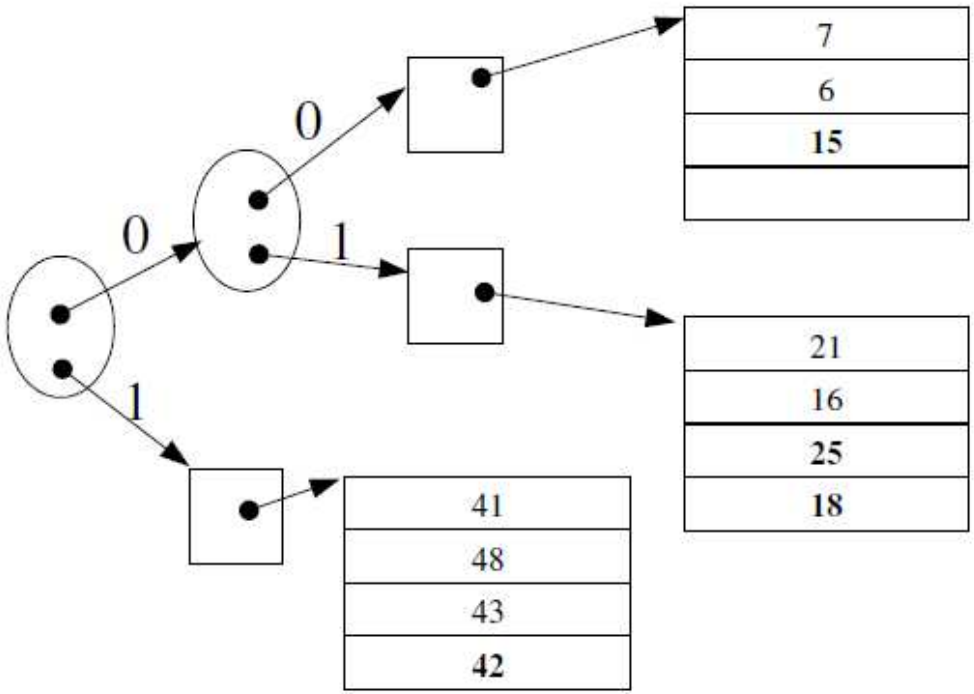
7	000111
41	101001
21	010101
48	110000
43	101011
6	000110
16	010000
15	001111
42	101010
25	011001
18	010010
20	010100



Al ingresar 43 el bucket entra en overflow. Además, es posible ingresar 6 y 16 sin problemas.

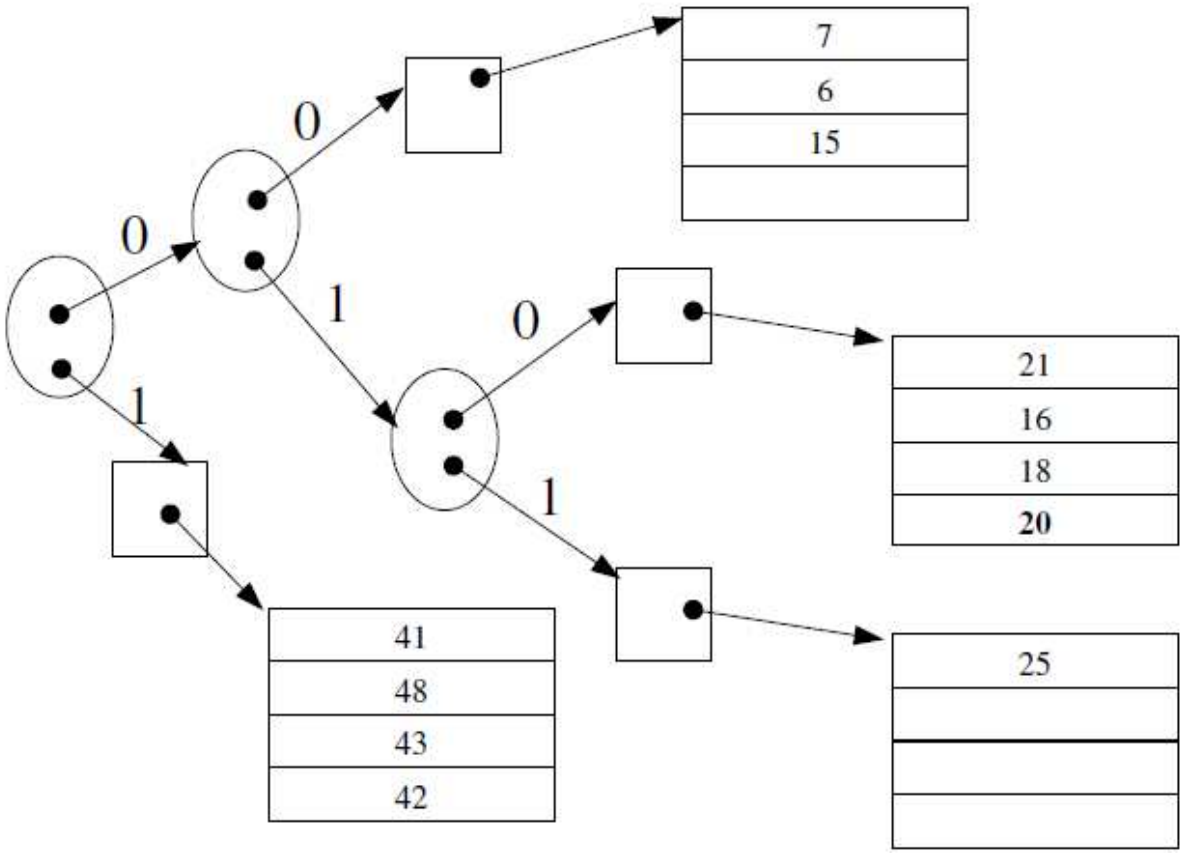


Al ingresar 15 hay overflow!



Además, es posible ingresar 42, 25 y 18 sin problemas.

Al ingresar 20 hay overflow!



Hashing extendido

Idea: Considerar una estructura de acceso tipo arreglo.Características:

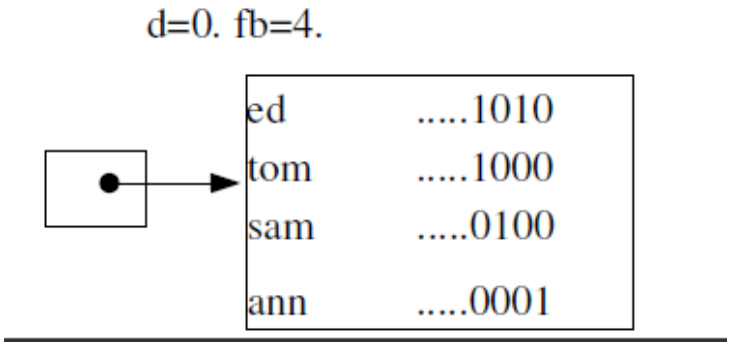
- La estructura de acceso es un arreglo de 2^d direcciones de buckets.
- d es la profundidad global del directorio.
- Se suponen buckets de tamaño fijo.
- En caso de overflow es necesario doblar el tamaño del directorio.
- La estructura de acceso se construye considerando el bit menos significativo.

Ejercicio

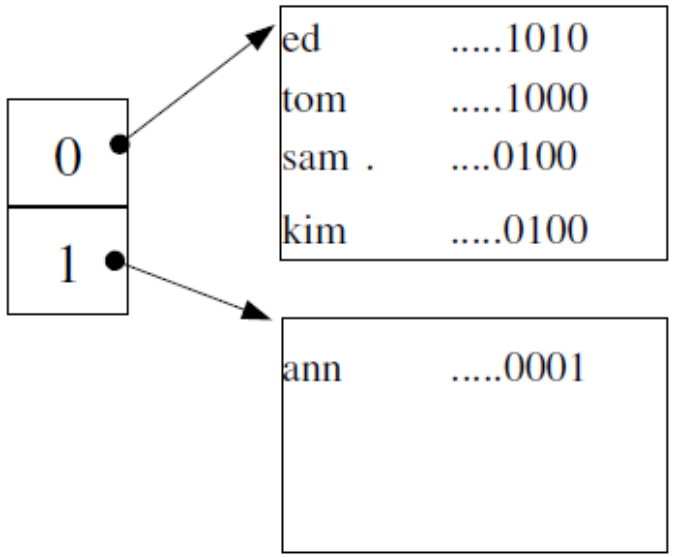
Considerando las siguientes transformaciones de claves, construya un archivo utilizando hashing extendido.

ed	111010001010
tom	001101101000
sam	111100010100
ann	010011010001
kim	100111010100
kely	011011011110
mindy	110111000101
mark	110001011001
chris	111001110011
laura	001011010101
sue	101110001100
jill	111000110010
amy	111000011101

Inicialmente $d = 0,\ fb = 4$

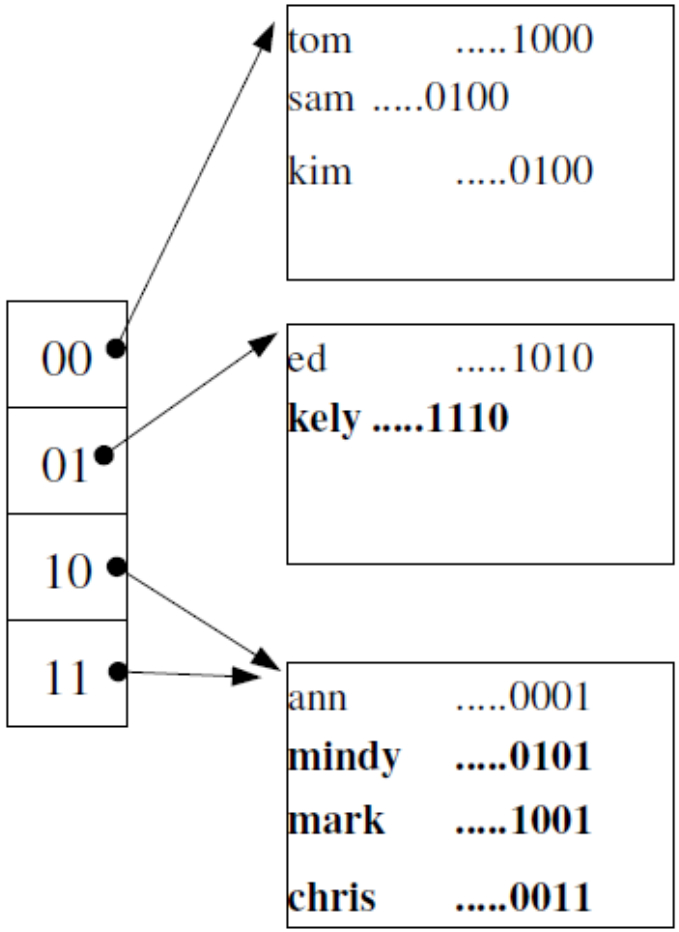


Al agregar kim existe overflow. El tamaño del directorio se debe doblar. $d = 1$



Al agregar kely hay overflow. Ahora $d = 2$

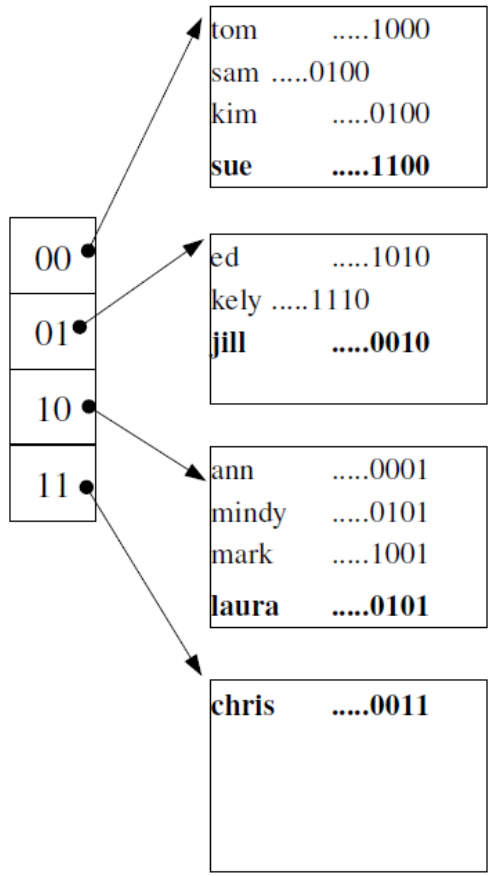
ed	111010001010
tom	001101101000
sam	111100010100
ann	010011010001
kim	100111010100
kely	011011011110
mindy	110111000101
mark	110001011001
chris	111001110011
laura	001011010101
sue	101110001100
jill	111000110010
amy	111000011101



Archivos directos

Al agregar laura hay overflow. Sin embargo, no es necesario doblar el directorio sino crear un nuevo bucket. Además, es posible agregar sue y jill sin problemas!!

ed	111010001010
tom	001101101000
sam	111100010100
ann	010011010001
kim	100111010100
kely	011011011110
mindy	110111000101
mark	110001011001
chris	111001110011
laura	001011010101
sue	101110001100
jill	111000110010
amy	111000011101



Archivos directos