ESERCIZI DI BASI DI DATI
Livello Fisico
(draft)

prof. Angelo Montanari
Michela Zamparini
Alberto Guglielmetti

12 marzo 2001

Indice

Utili riferimenti bibliografici per lo svolgimento degli esercizi relativi alla modellizzazione fisica sono i capitoli dedicati al modello fisico del libro *Fundamentals of Database Systems (3rd Edition)*, R. Elmasri and S. Navathe, Addison-Wesley, 2000, e il libro *Costruire sistemi per basi di dati*, A. Albano, Addison-Wesley Italia, 2001.

1 Indici di singolo livello e indici multilivello statici

Esercizio 1.1 Si descriva la tecnica di hashing lineare. In particolare, si spieghi perchè e come essa consente di espandere dinamicamente i file e si illustri la relativa procedura di ricerca.

Esercizio 1.2 Illustrare il legame esistente tra l'ordinamento fisico del file di dati e la presenza di indici primari, di cluster e/o secondari. In particolare, si risponda alle seguenti domende, motivando adeguatamente le risposte:

(i) Quali indici si possono associare ad un file ordinato? Sotto quali condizioni?

(ii) Quali indici si possono associare ad un file non ordinato? Sotto quali condizioni?

Esercizio 1.3 Si illustrino le differenze tra record di lunghezza fissa e record di lunghezza variabile. In particolare, si illustrino le ragioni alla base dell'uso di record di lunghezza variabile e le relative modalità di memorizzazione.

Esercizio 1.4 Sia dato un file contenente $r = 200000$ record, memorizzati su un disco con dimensione del blocco $B = 1024$ byte. I record abbiamo lunghezza fissa $R = 100$ byte e siano di tipo unspanned. La dimensione del campo chiave sia $V1 = 10$ byte; la dimensione del puntatore ad un blocco sia $P = 5$ byte.

(a) Si assuma che il file non sia ordinato rispetto ad alcun campo. Determinare il numero medio di accessi a blocco richiesto da una ricerca basata sulla chiave primaria.

(b) Si assuma che il file sia ordinato rispetto alla chiave primaria. Determinare il numero medio di accessi a blocco richiesto da una ricerca basata sulla chiave primaria.

(c) Si assuma che il file sia ordinato rispetto alla chiave primaria. Determinare il numero medio di accessi a blocco richiesto da una ricerca con indice primario.

(d) Si assuma che il file sia ordinato rispetto alla chiave primaria. Determinare il numero medio di accessi a blocco richiesto da una ricerca con indice secondario costruito su una chiave candidata (diversa dalla chiave primaria) di dimensione $V2 = 10$ byte.

Soluzione. Calcoliamo lo spazio occupato dal file sul disco.
Il blocking factor è:
$bfr = \lfloor B/R \rfloor = \lfloor 1024/100 \rfloor = \lfloor 10.24 \rfloor = 10$ record per ogni blocco.
Il numero di blocchi necessario a contenere l'intero file è:
$nb = \lceil r/bfr \rceil = \lceil 200000/10 \rceil = 20000$.

(a) Non essendoci alcun ordinamento, è necessaria una ricerca lineare. Il numero medio di accessi al disco è:

$nb/2 = 10000$.

(b) In questo caso si può fare una ricerca binaria. Il numero di accessi diventa:

$\lceil \log_2 nb \rceil = \lceil \log_2 20000 \rceil = \lceil 14.28 \rceil = 15$,

(c) Costruiamo l'indice primario. Ogni index entry ha dimensione: $R_{i1} = V1 + P = 10 + 5 = 15$ byte.

Il blocking factor dell'indice è:

$bfr_{i1} = \lfloor B/R_{i1} \rfloor = \lfloor 1024/15 \rfloor = \lfloor 68.26 \rfloor = 68$ index entry per ogni blocco.

Il numero totale di index entry è uguale al numero di blocchi del file: $r_{i1} = nb = 20000$.

Il numero di blocchi necessario per l'indice è:

$nb_{i1} = \lceil r_{i1}/bfr_{i1} \rceil = \lceil 20000/68 \rceil = \lceil 294.11 \rceil = 295$.

Dunque, una ricerca binaria sull'indice primario richiede in media:

$\lceil \log_2 nb_{i1} \rceil + 1 = \lceil \log_2 295 \rceil + 1 = \lceil 8.20 \rceil + 1 = 9 + 1 = 10$ accessi,

di cui nove per la ricerca nell'indice ed uno per l'accesso al blocco di dati cercato.

(d) Costruiamo l'indice secondario. Ogni index entry ha dimensione: $R_{i2} = V2 + P = 10 + 5 = 15$ byte.

Il blocking factor dell'indice è:

$bfr_{i2} = \lfloor B/R_{i2} \rfloor = \lfloor 1024/15 \rfloor = \lfloor 68.26 \rfloor = 68$ index entry per ogni blocco.

L'indice è denso perché ha un'entry per ogni record di dati: $r_{i2} = r = 200000$. Il numero di blocchi di indice è allora:

$nb_{i2} = \lceil r_{i2}/bfr_{i2} \rceil = \lceil 200000/68 \rceil = \lceil 2941.17 \rceil = 2942$.

In questo caso una ricerca binaria sull'indice secondario richiederà in media:

$\lceil \log_2 nb_{i2} \rceil + 1 = \lceil \log_2 2942 \rceil + 1 = \lceil 11.52 \rceil + 1 = 12 + 1 = 13$ accessi.

□

Esercizio 1.5 Sia dato un file ordinato con $r = 200000$ record memorizzati su un disco con dimensioni del blocco $B = 2056$ byte. I record abbiano lunghezza fissa $R = 400$ byte e siano di tipo unspanned. La dimensione del campo chiave primaria sia $V1 = 20$ byte, mentre la dimensione del campo chiave candidata (diversa dalla chiave primaria) sia $V2 = 15$ byte. La dimensione del puntatore a blocco sia $P = 5$ byte.

(a) Si assuma che il file sia ordinato rispetto al campo chiave primaria. Determinare il numero di accessi a blocco richiesti rispettivamente da una ricerca con indice primario e da una ricerca con indice secondario sulla chiave candidata (diversa dalla chiave primaria).

(b) Si assuma che il file non sia ordinato rispetto ad alcun campo. Determinare il numero di accessi a blocco richiesti rispettivamente da una ricerca con indice secondario sulla chiave primaria e da una ricerca con indice secondario sulla chiave candidata (diversa dalla chiave primaria).

(c) Confrontare tra loro le soluzioni proposte ai punti (a) e (b) e discuterne vantaggi e svantaggi.

Soluzione. Per i dettagli si veda la soluzione dell'esercizio 1.4.

(a) Il file è ordinato rispetto al campo chiave. Calcoliamo i blocking factor:

$bfr = \lfloor B/R \rfloor = \lfloor 2056/400 \rfloor = \lfloor 5.14 \rfloor = 5$ record per ogni blocco;

$bfr_{i1} = \lfloor B/R_{i1} \rfloor = \lfloor 2056/25 \rfloor = \lfloor 82.24 \rfloor = 82$ entry per ogni blocco;

$bfr_{i2} = \lfloor B/R_{i2} \rfloor = \lfloor 2056/20 \rfloor = \lfloor 102.8 \rfloor = 102$ entry per ogni blocco.

Quindi il numero di blocchi necessarî:

$nb = \lceil r/bfr \rceil = \lceil 200000/5 \rceil = \lceil 40000 \rceil = 40000;$

$nb_{i1} = \lceil r_{i1}/bfr_{i1} \rceil = \lceil 40000/82 \rceil = \lceil 487.80 \rceil = 488;$

$nb_{i2} = \lceil r_{i2}/bfr_{i2} \rceil = \lceil 200000/102 \rceil = \lceil 1960.78 \rceil = 1961.$

Una ricerca binaria sull'indice primario richiede in media:

$\lceil \log_2 nb_{i1} \rceil + 1 = \lceil \log_2 488 \rceil + 1 = \lceil 8.93 \rceil + 1 = 9 + 1 = 10$ accessi,

mentre una ricerca binaria sull'indice secondario (denso) richiede in media:

$\lceil \log_2 nb_{i2} \rceil + 1 = \lceil \log_2 1961 \rceil + 1 = \lceil 10.93 \rceil + 1 = 11 + 1 = 12$ accessi.

(b) Il file non è ordinato, quindi è possibile utilizzare solo gli indici secondarî.

$nb'_{i1} = \lceil 200000/82 \rceil = \lceil 2439.02 \rceil = 2440$ blocchi;

$nb'_{i2} = \lceil 200000/102 \rceil = 1961$ blocchi.

Una ricerca mediante indice secondario costruito sulla chiave primaria richiede in media:

$\lceil \log_2 nb'_{i1} \rceil + 1 = \lceil \log_2 2440 \rceil + 1 = \lceil 11.25 \rceil + 1 = 12 + 1 = 13$ accessi,

invece una ricerca basata su indice secondario costruito su un'altra chiave candidata richiede:

$\lceil \log_2 nb'_{i2} \rceil + 1 = \lceil \log_2 1961 \rceil + 1 = 12$ accessi in media.

(c) L'ordinamento del file di dati proposto al punto (a) permette l'uso dell'indice primario il quale, essendo non denso, prevede un'entry per ogni blocco di dati. Il file di indice ha dimensioni ridotte e richiede un numero di accessi limitato. Il prezzo da pagare è la necessità di mantenere ordinato il file di dati a fronte di inserimenti, cancellazioni e modifiche del campo ordinante (la chiave primaria).

Per un file di dati non ordinato, come quello del punto (b), i pro e i contro sono invertiti: aumentano il numero medio di accessi e le dimensioni dell'indice, ma non serve mantenere un ordinamento nel file. Visto il crollo dei prezzi delle memorie di massa avvenuto in questi ultimi anni, molte volte si preferisce l'uso di più indici secondari e nessun indice primario o di clustering.

□

Esercizio 1.6 Sia dato un file contenente $r = 100000$ record, memorizzati su un disco con dimensione del blocco $B = 1024$ byte. I record abbiamo lunghezza fissa $R = 100$ byte e siano di tipo unspanned. Si assuma che il file sia ordinato rispetto ad un campo chiave k. La dimensione del campo k sia $V1 = 15$ byte; la dimensione del puntatore ad un blocco sia $P = 5$ byte.

(a) Determinare il numero medio di accessi a blocco richiesto da una ricerca basata sulla chiave ordinante k.

(b) Determinare il numero medio di accessi a blocco richiesto da una ricerca con indice primario.

(c) Determinare il numero medio di accessi a blocco richiesto da una ricerca con indice secondario costruito su una chiave candidata (diversa dalla chiave k) di dimensione $V2 = 10$ byte.

Soluzione. Per i dettagli fare riferimento alla soluzione dell'esercizio 1.4.

(a) Una ricerca binaria sul file ordinato richiede in media:

$\lceil \log_2 nb \rceil = \lceil \log_2 10000 \rceil = 14$ accessi.

(b) Una ricerca binaria sull'indice primario richiede in media:

$\lceil \log_2 nb_{i1} \rceil + 1 = \lceil \log_2 197 \rceil + 1 = 9$ accessi.

(c) Infine, una ricerca binaria sull'indice secondario richiede in media:

$\lceil \log_2 nb_{i2} \rceil + 1 = \lceil \log_2 1471 \rceil + 1 = 12$ accessi.

□

Esercizio 1.7 Sia dato un file ordinato contenente $r = 100000$ record memorizzati su un disco con dimensioni del blocco $B = 1024$ byte. I record abbiano lunghezza fissa $R = 200$ byte e siano di tipo unspanned. La dimensione del campo chiave sia $V1 = 10$ byte. Il puntatore ad un blocco sia lungo $P = 5$ byte.

(a)] Si assuma che il file sia ordinato rispetto al campo chiave. Determinare il numero di accessi a blocco richiesti rispettivamente da una ricerca binaria e da una ricerca con indice primario.

(b) Si assuma che il file sia ordinato rispetto ad un campo non chiave di dimensione $V2 = 20$ byte. Determinare il numero di accessi a blocco richiesti rispettivamente da una ricerca lineare e da una ricerca con indice secondario denso costruito su un campo chiave di dimensione $V3 = 15$ byte.

(c) Si consideri l'indice secondario denso definito al punto precedente e lo si converta in un indice multilivello. Calcolare il numero di livelli dell'indice, il numero di blocchi per ogni livello ed il numero di accessi necessario per recuperare un blocco del file di dati.

Soluzione. Per i dettagli si faccia riferimento alla soluzione dell'esercizio 1.4.

(a) Una ricerca binaria sul file ordinato richiede in media:

$\lceil \log_2 nb \rceil = \lceil \log_2 20000 \rceil = 15$ accessi,

una ricerca binaria sull'indice primario invece richiede in media:

$\lceil \log_2 nb_i \rceil + 1 = \lceil \log_2 295 \rceil + 1 = 10$ accessi al disco.

(b) L'informazione su $V2$ è irrilevante. Una ricerca lineare comporta sempre un numero medio di accessi al disco pari a:

$nb/2 = 10000$,

mentre una ricerca binaria su un indice secondario denso necessita mediamente di:

$\lceil \log_2 nb_i \rceil + 1 = \lceil \log_2 1961 \rceil + 1 = 12$ accessi.

(c) Convertiamo l'indice secondario del punto (b). Il blocking factor bfr_i è anche il *fan-out* dell'indice multilivello:

$f_o = bfr_i = 51$.

Al *primo livello* ci sono i blocchi del vecchio indice, che sono $nb_1 = 1961$ e sono ordinati.

Al *secondo livello* si potrà usare un indice primario (con le ancore):

$nb_2 = \lceil nb_1/f_o \rceil = \lceil 1961/51 \rceil = \lceil 38.45 \rceil = 39$ blocchi;

analogamente per il *terzo livello*:

$nb_3 = \lceil nb_2/f_o \rceil = \lceil 39/51 \rceil = \lceil 0.76 \rceil = 1$ blocco.

Il terzo livello è il *top level* dell'indice e quindi $t = 3$.

Per accedere ai dati servono solo *quattro* accessi: tre all'indice (uno per ogni livello) ed uno al blocco dei dati.

□

2 Indici multilivello dinamici

Esercizio 2.1 Si consideri un disco con dimensione del blocco $B = 1024$ byte. Si consideri un B-albero che abbia come campo di ricerca un campo chiave non ordinante di dimensione $V = 9$ byte, dimensione dei puntatori ai record di dati $P_r = 7$ byte e dimensione dei puntatori ai blocchi $P = 6$ byte.

(a) Si calcoli l'ordine p del B-albero che massimizza l'occupazione dei nodi.

(b) Assumendo che ogni nodo del B-albero sia pieno al 70%, determinare il numero medio di entry (livello per livello e totale) di un B-albero a 2 livelli e di un B-albero a 3 livelli.

Si immagini ora di sostituire il B-albero con un B^+-albero.

(c) Si calcolino gli ordini p e p_{leaf} dei nodi interni e dei nodi foglia del B^+-albero, rispettivamente, che massimizzano l'occupazione dei nodi.

(d) Assumendo che ogni nodo del B^+-albero sia pieno al 70%, determinare il numero medio di entry di un B^+-albero a 2 livelli e di un B^+-albero a 3 livelli.

Soluzione.

(a) Ogni nodo del B-albero può contenere al più p *tree pointer* e $p-1$ entry, ognuna di esse costituita da un campo chiave di ricerca e da un *data pointer*. Ciascun nodo deve essere contenuto in un unico blocco:

$p * P + (p-1) * (V + P_r) \leq B$

$p * 6 + (p-1) * (9+7) \leq 1024$

$6p + 16p - 16 \leq 1024$

$22p \leq 1040$

Assegnamo a p il massimo valore che soddisfi la disuguaglianza propria (infatti è bene lasciare dello spazio per delle strutture dati ausiliarie):

$p = 47$.

(b) Assumendo un riempimento medio dell'albero dell'ordine del 70%, abbiamo:

$p_m = 0.70 * p = 32.9 \simeq 33$ puntatori ai figli, e 32 chiavi di ricerca per ogni nodo.

Radice:	1 nodo	32 entry	33 puntatori
Livello 1:	33 nodi	1056 entry	1089 puntatori
Livello 2:	1089 nodi	34848 entry	35937 puntatori
Livello 3:	35937 nodi	1149984 entry	

In totale:

2 livelli: 35936 entry;

3 livelli: 1185920 entry.

(c) Ogni nodo del B^+-albero può contenere al più p *tree pointer* e $p-1$ entry, ciascuna costituita da un campo chiave di ricerca. Ogni nodo deve essere contenuto in un unico blocco:

$p * P + (p-1) * V \leq B$

$p * 6 + (p-1) * 9 \leq 1024$

$6p + 9p - 9 \leq 1024$

$15p \leq 1033$

Scegliamo come valore di p il più alto valore che soddisfi la disuguaglianza propria: $p = 68$.

I nodi foglia del B^+-albero contengono al più p_{leaf} entry e un *next pointer* che punta alla foglia successiva. Ogni entry è costituita da un campo chiave di ricerca e da un data pointer. Ogni foglia deve essere contenuta in un unico blocco:

$p_{leaf} * (P_r + V) + P \leq B$

$p_{leaf} * (7+9) + 6 \leq 1024$

$16p_{leaf} \leq 1018$

Scegliamo $p_{leaf} = 63$.

(d) Assumendo un riempimento medio dell'albero dell'ordine del 70%, abbiamo:

$\overline{p} = 0.70 * p = 47.6 \simeq 48$ puntatori e 47 entry in media per ogni nodo,

$\overline{p_{leaf}} = 0.70 * p_{leaf} = 44.1 \simeq 44$ entry in media per ogni foglia.

- B^+-albero a un livello:

Radice:	1 nodo	47 entry	48 puntatori
Livello 1:	48 foglie	2112 entry	

- B^+-albero a due livelli:

Radice:	1 nodo	47 entry	48 puntatori
Livello 1:	48 nodi	2256 entry	2304 puntatori
Livello 2:	2304 foglie	101376 entry	

- B^+-albero a tre livelli:

Radice:	1 nodo	47 entry	48 puntatori
Livello 1:	48 nodi	2256 entry	2304 puntatori
Livello 2:	2304 nodi	108288 entry	110592 puntatori
Livello 3:	110592 foglie	4866048 entry	

□

Esercizio 2.2 Si consideri un disco con dimensione del blocco $B = 2048$ byte. Si consideri un B-albero che abbia come campo di ricerca un campo chiave non ordinante di dimensione $V = 15$ byte, dimensione dei puntatori ai record di dati $P_r = 12$ byte e dimensione dei puntatori ai blocchi $P = 9$ byte.

(a) Si calcoli l'ordine p del B-albero che massimizza l'occupazione dei nodi.

(b) Assumendo che ogni nodo del B-albero sia pieno al 65%, determinare il numero medio di entry (livello per livello e totale) di un B-albero a 2 livelli e di un B-albero a 3 livelli.

Si immagini ora di sostituire il B-albero con un B^+-albero.

(c) Si calcolino gli ordini p e p_{leaf} dei nodi interni e dei nodi foglia del B^+-albero, rispettivamente, che massimizzano l'occupazione dei nodi.

(d) Assumendo che ogni nodo del B^+-albero sia pieno al 60%, determinare il numero medio di entry di un B^+-albero a 2 livelli e di un B^+-albero a 3 livelli.

Soluzione. La soluzione è simile a quella dell'esercizio 2.1.

(a) Calcoliamo l'ordine p:

$p \cdot 9 + (p-1) \cdot 27 \leq 2048$

$p \cdot 36 \leq 2075 \Rightarrow p = 57.$

(b) Assumendo un riempimento medio dell'albero dell'ordine del 65%, abbiamo:

$\overline{p} = 0.65 * p = 37.05 \simeq 37$ puntatori ai figli, e 36 chiavi di ricerca per ogni nodo.

Radice:	1 nodo	36 entry	37 puntatori
Livello 1:	37 nodi	1332 entry	1369 puntatori
Livello 2:	1369 nodi	49284 entry	50653 puntatori
Livello 3:	50653 nodi	1823508 entry	

In totale:

2 livelli: 50652 entry;

3 livelli: 1874160 entry.

(c) Calcoliamo p e p_{leaf}:

$p * 9 + (p-1) * 15 \leq 2048$

$[\ldots] \Rightarrow p = 85.$

$p_{leaf} * 27 + 9 \leq 2048$

$[\ldots] \Rightarrow p_{leaf} = 75.$

(d) Assumendo un riempimento medio dell'albero dell'ordine del 60%, abbiamo:

$\overline{p} = 0.60 * 85 = 51,$

$\overline{p_{leaf}} = 0.60 * 75 = 45.$

- B^+-albero a un livello:

Radice:	1 nodo	50 entry	51 puntatori
Livello 1:	51 foglie	2295 entry	

- B^+-albero a due livelli:

Radice:	1 nodo	50 entry	51 puntatori
Livello 1:	51 nodi	2550 entry	2601 puntatori
Livello 2:	2601 foglie	117045 entry	

- B^+-albero a tre livelli:

Radice:	1 nodo	50 entry	51 puntatori
Livello 1:	51 nodi	2550 entry	2601 puntatori
Livello 2:	2601 nodi	130050 entry	132651 puntatori
Livello 3:	132651 foglie	5969295 entry	

$\square$