Lezione 11

Il file system

- File e directory
- Struttura logica del file system
- Montaggio di un file system
- Condivisione e protezione di file
- Struttura del file system
- Metodi di allocazione
- Gestione dello spazio libero
- Coerenza del file system

File

- Il maggior problema nella gestione della memoria secondaria e' la necessita' di fare riferimenti alla organizzazione fisica dei dispositivi.
- Esempio:
  - un programma in linguaggio macchina puo' risiedere nei blocchi 45, 51, 68, 73 e 99 di un disco
  - Una modifica al codice puo' produrre una diversa memorizzazione
  - Necessita' di fare riferimento ai blocchi fisici anche se l'entita' logica (il programma) e' sempre la stesso

FILE

Una **raccolta di dati associata ad un nome**, residente in memoria secondaria e manipolabile come una unica entita'

File

Dal punto di vista dell'utente, il file e' la piu' piccola unita' di memoria secondaria utilizzabile

Tutti i dati presenti nella memoria secondaria devono essere organizzati in file

Tipi di file:

- testo (sequenza di caratteri)
- sorgente (sequenza di procedure e funzioni)
- eseguibile (sequenza di moduli in linguaggio macchina che il loader puo' caricare in memoria)
- ...

Attributi dei file

- **Nome** — unica informazione conservata in formato direttamente leggibile.
- **Tipo** — descrizione dell'utilizzo del file.
- **Locazione** — puntatore alla posizione del file sul dispositivo.
- **Dimensione** — dimensione attuale del file.
- **Protezione** — controlla l'accesso in lettura, scrittura ed esecuzione del file.
- **Ora, data, e identificazione dell'utente** — dati necessari alla protezione e sicurezza del sistema, e per il controllo d'uso.

Struttura e tipi di file

- **Struttura**:
  - Nessuna — sequenza di parole o byte (es. file di testo)
  - Struttura a record semplice
    - Linee
    - Record a lunghezza fissa
    - Record a lunghezza variabile
  - Struttura complessa
    - Documento formattato
    - File rilocabile
- **Tipi**
  - Eseguibile, di testo, archivi, musicali, …
  - Di solito identificati da estensioni al nome

Tipiche estensioni ai nomi di file

Extension	Meaning
file.bak	Backup file
file.c	C source program
file.gif	Compuserve Graphical Interchange Format image
file.hlp	Help file
file.html	World Wide Web HyperText Markup Language document
file.jpg	Still picture encoded with the JPEG standard
file.mp3	Music encoded in MPEG layer 3 audio format
file.mpg	Movie encoded with the MPEG standard
file.o	Object file (compiler output, not yet linked)
file.pdf	Portable Document Format file
file.ps	PostScript file
file.tex	Input for the TEX formatting program
file.txt	General text file
file.zip	Compressed archive

- In MS-DOS possono essere eseguiti solo file con estensione .COM, .EXE, .BAT.
- In Apple Macintosh ciascun file possiede un attributo di creazione contenente il nome del programma che lo ha creato.
- UNIX memorizza un *magic number* per indicare il tipo di file. Usa però le estensioni solo come suggerimento, non vengono imposte né dipendono dal SO.

Operazioni definite sui file

- Creazione — crea una associazione tra un nome di un file e un insieme di blocchi del disco
- Posizionamento — spostamento all'interno del file system
- Distruzione — elimina l'associazione nome-blocchi e libera lo spazio assegnato al file
- Apertura — rende disponibile il file ad un programma in esecuzione.
- Chiusura — evita ulteriori referenze al file
- Elenco — stampa o mostra il contenuto
- Copia — copia il contenuto di un file in un altro file

Operazioni definite sugli elementi di un file

- Scrittura – copia dati dall'area dati di un processo nel file
- Lettura – copia dati dal file nell'area dati di un processo
- Modifica - modifica di dati in un file
- Cancellazione – rimozione di dati dal file

Per le operazioni sugli elementi di un file e' necessario conoscere la posizione dell'elemento sui cui operare.

Ad ogni file e' associato un puntatore al file (file pointer)

Tabella dei file aperti

I programmi possono fare numerosi riferimenti ad uno stesso file

Problema: numerosi accessi al file
e quindi numerose ricerche del file

Il sistema operativo definisce una tabella dei file aperti

open ➔ crea una entry nella tabella dei file aperti
read/write ➔ il file viene individuato in base all'indice della tabella
close ➔ viene rimossa la entry nella tabella dei file aperti

Esempio: Unix

- Esempio di indice ottenuto con la s.c. open

```
#include<stdio.h>
main( ){
int fd;
fd=open("pippo", 0644);
printf("file descriptor = %d\n", fd);
close(fd);
}
```

```
prompt  ->  ./a.out
file descriptor = 3
prompt ->
```


Struttura interna dei file

- Un file e' costituito da sequenze di dati piu' o meno strutturati (byte, sequenze o record)
- Tali dati (record logici) devono essere impacchettati nei blocchi fisici del disco che hanno dimensioni fissate
- Esempio:
  - Un file di 1949 byte richiede 4 blocchi di 512 byte per un totale di 2048 byte
  - I 99 byte dell'ultimo blocco non utilizzati costituiscono la frammentazione interna del disco

Maggiore e' la dimensione del blocco
maggiore e' la frammentazione !!

Metodi di accesso

Indipendentemente dalla sua memorizzazione un file puo' avere differenti metodi di accesso

- Accesso sequenziale: i dati si elaborano ordinatamente un record dopo l'altro. Si incrementa il file pointer dopo ogni lettura o scrittura. Sono possibili operazioni esplicite di spostamento avanti o indietro
- Accesso diretto: si puo' accedere ai dati in un ordine qualunque specificando la posizione del dato.
- Accesso indicizzato: si accede al file mediante una tabella con gli indici dei blocchi. Per reperire un elemento del file occorre prima cercare nell'indice l'elemento corrispondente e utilizzare il puntatore in esso contenuto per accedere ai dati. Se la tabella e' grande si usa un indice a piu' livelli (es. Indexed Sequential Access Method dell'IBM)

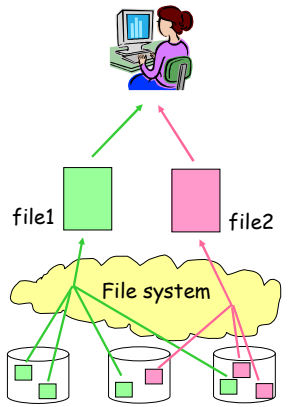
Il file system

- Un sistema operativo puo' gestire
  - milioni di file,
  - di differenti utenti
  - di tipi differente

Necessita' di organizzare logicamente i file

L'organizzazione logica dei file di un sistema operativo e' chiamata FILE SYSTEM

Il file system:



- Organizza i file e gestisce l'accesso ai dati
- Responsabile della integrita' dei file, della gestione dei metodi di accesso (lettura/scrittura) e della gestione della memoria secondaria

Visione dei file indipendente dai dispositivi fisici !!!

Dispositivi fisici

Organizzazione del file system

L'organizzazione del file system deve garantire…

- **Efficienza** — capacità di reperire file rapidamente.
- **Nominazione** — conveniente per gli utenti.
  - Due utenti possono utilizzare nomi uguali per file diversi.
  - Lo stesso file può avere diversi nomi.
- **Grouping** — Raggruppamento dei file sulla base di proprietà logiche, (ad esempio, tutti i programmi Java, tutti i giochi, etc.).

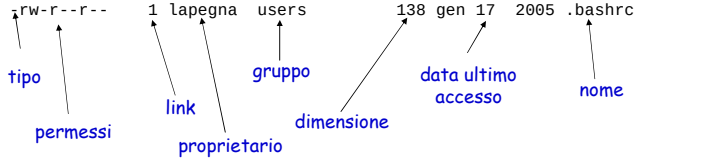
Directory

- Lo strumento piu' comune usato per realizzare un file system e' la directory *(elenco o cartella)*
- Una directory e' un file che contiene nomi e locazioni di file presenti nel file system (non possono contenere dati utente)
- Un elemento di una directory conserva informazioni come:
  - Nome file
  - Posizione
  - grandezza
  - Tipo
  - Privilegi di accesso
  - Tempo di creazione e modifica

esempio: Unix

- Il comando Unix ls elenca le caratteristiche degli elementi di una directory

```
[lapegna@renato lapegna]$ ls -la
totale 27492
drwx------   30 lapegna  users        4096 gen 11 11:04 .
drwxr-xr-x   20 root     root         4096 dic 12 13:16 ..
-rw-r--r--    1 lapegna  users           0 set 26 10:20 aaa
drwxr-xr-x    2 lapegna  users        4096 set 26 12:44 anastasia
-rwxr-xr-x    1 lapegna  users       13961 gen 11 11:02 a.out
-rw-------    1 lapegna  users       14403 nov 29 17:15 .bash_history
-rw-r--r--    1 lapegna  users          24 lug  9  2001 .bash_logout
-rw-r--r--    1 lapegna  users         390 mar 30  2005 .bash_profile
-rw-r--r--    1 lapegna  users         138 gen 17  2005 .bashrc
```

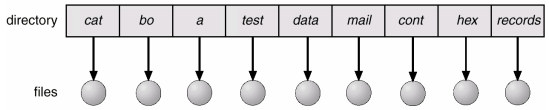


Operazioni sulle directory

Poiche' una directory e' un file, sui suoi elementi e' possibile fare le stesse operazioni che e' possibile fare sugli elementi di un file

- Ricerca di un file
- Creazione di un file
- Cancellazione di un file
- Elenco dei contenuti di una directory
- Ridenominazione di un file
- Attraversamento del file system

File system monolivello (file system piatto)



- gli elementi di una directory sono solamente file
- Una directory unica per tutti i file.
- Problemi di nominazione: occorre scegliere un nome diverso per ogni file.
- Problemi di performance: ricerca lineare nella directory
- Nessun raggruppamento logico.

- raramente implementato

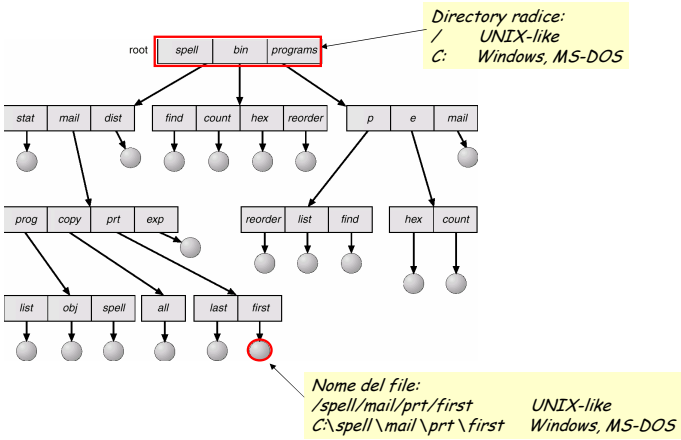
File system gerarchico (ad albero)

- Gli elementi di una directory possono essere file o directory
- Una directory principale (root directory, master file directory) indica la radice dell'albero dalla quale partono le directory di secondo livello
- L'unicita' dei nomi di file e' realizzata mediante il "pathname assoluto" costituito dai nomi delle directory incontrate dalla radice fino al file

- I nomi dei file possono essere unici solo all'interno della stessa directory (pathname relativo) ➔ nominazione
- Ricerca solo nella directory corrente ➔ efficienza
- Possibilita' di raggruppamenti logici ➔ grouping

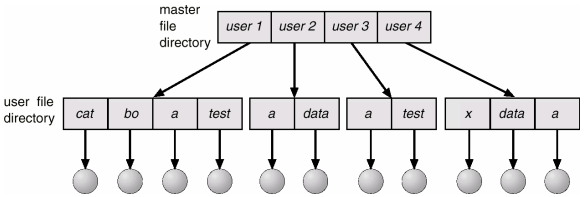
- Implementato nella quasi totalita' dei s.o.

File system gerarchico (ad albero)



Soluzione intermedia (f.s. a due livelli)

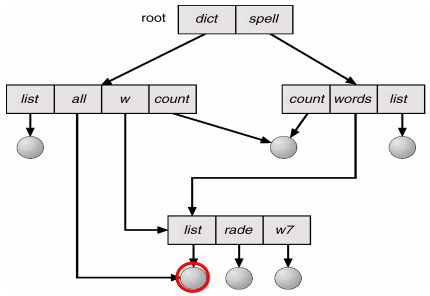
- Directory separate per ciascun utente.



- Ammessi nomi uguali per file di utenti diversi.
- Ricerca efficiente.
- limitata capacità di raggruppamento logico (solo in base ai proprietari).
- Nome di percorso (permette ad un utente di vedere i file degli altri utenti).

File system gerarchico (grafo aciclico)

- Permettono la condivisione di file e sottodirectory.
- La condivisione può essere implementata per duplicazione o tramite *link*.

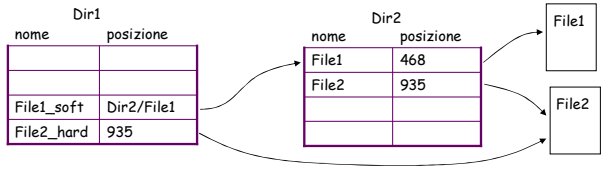


Es: Unix
- /dict/all
- /dict/w/list
- /spell/word/list

sono tutti link allo stesso file

Link

I link possono essere:
- Soft link: collegamento al nome di un file
  - Link simbolico (Unix); Alias (MacOS); collegamento (Windows)
- Hard link: collegamento ai dati di un file



- Rischio di "link invalido" se cambia il nome o la posizione dei file.
- Soluzioni:
  - Conservazione del file fino a quando non esistono più hard link.
  - È sufficiente mantenere il numero di riferimenti: quando il contatore è 0 il file può essere cancellato.

Montaggio di un file system

- Puo' capitare di voler accedere a dati e informazioni che fanno parte di un altro file system
  - (secondo disco rigido, DVD, file system di altri calcolatori connessi in rete)
- Operazione di montaggio
  - Combina piu' file system in un singolo file system, in maniera che possono essere referenziati da una singola root directory
  - Assegna una directory (mount point) del file system nativo alla root del file system da montare
- Il S.O. gestisce le directory montate con una "mount tables":
  - Contiene informazioni sulla posizione del mount point e il dispositivo che memorizza il file system montato

Semantica per il montaggio

- Modalita'
  - Automatico quando il sistema rileva un nuovo dispositivo (Macintosh, Windows)
  - Esplicito mediante comando (Unix-like)
- Mount point
  - accessibile mediante etichetta (Windows)
  - a livello di root (Macintosh)
  - Directory qualunque (Unix)
- Possibilita'
  - Vietare il montaggio su directory piene
  - Nascondere temporaneamente il contenuto di directory piene

Protezione

- Il possessore di un file deve poter controllare gli accessi al file.
  - Cosa puo' essere fatto
  - Da chi puo' essere fatto:

Una soluzione: associare ad ogni file un elenco degli accessi dove sono specificati i nomi degli utenti e le operazioni consentite (access control list)

- Implementato nel s.o. VMS
- Inconvenienti:
  - Elenco lungo e di difficile gestione
  - Campo della directory relativo al file di dimensione variabile

Protezione: esempio Unix

- Esempio Unix: Tre classi di utenti e tre tipi di accesso definiti mediante bit
  - 1 si
  - 0 no

	RWX	
• Proprietario	1 1 1	
• Gruppo	1 1 0	➔761
• Altri	0 0 1	

- Al momento dell'apertura del file il s.o. ritorna un identificativo o un errore a secondo se sono soddisfatti i requisiti di sicurezza o meno
- Solaris 2.6 usa uno schema ibrido

Condivisione di file

- Un problema comune nei file system in ambiente multiutente e' la condivisione dei file
  - Condivisione all'interno di uno stesso file system
  - Condivisione in un Network File System (file system di rete)
- Stabilito chi e come puo' accedere ai file sono necessarie delle regole di sincronizzazione alle risorse (semantica della consistenza)
- Esempio tipo: problema lettori/scrittori

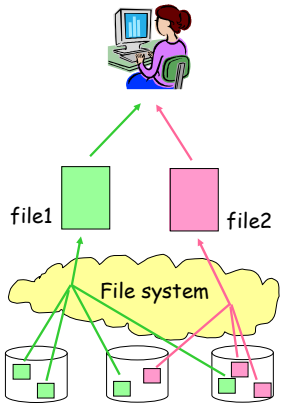
Semantica della consistenza: Esempi

- Unix file system (UFS) :
  - Scritture su un file aperto sono visibili immediatamente ad altri utenti che hanno aperto lo stesso file
  - Realizzato mediante la condivisione del puntatore alla stessa area dati
  - Sincronizzazione a carico del s.o.
  - Possibile inefficienza a causa delle sincronizzazioni
- Andrews file system (AFS) :
  - Le scritture sono visibili agli altri utenti solo dopo la chiusura del file
  - Sincronizzazione a carico dei processi che accedono ai file
  - Maggiore efficienza soprattutto in ambienti distribuiti

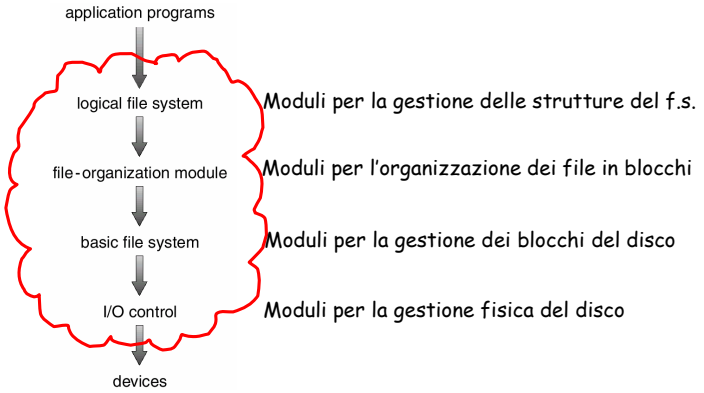
Problema

Un problema fondamentale nella realizzazione di un f.s. e' la definizione di algoritmi e strutture dati per far corrispondere il file system logico (file, directory,...) al file system fisico (dispositivi fisici della memoria secondaria)

Organizzazione a livelli

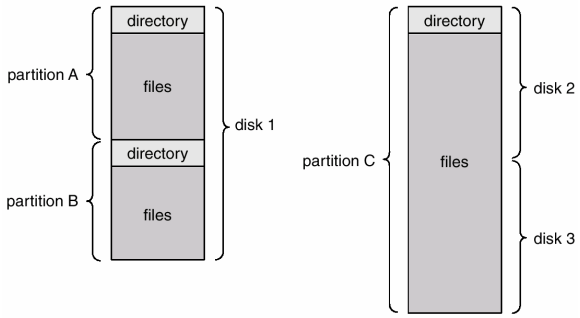


Struttura del file system



Struttura del disco

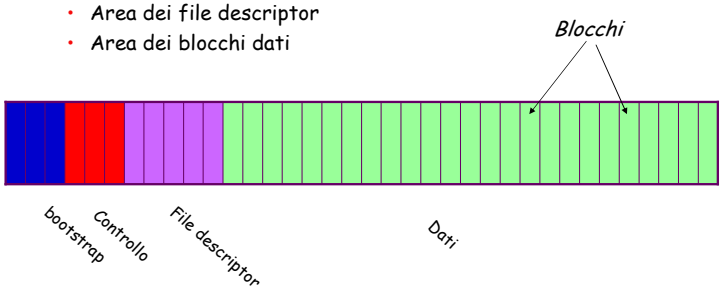
- Il disco e' suddiviso in partizioni (minidischi, volumi, …)
- Ogni file system e' contenuto in un volume e un volume puo' contenere un solo file system



Corrispondenza 1-1 tra file system e partizione

Organizzazione di una partizione

- Dipende dal s.o. e dal file system
- In generale una partizione e' divisa in 4 aree distinte
  - Area di bootstrap
  - Area di controllo
  - Area dei file descriptor
  - Area dei blocchi dati



Oranizzazione di una partizione

- Area di bootstrap
  - Contiene il programma per l'inizializzazione del sistema
  - Di solito e' il primo blocco di una partizione
  - Per uniformita' ce n'e' una in ogni partizione
  - Diversi nomi
    - Boot block (Unix file system)
    - Partition boot sector (NTFS)
- Area dei file descriptor
  - Contiene le strutture dati e gli indirizzi dei blocchi dei dati dei file
  - Diversi nomi
    - i-list (UFS)

Organizzazione di una partizione

- Area di controllo
  - Contiene informazioni su tutta la partizione
    - Dimensione della partizione
    - Numero e posizione dei blocchi liberi
    - Dimensione della i-list (area dei file descriptor)
  - Diversi nomi
    - Superblocco (UFS)
    - Master file table (NTFS)
- Area dei dati
  - Contiene i blocchi dei dati dei file e i blocchi liberi

Area dei blocchi dei dati

- Problema: come allocare lo spazio disco ai file in modo da avere spreco minimo di memoria e rapidità di accesso?

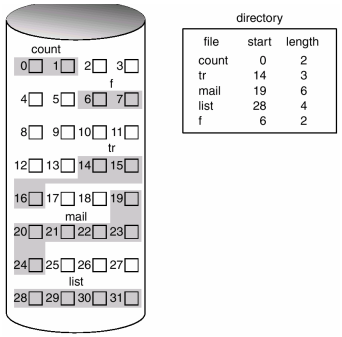
- Il metodo di allocazione dello spazio su disco descrive come i blocchi fisici del disco vengono allocati ai file:
  - Allocazione contigua
  - Allocazione concatenata
  - Allocazione indicizzata

Allocazione contigua

- Ciascun file occupa un insieme di blocchi contigui sul disco.
- Per reperire il file occorrono solo la locazione iniziale (# blocco iniziale) e la lunghezza (numero di blocchi).

Principali problemi

- i file non possono crescere
- frammentazione



directory

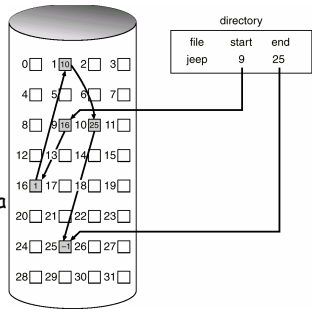
file	start	length
count	0	2
tr	14	3
mail	19	6
list	28	4
f	6	2

Allocazione concatenata

- Ciascun file è una lista concatenata di blocchi su disco:
- i blocchi possono essere sparsi ovunque nel disco.
- La directory contiene l'indirizzo del blocco iniziale.
- Ogni blocco possiede un'area puntatore

Principali problemi

- per trovare un blocco e' necessario scorrere tutta la lista sul disco con numerosi posizionamenti della testina
- Perdita di efficienza



directory

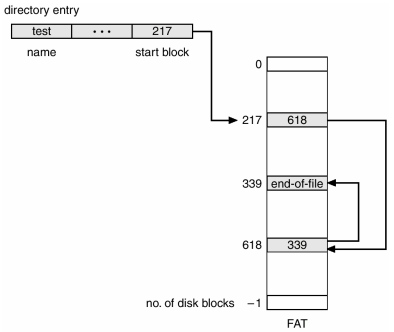
file	start	end
jeep	9	25

Allocazione tabellare

- Rappresenta una possibile soluzione al problema dello scorrimento della lista sul disco
- Mantenere una tabella contenuta in un unico blocco che contiene la lista dei puntatori ai blocchi del disco
- Per trovare un blocco si scorre la lista nella tabella e non sul disco

File Allocation Table
FAT

Implementata nei f.s. di
- MS-DOS
- OS/2



FAT (Microsoft)

- Prima versione della FAT dell'MS-DOS 1.0 (FAT12) usava 12 bit per l'indirizzamento dei blocchi
- 2^{12} = 4096 blocchi

Dim. del disco	Dim. dei blocchi
4 MB	1K
16 MB	4K
64 MB	16K

Blocchi grandi = frammentazione

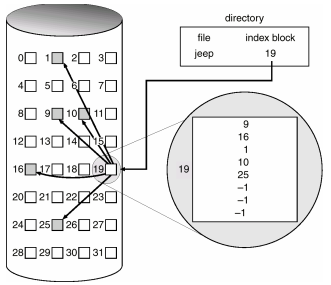
- Successive versioni a 16 e 32 bit

Allocazione indicizzata

- Colleziona tutti i puntatori in un unico *blocco indice*.
- La directory contiene l'indirizzo del blocco indice
- Richiede una tabella indice.
- Permette l'accesso dinamico senza frammentazione esterna;

Principali problemi

- overhead per l'accesso al blocco indice (comunque inferiore all'allocazione concatenata)



Esempio

- Area dati = 64 Mbyte
- Blocchi di 1 Kbyte

65536 blocchi

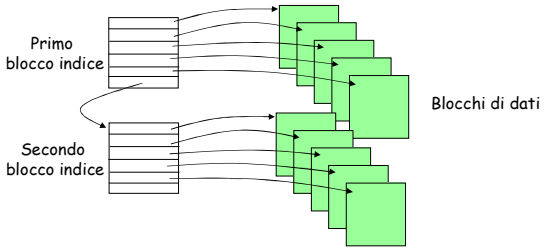
Necessari 16 bit (2 byte) per indirizzarli

512 entry di 16 bit nel blocco indice

Massima dimensione di un file
512 * 1 Kbyte = 512 Kbyte

File di grandi dimensioni

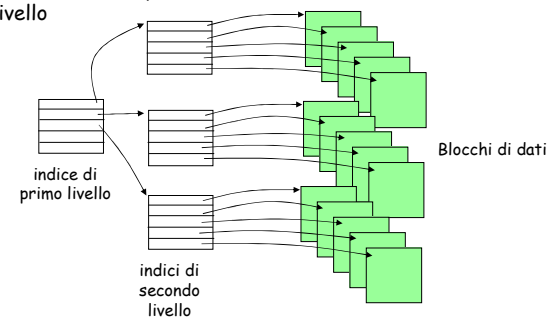
- Nell'esempio precedente se il file ha dimensione maggiore di 256Kbyte e' possibile concatenare i blocchi indice



Allocazione indicizzata – schema concatenata

Una alternativa

- Il blocco indice punta ad altri blocchi indice di secondo (e terzo) livello



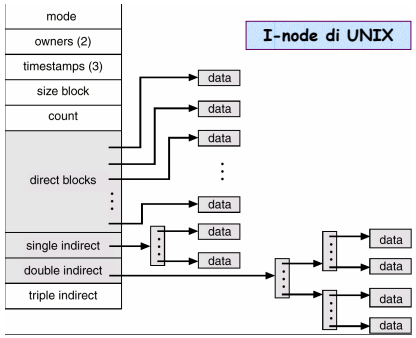
Allocazione indicizzata – schema multilivello
(e multiaccesso !!)

Esempio:

- Blocchi di 1 Kbyte e puntatori di 32 bit (4byte)
- Indirizzamento a 1 livello
  - 1024/4 = 256 blocchi indirizzabili
  - Dimensione massima di un file = 256 Kbyte
- Indirizzamento a 2 livelli
  - 256*256 = 65536 blocchi indirizzabili
  - Dimensione massima di un file = 64 Mbyte
- Indirizzamento a 3 livelli
  - 256*256*256 = ~ 16 milioni di blocchi indirizzabili
  - Dimensione massima di un file = 16 Gbyte (! ! !)

Schema combinato: Unix

- Nel Unix File System, l'area dei descrittori (chiamata I-list: index list) contiene una tabella di descrittori chiamati i-node (index node)
- Ogni i-node e' accessibile attraverso l'indice della tabella (i-number)
- Un i-node contiene gli attributi di un file
- Indirizzi dei blocchi
  - 12 diretti
  - 1 indiretto singolo
  - 1 indiretto doppio
  - 1 indiretto triplo



Indirizzamento

- Blocchi di 1 kbyte
- Indirizzi di 4 byte
- 1024/4 = 256 indirizzi per blocco
- Dimensione massima per un file
  - 12 * 1 Kbyte = 12 Kbyte +
  - 256 * 1 Kbyte = 256 Kbyte +
  - 256^2 * 1 Kbyte = 64 Mbyte +
  - 256^3 * 1 Kbyte = 16 Gbyte =
  - Totale > 16 Gbyte
- Indirizzamenti a 64 bit

Superiore alla capacita' di indirizzamento di 32 bit

osservazione

- Qual'e' l'idea alla basedi tale organizzazione?
- Il numero di accessi al disco e' proporzionale al livello di indirizzamento

- File piccoli (< 12 Kbyte) 2 accessi
- File medi (< 256 Kbyte) 3 accessi
- File grandi (< 64 Mbyte) 4 accessi
- File molto grandi (< 16 Gbyte) 5 accessi

Gestione dello spazio libero

- Poiche' i file sono entita' estremamente dinamiche (aumentano e diminuiscono in numero e in dimensione) e' necessario assicurarsi una gestione efficiente dello spazio libero
- L'area di controllo (superblocco) di una partizione contiene le informazioni sulla posizione dei blocchi liberi
- Tre possibili gestioni:
  - Bitmap
  - Lista dello spazio libero
  - raggruppamento

Bitmap

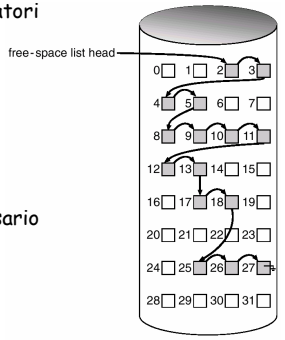
- Vettore di bit (*n* blocchi)

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

- Vantaggi: Buone prestazioni se il vettore è conservato in memoria centrale.
- Svantaggi: per trovare un blocco libero puo' essere necessario esaminare gran parte della bitmap

Lista concatenata

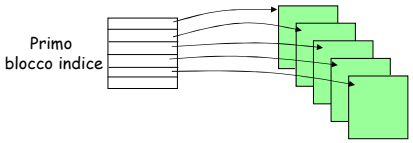
- Lista di blocchi liberi legati da puntatori
- Puntatori alla testa e alla coda sono memorizzati nel superblocco



- Vantaggio: 1 accesso per trovare un blocco libero
- Svantaggio: inefficiente se e' necessario trovare molti blocchi

Raggruppamento

- Raggruppamento: memorizzazione degli indirizzi di *n* blocchi liberi sul primo di tali blocchi. Sull'ultimo sono indirizzati altri blocchi, etc.

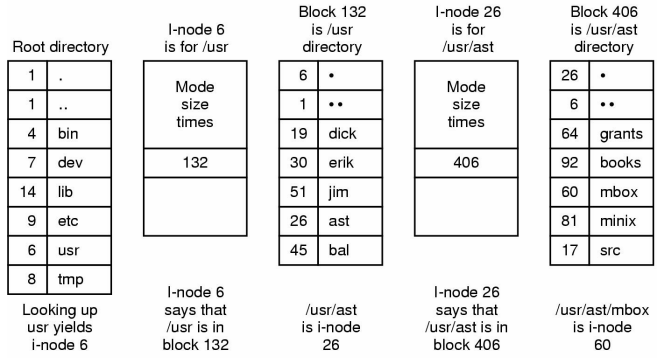


- Vantaggio: rapida ricerca di blocchi liberi
- Svantaggio: gestione inefficiente se il numero di blocchi e' grande

Esempio di implementazione di file system

UNIX V7

Qual'e' l'i-node del file /usr/ast/mbox?

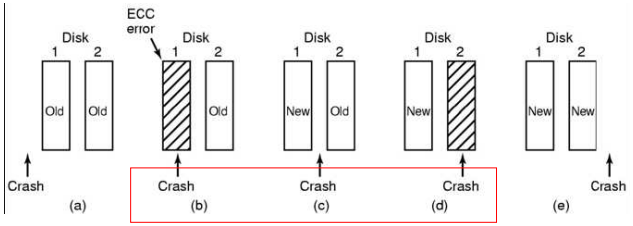


Efficienza e prestazioni

- Per migliorare le prestazioni, parte dei blocchi su disco sono tenuti in un buffer in memoria (cache di disco)
  - Sezione della memoria centrale impiegata per blocchi molto utilizzati
  - l'implementazione è analoga alla gestione della paginazione
- Quante volte un blocco viene aggiornato?
  - periodicamente (UNIX)
  - ad ogni richiesta di scrittura (MS-DOS)
    - estrarre un dischetto sotto MS-DOS non è critico, con UNIX occorre *smontare* il dispositivo
- Blocchi utilizzati in sequenza dovrebbero essere memorizzati vicino per minimizzare i tempi di posizionamento delle testine (deframmentazione del disco - richiede però molto tempo)

Consistenza del file system

- A seguito di un crollo del sistema, se alcune informazioni nella memoria centrale non sono state salvate, si può verificare un'incoerenza dei dati memorizzati sul disco



Incoerenza

- A seguito di un crollo del sistema, se alcune informazioni nella memoria centrale non sono state salvate, si può verificare un'incoerenza dei dati memorizzati sul disco
- Uso tipico dei file:
  - si legge un blocco da disco
  - si modifica il blocco in memoria
  - lo si riscrive
- Se si verifica un crash:
  - alcuni blocchi non sono scritti
  - il file system diventa incoerente

Tecniche per garantire la coerenza

- Il problema della consistenza e' particolarmente rilevante per "blocchi critici" (blocchi che contengono FAT, i-node, bitmap,…)
- Due possibili soluzioni
  - curare (file system checker)
    - Strumenti come fsck (Unix), scandisk (windows)
  - prevenire (file system con annotazioni)
    - Ext3, ntfs

Controlli di coerenza

- Automatico o su richiesta dell'utente
- Si costruiscono due tabelle used e free che contengono un elemento per ogni blocco dati del sistema
  - Tabella used
    - used[i]: Conta quante volte l'i-mo blocco e' presente nell'elenco dei blocchi usati
    - Si inizializza a 0, si incrementa se il blocco viene assegnato ad un file e si decrementa se viene rilasciato
  - Tabella free
    - free[i]: conta quante volte l'i-mo blocco compare tra i blocchi liberi del sistema
    - Si inizializza a 1, si decrementa quando il blocco viene assegnato ad un file e si incrementa quando ritorna libero

Controlli di coerenza (1)

- free[i]+used[i]=1

1	1	0	1	0	1	1	1	1	0	0	1	1	1	0	0	used
0	0	1	0	1	0	0	0	0	1	1	0	0	0	1	1	free

  - Nessun errore. Ogni blocco e' correttamente elencato nel sistema
- free[i]+used[i] = 0

1	1	0	1	0	1	1	1	1	0	0	1	1	1	0	0	used
0	0	0	0	1	0	0	0	0	1	1	0	0	0	1	1	free

  - Blocco mancante. Nessun problema di coerenza ma spazio sprecato. Soluzione: si pone free[i]=1

Controlli di coerenza (2)

- free[i] > 1

1	1	0	1	0	1	1	1	1	0	0	1	1	1	0	0	used
0	0	2	0	1	0	0	0	0	1	1	0	0	0	1	1	free

  - Nessun problema di coerenza ma il blocco potrebbe essere assegnato due volte. Soluzione: si pone free[i]=1

- free[i]=used[i] = 1

1	1	1	1	0	1	1	1	1	0	0	1	1	1	0	0	used
0	0	1	0	1	0	0	0	0	1	1	0	0	0	1	1	free

  - Blocco libero o occupato? Nessun problema ma il blocco potrebbe essere assegnato due volte . Soluzione: si pone free[i]=0

Controlli di coerenza (3)

- used[i] > 1

1	1	2	1	0	1	1	1	1	0	0	1	1	1	0	0	used
0	0	0	0	1	0	0	0	0	1	1	0	0	0	1	1	free

  - Il blocco e' assegnato a due file. Gravi problemi di coerenza. Soluzione: si pone used[i]=1 ma un file risultera' corrotto

File system annotati (es. ext3/Linux e NTFS)

- I file system annotati (*journaling file system*) annotano tutte le modifiche in un file di log (*giornale delle modifiche).*
  - si annotano le modifiche da effettuare al file system in un file di log
  - Si aggiorna il file system
  - Quando le operazioni si considerano portate a termine, si rimuove l'annotazione e le chiamate di sistema ritornano.
- Nel caso di caduta del sistema durante la modifica, si vede nel file di log cosa non e' stato portato a termine e si ripristina la coerenza del file system
- Tali tecniche hanno un impatto anche sulle prestazioni del sistema, in quanto velocizzano gli aggiornamenti percepiti dalle applicazioni.
  - Le modifiche avvengono con accessi sequenziali, invece che con accessi diretti e sincroni alle strutture dati del file system.

Sommario

- Il file system risiede permanentemente nella memoria secondaria, che di solito è un disco, che viene partizionato per controllarne l'uso e consentire la coesistenza di file system differenti.

- Ogni file system utilizza i suoi algoritmi e le sue strutture dati; ai file si può assegnare spazio in maniera contigua, concatenata ed indicizzata.

- I metodi di assegnazione dello spazio libero influenzano l'efficienza d'uso dei dischi, le prestazioni e l'affidabilità.