Lezione 22– File hash: caratteristiche ed esercizi

Prof.ssa Maria De Marsico
demarsico@di.uniroma1.it

SAPIENZA
UNIVERSITÀ DI ROMA

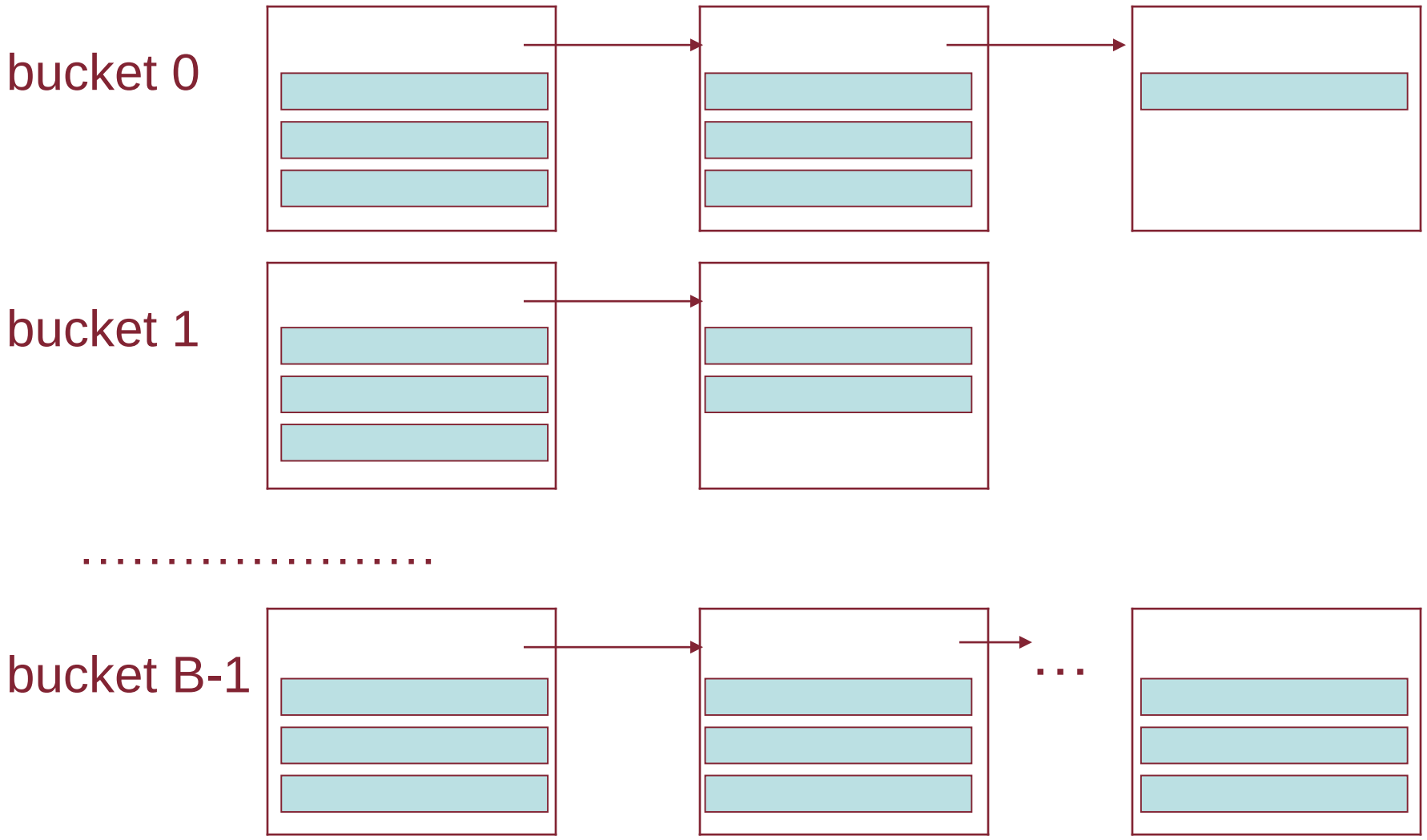
Parte del materiale è tratto dal materiale preparato dalla prof.ssa Moscarini per il canale A-L

File hash

- Il file è suddiviso in ***bucket*** (secchi) numerati da *0* a *B-1*

- ciascun **bucket** è costituito da **uno o più blocchi** (collegati mediante **puntatori**) ed è organizzato come un **heap**

Bucket



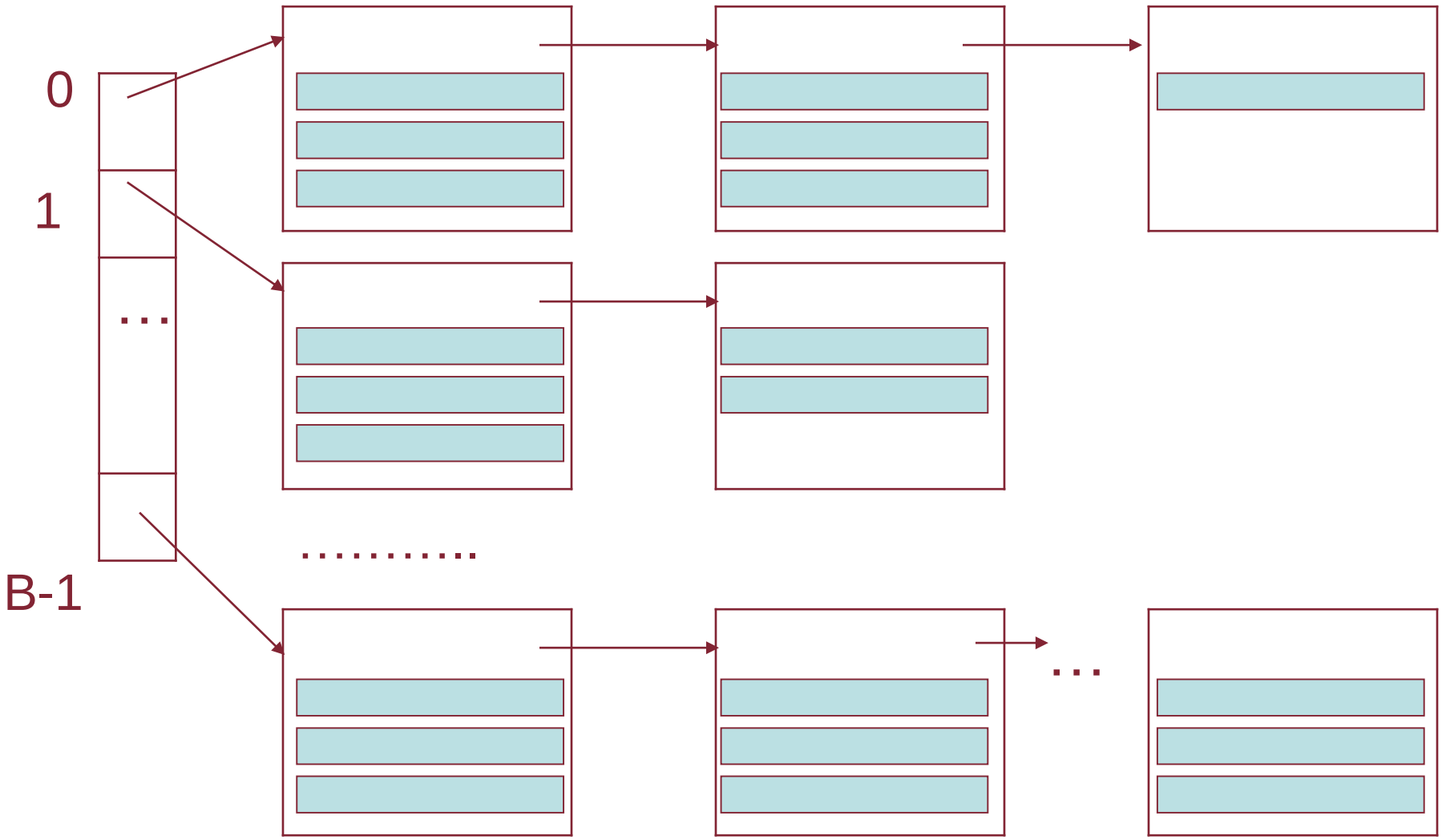
Bucket directory

- L'accesso ai bucket avviene attraverso la ***bucket directory*** che contiene ***B* elementi**

- L'i-esimo elemento contiene l'indirizzo (**bucket header**) del **primo** blocco **dell'i-esimo bucket**.

Bucket directory

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

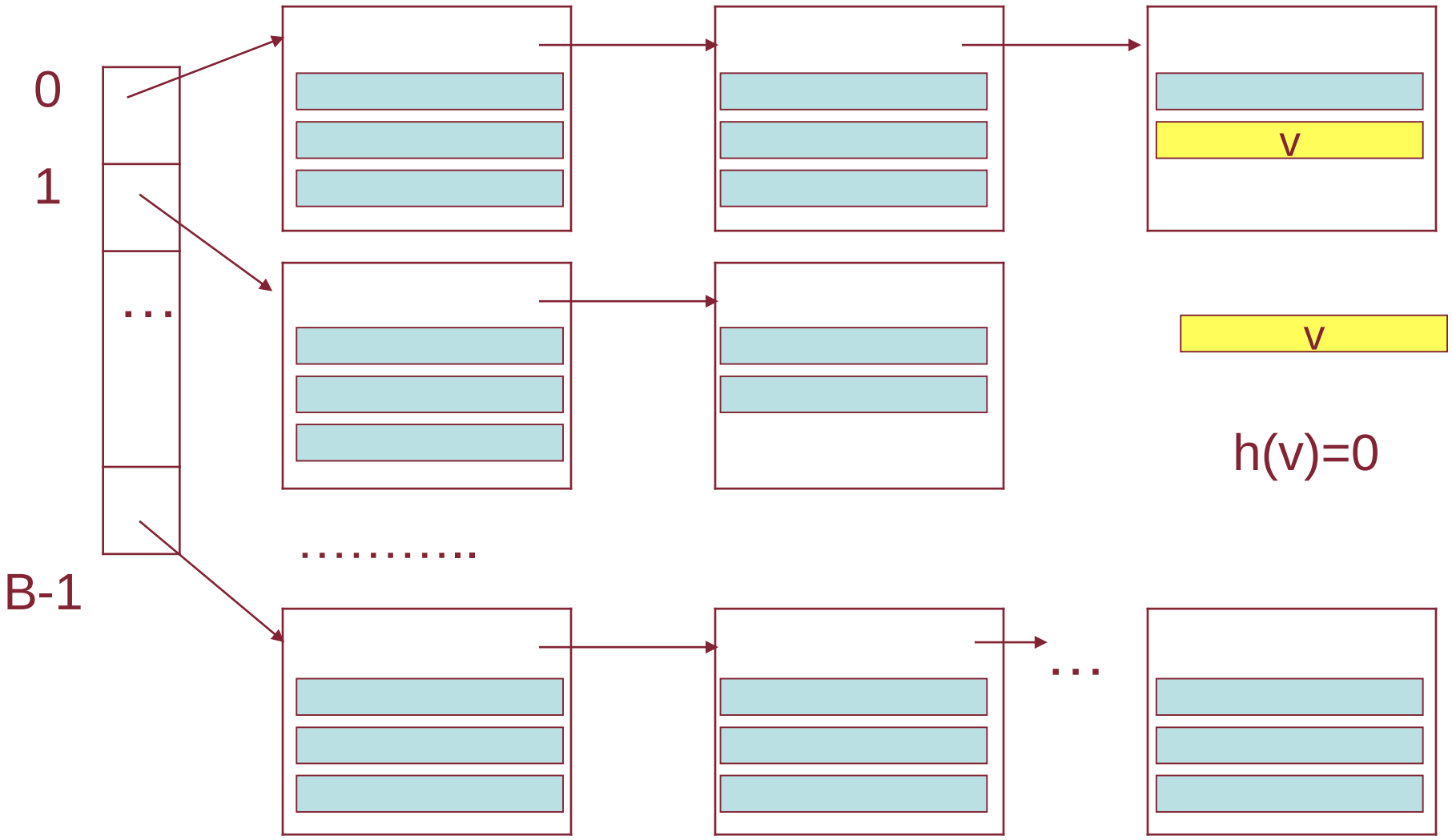


Funzione hash

Dato un valore ***v*** per la chiave **il numero del bucket** in cui deve trovarsi un record con chiave *v* è **calcolato** mediante una funzione che prende il nome di ***funzione hash***

Inserimento

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA



Operazioni

- Una qualsiasi operazione (ricerca, inserimento, cancellazione, modifica) su un file hash richiede
  - la **valutazione di *h(v)*** per **individuare** il bucket
  - esecuzione dell'operazione sul bucket che è **organizzato** come un **heap**.

Poiché l'inserimento di un record viene effettuato sull'ultimo blocco del bucket, è opportuno che la bucket directory contenga **anche**, per ogni bucket, un puntatore **all'ultimo record** del bucket.

Costi operazioni

Pertanto:

Se la funzione hash distribuisce uniformemente i record nei bucket:

ogni bucket è costituito da n/B blocchi

e quindi

il costo richiesto di un'operazione è **approssimativamente *1/B-esimo*** del costo della stessa operazione se il file fosse organizzato come un heap.

Funzione hash

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

- Una funzione hash per essere "**buona**" deve ripartire **uniformemente** i record nei bucket, cioè al **variare** del valore della chiave deve assumere con la "**stessa" probabilità** uno dei valori compresi tra ***0* e *B-1.***

- In genere, una funzione hash trasforma la chiave in un **intero**, **divide** questo intero per B, e fornisce il **resto** della divisione come **numero del bucket** in cui deve trovarsi il record con quel valore della chiave.

Esempio funzione hash

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

1	0	0	1	0	1	1	1	1	0	1	0

v

- 1. trattare il valore *v* della chiave come una sequenza di bit

- 2. suddividere tale sequenza in gruppi di bit di **uguale** lunghezza e **sommare** tali gruppi trattandoli come **interi**

1	0	0	1	0	1	1	1	1	0	1	0

9 + 7 + 10 = 26

Esempio funzione hash

- 3. dividere il risultato per il numero dei bucket (cioè per *B*) e prendere il **resto** della divisione come numero del bucket in cui deve trovarsi il record con chiave *v.*

- *Es: se B=3 allora il record con chiave v si deve trovare nel bucket 2 in quanto*
  - *26=3*8+2*

Considerazioni

Da quanto detto appare evidente che **quanti più sono i bucket tanto più basso è il costo di ogni operazione**. D'altra parte limitazioni al numero dei bucket derivano dalle seguenti considerazioni:

- ogni bucket deve avere **almeno** un blocco
- se le dimensioni della bucket directory sono tali che **non può essere mantenuta in memoria principale** durante l'utilizzo del file, **ulteriori accessi** sono necessari per leggere i blocchi della **bucket directory.**

Nota

Negli esempi che seguono, così come negli esercizi di esame, **a meno che non venga specificato diversamente** assumeremo **sempre** che:

- ogni record deve essere contenuto completamente in un blocco (non possiamo avere record a cavallo di blocchi)
- i blocchi vengono allocati per intero (non possiamo allocare frazioni di blocco)

Esempio 1

Supponiamo di avere un file di 250.000 record. Ogni record occupa 300 byte, di cui 75 per il campo chiave. Ogni blocco contiene 1024 byte. Un puntatore a blocco occupa 4 byte.

a) Se usiamo una organizzazione hash con 1200 bucket, quanti blocchi occorrono per la bucket directory ?

b) Quanti blocchi occorrono per i bucket, assumendo una distribuzione uniforme dei record nei bucket?

c) Assumendo ancora che tutti i bucket contengano il numero medio di record, qual è il numero medio di accessi a blocco per ricercare un record che sia presente nel file?

d) Quanti bucket dovremmo creare per avere invece un numero medio di accessi a blocco inferiore o al massimo uguale a 10, assumendo comunque una distribuzione uniforme dei record nei bucket?

Esempio 1

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

Abbiamo i seguenti dati:

- il file contiene 250.000 record: NR = 250.000
- ogni record occupa 300 byte: R = 300
- il campo chiave occupa 75 byte: K = 75
- ogni blocco contiene 1024 byte: CB = 1024
- un puntatore a blocco occupa 4 byte: P = 4

Attenzione: un calcolo del tipo (NR×R)/CB per calcolare l'occupazione totale è sbagliato per tre motivi:

- Avremmo record a cavallo di blocchi (se la taglia non è divisibile)
- Avremmo blocchi a cavallo di bucket (se gli ultimi blocchi dei bucket non sono riempiti per intero)
- Mancano i puntatori al prossimo blocco nel bucket

Esempio 1

a) Indichiamo con B il numero di bucket e con BD il numero di **blocchi per la bucket directory**. La bucket directory è essenzialmente un **array di puntatori** indicizzato da 0 a B-1.

- Vediamo prima **quanti puntatori entrano in un blocco**: $PB=\lfloor CB/P \rfloor$ (prendiamo la parte intera inferiore perché assumiamo che i record siano contenuti interamente nel blocco) i record devono essere interi; in questo caso **CB (=1024) è multiplo di P (=4)** ma questo potrebbe non essere vero; $RB=\lfloor 1024/4 \rfloor = 256$
- Ci occorreranno $BD = \lceil 1200/256 \rceil = \lceil 4{,}69 \rceil = 5$ blocchi per la bucket directory (prendiamo la **parte intera superiore** perché, non essendo stato specificato diversamente dall'esercizio, i **blocchi vengono allocati interamente**, e quindi la frazione di blocco **va arrotondata ad un blocco intero**).
- **Nota**: se viene chiesto che nella bucket directory venga memorizzato anche il puntatore **all'ultimo** blocco del bucket occorre considerare **coppie** intere di puntatori (non possiamo spezzare in due blocchi la coppia di puntatori per un bucket) $PB=\lfloor CB/(2 \times P) \rfloor$

Esempio 1

- b) Abbiamo record a lunghezza **fissa**, quindi supponiamo di **non avere** un **direttorio di record all'inizio del blocco (TUTTO lo spazio è occupato dai dati)**. Serve però un **puntatore** per ogni blocco per linkare i blocchi **dello stesso bucket.** In un blocco dobbiamo quindi memorizzare **il maggior numero possibile di record e in più un puntatore** per un eventuale prossimo blocco nel bucket. Se indichiamo con **M il massimo numero di record memorizzabili** in un blocco, avremo $M \times R$ **+ P** $\leq CB$, cioè $300M + 4 \leq 1024$, quindi $M \leq 1020/300 = 3,4$. M deve essere **intero**, perché non essendo stato detto altrimenti nella traccia, assumiamo che i record **non possano trovarsi a cavallo di due o più blocch**i, quindi assumiamo M = 3.
  - In alternativa possiamo prima **sottrarre la taglia del puntatore** dallo spazio utile e poi prendere la **parte intera inferiore della divisione dello spazio rimanente per la taglia dei record**
  - **M=** $\lfloor$**(**CB-P)/R$\rfloor = \lfloor 1020/300 \rfloor = \lfloor 3,4 \rfloor = 3$
  - **Nota:** si potrebbe chiedere che ogni blocco abbia anche un puntatore al blocco precedente quindi $M \times R$ **+** $2\times$ **P** $\leq CB$ oppure **M=** $\lfloor$**(**$CB- 2 \times P)/(R)\rfloor$

Esempio 1

- b cont.)
  - Se la distribuzione dei record nei bucket è uniforme, indicando con RB il numero di record in un bucket, avremo RB = ⌈NR/B⌉ = ⌈250.000/1200⌉ = ⌈208,3⌉ = 209 record per ogni bucket (prendiamo la parte **intera superiore** perché i record devono essere **inseriti tutti**, quindi la frazione di record **va considerata per non tralasciare alla fine una parte dei record stessi**).
  - Indicando con NB il numero di blocchi per ogni bucket, occorrono quindi NB = ⌈RB/M⌉ = ⌈209/3⌉ = 70 blocchi **per ogni bucket**; indicando con BB il numero complessivo di blocchi per il file hash, avremo BB = NB × B = 70 × 1200 = 84.000 blocchi;

Esempio 1

- c) Se la distribuzione dei record nei bucket è **uniforme**, in un bucket avremo come detto NB = ⌈RB/M⌉ = 70 blocchi per bucket. Poiché la ricerca avviene **solo sul bucket individuato** in base al risultato dell'applicazione della **funzione hash** alla chiave del record, avremo un numero di accessi a blocco pari a quello **che si avrebbe su un heap della stessa dimensione del bucket (cioè 1/B rispetto alla dimensione originale).** In media accederemo **alla metà di questi blocchi**, quindi indicando con MA il numero medio di accessi, avremo MA, pari a ⌈NB/2⌉. Nel nostro caso quindi **MA = ⌈70/2⌉ = 35**; a questi occorrerà aggiungerne 1 se il blocco della bucket directory relativo al bucket in cui si trova il record non si trova già in memoria principale;

Esempio 1

- d) per avere un numero di accessi a blocco inferiore o al massimo uguale a 10, riscriviamo l'espressione di MA in modo che vi **compaia esplicitamente il numero di bucket B**, e **tralasciando per semplicità gli arrotondamenti** che abbiamo effettuato via via nei calcoli **tranne l'ultimo**. Avremo $MA = \lceil NB/2 \rceil = \lceil (RB/M)/2 \rceil = \lceil (NR/B)/M/2 \rceil = \lceil NR/2(B \times M) \rceil$. Vogliamo calcolare quindi B in modo che $\lceil NR/2(B \times M) \rceil \leq 10$, cioè deve essere $B \geq NR/20M$ (l'inversione della disuguaglianza deriva da fatto che il numero medio di accessi decresce con l'aumentare del numero di bucket). Nel nostro caso dovremo avere allora $B \geq 250.000/(20 \times 3) = 4167$. Verifichiamo infatti che in questo caso avremo $RB = \lceil NR/B \rceil = \lceil 250.000/4167 \rceil = 60$ record per ogni bucket, quindi $NB = \lfloor RB/M \rfloor = 20$ record per bucket, e infine $MA = \lceil NB/2 \rceil = 10$ accessi a blocco.

Esempio 1

- d cont.)
  - Si poteva anche ragionare in un altro modo. Siccome MA = MA = $\lceil NB/2 \rceil$, per avere MA $\leq$ 10 dobbiamo fare in modo che sia NB $\leq$ 20 (ricordiamo che NB è il numero di blocchi in un bucket). Dobbiamo allora avere RB = M $\times$ NB $\leq$ M $\times$ 20, cioè nel nostro caso RB $\leq$ 60 (ricordiamo che RB è il numero di record per bucket). Per avere un numero di record per bucket inferiore a 60, deve essere NR/B $\leq$ 60, e quindi B $\geq$ (250.000/60), ottenendo lo stesso risultato.

Esempio 2

•Supponiamo di avere un file di 780.000 record. Ogni record occupa 250. Ogni blocco contiene 1024 byte. Un puntatore a blocco occupa 4 byte. Usiamo una organizzazione hash con 2500 bucket.

a) Quanti blocchi dobbiamo utilizzare complessivamente per la bucket directory e per i bucket, assumendo una distribuzione uniforme dei record nei bucket ?

b) Quanti blocchi dobbiamo utilizzare complessivamente per i bucket, assumendo che il 30% dei record sia distribuito in modo uniforme su 1000 bucket, e che il restante 70% dei record sia distribuito in modo uniforme sui 1500 bucket rimanenti ?

Esempio 2

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

- Numero record NR = 780.000
- Taglia record R = 250 byte
- Taglia puntatore P = 4 byte
- Capacità blocco CB = 1024 byte
- Numero Bucket B = 2500

a) Calcoliamo innanzitutto quanti puntatori entrano in ogni blocco della bucket directory

- $PB = \lfloor CB/P \rfloor = \lfloor 1024/4 \rfloor = 256$

Si prende la parte intera inferiore perché i puntatori devono essere memorizzati per intero.

- Calcoliamo quindi quanti blocchi occorrono per la bucket directory, cioè per memorizzare 2500 puntatori

Si prende la parte intera superiore perché i blocchi devono essere allocati per intero

- $BD = \lceil B/ PB \rceil = \lceil 2500/ 256 \rceil = \lceil 9,7 \rceil = 10$

Esempio 2

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

- Assumendo una distribuzione uniforme, dobbiamo prima di tutto calcolare quanti record devono essere memorizzati in ogni bucket

$RB = \lceil NR/ B \rceil = \lceil 780.000/ 2500 \rceil = \lceil 780.000/ 2500 \rceil = \lceil 312 \rceil = 312$

Si prende la parte intera superiore perché tutti i record devono essere memorizzati nei bucket .

Vediamo ora quanti record possono essere memorizzati in un blocco, tenendo conto del fatto che ogni blocco di un bucket deve contenere anche un **puntatore al blocco successivo**

$RBL = \lfloor (CB - P)/R \rfloor = \lfloor 1020/250 \rfloor = \lfloor 4,08 \rfloor = 4$

Si prende la parte intera inferiore perché i record devono essere memorizzati per intero..

Calcoliamo infine quanti blocchi occorrono per ogni bucket

$BB = \lceil RB/ RBL \rceil = \lceil 312/ 4 = \lceil NR/ B \rceil = \lceil 78 \rceil = 78$

Si prende la parte intera superiore perché i blocchi devono essere allocati per intero.

Complessivamente ci occorrono

$BD + BB \times B = 10 + 78 \times 2500 = 195.010$ BLOCCHI

Esempio 2

b) I calcoli per la bucket directory e per il numero di record che possono essere memorizzati in un blocco rimangono validi anche in questo caso.

Cambia la distribuzione dei record nei bucket, e quindi il numero di blocchi che occorrono per ogni bucket.

Il 30% dei record è distribuito uniformemente su 1000 bucket, il rimanente 70% dei record è distribuito uniformemente su 1500 bucket.

Quindi il numero di record che va distribuito uniformemente su 1000 bucket è

N1000 = (780.000 X 30)/100 = 234.000

Esempio 2

b cont.)

Per ogni bucket dei 1000 avremo quindi RB1000 = $\lceil N1000/ 1000 \rceil = 234$ record

Quindi per ognuno di questi 1000 bucket occorrono

B1000 = $\lceil RB1000/ RBL \rceil = \lceil 234/ 4 \rceil = \lceil 58,5 \rceil = 59$ blocchi

Il numero di record che va distribuito uniformemente su 1500 bucket è

N1500 = **N - N1000** = 780.000 - 234.000 = 546.000 record

Calcoliamo direttamente per differenza i record rimanenti che sono comunque quelli da memorizzare

Per ogni bucket dei 1500 avremo quindi RB1500 = $\lceil N1500/ 1500 \rceil = 364$ record

Quindi per ognuno di questi 1500 bucket occorrono quindi

B1500 = $\lceil RB1500/ RBL \rceil = \lceil 364/ 4 \rceil = \lceil 91 \rceil = 91$ blocchi

In totale avremo B1000 X 1000 + B1500 X 1500 = 59X1000+91X1500 = 195.500 blocchi per i bucket.

Esempio 2

- Notare che a parità di spazio totale occupato, nella prima parte del file ho un tempo di ricerca medio che è quasi la metà che nella seconda parte (⌈59/2⌉ contro ⌈91/2⌉)