
Esercitazione E3

File System

Indici ISAM

Si consideri un file di 60.000 record di 80 byte ciascuno, con un campo chiave di 16 byte, sul quale è stato costruito un indice ISAM. Supponendo di disporre di blocchi di 512 byte con indirizzi di 4 byte, e di non spezzare mai un record su due blocchi, calcolare:

- a) la dimensione complessiva in byte del file dati e del file indice;
- b) la frazione di spazio inutilizzata rispettivamente nei blocchi del file dati e del file indice.
- c) il numero medio di accessi a disco necessari per accedere ad un record del file dati senza utilizzare l'indice;
- d) il numero minimo e massimo di accessi a disco necessari per accedere ad un record del file dati utilizzando l'indice;

Indici ISAM

a)

a) la dimensione complessiva in byte del file dati e del file indice;

- Ciascun blocco del file dati contiene $R = \lfloor B/r \rfloor = \lfloor 512/80 \rfloor = 6$ record/blk. Pertanto il file dati è costituito da $F = \lceil n/R \rceil = \lceil 60.000/6 \rceil = 10.000$ blocchi.
- I record del file indice sono in numero pari ai blocchi del file dati, cioè: $F = 10.000$. Ciascuno di essi è costituito da $(c+b) = 16+4 = 20$ byte.
- Ciascun blocco del file indice contiene $R_1 = \lfloor B / (c+b) \rfloor = \lfloor 512 / 20 \rfloor = 25$ rec/blk. Quindi il file indice è costituito da $I = \lceil F/R_1 \rceil = \lceil 10.000 / 25 \rceil = 400$ blocchi.

Indici ISAM

b)

b) la frazione di spazio inutilizzata rispettivamente nei blocchi del file dati e del file indice.

– Ciascun blocco del file dati contiene $R = 6$ record.

– Pertanto sono utilizzati $R \cdot r = 6 \cdot 80 = 480$ byte. Ne restano quindi inutilizzati $B - R \cdot r = 512 - 480 = 32$ byte.

– La frazione di spazio inutilizzata nei blocchi del file dati è quindi:

$$(B - R \cdot r) / B = 32 / 512 \approx 0.06 = 6\%$$

– Ciascun blocco del file indice contiene $R_1 = 25$ record.

– Pertanto sono utilizzati $R_1 \cdot (c+b) = 25 \cdot 20 = 500$ byte. Ne restano quindi inutilizzati $B - R_1 \cdot (c+b) = 512 - 500 = 12$ byte.

– La frazione di spazio inutilizzata nei blocchi del file dati è quindi:

$$(B - R_1 \cdot (c+b)) / B = 12 / 512 \approx 0.02 = 2\%$$

Indici ISAM

c)

- c) il numero minimo, massimo e medio di accessi a disco necessari per accedere ad un record del file dati senza utilizzare l'indice;
- Occorre effettuare una ricerca sequenziale su tutto il file dati costituito da **$F = 10.000$** blocchi
 - Il costo minimo si ha nel caso che il record di chiave data venga rinvenuto nel primo blocco: **$C_{\min} = 1$** .
 - Il costo massimo si ha nel caso che il record di chiave data venga rinvenuto nell'ultimo blocco: **$C_{\max} = F = 10.000$** .
 - Il costo medio, nell'ipotesi di distribuzione uniforme della probabilità di accesso, corrisponde alla lettura di metà dei blocchi del file, quindi **$C_{\text{med}} = F / 2 = 5.000$** .

Indici ISAM

d)

d) il numero minimo, massimo e medio di accessi a disco necessari per accedere ad un record del file dati utilizzando l'indice;

- Occorre effettuare preliminarmente una ricerca sequenziale sul file indice costituito da $I = 400$ blocchi. Questa permette di individuare l'indirizzo del blocco contenete il record, al quale si accede poi direttamente.
- Il costo minimo si ha nel caso che il record di chiave data venga rinvenuto nel primo blocco del file indice: $C_{\min} = 1+1 = 2$.
- Il costo massimo si ha nel caso che il record di chiave data venga rinvenuto nell'ultimo blocco del file indice: $C_{\max} = I+1 = 401$.
- Il costo medio, nell'ipotesi di distribuzione uniforme della probabilità di accesso, corrisponde alla lettura di metà dei blocchi del file indice, quindi $C_{\text{med}} = I / 2 + 1 = 201$.

File hash

Si consideri un file di 72.000 record di 80 byte ciascuno, con un campo chiave di 16 byte. Supponendo di disporre di blocchi di 512 byte con indirizzi di 4 byte, e di non spezzare mai un record su due blocchi. Costruire un'organizzazione hash usando una funzione hash con codominio di cardinalità pari a pari 1000;

- a) Calcolare la lunghezza media delle liste di trabocco.
- b) Calcolare il costo medio di accesso.
- c) Calcolare quanto occorre che sia la cardinalità del codominio della funzione hash per garantire un costo di accesso medio pari a 1.

File hash

a)

a) Calcolare la lunghezza media delle liste di trabocco.

- Una funzione hash con cardinalità del codominio pari ad **S** suddivide i record del file dati in **S** sottoinsiemi. Nella fattispecie in $S = 1000$ sottoinsiemi.
- Ogni sottoinsieme contiene, in media $n / S = 72.000 / 1000 = 72$ record.
- I record di ciascun sottoinsieme vengono impaccati in una sequenza di blocchi che costituisce una *lista di trabocco*.
- Il fattore di impaccamento è, al solito, $R = \lfloor B/r \rfloor = \lfloor 512/80 \rfloor = 6$ record/blk
- Una lista di trabocco è composta da $L = \lceil (n / S) / R \rceil = F / S$ blocchi. Nella fattispecie $L = F / S = 12.000 / 1000 = 12$ blocchi.

File hash

b)

b) Calcolare il costo medio di accesso.

– L'accesso consta di due fasi:

I. Calcolo della funziona hash: determina in quale lista di trabocco occorre effettuare al ricerca

II. Ricerca del record: viene effettuata sulla lista di trabocco individuata tramite ricerca sequenziale

– Il costo medio di accesso, nell'ipotesi di probabilità uniforme di accesso è pari alla metà della lunghezza media di una lista di trabocco:

$$C_{\text{med}} = F / (2 \cdot S) = 12.000 / (2 \cdot 1.000) = 6$$

– Il costo massimo di accesso è pari alla lunghezza media di una lista di trabocco:

$$C_{\text{max}} = F / S = 12.000 / 1.000 = 12$$

File hash

b)

c) Calcolare quanto occorre che sia la cardinalità del codominio della funzione hash per garantire un costo di accesso medio pari a 1.

– Occorre scegliere **S** in modo che:

$$C_{\text{med}} = F / (2 \cdot S) = 12.000 / (2 \cdot S) = 1$$

– Quindi risolvendo per S, si ottiene:

$$S = 12.000 / 2 = 6.000$$

i-node

In un sistema Unix si consideri un file di 400.000 record di 90 byte ciascuno, con un campo chiave di 40 byte, sul quale è stato costruito un indice ISAM. Supponendo di disporre di blocchi di 256 byte con indirizzi di 4 byte, e di non spezzare mai un record su due blocchi:

- a) Calcolare quanti byte occupano rispettivamente il file dati ed il file indice
- b) Calcolare la frazione percentuale di spazio inutilizzato nel file dati e nel file indice
- c) Calcolare quanti blocchi a ciascun livello di indizione dell'i-node occupano rispettivamente il file indice ed il file dati

i-node

a)

a) Calcolare quanti byte occupano rispettivamente il file dati ed il file indice

- Con i consueti ragionamenti: $R = \lfloor B/r \rfloor = \lfloor 256 / 90 \rfloor = 2$ record/blk.
- Pertanto il file dati occupa $F = \lceil n/R \rceil = \lceil 400.000 / 2 \rceil = 200.000$ blocchi, e quindi complessivamente $F \cdot 256 = 200.000 \cdot 256$ byte ≈ 50 Mbyte
- Per il file indice: $R_1 = \lfloor B / (c+b) \rfloor = \lfloor 256 / 44 \rfloor = 5$ rec/blk
- Quindi il file indice occupa $I = \lceil F/R_1 \rceil = \lceil 200.000 / 5 \rceil = 40.000$ blocchi, e quindi complessivamente $I \cdot 256 = 40.000 \cdot 256$ byte ≈ 10 Mbyte.

b) Calcolare la frazione percentuale di spazio inutilizzato nel file dati e nel file indice

- Ciascun blocco del file dati contiene $R = 2$ record.
- Pertanto sono utilizzati $R \cdot r = 2 \cdot 90 = 180$ byte. Ne restano quindi inutilizzati $B - R \cdot r = 256 - 180 = 76$ byte.

- La frazione di spazio inutilizzata nei blocchi del file dati è quindi:

$$(B - R \cdot r) / B = 76 / 256 \approx 0.3 = 30\%$$

- Ciascun blocco del file indice contiene $R_1 = 5$ record.
- Pertanto sono utilizzati $R_1 \cdot (c+b) = 5 \cdot 44 = 220$ byte. Ne restano quindi inutilizzati $B - R_1 \cdot (c+b) = 256 - 220 = 36$ byte.

- La frazione di spazio inutilizzata nei blocchi del file dati è quindi:

$$(B - R_1 \cdot (c+b)) / B = 36 / 256 \approx 0.15 = 15\%$$

i-node

c)

- c) Calcolare quanti blocchi a ciascun livello di indirizzatura dell'i-node occupano rispettivamente il file indice ed il file dati
- Direttamente l'i-node indirizza **10** blocchi.
 - Al I livello di indirizzatura indirizza $256/4 = 64 = 2^6$ blocchi
 - Al II livello di indirizzatura indirizza $2^6 \cdot 2^6 = 2^{12} = 4 \text{ K}$ blocchi
 - Al III livello di indirizzatura indirizza $2^6 \cdot 2^6 \cdot 2^6 = 2^{18} = 256 \text{ K}$ blocchi
 - Per il file dati $4 \text{ K} \leq 200.000 \leq 256 \text{ K}$ occorre andare al III livello
 - Per il file indice $4 \text{ K} \leq 40.000 \leq 256 \text{ K}$ occorre andare al III livello

Frammentazione dei file

Si consideri un file di 64 KB, allocato in blocchi da 2 KB su di un disco con:

- Seek time medio 5 ms
- Rotazione 3.000 giri/min
- Tracce da 32 KB
- Cilindri da 256 KB

Supponendo che il disco usi scheduling FIFO, calcolare quanto tempo occorre in media per caricare il file in memoria nei due seguenti casi:

- File frammentato: tutti i blocchi si trovano su cilindri diversi
- File deframmentato: i blocchi occupano settori consecutivi

Caratteristiche del disco

- Tempo medio di seek : $t_{\text{seek}} = 5$ ms
- Velocità di rotazione **3.000** giri/min = **50** giri/s
- Tempo di rotazione: $T_{\text{rot}} = 1/50 = 0.02$ s = **20** ms
- Tempo di latency medio: $t_{\text{lat}} = T_{\text{rot}} / 2 = 20 / 2 = 10$ ms
- Tempo di accesso medio: $t_{\text{acc}} = t_{\text{seek}} + t_{\text{lat}} = 5 + 10 = 15$ ms
- Dato che una rotazione prende **20** ms e che una traccia (letta in una rotazione) contiene **32** Kbyte, ne consegue che al velocità di trasferimento è : $V = 32 \text{ Kbyte} / T_{\text{rot}} = 32 \text{ K} / 0.02 = 1.6 \text{ Mbyte /sec}$

File frammentato

- Il file di 64 KB, allocato in blocchi da 2 KB, occupa 32 blocchi.
- Essendo il file frammentato, nel caso peggiore, due blocchi successivi non sono mai contigui
- La lettura del file comporta complessivamente:
 - 32 tempi di accesso: $T_A = 32 \cdot t_{acc} = 32 \cdot 15 \text{ ms} = 480 \text{ ms} = 0.48 \text{ s}$
 - Il trasferimento di 64 Kbyte: $T_T = 64 \text{ Kbyte} / 1.6 \text{ Mbyte /sec} = 40 \text{ ms}$
- Complessivamente:

$$T_{read} = T_A + T_T = 0.48 \text{ s} + 40 \text{ ms} = 0.52 \text{ s}$$

File deframmentato

- Il file di 64 KB, allocato in blocchi da 2 KB, occupa 32 blocchi.
- Essendo il file deframmentato possiamo supporre che tutti i blocchi siano contigui, e sullo stesso cilindro
- La lettura del file comporta complessivamente:
 - Un solo tempo di accesso: $T_A = t_{acc} = 15$ ms
 - Il trasferimento di 64 Kbyte: $T_T = 64 \text{ Kbyte} / 1.6 \text{ Mbyte /sec} = 40$ ms
- Complessivamente:

$$T_{read} = T_A + T_T = 15 \text{ ms} + 40 \text{ ms} = 55 \text{ ms}$$