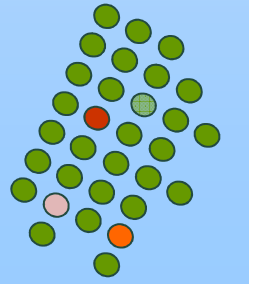
Implementazione del File System

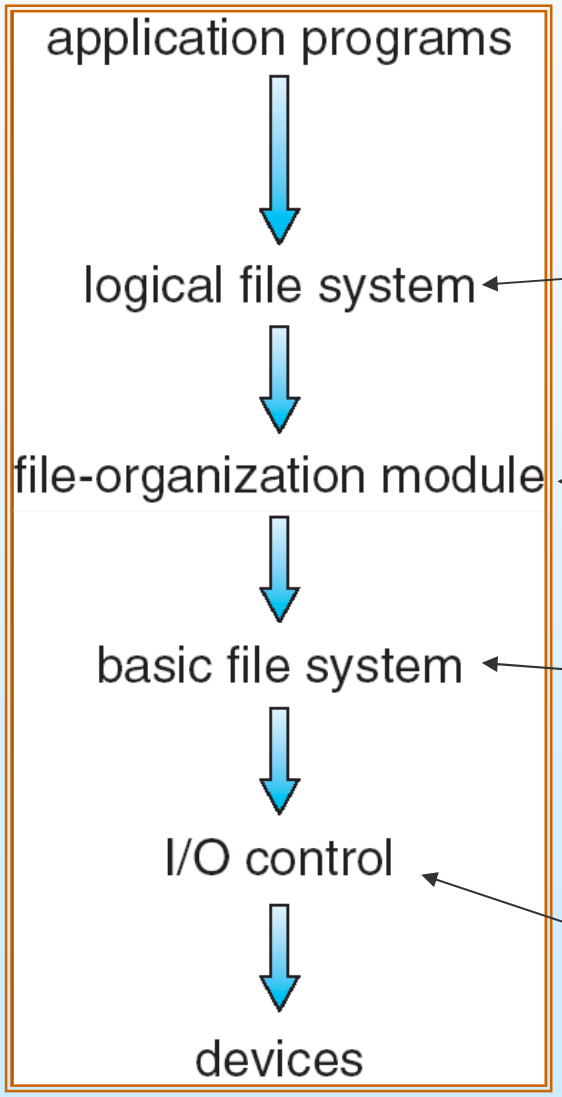
Capitolo 11 -- Silberschatz



Implementazione del File System

- File system:
  - Definizione dell'aspetto del sistema agli occhi dell'utente
  - Algoritmi e strutture dati che permettono di far corrispondere il file system logico ai dispositivi fisici
- Il file system risiede in un'unità di memorizzazione secondaria (disco)
  - Si può scrivere localmente
  - Accesso diretto
- Il file system è organizzato in livelli.

File System a strati



File system logico: gestisce i metadati (tutte le strutture dei file eccetto i dati): directory, File control block (FCB), protezione e sicurezza

Modulo di organizzazione dei file: traduce gli indirizzi dei blocchi logici in quelli dei blocchi fisici; gestisce spazio libero

File system di base: deve inviare comandi generici al driver del dispositivo per leggere o scrivere blocchi fisici nel disco

Controllo I/O (driver dei dispositivi e gestori di interrupt): si occupa del trasferimento delle info tra memoria centrale e secondaria

Strutture presenti sul disco

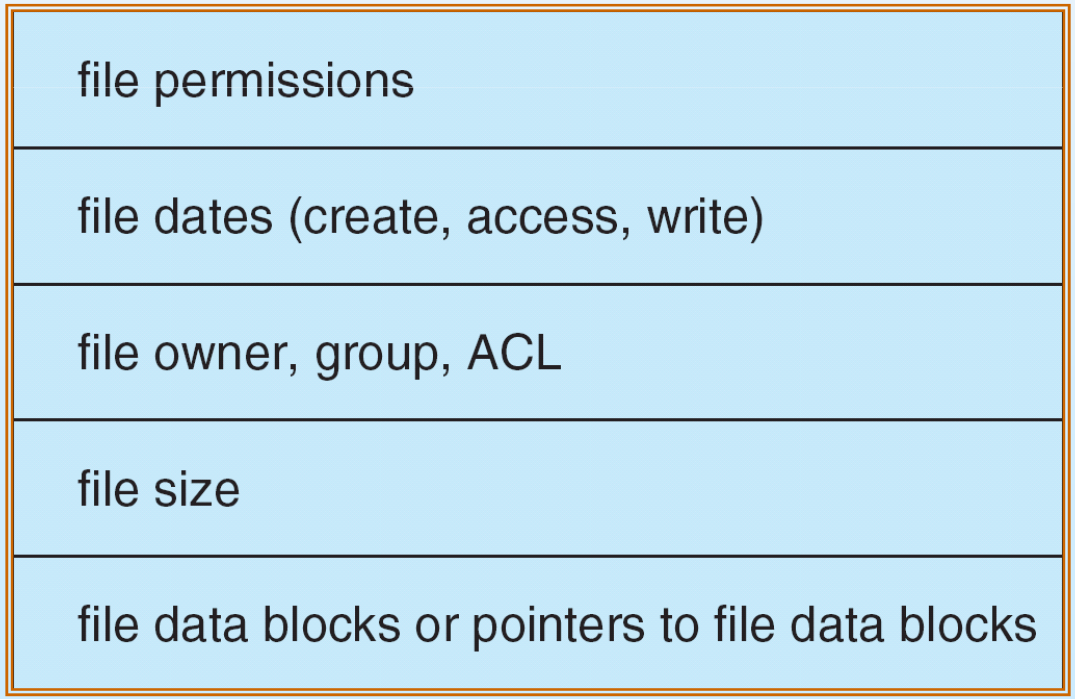
- Boot control block – Informazioni necessarie per il caricamento del sistema operativo (UFS - boot block, NTFS - partition boot sector)
- Volume control block – contiene dettagli del volume, numero di blocchi per partizione, taglia dei blocchi, blocchi liberi …(UFS – superblock, NTFS – master file table)
- Struttura delle directory – usata per organizzare i file …(UFS – nomi file e i-node associato, NTFS – master file table)
- File control block (FCB) – informazioni sul file (UFS – inode)

Strutture del file system in memoria

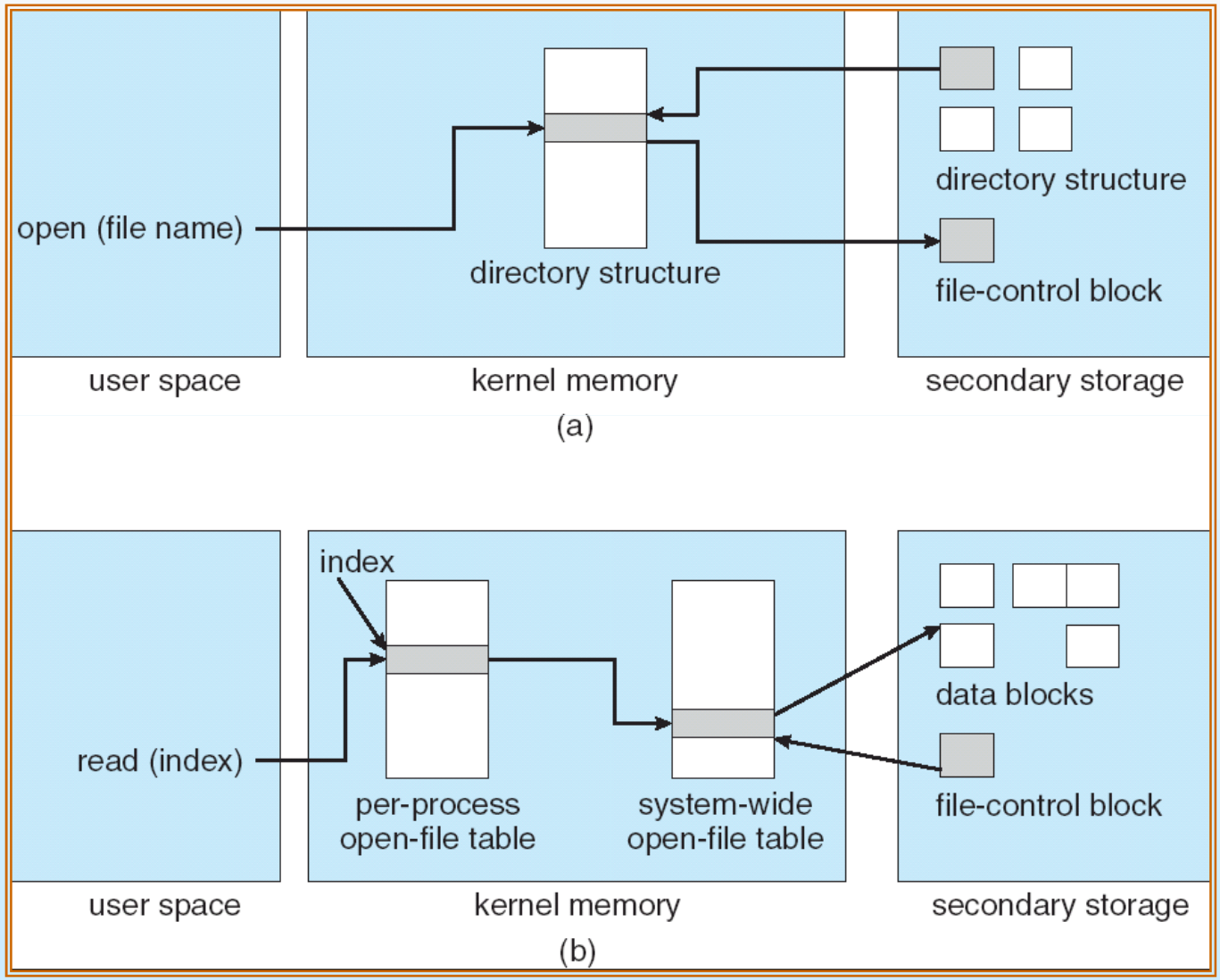
- Quando si monta un file system vengono caricate in memoria questa serie di informazioni che si eliminano solo allo smontaggio:
  - Tabella di montaggio (mount table) – contiene informazioni circa i volumi montati
  - Cache per le strutture delle directory recentemente accedute
  - Tabella dei file aperti di sistema
  - Tabella dei file aperti per processo

File control block

- Creazione di un file
  - Creazione (o allocazione) di un nuovo FCB
  - Carica in memoria la directory appropriata
  - Modifica la directory (nome file e FCB) e riscrittura su disco



open e read


open (file name)
directory structure
directory structure
file-control block
user space
kernel memory
secondary storage
(a)
index
read (index)
per-process
open-file table
system-wide
open-file table
data blocks
file-control block
user space
kernel memory
secondary storage
(b)

Partizioni e montaggio

- Un disco può presentare diverse partizioni:
  - raw : non contiene alcun file system; è una parte del disco privo di struttura logica
  - cocked : parte del disco con una struttura logica
- Root partition – contiene il kernel del sistema operativo e altri file. Viene montata in fase di boot
- La tabella di montaggio tiene traccia di tutti i file system montati
  - Windows monta ciascun volume in uno spazio di nomi separato (E:, F:, G: …)
  - In Unix un file system può essere montato su ogni directory (un flag nell'i-node della directory indica che quella directory è un "punto di montaggio")

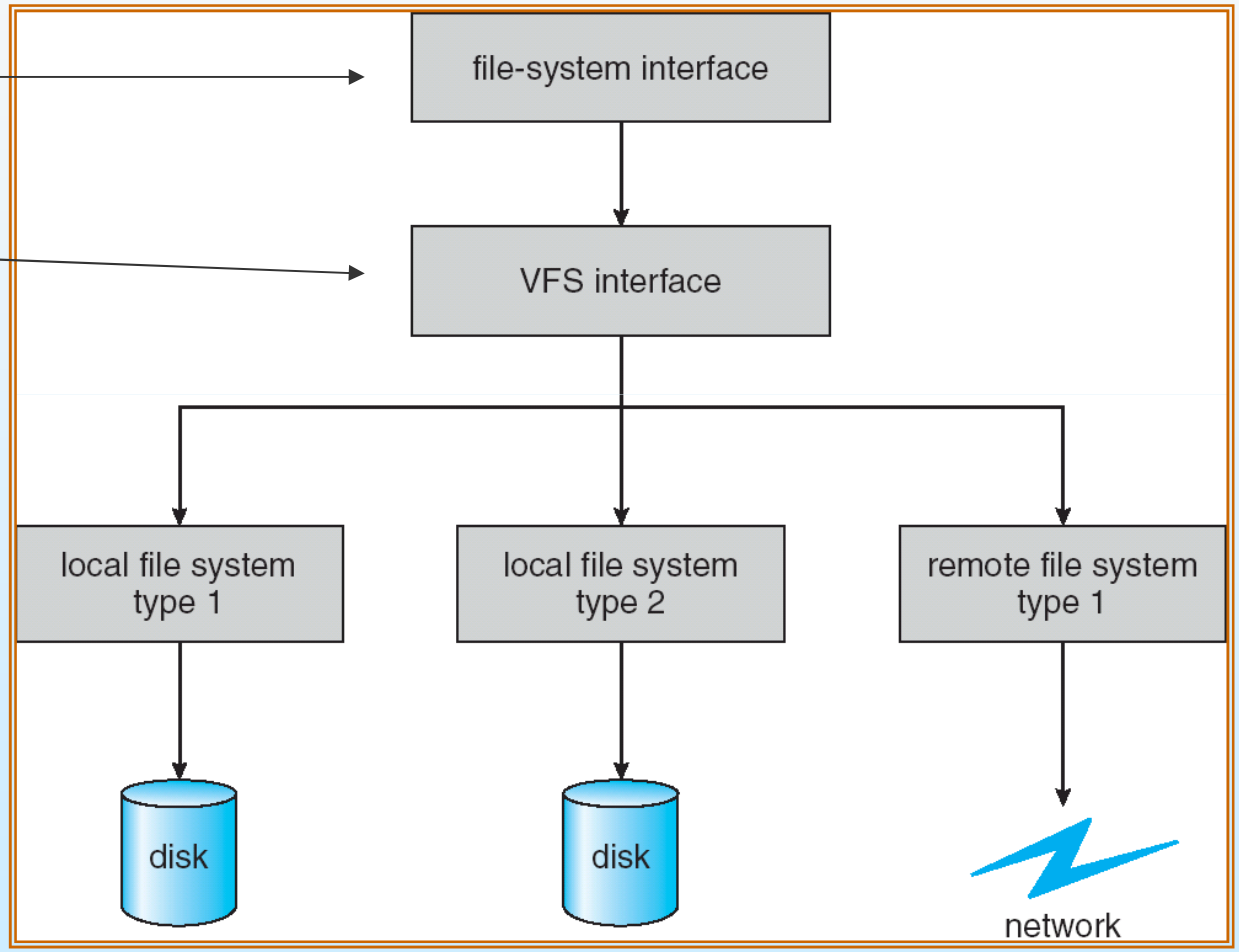
File System Virtuali

- Un moderno sistema operativo può supportare diversi tipi di file system. Come?
- Un Virtual File System (VFS) utilizza una tecnica orientata agli oggetti.
- VFS permette che la stessa interfaccia, cioè chiamate di sistema e API, sia usata per differenti tipi di file system.
- L'API usa le funzioni VFS piuttosto che quelle di un tipo specifico di file system. VFS si occupa della traduzione della chiamata "astratta" nella chiamata all'opportuno file system

Vista schematica di un VFS

Open, close, read, write

Rapp. univoca - vnode



Implementazione di una directory

- Lista di nomi di file con puntatori ai blocchi dei dati:
  - semplice da programmare,
  - richiede tempo per la ricerca,
  - Lista ordinata, B-albero.
- Tabella hash – lista lineare con una tabella hash:
  - diminuisce il tempo di ricerca nella directory;
  - collisioni – due nomi di file vengono associati alla stessa posizione dalla funzione hash;
  - dimensione fissa.

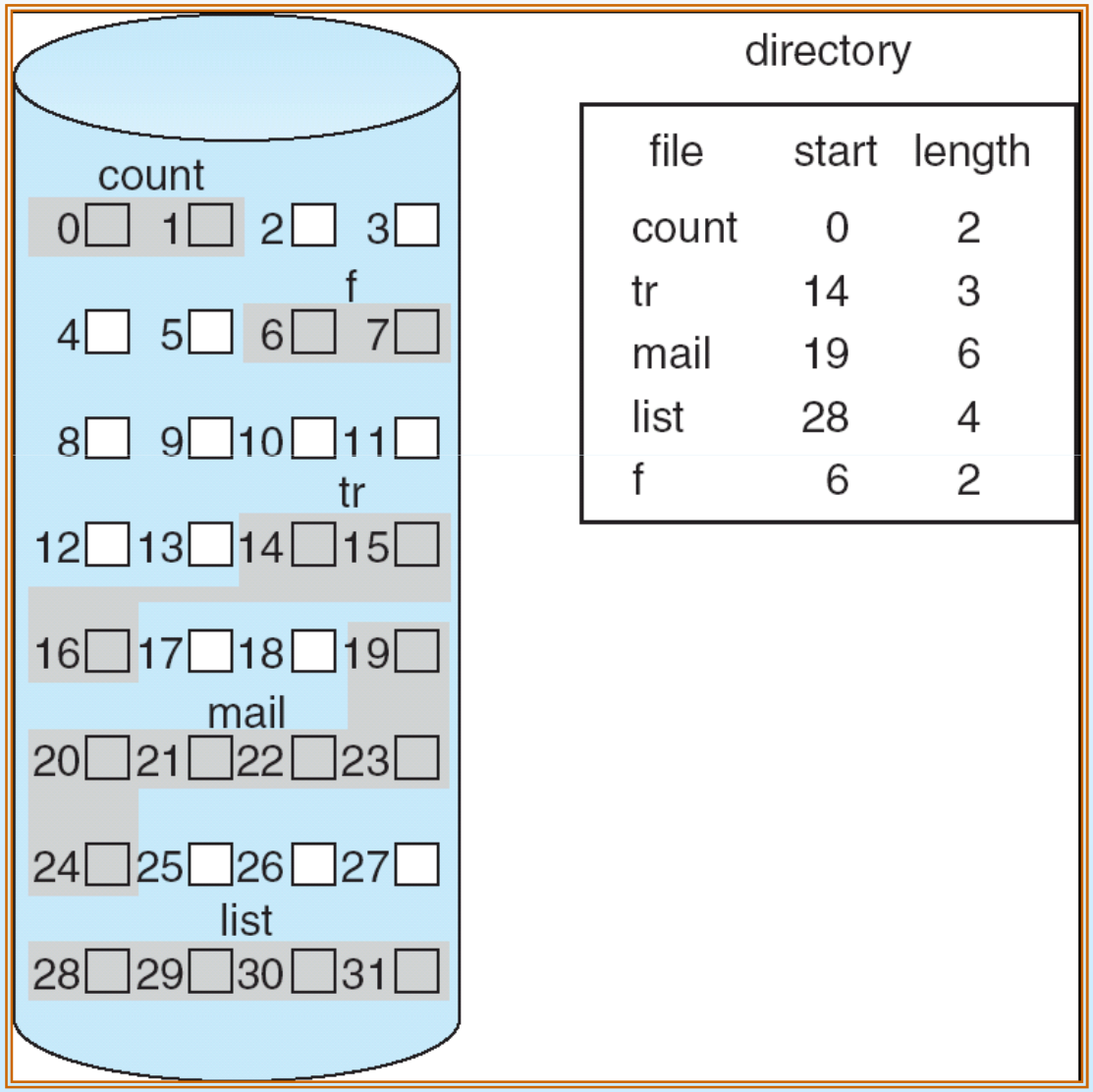
Metodi di allocazione

- Un metodo di allocazione specifica il modo in cui i blocchi di un file vengono allocati nel disco:
  - Allocazione contigua.
  - Allocazione linkata.
  - Allocazione indicizzata.

Allocazione contigua

- Ogni file occupa un certo numero di blocchi contigui su disco.
- Facile – è definita dall'indirizzo del primo blocco del file su disco e dalla lunghezza (numero di blocchi).
- Accesso
  - Accesso sequenziale: il FS memorizza l'indirizzo dell'ultimo blocco a cui si è acceduto → un nuovo accesso è immediato o al più necessita di accedere al blocco successivo
  - Accesso diretto: se si vuole accedere all' ***i*** -mo blocco di un file che comincia al blocco ***b***, si accede direttamente al blocco ***b*** + ***i***

Allocazione contigua dello spazio del disco

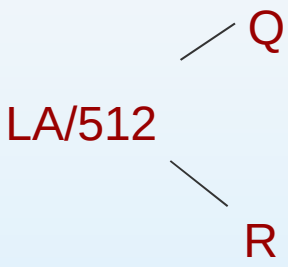


directory

file	start	length
count	0	2
tr	14	3
mail	19	6
list	28	4
f	6	2

Allocazione contigua

- Mappatura da logica a fisica



Blocco al quale accedere = Q + blocco di partenza
Spostamento nel blocco = R

LA = Logical Address

Allocazione contigua

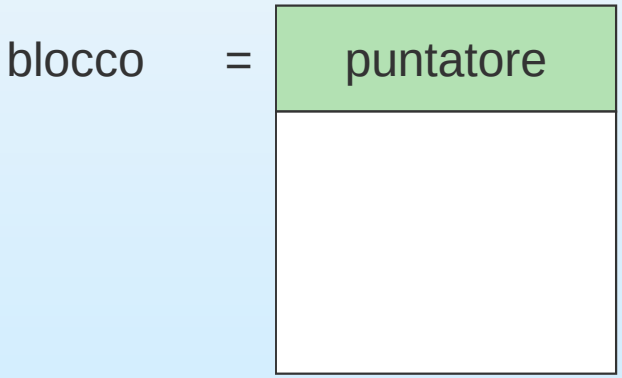
- Svantaggi:
  - Frammentazione esterna (problema dell'allocazione dinamica della memoria): assegnando e liberando lo spazio per i file, lo spazio libero del disco viene frammentato in tanti buchi.
  - La taglia dei file non può crescere.

Estensione

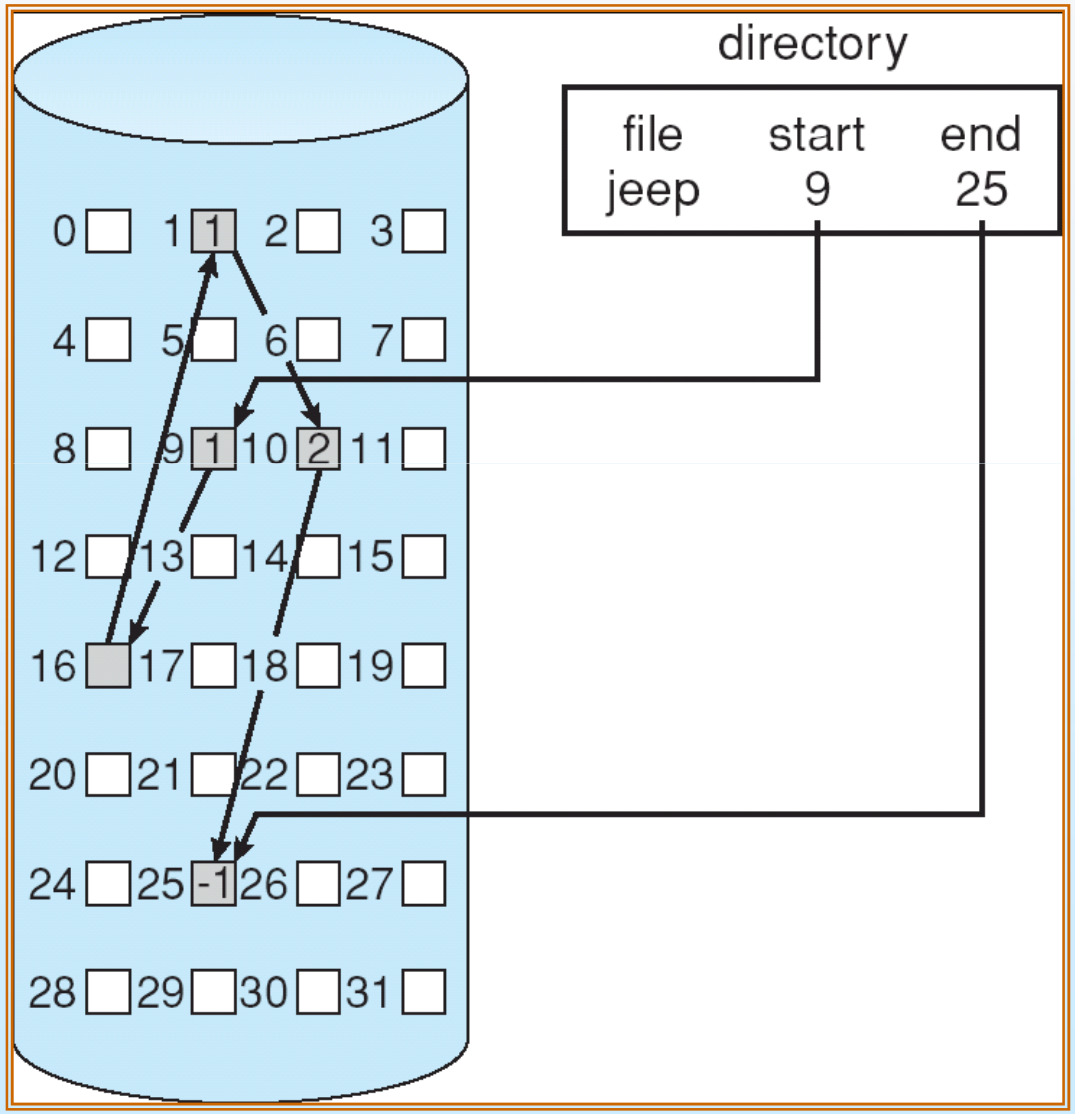
- Molti nuovi file system (e.g., Veritas File System) usano uno schema di allocazione contigua modificato.
- Inizialmente viene allocato un pezzo contiguo di spazio e poi, quando il pezzo non è più sufficientemente, viene aggiunta un'estensione.
- Un'**estensione** è un altro pezzo di spazio contiguo. Un file consiste in una o più estensioni.

Allocazione linkata

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



Allocazione Linkata

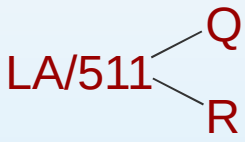


Allocazione linkata

- Una directory entry può consistere in:
  - puntatore al primo blocco e size del file.
- Gestione dello spazio libero – assenza di sprechi.
  - Creazione di un file: si crea un nuovo elemento nella directory, si mette a *nil* il puntatore al primo blocco e si inizializza la size a 0
  - Scrittura in un file: se quello da scrivere va oltre le dimensioni dell'ultimo blocco si cerca un nuovo blocco e lo si concatena alla fine

Allocazione linkata

- Mapping indirizzi logici/fisici.



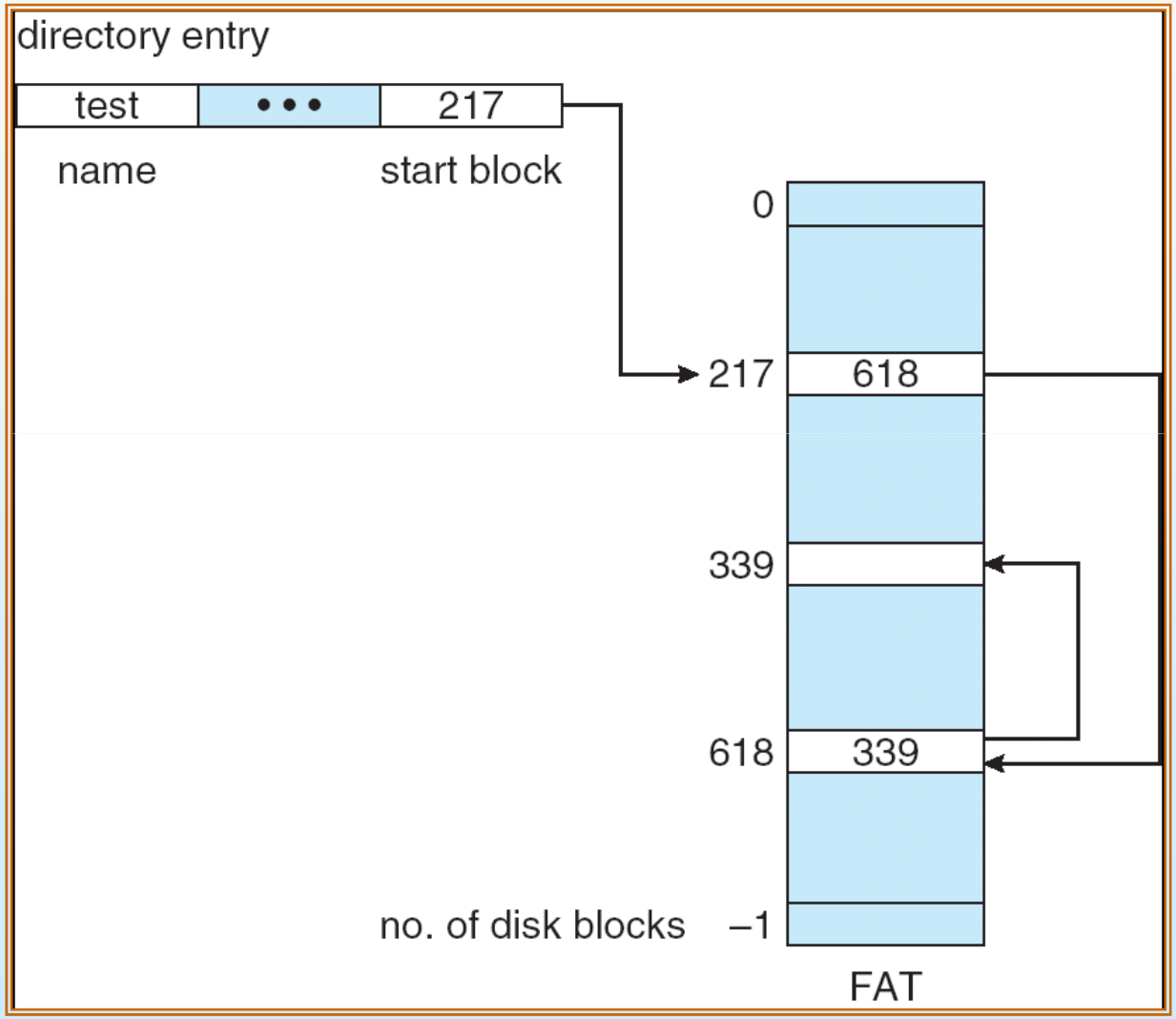
• Il blocco al quale accedere è il Q-esimo blocco nella lista linkata di blocchi che rappresentano il file.

• Spostamento nel blocco = R + 1

Allocazione linkata

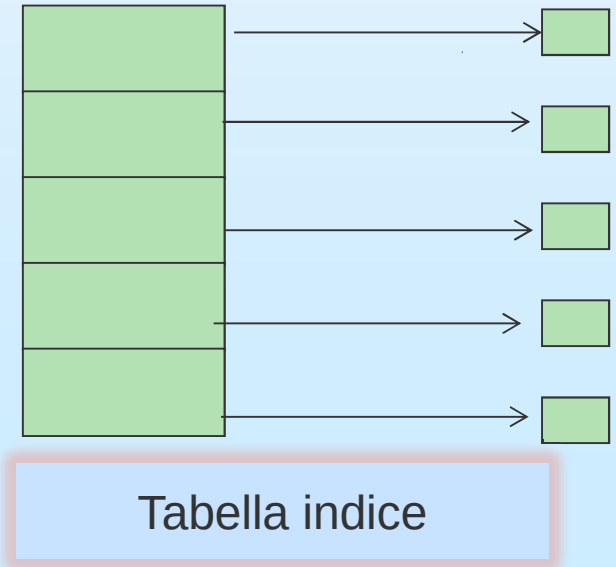
- Svantaggi
  - Inefficiente l’accesso diretto.
  - Spreco di parte di un blocco per il puntatore.
  - Poca affidabilità.

Tabella di allocazione dei file (FAT)

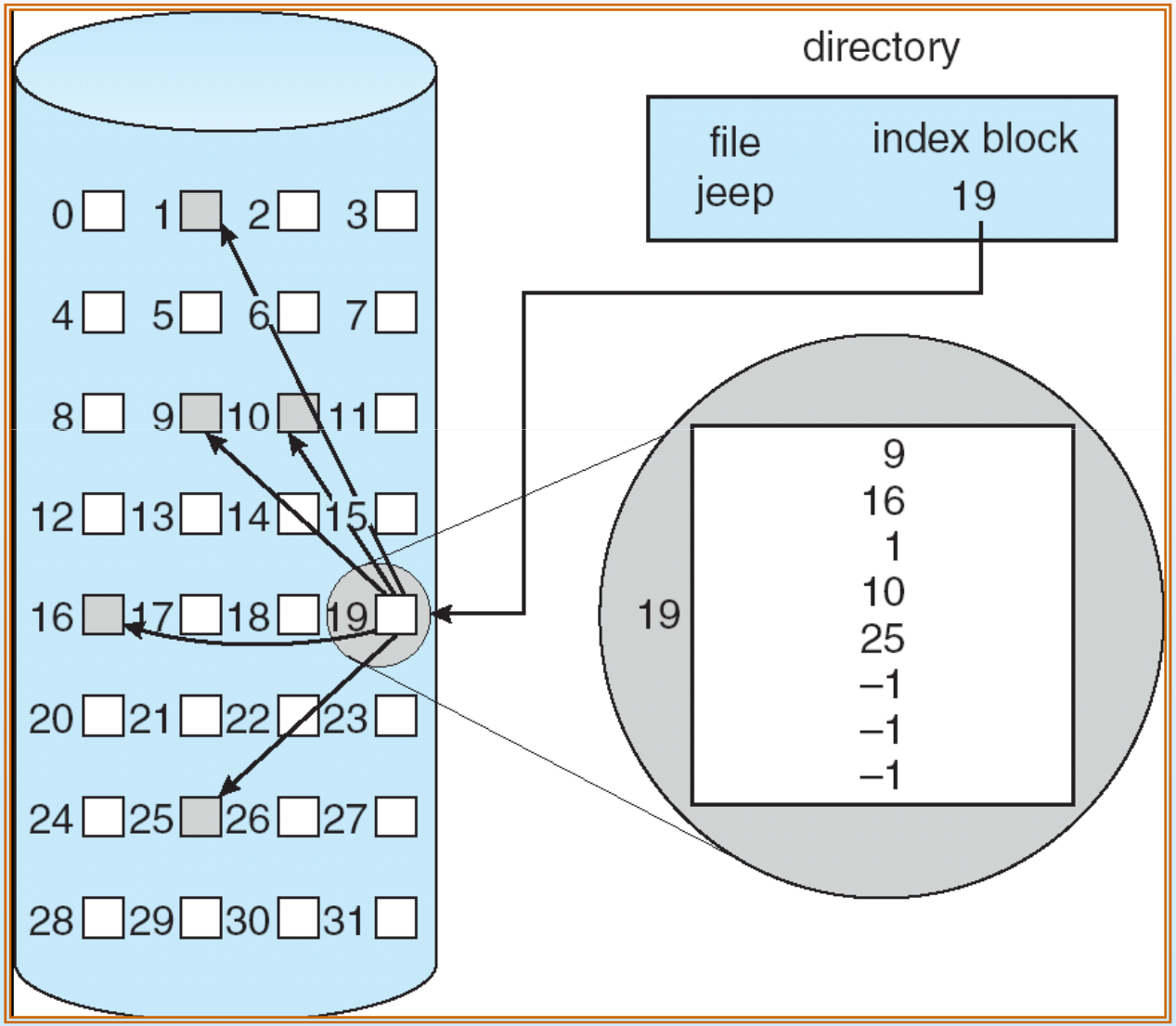


Allocazione Indicizzata

- Contiene tutti puntatori nel blocco indice.
- Vista logica.
- Ogni file ha il proprio blocco indice, cioè un array contenente gli indirizzi dei blocchi di cui il file è costituito

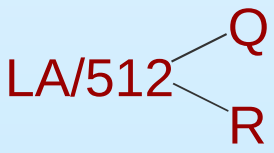


Esempio di allocazione indicizzata



Allocazione indicizzata

- Necessita di una tabella indice.
  - Creazione di un file: si alloca il blocco indice e tutto il suo contenuto è a *nil*
  - Scrittura in un file: se c'è necessità di un nuovo blocco, diciamo l' i -mo, si alloca un nuovo blocco e si mette il suo indirizzo nell' i –ma posizione del blocco indice.
- Mapping da logico a fisico. Blocco indice della dimensione di 512 parole.



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

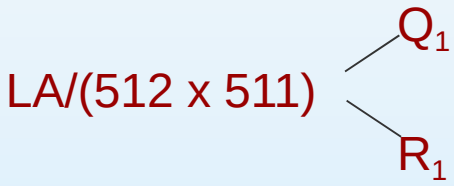
Allocazione indicizzata

- Accesso diretto senza frammentazione esterna, ma
  - c'è l'overhead del blocco indice se il file è piccolo;
  - può essere troppo piccolo se il file è grande

Allocazione indicizzata – schema concatenato

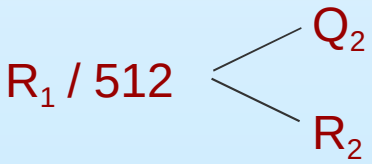
- Per permettere la presenza di file lunghi vengono collegati tra loro parecchi blocchi indice; ciò è fatto ponendo come ultima parola di un blocco indice il puntatore al prossimo blocco indice.

- Mappatura da logico a fisico



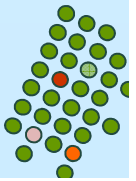
Q_1 = blocco della tabella indice,

R_1 è usato come segue:

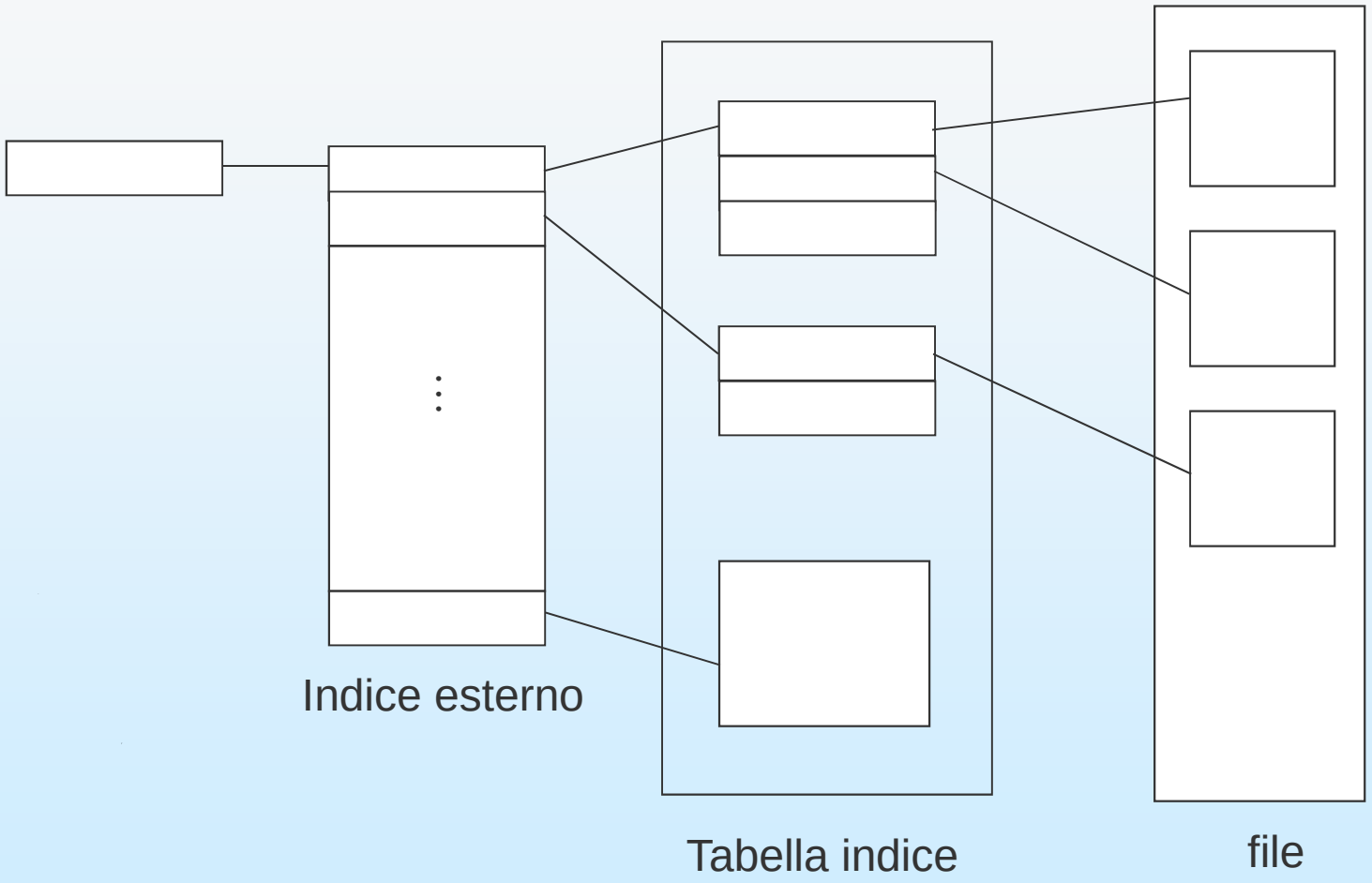


Q_2 = spostamento nel blocco della tabella indice.

R_2 = spostamento nel blocco di file.

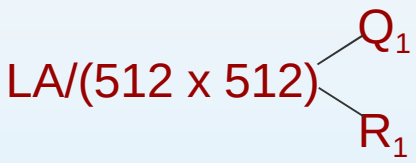


Allocazione indicizzata – 2 livelli



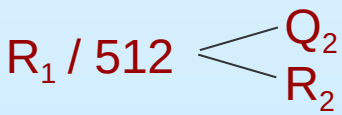
Allocazione indicizzata – 2 livelli

- Mappatura da logico a fisico
- Indice a due livelli (la dimensione massima del file è 512^3).



Q_1 = spostamento nell'indice esterno,

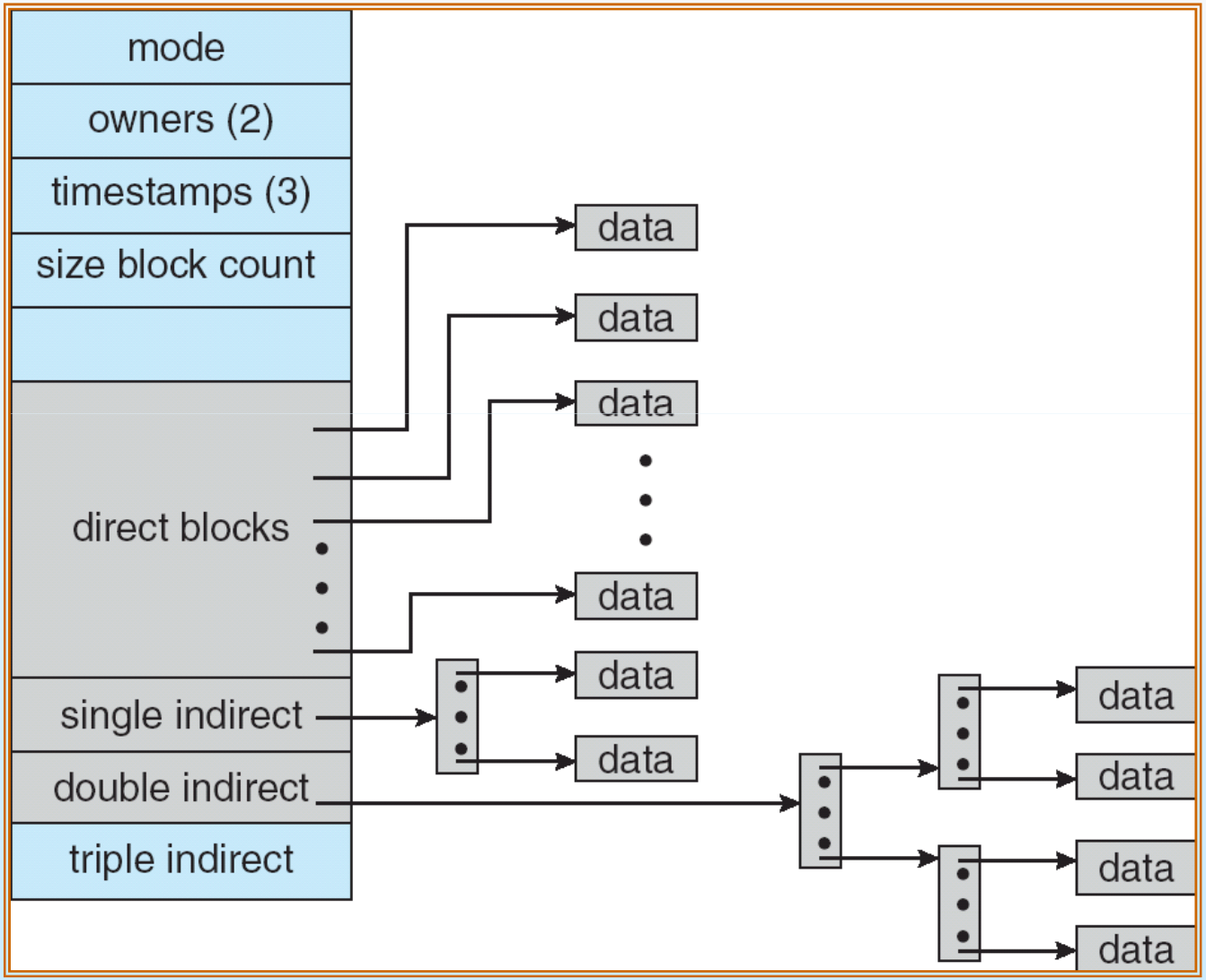
R_1 è usato come segue:



Q_2 = spostamento nel blocco della tabella indice.

R_2 = spostamento nel blocco di file.

Schema combinato: UNIX (4K byte per blocco)

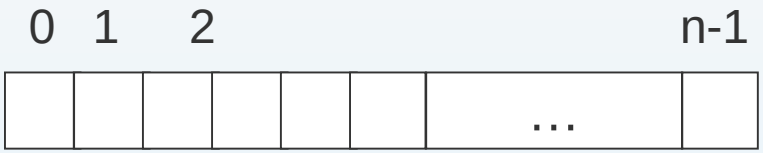


Gestione dello spazio libero

- È necessario tener memoria dei blocchi di spazio libero per poterli utilizzare quando si ha bisogno di ingrandire un file
- Bisogna ricordare i blocchi rilasciati quando si cancella un file

Gestione dello spazio libero – vettore di bit

- Vettore dei bit (*n* blocchi)



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

Calcolo del primo numero di blocco libero:

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

Gestione dello spazio libero – vettore di bit

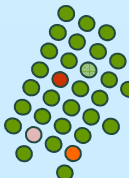
- Il vettore di bit richiede spazio extra.
  - Esempio:

    dimensione blocco = 2^{12} byte

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

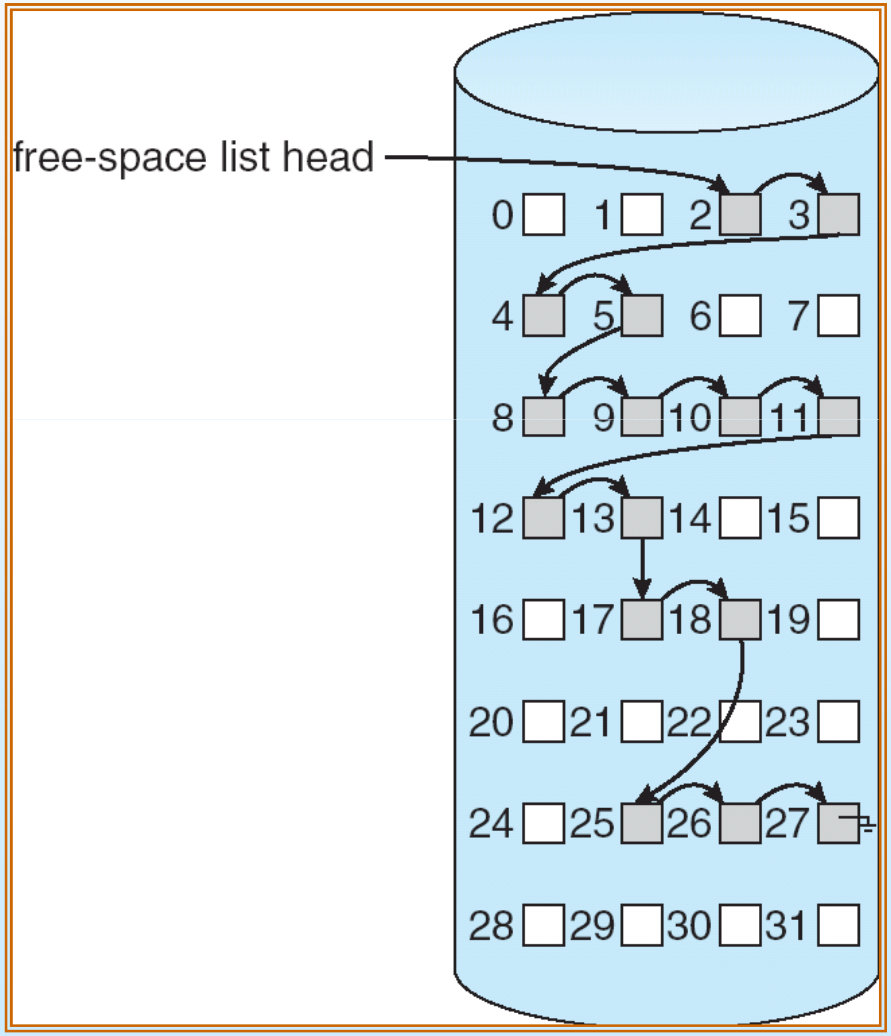
    n = $2^{30}/2^{12} = 2^{18}$ bit (o 32Kb)

- Semplicità di trovare blocchi liberi consecutivi sul disco.



Gestione dello spazio libero – Lista dei blocchi liberi

Si conserva il puntatore al primo blocco in una locazione speciale del disco che viene caricato in memoria quando si vuole accedere ad un blocco libero



- Assenza di spreco di spazio.
- Non facile trovare blocchi contigui.

Gestione dello spazio libero – altri metodi

Raggruppamento:

- Si memorizzano in un blocco gli indirizzi di n blocchi: di questi i primi n-1 sono realmente liberi mentre l'ultimo contiene gli indirizzi di altri n blocchi, e così via
- Permette di trovare rapidamente molti blocchi liberi.

Conteggio:

- Spesso più blocchi contigui possono essere rilasciati
- Quindi ogni elemento della lista dello spazio libero è formato da un indirizzo ed un contatore: l'indirizzo punta al primo dei blocchi liberi mentre il contatore dice quanti blocchi contigui ci sono

Gestione dello spazio libero

Bisogna fare attenzione a:

- Nella lista dei blocchi liberi: Proteggere il puntatore alla lista
- Nel vettore di bit : Consistenza
  - 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] = 0 in memoria e bit[i] = 1 su disco non è accettabile.
  - Soluzione:
    - impostare bit[i] = 0 nel disco;
    - allocare il blocco[i];
    - impostare bit[i] = 0 in memoria.