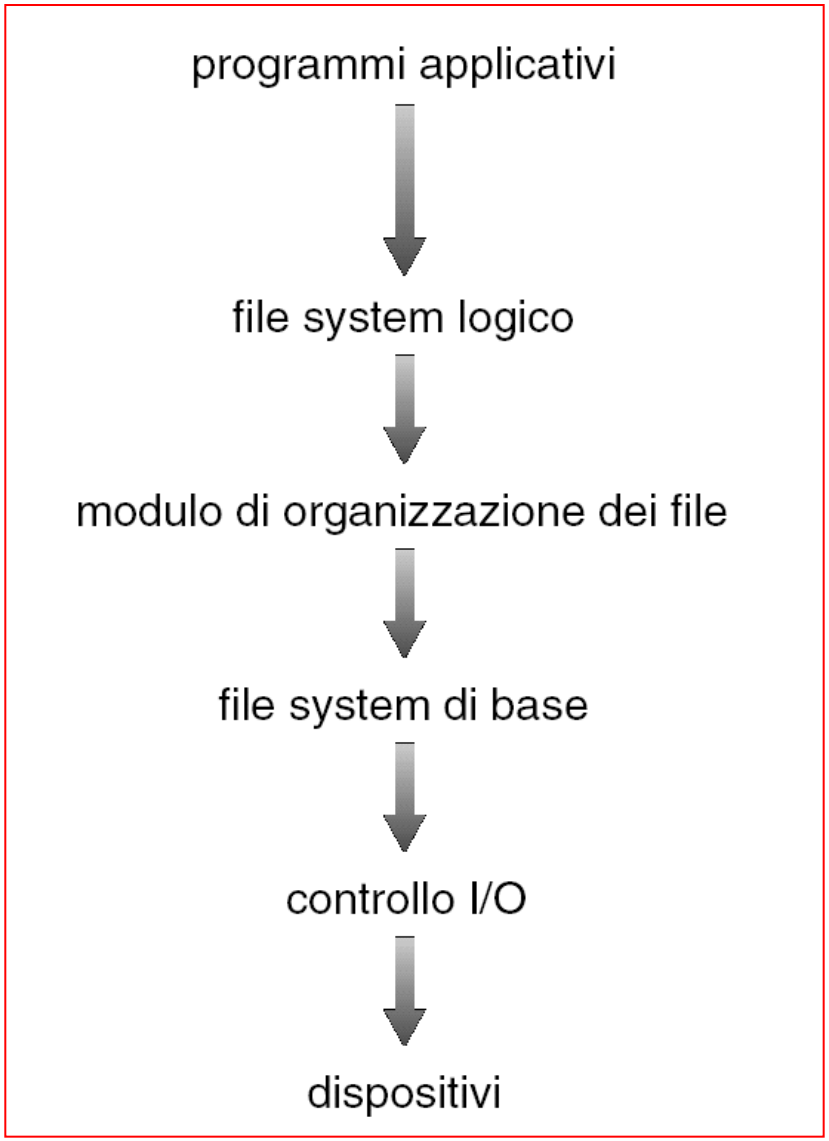
Capitolo 12: Implementazione del file system

- Struttura del file system.
- Realizzazione del file system.
- Realizzazione della directory.
- Metodi di allocazione.
- Gestione dello spazio libero.
- Efficienza e prestazioni.
- Recupero del file system.
- File system basato sulla registrazione delle attività.

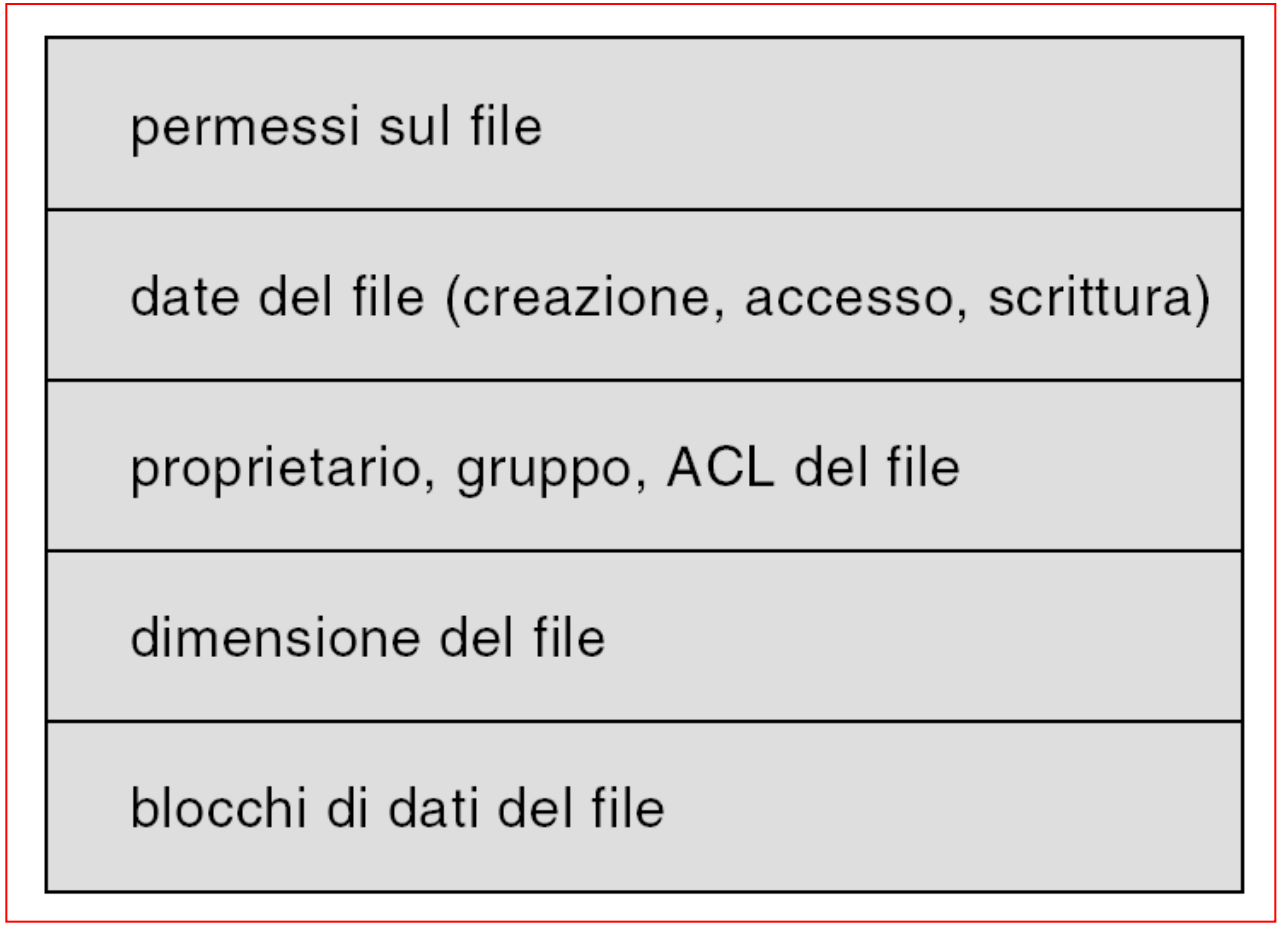
Struttura del file system

- Struttura del file:
  - Unità logica di memorizzazione.
  - Insieme di informazioni collegate.
- Il file system risiede in un'unità di memorizzazione secondaria (dischi).
- Il file system è organizzato in livelli.
- *Blocco di controllo di file* – struttura di memorizzazione che contiene informazioni sul file.

File System a strati



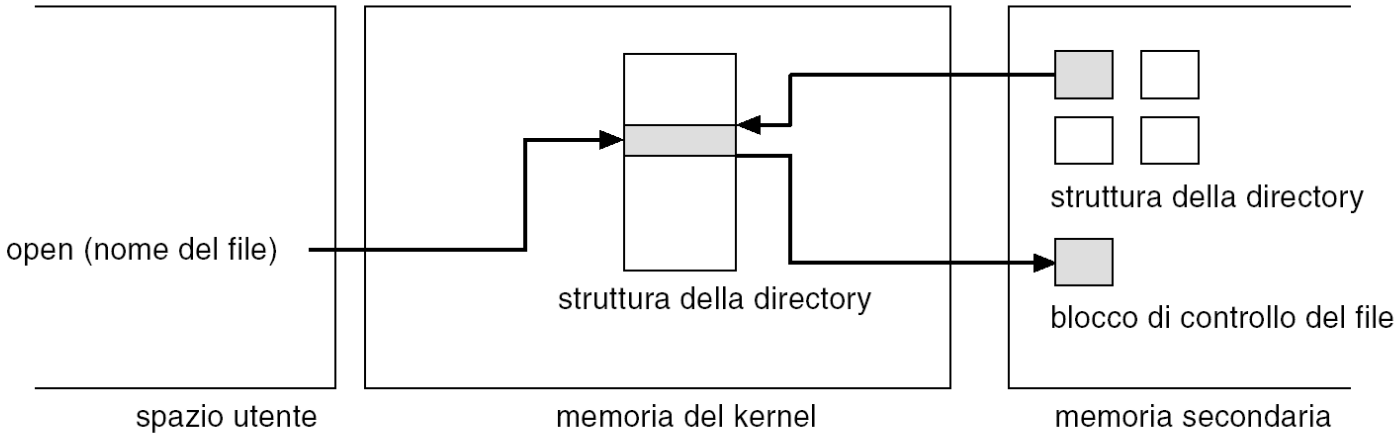
Un tipico blocco di controllo di file



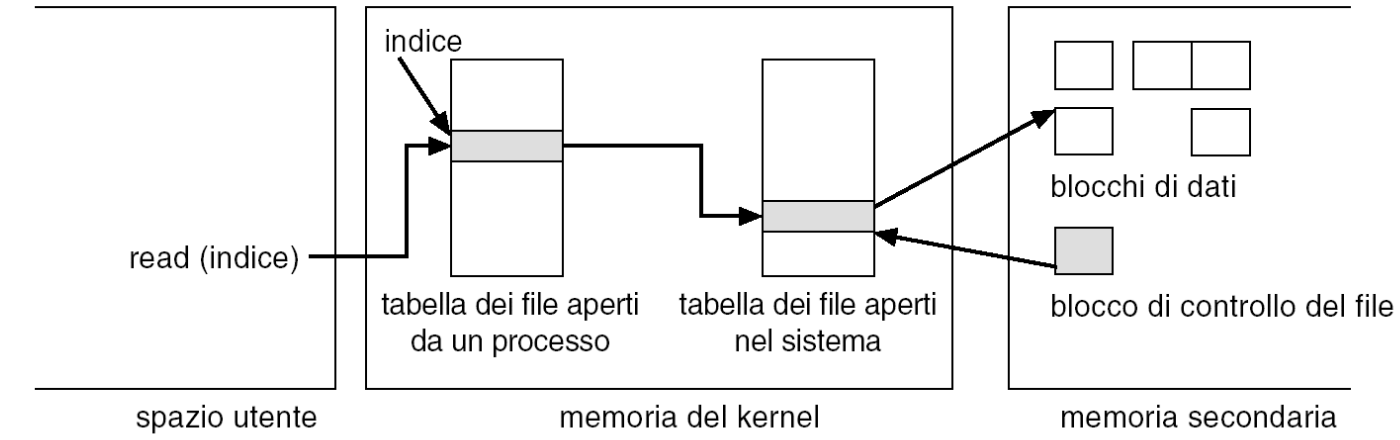
Strutture del file system in memoria

- Le seguenti figure illustrano le strutture operative di una implementazione di un file system.
- La figura 12-3(a) si riferisce all'apertura di un file.
- La figura 12-3(b) si riferisce alla lettura di un file.

Strutture del file system in memoria (Cont.)



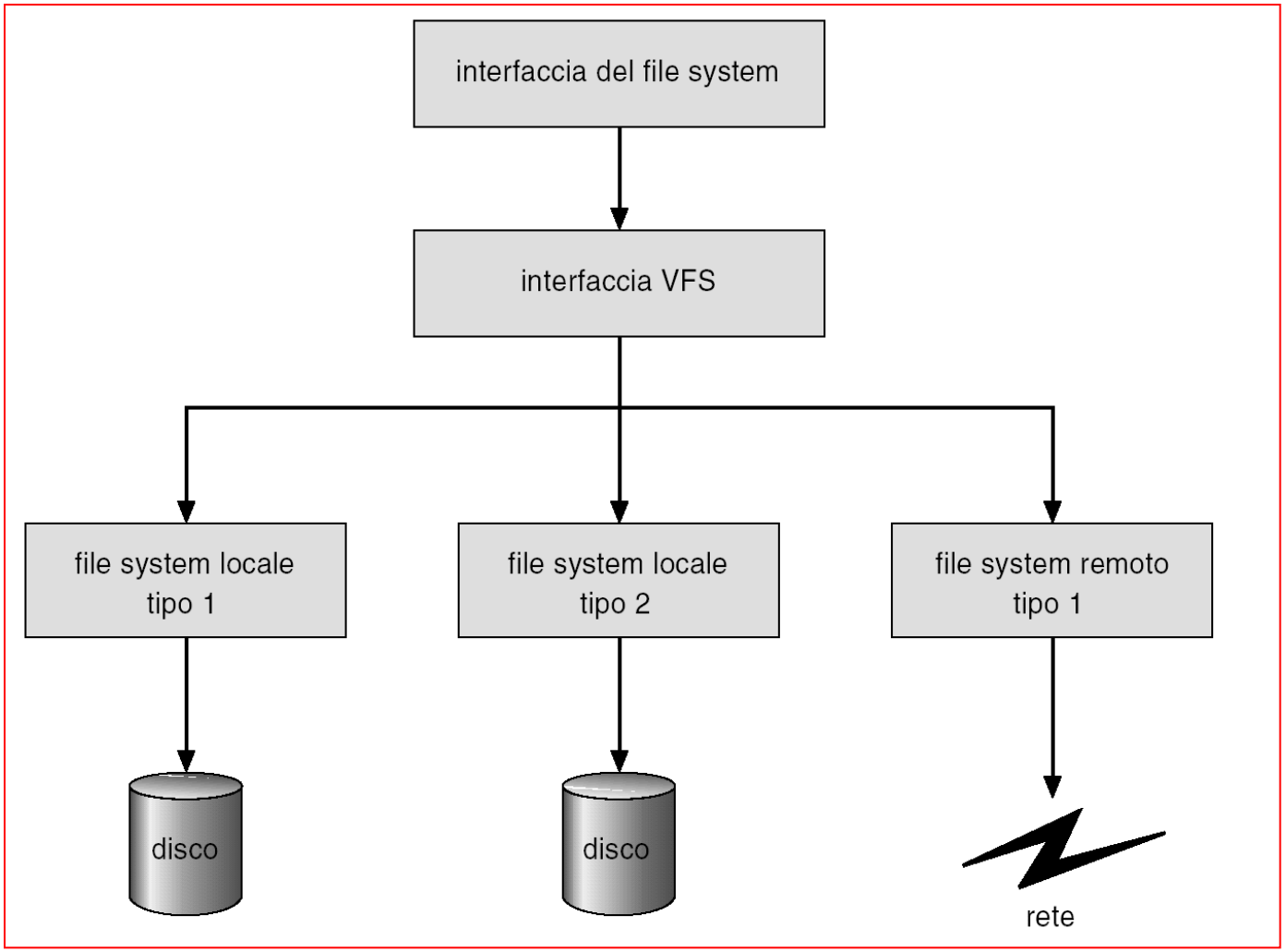
(a)



File System virtuali

- Il Virtual File Systems (VFS) utilizza una tecnica orientata agli oggetti per implementare il file system.
- VFS permette che la stessa interfaccia di sistema (API) sia usata per differenti tipi di file system.
- API è un'interfaccia di VFS piuttosto che uno specifico tipo di sistema operativo.

Vista schematica di un file system virtuale



Realizzazione delle directory

- Lista di nomi di file con puntatori ai blocchi dei dati:
  - semplice da programmare,
  - richiede tempo per l'esecuzione.

- Tabella di hash – metodo che utilizza una lista per memorizzare gli elementi della directory ed una tabella di hash:
  - diminuisce il tempo di ricerca nella directory;
  - *collisioni* – situazioni in cui due file sono identificate dalla stessa posizione generata dalla funzione di hash;
  - dimensione fissa.

Metodi di allocazione

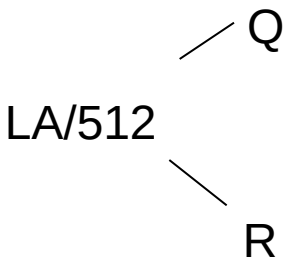
- Un metodo di allocazione indica come i file devono essere allocati ai blocchi del disco:
- Allocazione contigua.
- Allocazione collegata.
- Allocazione indicizzata.

Allocazione contigua

- Ogni file occupa un certo numero di blocchi contigui su disco.
- Facile – è definita dall'indirizzo del disco (blocco #) e dalla lunghezza (numero di blocchi).
- Accesso sequenziale e accesso diretto.
- Spreco di spazio (problema di allocazione dinamica della memoria).
- I file non possono essere ampliati.

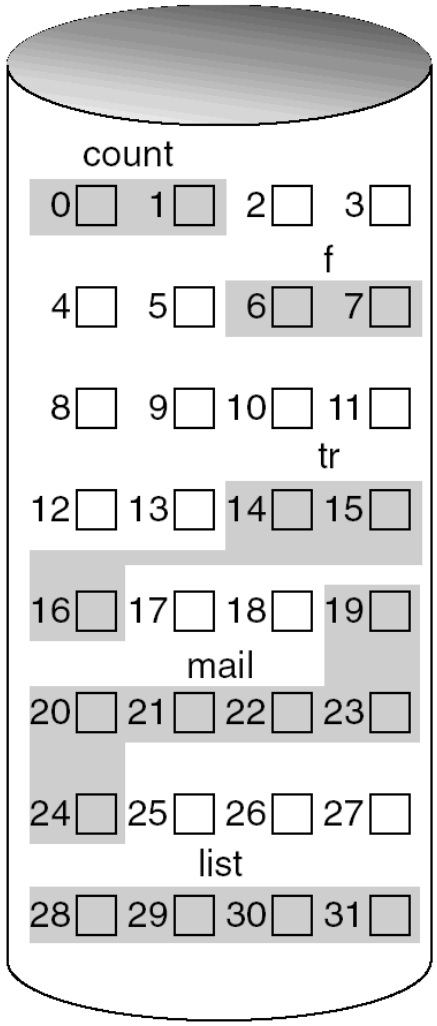
Allocazione contigua (Cont.)

- Mappatura da logico a fisico.



  - Blocco al quale accedere = ! + indirizzo del disco.
  - Spostamento nel blocco = R

Allocazione contigua dello spazio del disco



directory

file	inizio	lunghezza
count	0	2
tr	14	3
mail	19	6
list	28	4
f	6	2

Estensione

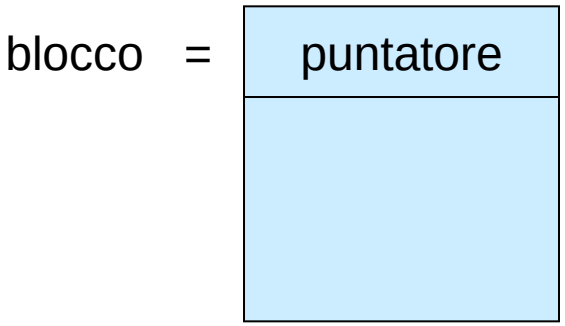
- Molti nuovi sistemi operativi (ad esempio Veritas File System) usano uno schema di allocazione contigua modificato.

- Schema cui viene inizialmente allocato un pezzo contiguo di spazio e poi, quando la quantità non è sufficientemente grande, si aggiunge un estensione.

- Un'**estensione** è un altro pezzo di spazio contiguo. Le estensioni sono allocate per l'allocazione dei file. Un file consiste in una o più estensioni.

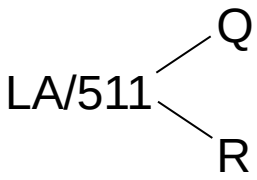
Allocazione collegata

- Ogni file è una lista collegata di blocchi del disco: i blocchi possono essere sparpagliati ovunque nel disco.



Allocazione collegata (Cont.)

- Facile – richiede solo l'indirizzo del disco.
- Libera gestione dello spazio – no sprechi di spazio.
- No accesso casuale.
- Mappatura.



Il blocco al quale accedere è il Qth blocco nella catena collegata di blocchi che rappresentano il file.
Spostamento nel blocco = R + 1

Tabella di allocazione dei file (FAT) – metodo di allocazione usato da MS-DOS e OS/2.

Allocazione collegata (Cont.)

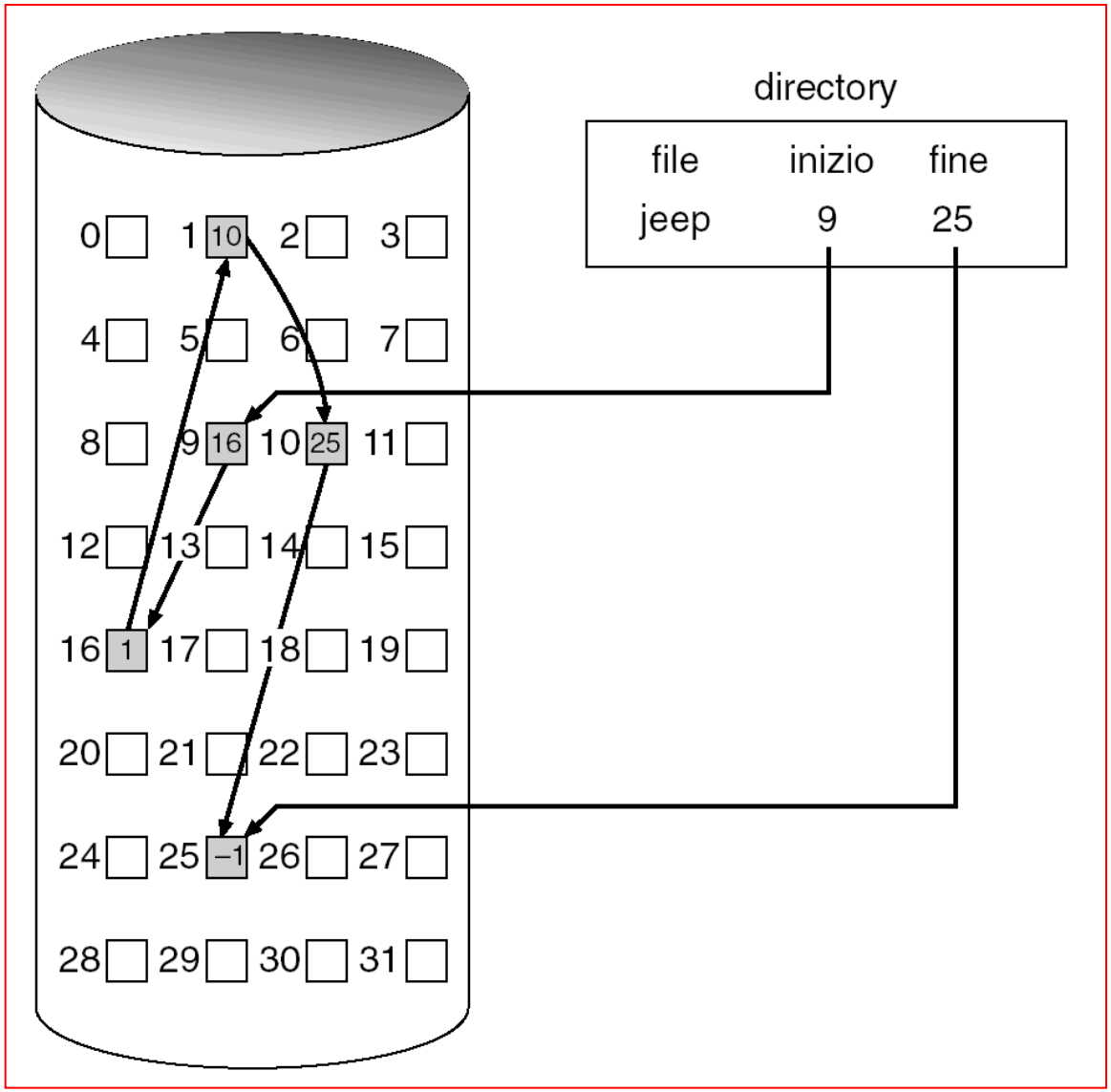
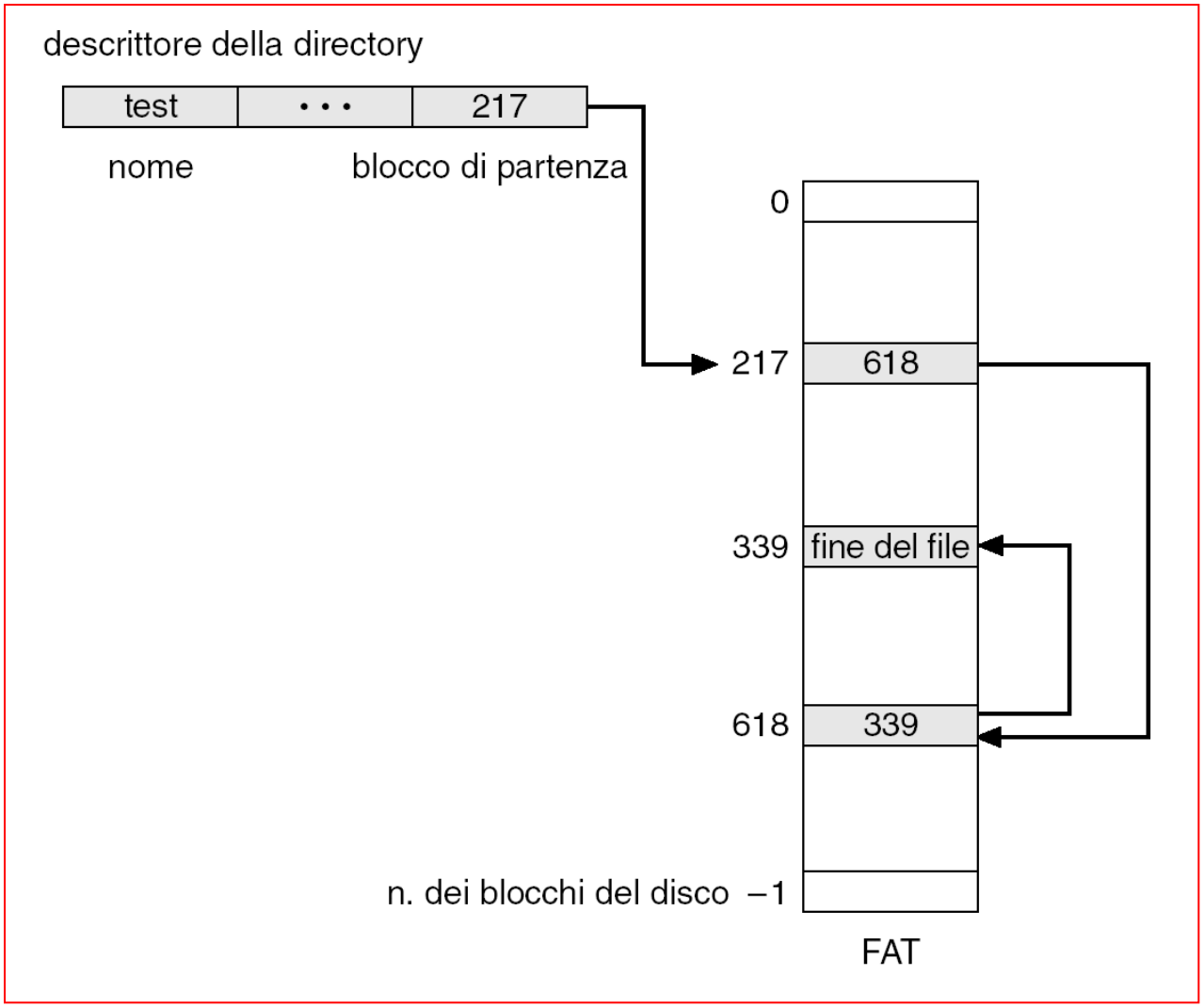


Tabella di allocazione dei file



Allocazione indicizzata

- Porta tutti puntatori nel *blocco indice.*
- Vista logica.

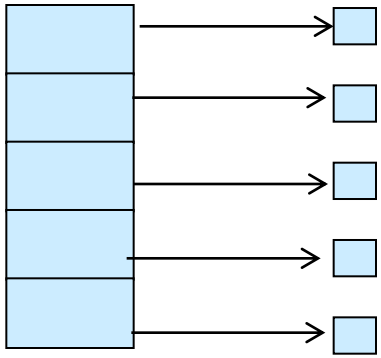
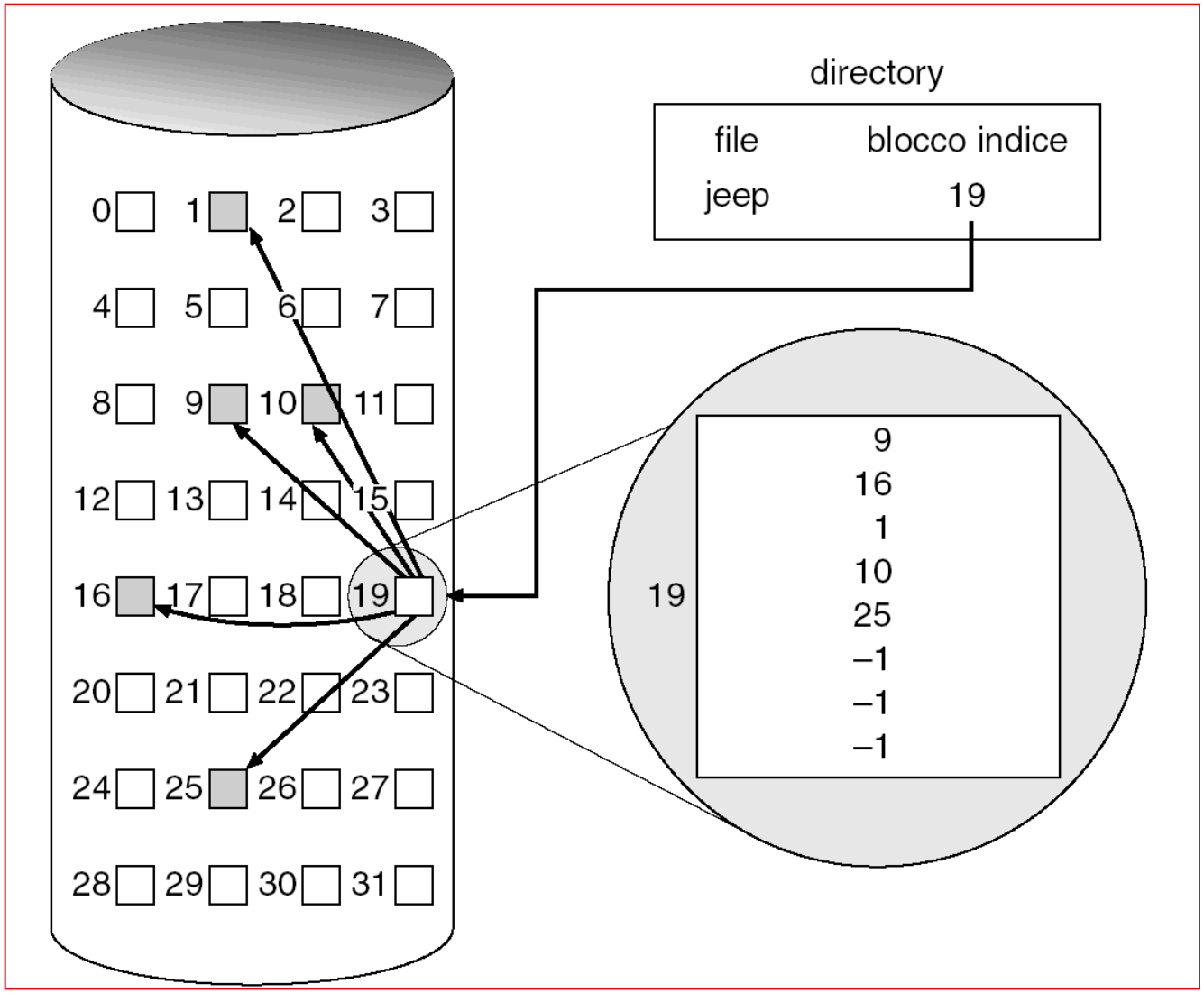


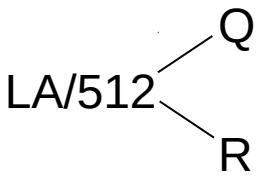
tabella indice

Esempio di allocazione indicizzata



Allocazione indicizzata (Cont.)

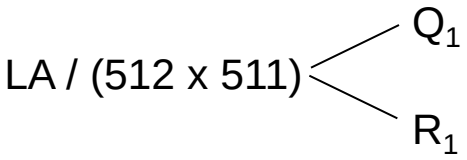
- Necessita di una tabella indice.
- Accesso casuale.
- Accesso diretto senza frammentazione esterna, ma ha overhead del puntatore al blocco indice.
- Mappatura da logico a fisico in un file della dimensione massima di 256K parole e blocco della dimensione di 512 parole. È necessario un solo blocco per la tabella indice.



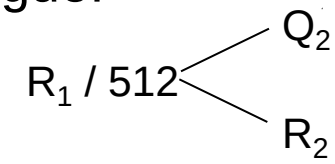
Q = spostamento nella tabella indice,
R = spostamento nel blocco.

Allocazione indicizzata – Mappatura (Cont.)

- Mappatura da logico a fisico in un file di lunghezza illimitata (dimensione blocco di 512 parole).
- Schema collegato – collega i blocchi alla tabella indice (nessun limite di dimensione).



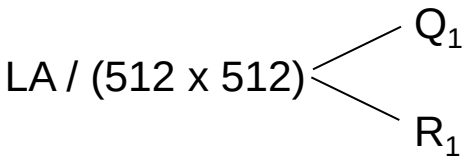
Q_1 = blocco della tabella indice.
R_1 è usato come segue:



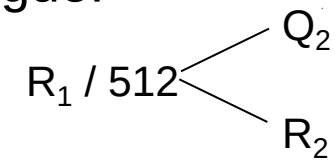
Q_2 = spostamento nel blocco della tabella indice.
R_2 spostamento nel blocco di file.

Allocazione indicizzata – Mappatura (Cont.)

- Indice multi-livello (la dimensione massima del file è 512^3).

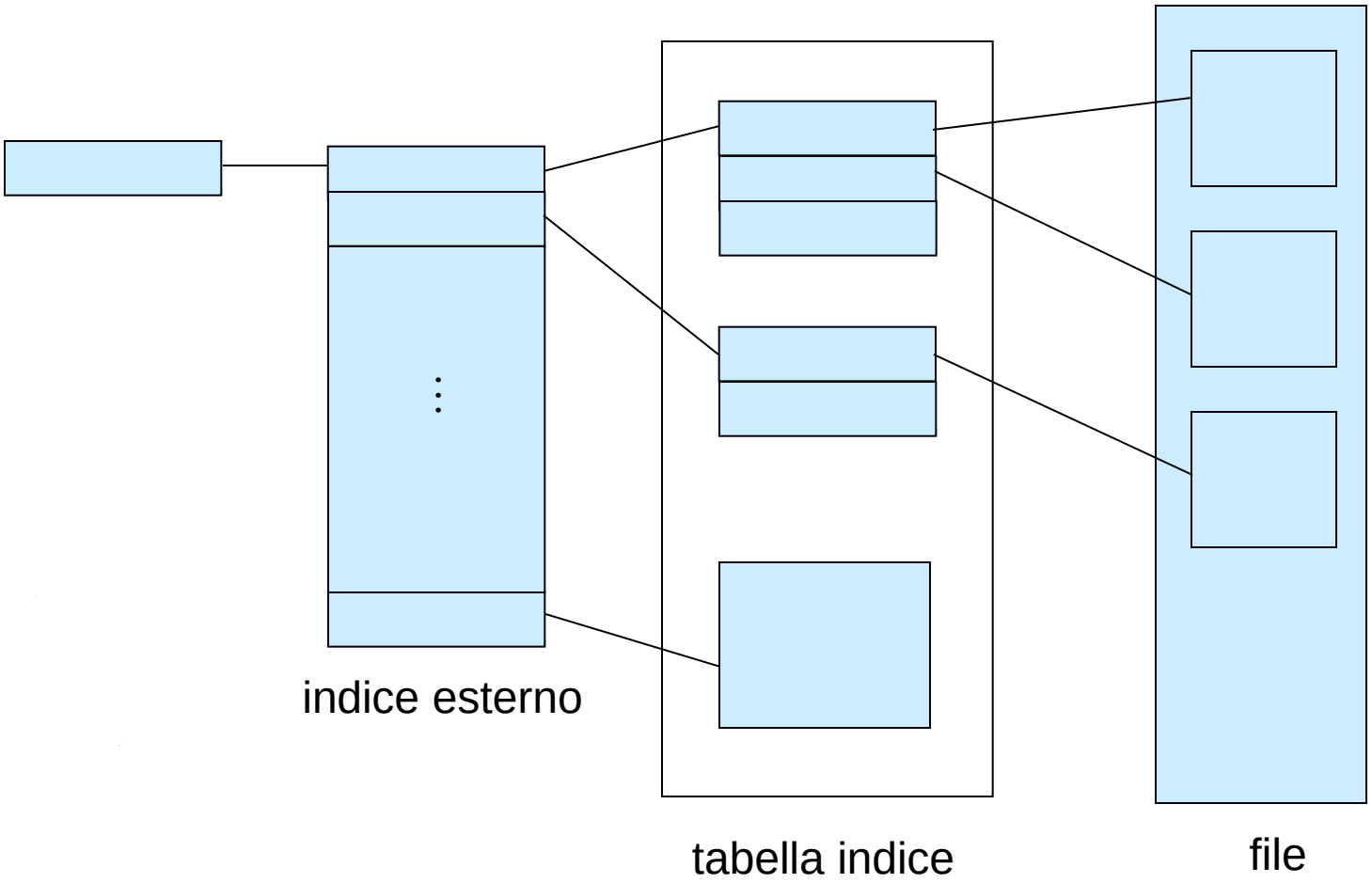


Q_1 = spostamento nell'indice esterno,
R_1 è usato come segue:

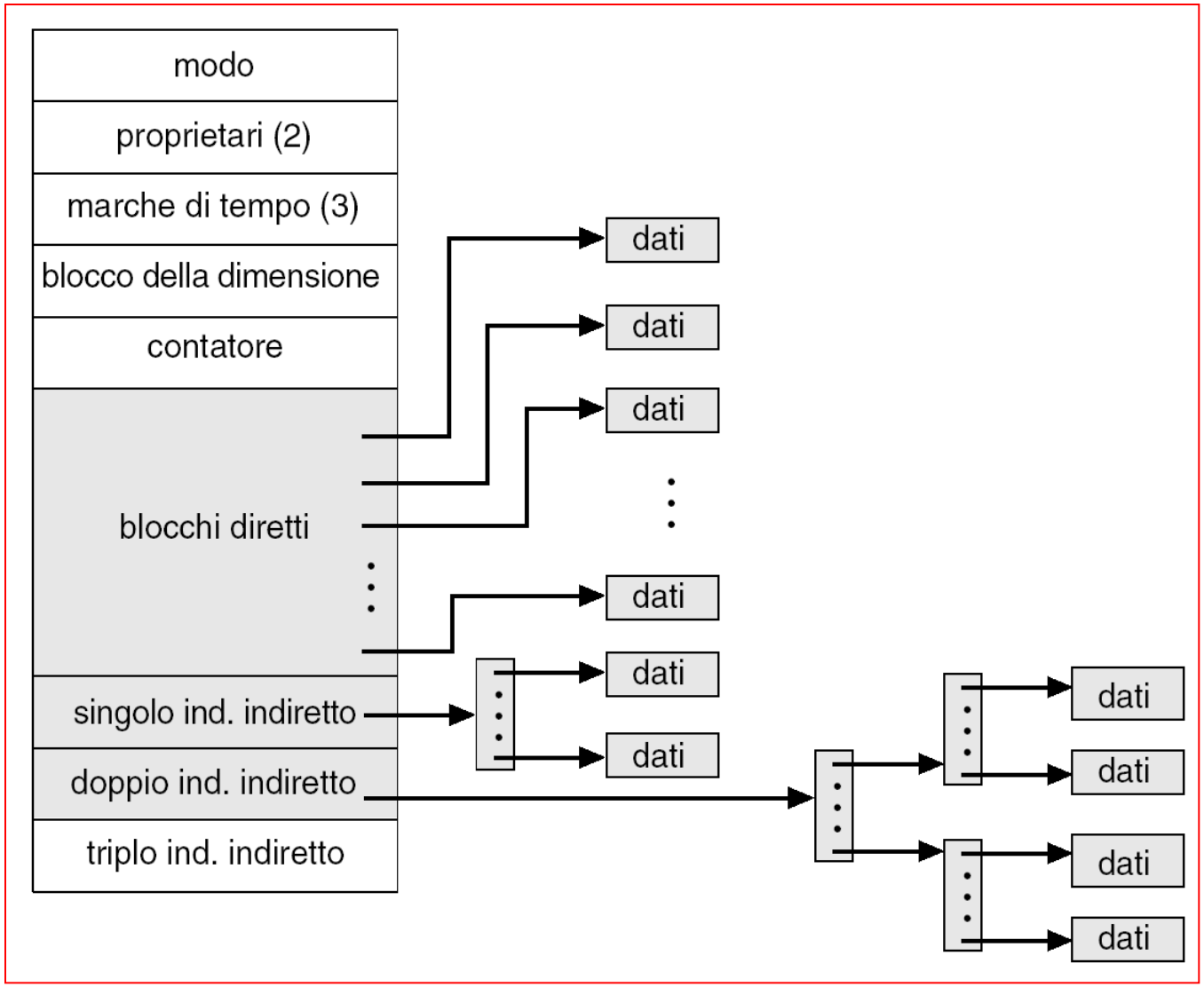


Q_2 = spostamento nel blocco della tabella indice.
R_2 spostamento nel blocco del file.

Allocazione indicizzata – Mappatura (Cont.)

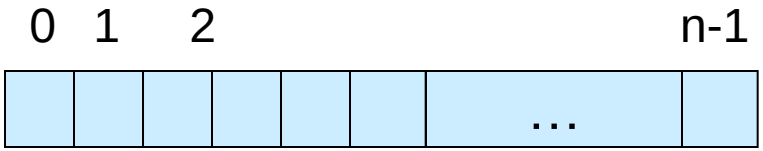


Schema combinato: UNIX (4K bytes per blocco)



Gestione dello spazio libero

- Vettore di bit (*n* blocchi).



$$\text{bit}[i] = \begin{cases} 0 \Rightarrow \text{blocco}[i] \text{ libero} \\ 1 \Rightarrow \text{blocco}[i] \text{ occupato} \end{cases}$$

Calcolo del numero di blocco:

(numero di bit per parola) *
(numero di parola con valore 0) +
spiazzamento del primo bit a 1

Gestione dello spazio libero (Cont.)

- La mappa di bit richiede spazio extra. Esempio:

  dimensione blocco = 2^{12} bytes

  dimensione disco = 2^{30} bytes (1 gigabyte)

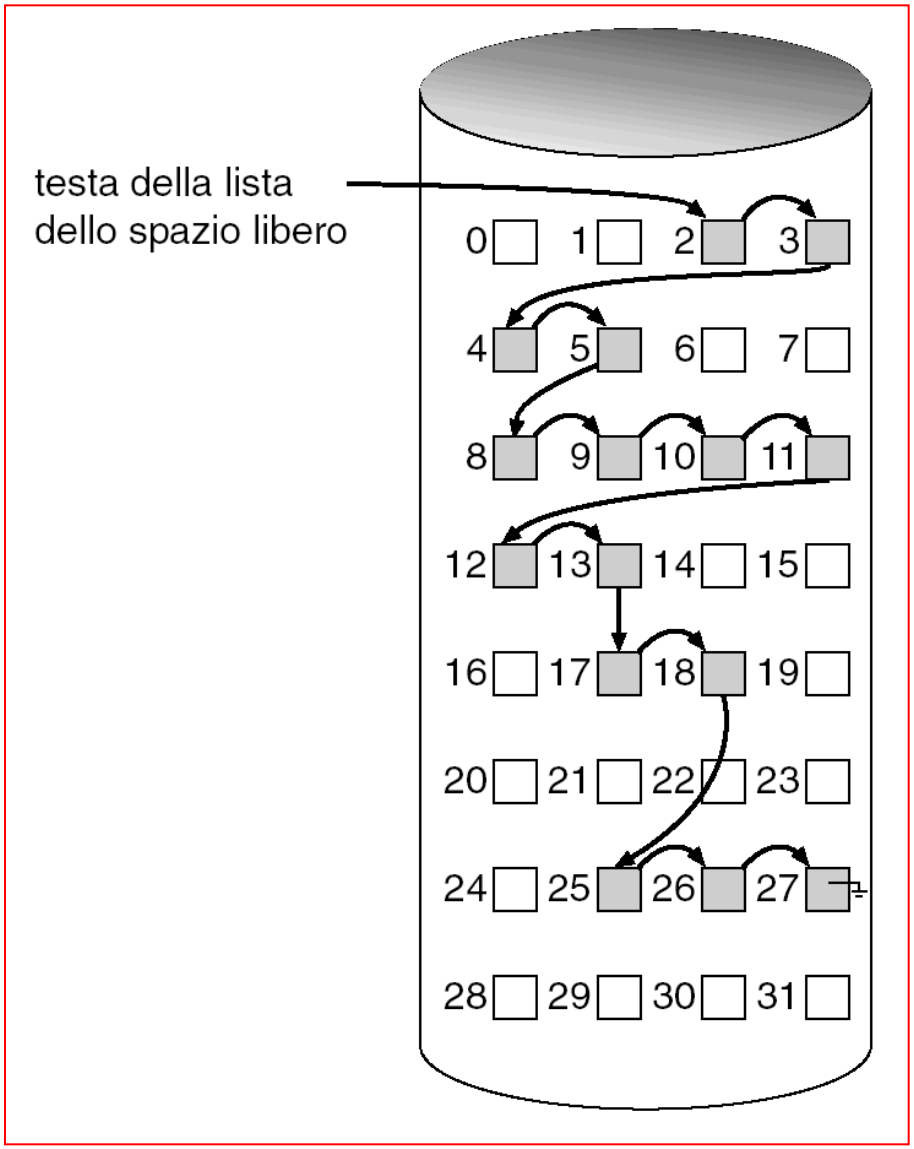
  $n = 2^{30}/2^{12} = 2^{18}$ bits (or 32K bytes)

- Semplicità di trovare blocchi liberi consecutivi sul disco.
- Lista collegata (lista libera)
  - Non trova facilmente spazi contigui.
  - No spreco di spazio.
- Raggruppamento.
- Conteggio.

Gestione dello spazio libero (Cont.)

- Bisogna proteggere:
  - **Pointer to free list**
  - Bit map
    - deve essere tenuto nel disco;
    - la copia in memoria e su disco possono essere diverse;
    - la situazione in cui un blocco[*i*] ha bit[*i*] = 1 in memoria e bit[*i*] = 0 su disco non è accettabile.
  - Soluzione:
    - impostare bit[*i*] = 1 nel disco;
    - allocare il blocco[*i*];
    - Impostare bit[*i*] = 1 in memoria.

Lista dello spazio libero collegato su disco

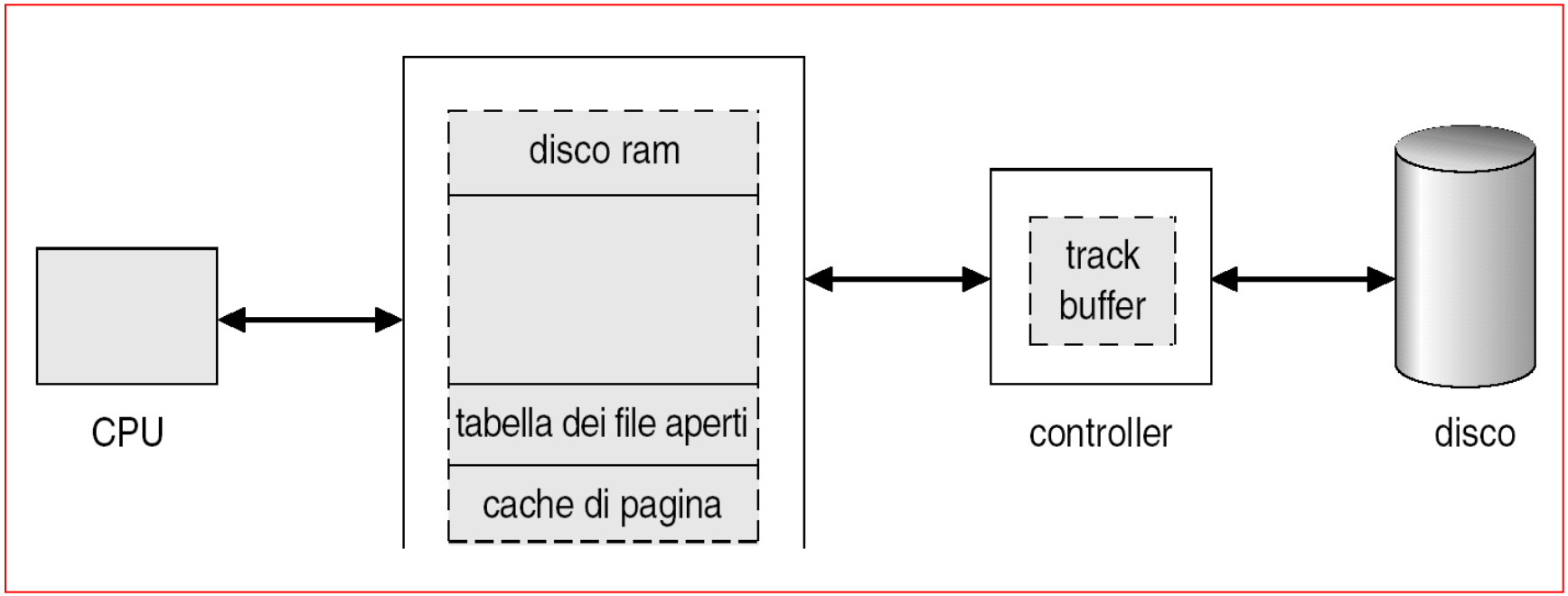


Efficienza e prestazioni

- L'efficienza dipende da:
  - algoritmi usati per l'allocazione del disco e delle directory;
  - tipi di dati conservati nel descrittore della directory.

- Prestazioni:
  - cache del disco – sezione separata della memoria centrale dove sono mantenuti i blocchi;
  - free-behind a read-ahead – tecniche per ottimizzare l'accesso sequenziale;
  - migliorare le prestazioni del PC separando una parte della memoria e trattarla come un disco virtuale o un disco RAM.

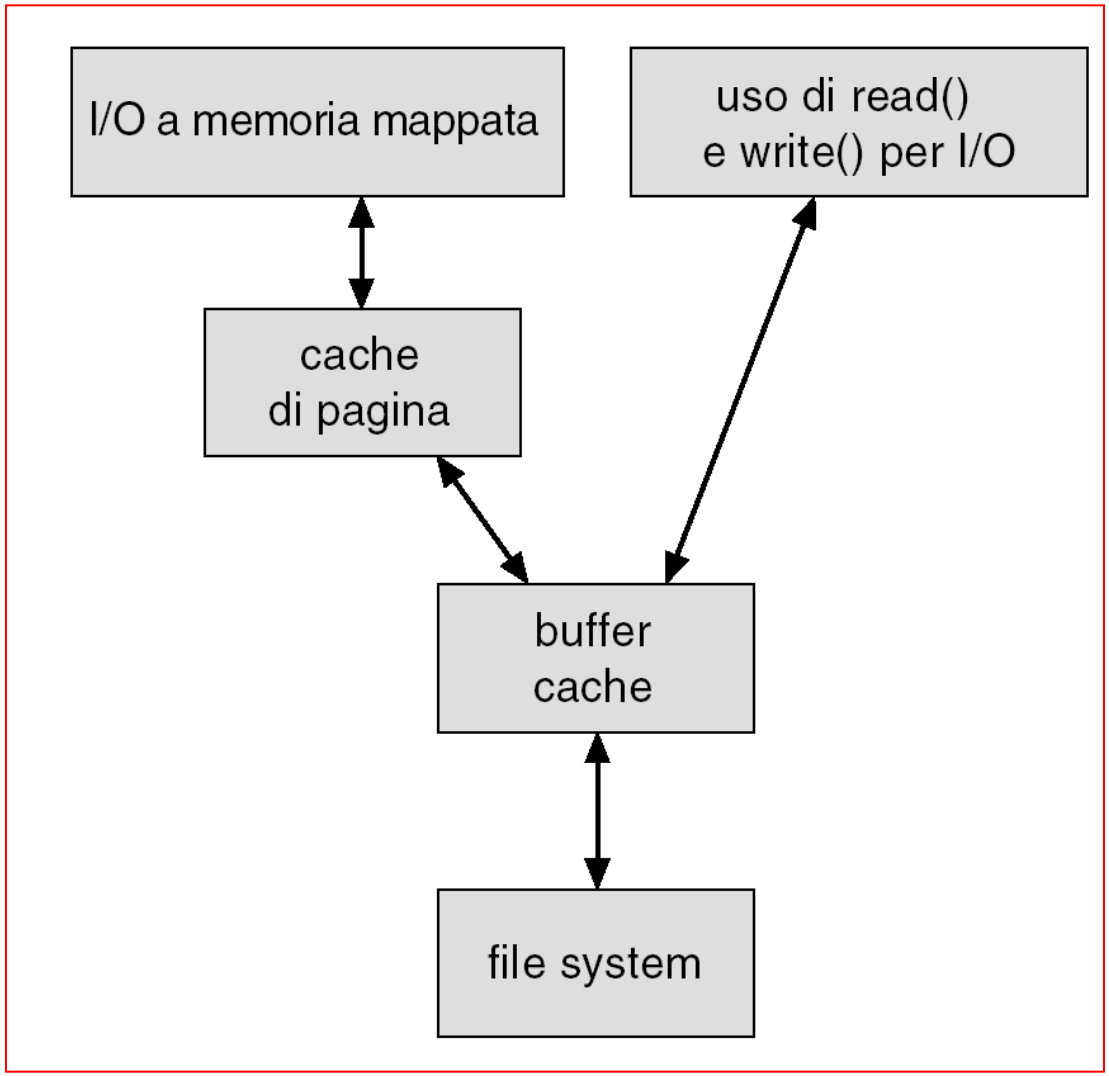
Varie locazioni della cache del disco



Cache delle pagine

- Una **cache delle pagine** usa tecniche di memoria virtuale per memorizzare i dati del file come pagine invece che come blocchi per il file system.
- La memoria mappata I/O usa una cache delle pagine.
- La procedura di I/O attraverso il file system usa il buffer (disk) cache.
- Questo porta alla seguente figura.

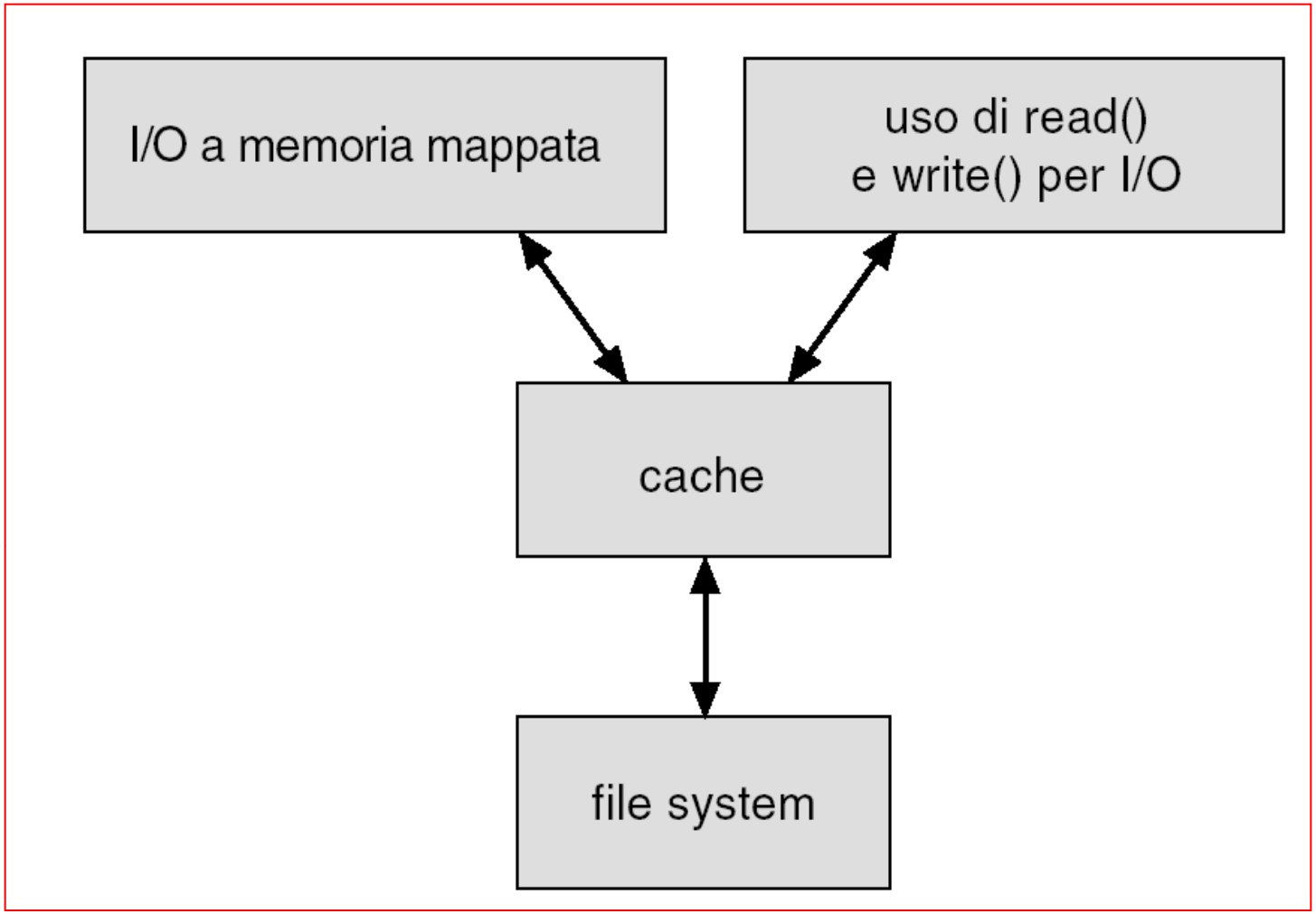
I/O senza buffer cache unificata



Buffer Cache unificata

- Con un buffer cache unificata la mappatura della memoria e le chiamate di sistema *read* e *write* usano la stessa cache di pagina, con il vantaggio di evitare la doppia cache e di permettere alla memoria virtuale di gestire i dati del file system.

I/O con buffer cache unificata



Recupero del file system

- Controllore della coerenza – confronta i dati nella struttura delle directory con i blocchi di dati su disco e cerca di riparare tutte le incoerenze che trova.
- Usare programmi di sistema per effettuare il salvataggio di sicurezza (*back up*) dei dati dal disco fisso ad un altro dispositivo di memorizzazione (floppy disk, DVD).
- Recuperare i file persi o l'intero disco attraverso il *ripristino* dei dati salvati sul supporto di backup.

File system basato sulla registrazione delle attività

- **I file system orientati alle transazioni basate su log** (o file system basati sulla registrazione delle attività) registrano ogni aggiornamento del file system come una transazione.

- Tutte le transazioni sono scritte in un registro (**log)**. Una transazione è considerata effettuata quando è scritta nel log. Comunque, il file system non può ancora essere aggiornato.

- Le transazioni nel log sono scritte in modo asincrono rispetto al file system. Quando il file system viene modificato, la transazione viene rimossa da log.

- Se il file system si blocca, tutte le transazioni rimanenti nel log devono ancora essere completate.