

Aux #3

2015年9月28日 (月)

Memoria externa

$$\text{Ordenar: } \Theta(N \log N) \rightarrow \Theta\left(\frac{N}{B} \log \frac{n}{B} \frac{N}{B}\right) = \Theta(n \log_m n)$$

N : n° de eltos

B : unidad (de bloque) de I/O al disco

M : tamaño de la memoria

$$\frac{N}{B} = n, \quad \frac{M}{B} = m$$

PI

$$A[1..N] \Rightarrow C[i] = A[B[i]]$$
$$B[1..N] \quad i=1..N$$

contienen
todos los números
 $\in \{1, \dots, N\}$

$\Theta(N)$ **no queremos!**

queremos que quede en
función de n y m
involucra bloques.

* Sea un arreglo $X[1..N]$

• Al ordenar, obtengo $[1..N]$

• Si considero a X como una permutación ϕ_X , ordenar aplica ϕ_X^{-1}

• Algoritmo:

• Copiar A a A' tq $A'[i] = (i, A[i])$ // $\Theta(\frac{N}{B}) = \Theta(n)$

• Ordeno A' por el segundo argumento $\Rightarrow (\phi_A^{-1}, \text{id})$ // $\Theta(n \log_m n)$

• Copio B en el 2° argumento $\Rightarrow (\phi_A^{-1}, \phi_B)$ $\Theta(n)$

• Ordeno A' por el primer argumento $\Rightarrow (\text{id}, \phi_A \circ \phi_B)$ // $\Theta(n \log_m n)$

• Copio el 2° argumento en C
 $\Rightarrow \phi_A \circ \phi_B$ // $\Theta(n)$
 $A[B[i]]$

Tiempo total: $\Theta(n \log_m n) \ll \Theta(N)$

E # xVA

Hacer dibujo!! ppto, revisar si $A \leftrightarrow B$ estuvo bien.

$$A = [2 \ 4 \ 1 \ 5 \ 3]$$

$$1) A' = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 4 & 1 & 5 & 3 \end{bmatrix}$$

$$B = [3 \ 5 \ 2 \ 1 \ 4]$$

$$2) A' = \begin{bmatrix} 3 & 1 & 5 & 2 & 4 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix}$$

$$\Rightarrow C = [1 \ 3 \ 4 \ 2 \ 5]$$

$$3) A' = \begin{bmatrix} 3 & 1 & 5 & 2 & 4 \\ 3 & 5 & 2 & 1 & 4 \end{bmatrix}$$

$$4) A' = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 5 & 1 & 3 & 4 & 2 \end{bmatrix}$$

5) Esto es $B[A[i]]$ \times

P

$$(n) \theta = \left(\frac{n}{2}\right) \theta$$

$$(n \log n) \theta$$

$$(n) \theta$$

$$(n \log n) \theta$$

$$(n) \theta$$

$$(n \log n) \theta = \text{total cost}$$

P2

Relación Binaria R
($R \subseteq \{1..n\} \times \{1..n\}$) } $[(a_1, b_1), (a_2, b_2), \dots]$
 $|R| \gg M$

Simetría: " $(a, b) \in R \Leftrightarrow (b, a) \in R$ "
 $\quad \quad \quad R \quad \quad \quad R'$

- \Rightarrow
- Hacer copia de R : R'
 - Ordenar R por el primer argumento \rightarrow y luego por el 2°
 - Ordenar R' por el segundo \rightarrow luego por el 1°
 - Comparar secuencialmente (invirtiendo los pares de R')
 - Si una comp. falla \Rightarrow no es simétrica.

P3

A , $|A|=N$, $A = a_1, a_2, \dots, a_n$

Mayoría (a_1, \dots, a_n)

```
X ← a1
c ← 1
for y in a2 ... aN:
    if x == y
        c ← c + 1
    else c ← c - 1
    if c == 0:
        c ← 1
        x ← y
return x.
```

Dem: por contradicción.

Supongamos que $\exists a_k$ con $> \frac{N}{2}$ apariciones

Si $a_k \neq x$, todas las instancias de a_k deben haber sido "quemadas" por otro elemento

no necesariamente el mismo en todos los casos.

\Rightarrow hay $> \frac{N}{2}$ a_i 's $\neq a_k \Rightarrow \Leftarrow$

Luego Mayoria retorna el a_k correcto si \exists ; si no

\rightarrow verificar (x, a_1, \dots, a_N)

Compara cada a_i con x y cuenta el # de aciertos

\Rightarrow si son más de $N/2 \rightarrow$ retorno el conte


\Rightarrow else retorno "No"

* ojo con el funcionamiento del contador, en casos extremos

PPTOS

para 1 lado y otro

[P1] a) ordenar y contar

b)  pares de nodos en los que existe un camino de largo 2.


c)

2015年9月29日 (火)

Colas de prioridad en memoria secundaria

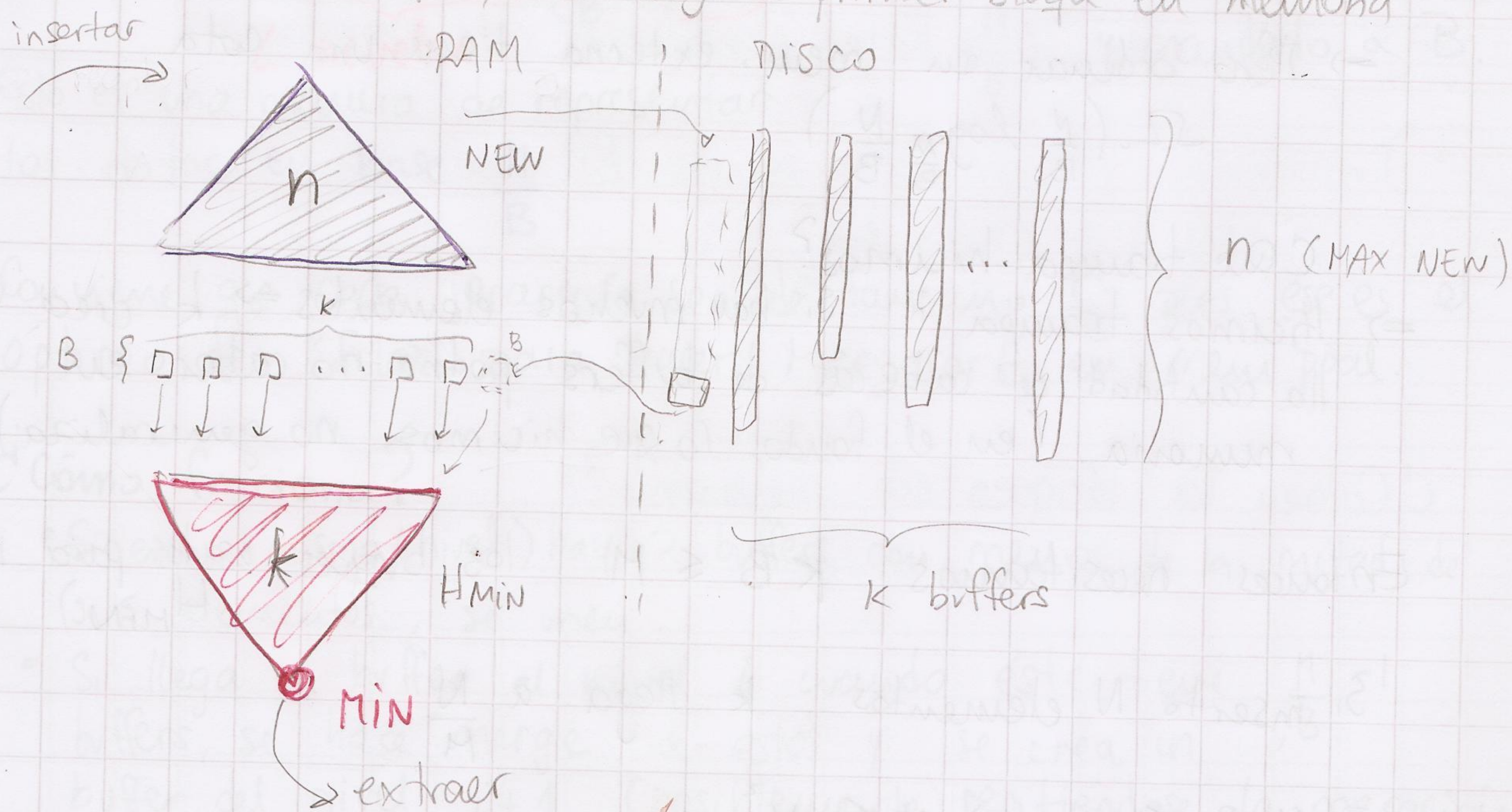
OPERACIONES:

- inicializar
- insertar elementos
- extraer mínimo.

En mem principal, tenemos el HEAP BINARIO 

Idea:

- tener un heap en memoria principal
- Si se llena, ordeno sus elementos y los escribo a disco en arreglo como un arreglo: "buffer"
- De cada buffer, mantengo el primer bloque en memoria



- En otro heap en memoria se almacenan estos bloques (Ojo al insertar, comparar con el mínimo del H_{min})

Análisis:

¿Cuánto paga un elemento al insertarlo y eventualmente extraerlo?

(en I/O)

- $\Theta(1/B)$ {
- entra a NEW $\rightarrow 0$
 - viaja a un buffer en Disco $\rightarrow 1/B$ (promediado cuando entran B elementos a la vez)
 - pasa a un minibloque $\rightarrow 1/B$
 - llega a Hmin $\rightarrow 0$
 - sale $\rightarrow 0$
- no se pondera por M (tamaño buffer)
pres en el modelo no se considera secuencialidad.

¿Qué pasa si inserto N elementos y dp los saco todos?

$$\Rightarrow \Theta(1/B) \cdot N = \Theta\left(\frac{N}{B}\right), \text{ pero salen en orden}$$

\Rightarrow Pero ordenar en mem. externa tiene una cota

$$\Omega\left(\frac{N}{B} \log_{\frac{M}{B}} \frac{N}{B}\right)$$

¿Qué trampa hicimos?

\Rightarrow Hicimos trampa \because Si hay muchos elementos $\Rightarrow k$ crece y la cantidad de cabezas de buffers podría no caber en memoria. (en el fondo, lo que hicimos no generaliza)

Entonces necesitamos: $kB \leq M$ (los bloques usados para rellenar Hmin)

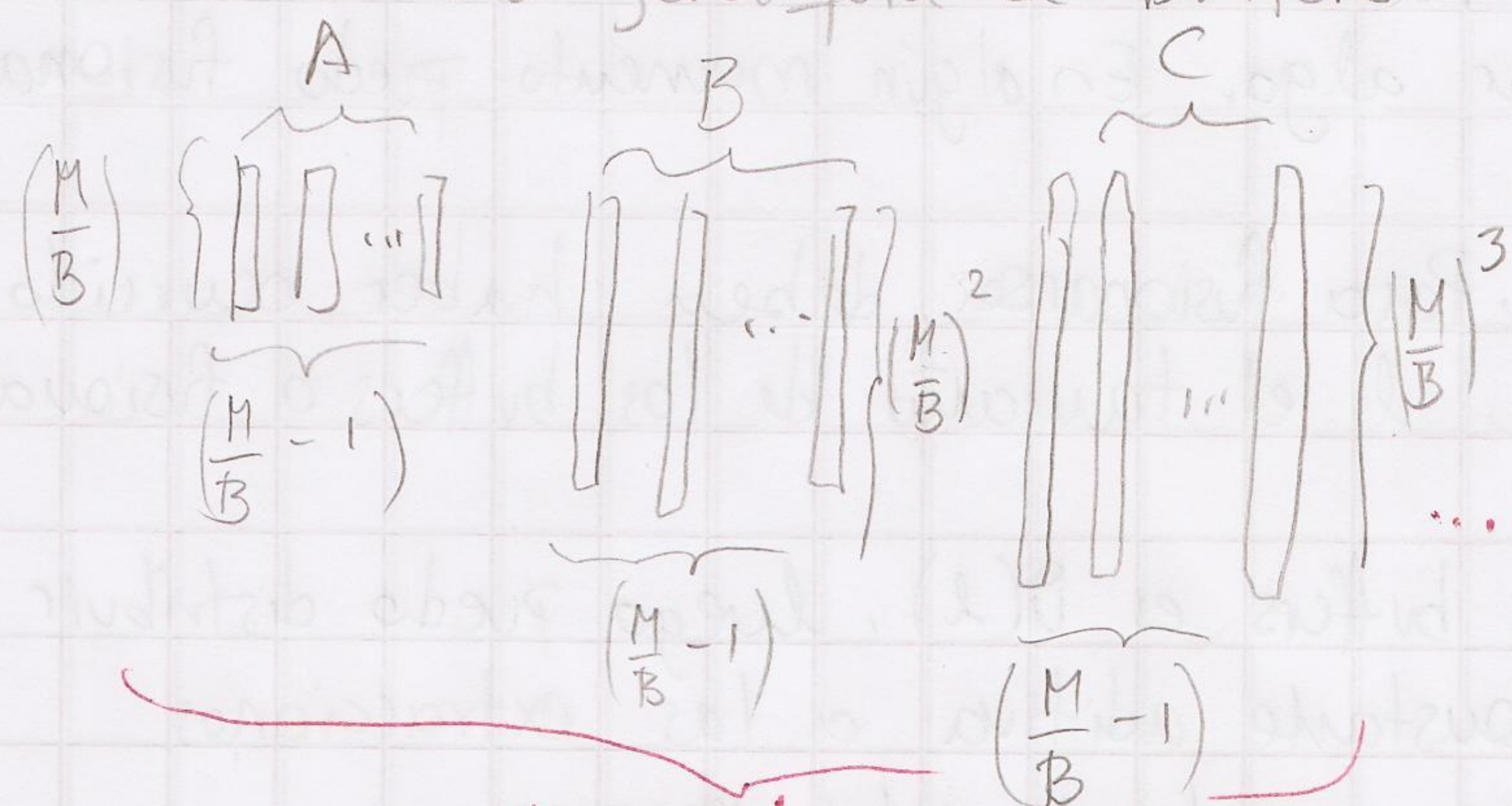
Si inserto N elementos, k llega a $\frac{N}{M}$

$$\Rightarrow \frac{N \cdot B}{M} \leq M \Rightarrow N \leq \frac{M^2}{B}$$

• En este caso $\log_{\frac{M}{B}} \frac{N}{B} \approx 1$, luego $\Theta\left(\frac{N}{B} \log_{\frac{M}{B}} \frac{N}{B}\right) \approx \Theta\left(\frac{N}{B}\right)$

¿Qué hacemos si $N > \frac{M^2}{B}$? es decir, los bloques para H_{\min} no caben en mem. ppal.

La idea es limitar el número de cabezas de buffers en mem. ppal. Haremos una jerarquía de buffers.



Por ejemplo, si llega un $\frac{M}{B}$ -ésimo buffer a A,

se hace merge de todos los buffers para crear un nuevo buffer de tamaño $\left(\frac{M}{B}\right)^2$ y mandarlo a B.

Esto es una manera de representar los datos en base $\frac{M}{B}$.

Conviene que cada jerarquía sea de tamaño $\frac{M}{B}$ pues ese es el óptimo de "filas" para hacer Mergesort en mem. ppal.

¿Cómo funciona?

- Si en un mismo nivel hay 2 buffers con menos de la mitad de sus elementos, se unen.
- Si llega un buffer al nivel i cuando este tiene $\frac{M}{B} - 1$ buffers, se hace merge de éstos y se crea un buffer del nivel $i+1$ (posiblemente repitiéndose la operación).

¿Cuántos niveles hay?

El máx número de niveles es tal que

$$\frac{N}{B} = \Theta\left(\left(\frac{M}{B}\right)^L\right) \Rightarrow L = \Theta\left(\log_{\frac{M}{B}} \frac{N}{B}\right)$$

el ipc del elemento



Evaluemos el costo de la vida de un elemento.

- Supongamos que pasa por todos los niveles

- Entrar al 1^{er} nivel: $1/B$

- Pasar del $i \rightarrow i+1$: $2/B \Rightarrow \Theta\left(\frac{L}{B}\right)$ (leer, merge en mem ppal, escribir)

- Salir a mem ppal: $1/B$

Wait! Nos faltó contar algo. En algún momento puedo fusionar buffers semivacíos.

* Los buffers hacen llenos. Para fusionarse, deben haber ocurrido $\frac{l}{2} + \frac{l}{2}$ extracciones (con l el tamaño de los buffers a fusionar)

Hacer el merge de los buffers es $\Theta(l)$, luego puedo distribuir este costo como una constante aditiva a las extracciones
 \Rightarrow "está bien" no haber considerado estas merges.

- Insertemos y luego extraigamos N elementos

$$\Theta\left(\frac{NL}{B}\right) = \Theta\left(\frac{N}{B} \log_{\frac{M}{B}} \frac{N}{B}\right)$$

es L !

¿Caben los bloques en memoria?