Il File-System

I direttori in ambiente Linux

Stefano Quer
Dipartimento di Automatica e Informatica
Politecnico di Torino

Direttori

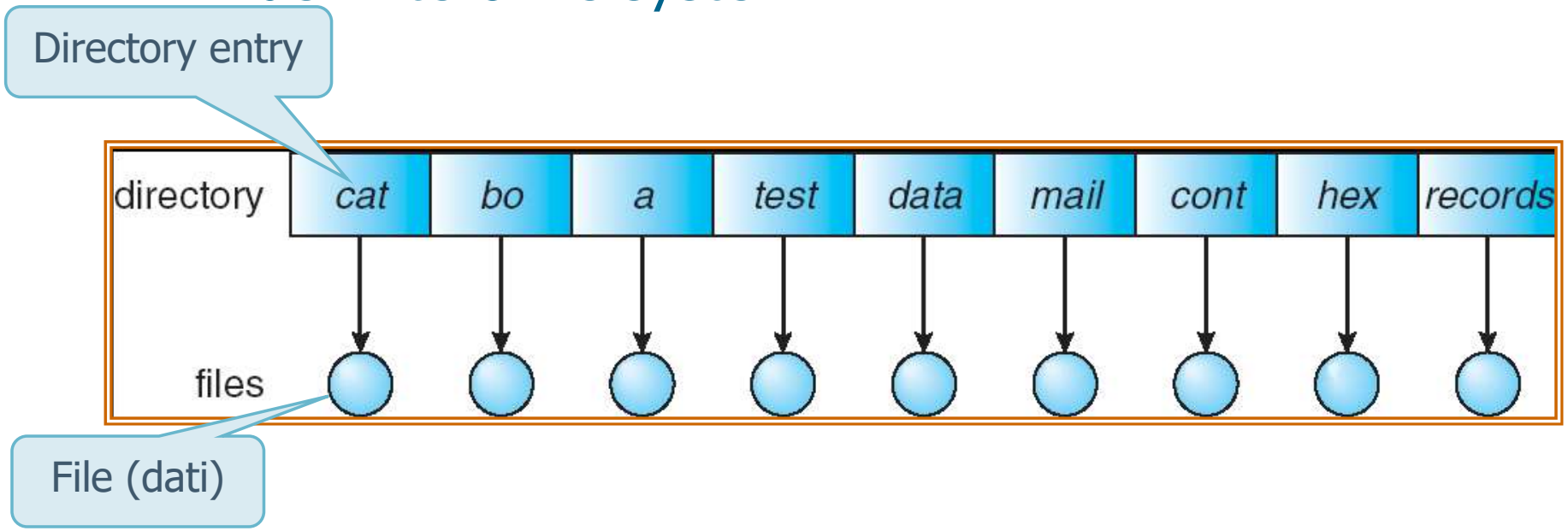
- Nessun sistema di memorizzazione contiene un unico file
- I file sono organizzati in direttori
  - Un direttorio è un nodo (di albero) o vertice (di grafo) contenente informazioni sugli elementi in esso contenuti
  - Direttori e file risiedono entrambi su memoria di massa
- Su un direttorio sono effettuabili operazioni simili a quelle dei file
  - Creazione, cancellazione, elenco del contenuto, ri-denominazione, visita, ricerca, etc.

Struttura

- La struttura di un direttorio dipende da ragioni di
  - Efficiency (efficienza)
    - Velocità nel manipolare il file system, e.g., localizzare un file
  - Naming (convenienza)
    - Semplicità per un utente di identificare i propri file
    - Evitare che lo stesso nome attribuito a più file crei problemi
  - Grouping (organizzazione)
    - Raggruppare le informazioni in base alle relative caratteristiche
      - Programmi di editing, compilatori, giochi, etc.

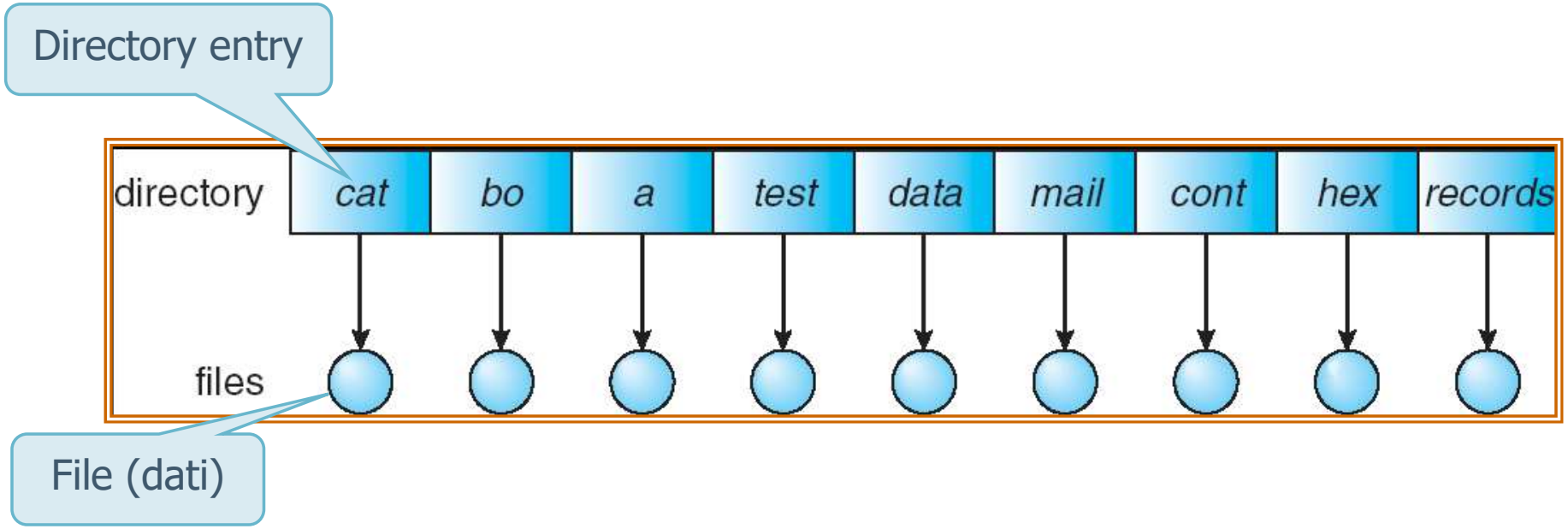
Direttori a un livello

- La struttura più semplice è quella a livello singolo
- Tutti i file del file system sono contenuti all'interno dello stesso direttorio
  - I file sono differenziati dal solo nome
  - Ciascun nome deve essere univoco all'interno dell'intero file system



Direttori a un livello

- Per ciascun file si evidenziano
  - La directory entry: individua il nome e eventualmente altre informazioni del file
  - I dati: separati dalla directory entry, sono da essa individuati tramite puntatore

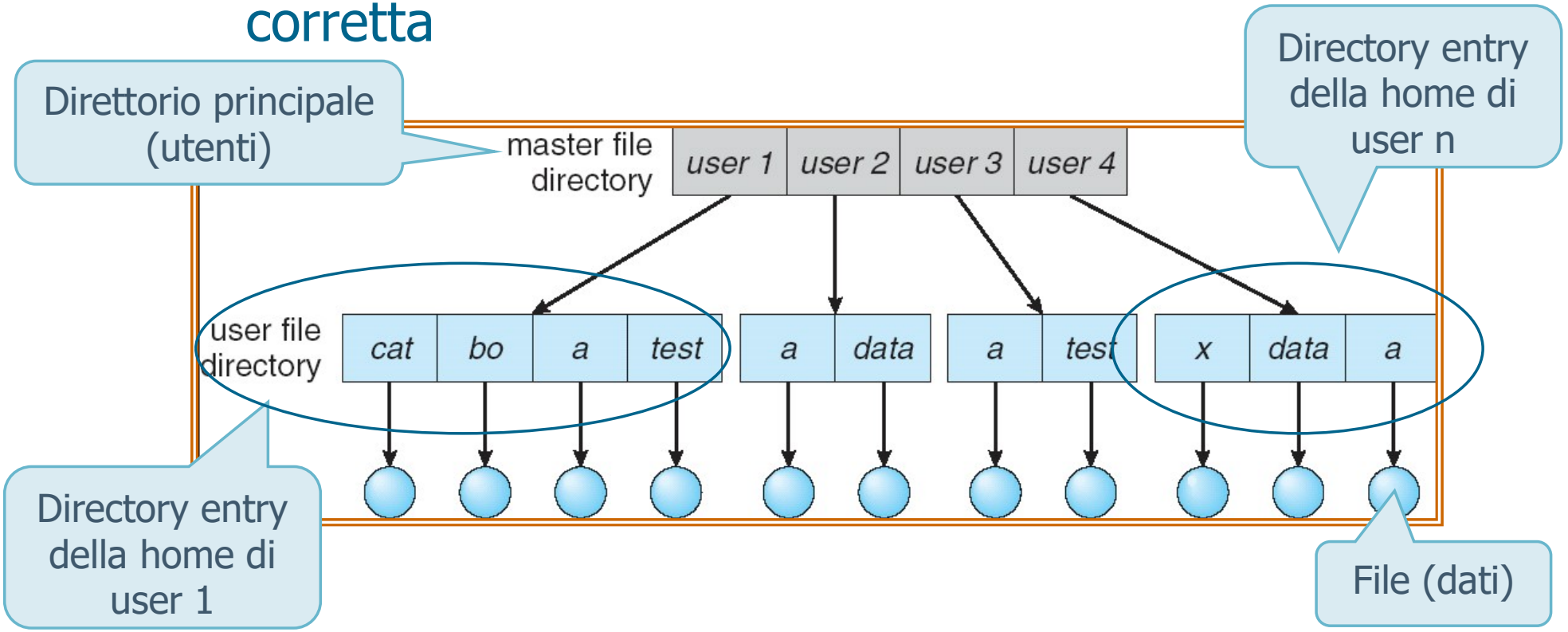


Direttori a un livello

- Prestazioni
  - Efficiency
    - Struttura facilmente comprensibile e gestibile
    - Gestione del file system semplice ed efficiente
  - Naming
    - I file devono avere nomi univoci
    - Presenta limiti evidenti all'aumentare del numero di file memorizzati
  - Grouping
    - La gestione dei file di un utente singolo è complessa
    - La gestione di utenti multipli è praticamente impossibile

Direttori a due livelli

- I file sono contenuti in un albero a due livelli
- Ogni utente può avere il proprio direttorio
  - Ogni user ha la sua home directory
  - Tutte le operazioni sono eseguite solo sulla home corretta

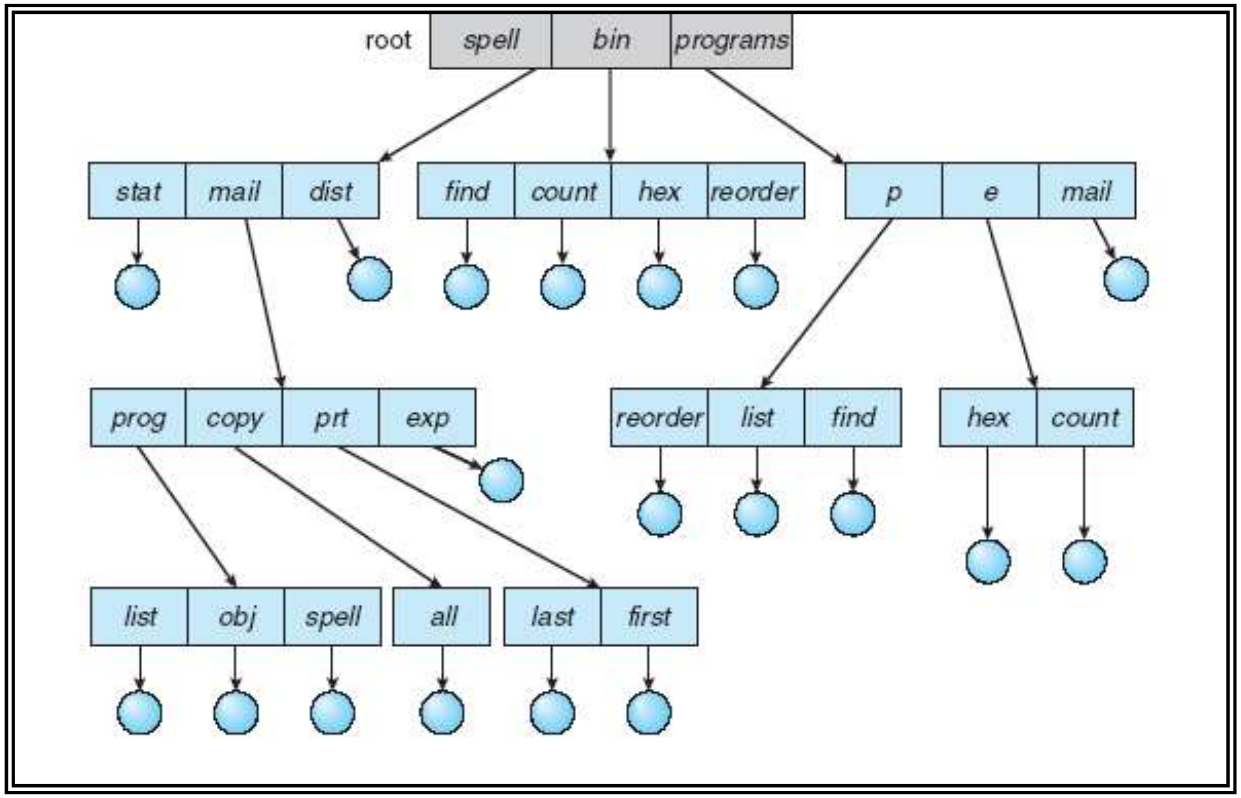


Direttori a due livelli

- Prestazioni
  - Efficiency
    - Visione del file-system "user oriented"
    - Ricerche semplificate e efficienti agendo su utenti singoli
  - Naming
    - È possibile avere file con lo stesso nome purchè appartenenti a utenti diversi
    - Occorre specificare un path-name per ogni file
  - Grouping
    - Semplificato tra utenti diversi
    - Complesso per ciascun utente singolo

Direttori ad albero

- Generalizza i precedenti
- I file sono contenuti in un albero
  - Ogni nodo/vertice può contenere come entry un altro nodo/vertice dell'albero



Direttori ad albero

- Ogni utente può gestire tanto file quanto direttori e sotto-direttori
  - Nascono i concetti di working directory, cambio direttorio, path assoluto e relativo, etc.
- Prestazioni
  - Efficiency
    - Ricerche vincolate alla struttura ad albero e quindi alla sua profondità e ampiezza
  - Naming
    - Permesso in maniera estesa
  - Grouping
    - Permesso in maniera estesa

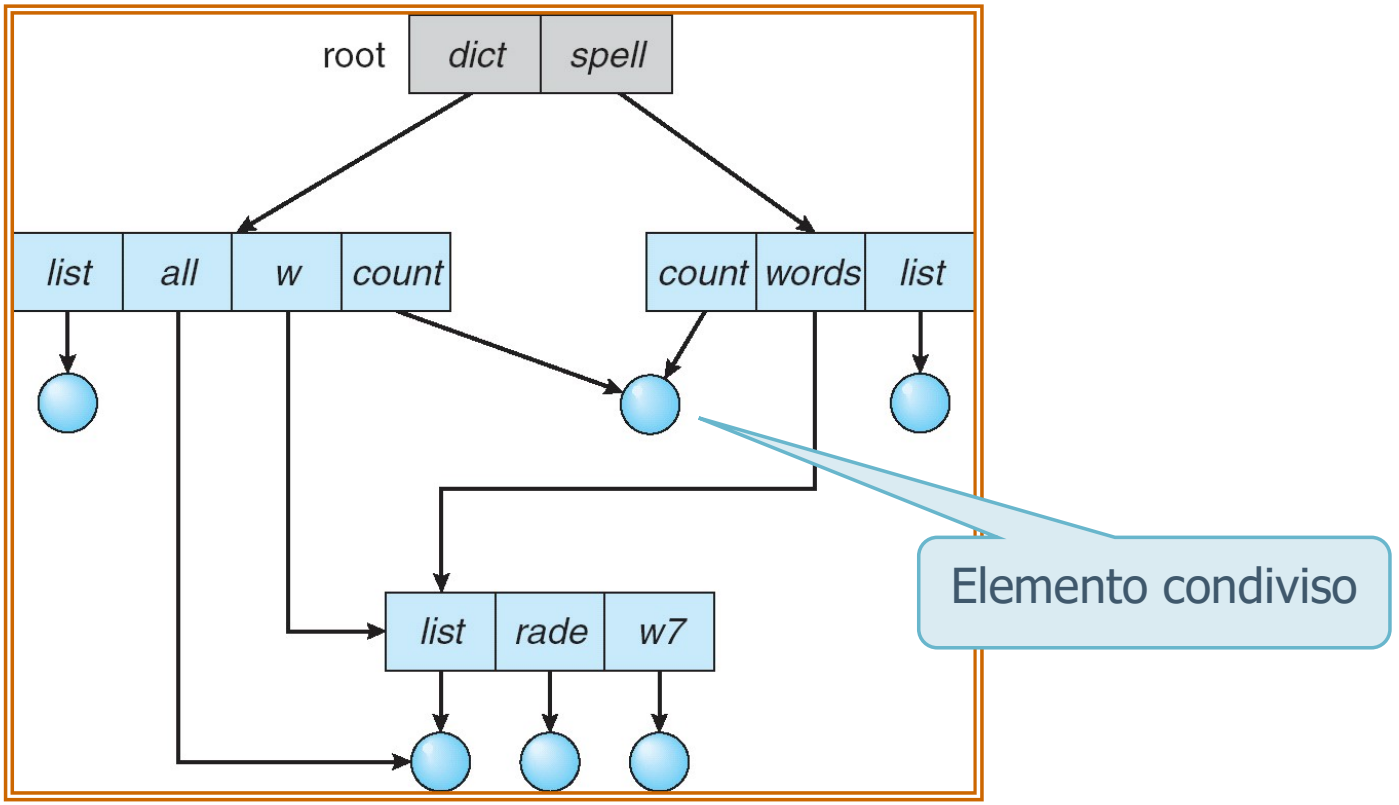
Concetti analizzati nella parte sperimentale Linux

Direttori a grafo aciclico

- File system a albero non permettono la **condivisione** di informazioni
- Spesso è utile individuare lo stesso oggetto con nomi diversi
  - Da parte dello stesso utente con path diversi
  - Da parte di utenti diversi
  - Si osservi che la duplicazione delle informazioni (copia) non risolve tale problema a causa di
    - Aumento dello spazio occupato dal file system
    - Coerenza delle informazioni presenti nelle varie copie

Direttori a grafo aciclico

- I file system a grafo orientato aciclico (cioè senza cicli)
  - Permettono di condividere informazioni, rendendola visibile con path diversi



Direttori a grafo aciclico

- La presenza di link aumenta la difficoltà di gestione dei file system
  - Occorre distinguere gli oggetti nativi dai relativi collegamenti in fase di creazione, manipolazione e cancellazione
  - Creazione
    - La condivisione di un oggetto può essere ottenuta in diversi modi
    - Nei sistemi UNIX-like la strategia standard è quella della creazione di **collegamenti** o **link**
      - Un link è un riferimento (puntatore) a un oggetto pre-esistente

Direttori a grafo aciclico

- Visita e ricerca
  - Se l'oggetto in realtà è un link occorre accorgersene e effettuare un indirizzamento indiretto, ovvero "risolvere" il collegamento, utilizzandolo per raggiungere l'oggetto originario
  - Tramite link ogni oggetto del file system può essere raggiungibile con più path assoluti e con nomi diversi
    - Analisi del file system (statistiche, e.g., quanti file di estensione ".c" sono presenti?) diventano molto più complesse

Direttori a grafo aciclico

- Cancellazione
  - Occorre stabilire come gestire il link e come gestire l'oggetto riferito
    - La cancellazione di un link in genere viene effettuata in maniera immediata e non influisce sull'oggetto originale
    - Occorre però decidere come effettuare la cancellazione dell'oggetto
      - Se si cancella l'oggetto che cosa si fa dei link che lo riferiscono?
      - Quando si riutilizza lo spazio ad esso riservato?

Direttori a grafo aciclico

- Cancellazione immediata dei dati
  - È possibile lasciare dei link pendenti (dangling)
  - Quando si cercherà di utilizzare il link in futuro ci si accorgerà che l'oggetto riferito è scomparso

Soft-link UNIX

- Cancellazione dei dati alla cancellazione dell'ultimo link
  - Per evitare link pendenti occorre tenerne traccia, ovvero occorre gestire la presenza di oggetti e link multipli
    - Mantenere l'elenco di tutti i link è costoso (lista di lunghezza variabile)
    - Cancellare tutti i link (le entry) alla cancellazione dell'oggetto è costoso, in quanto occorrerebbe ricercarli

Hard-link UNIX

  - Conviene memorizzare solo il contatore (numero di link)
    - Nei sistemi UNIX tale contatore viene memorizzato negli i-node
    - Esso viene incrementato e decrementato in maniera opportuna

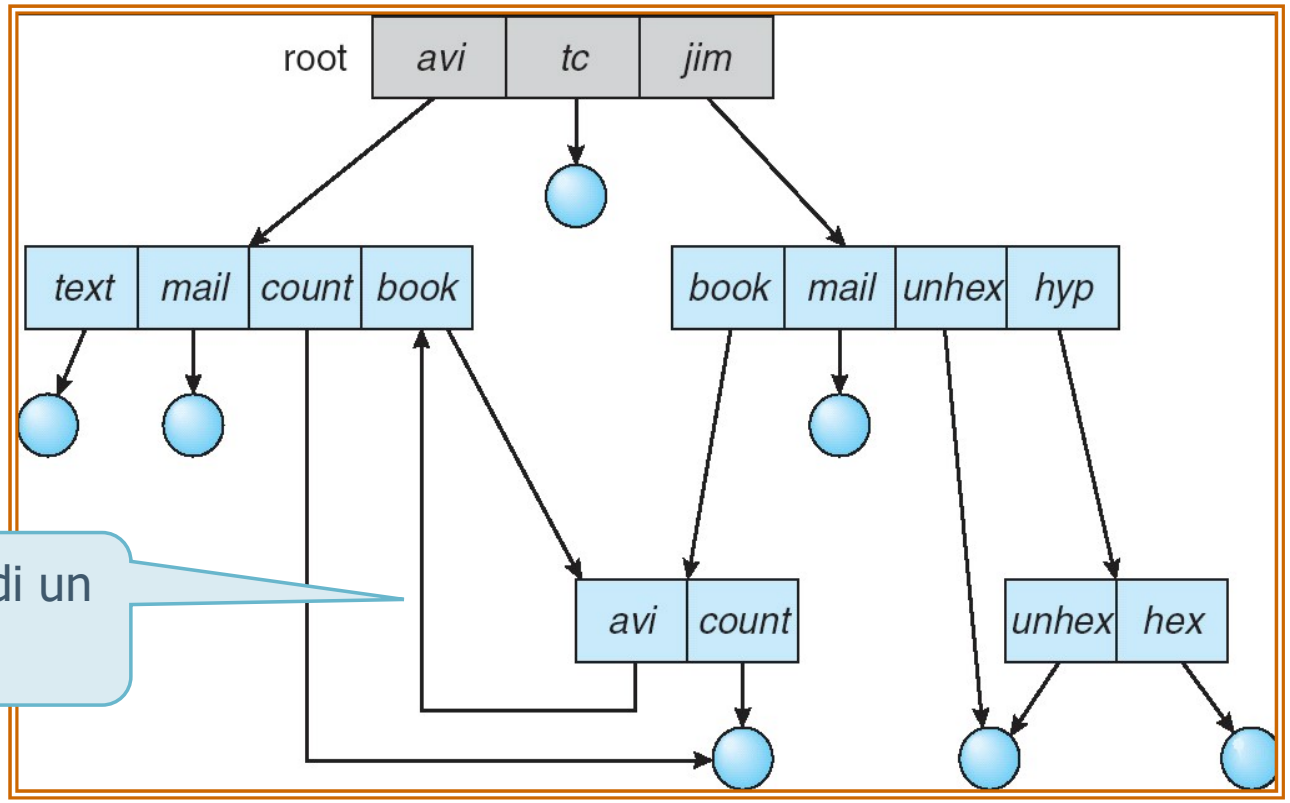
Comando "ls –i"

Direttori a grafo aciclico

- Inoltre la creazione di un link a un direttorio potrebbe causare la nascita di un ciclo nel file system
  - La gestione di un grafo ciclico richiede operazioni di maggiore complessità
    - Un comando di ricerca o di visita dovrebbe verificare la presenza di cicli per evitare ricorsioni infinite
  - Tra le possibili strategie la più semplice è non permette la creazione di link a direttori

Direttori a grafo ciclico

- L'alternativa al grafo aciclico e quella del grafo ciclico ovvero occorre
  - Permettere la creazione di cicli
  - Gestire opportunamente i cicli esistenti in tutte le fasi

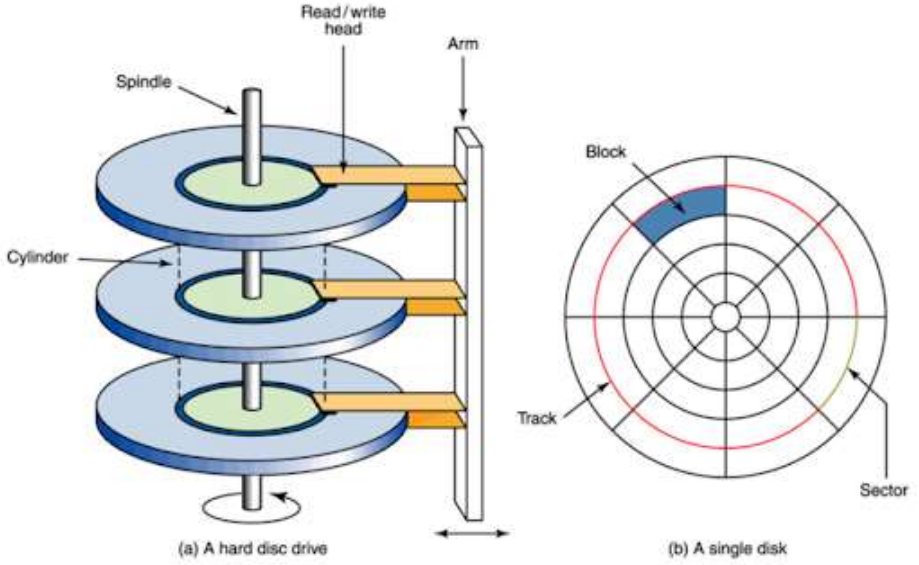


Direttori a grafo ciclico

- La gestione può essere effettuata con diversi approcci che dovrebbero tenere conto di diverse problematiche
  - Un elemento potrebbe auto-referenziarsi e non essere mai cancellato e/o rilevato
  - La gestione più semplice consiste nel non visitare **mai** i link (oppure sotto-categorie dei link)

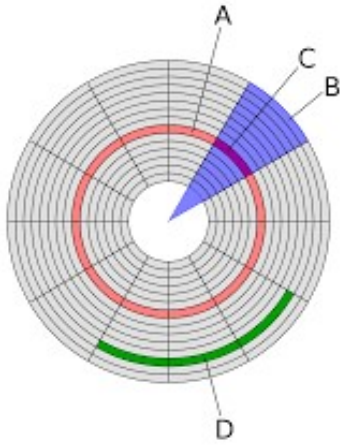
Allocazione

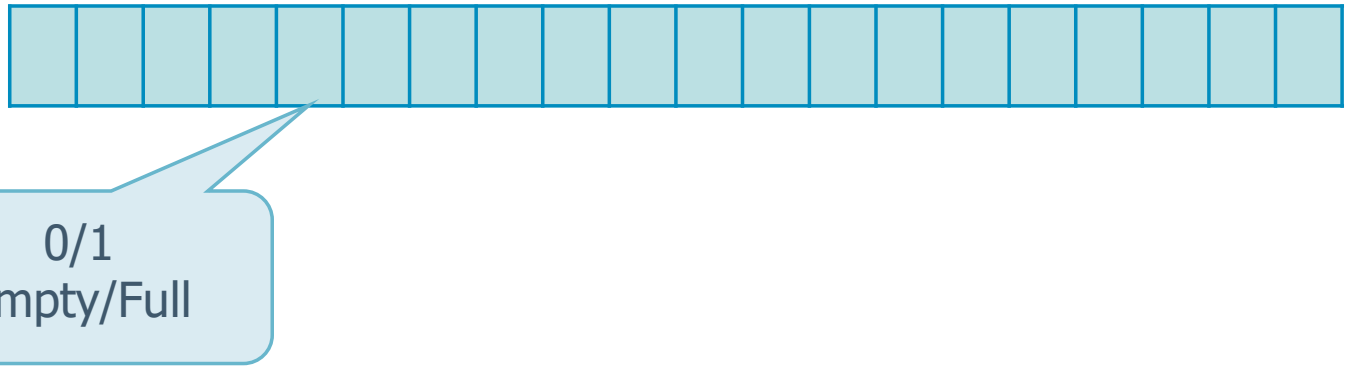
- Per **allocazione** si intendono le tecniche di utilizzo dei blocchi dei dischi per la memorizzazione di file
  - Non ci occuperemo della struttura delle unità di memorizzazione
  - In ogni caso tali unità possono essere viste come un insieme indicizzabile lineare (vettore) di blocchi



Allocazione

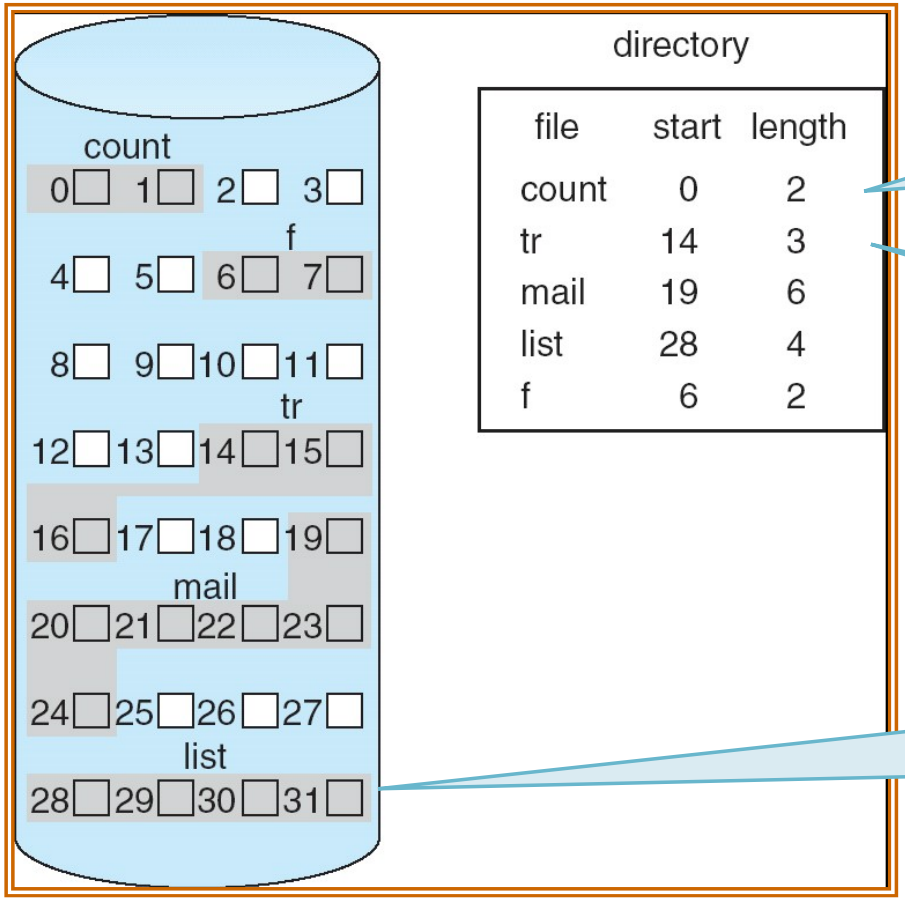
- Metodologie di allocazione principali
  - Contigua (contiguous)
  - Concatenata (linked)
  - Indicizzata (indexed)





Allocazione contigua

- Ogni file occupa un insieme contiguo di blocchi



Per ciascun file il direttorio specifica l'indirizzo del primo blocco (b) e la lunghezza del file (n)

Il file occupa i blocchi b, b+1, b+2, … , b+n-1

Ogni file presenta frammentazione interna (ultimo blocco parzialmente occupato)

Allocazione contigua

- Vantaggi
  - Strategia di allocazione molto semplice
    - Per ogni file si memorizzano poche informazioni
  - Permette accessi sequenziali immediati
    - Ogni blocco si trova dopo il precedente e prima del successivo
  - Permette accessi diretti semplici
    - L'i-esimo blocco a partire dal blocco b si trova all'indirizzo b+i-1

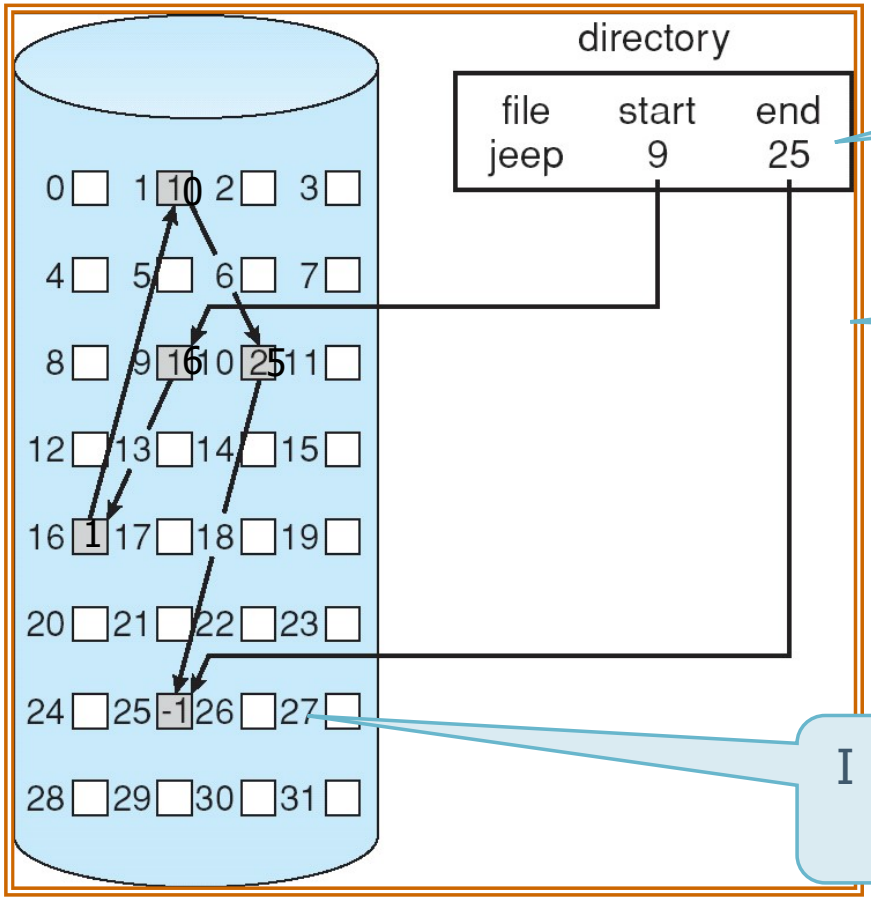
Allocazione contigua

- Svantaggi
  - Occorre decidere la politica di allocazione
    - Dove vengono allocati i file nuovi?
      - Algoritmi di first-fit, best-fit, worst-fit, etc.
      - Come è possibile determinare lo spazio necessario?
  - Nessun algoritmo di allocazione risulta privo di difetti quindi la tecnica spreca spazio
    - Tale spreco è noto come **frammentazione esterna** (insieme dei blocchi non utilizzati)
    - Possibile la ri-compattazione (on-line e off-line)
  - Problemi di allocazione dinamica
    - I file non possono crescere liberamente in quanto lo spazio disponibile è limitato dal file successivo

Occorre ricercare uno spazio libero di dimensione sufficiente

Allocazione concatenata

- Ogni file può essere allocato gestendo una lista concatenata di blocchi



Il direttorio contiene un puntatore al primo e uno all'ultimo blocco del file

Ogni blocco contiene un puntatore al blocco successivo

I blocchi di ciascun file sono sparsi per l'intero disco

Allocazione concatenata

- Vantaggi
  - Risolve i problemi dell'allocazione contigua
    - Permette l'allocazione dinamica dei file
    - Elimina la frammentazione esterna
    - Evita l'utilizzo di algoritmi di allocazione complessi

Allocazione concatenata

- Svantaggi
  - Ogni lettura implica un accesso sequenziale ai blocchi
  - Risulta efficiente solo per accessi sequenziali
    - Un accesso diretto richiede la lettura di una catena di puntatori sino a raggiungere l'indirizzo desiderato
    - Ogni accesso a un puntatore implica una operazione di lettura dell'intero blocco
  - La memorizzazione dei puntatori
    - Richiede spazio
    - È critica dal punto di vista dell'affidabilità
    - Rende lo spazio utile minore
      - Lo spazio disponibile non è più una potenza di 2

Allocazione concatenata: FAT

- File Allocation Table (FAT)
  - File system sviluppato inizialmente da IBM e Digital Equipment Corporation e poi da Bill Gates e Marc McDonald per MS-DOS
  - È il file system primario per diversi sistemi Microsoft Windows fino alla versione Windows ME
    - Windows NT e le successive versioni hanno introdotto **l'NTFS** e mantenuto la compatibilità con la FAT così come molti altri sistemi operativi moderni
  - È una variante del metodo di allocazione concatenato

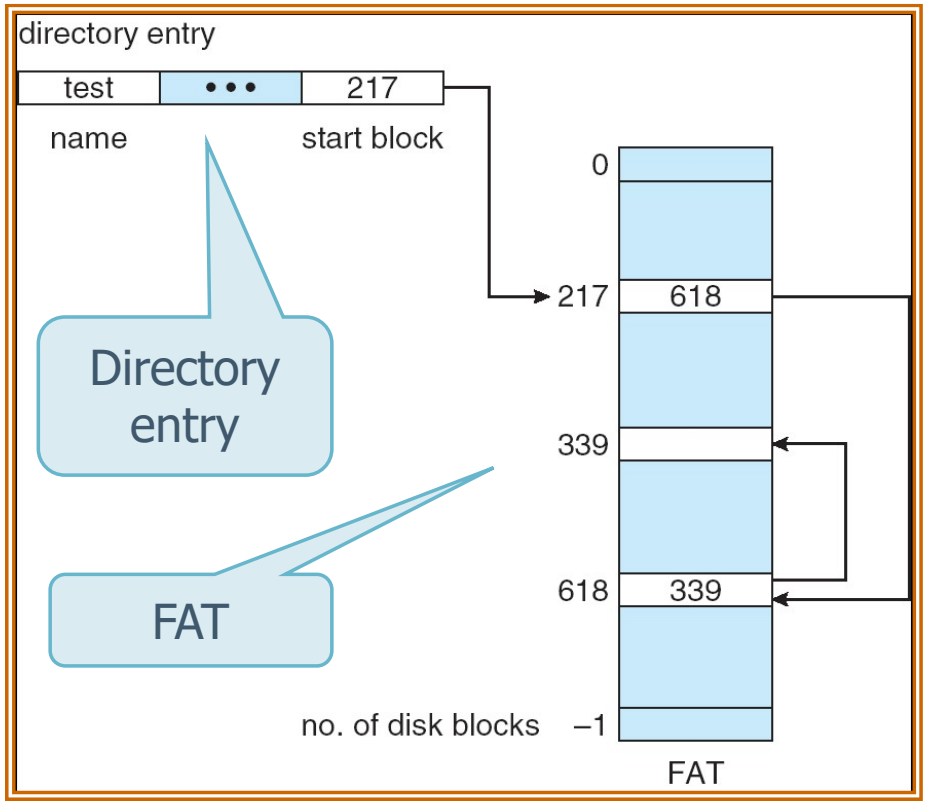
Allocazione concatenata: FAT

- Il blocco degli indici contiene la FAT
  - Tabella con un elemento per ciascun blocco presente sul disco
  - La sequenza dei blocchi appartenenti a un file è individuata a partire dalla directory mediante
    - Elemento di partenza del file nella FAT
    - Sequenza di puntatori presenti (direttamente) nella FAT (non più nei blocchi)

Sposta i puntatori dai vari blocchi a un blocco specifico

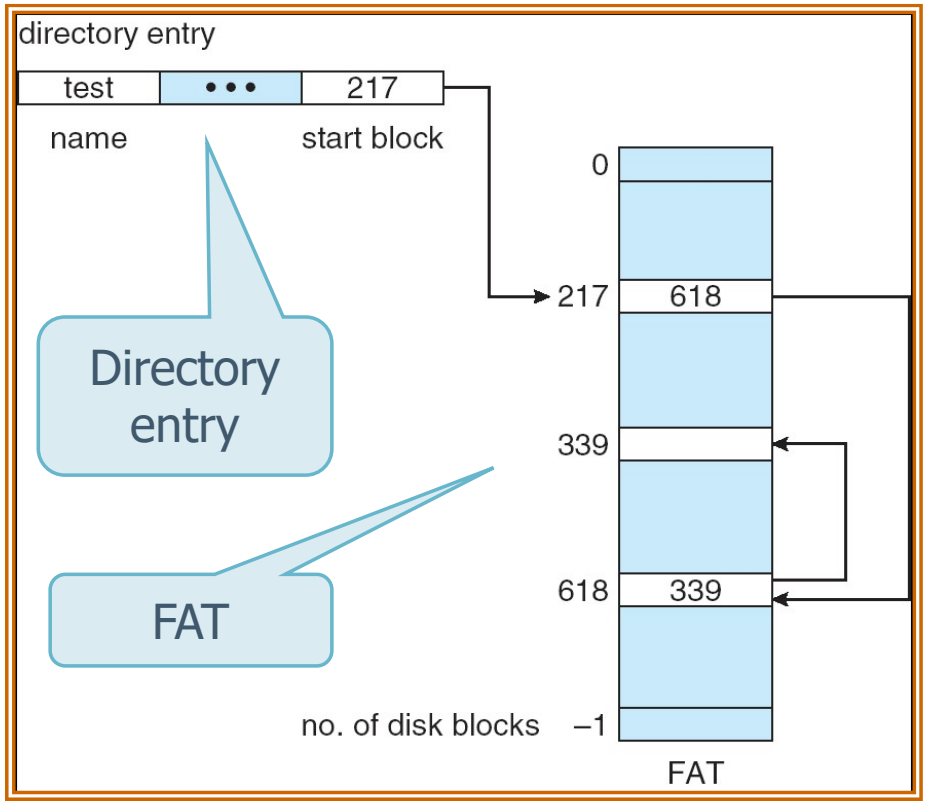
Allocazione concatenata: FAT

- I riferimenti non sono memorizzati nei blocchi su disco ma direttamente negli elementi della FAT
- La lettura di ogni blocco richiede due accessi a disco
  - Il primo accesso viene effettuato alla FAT
  - Il secondo, al blocco dati



Allocazione concatenata: FAT

- Limiti
  - Accesso lento
  - L'affidabiità è critica (persa la FAT si perde tutto)
  - La dimensione della FAT è critica

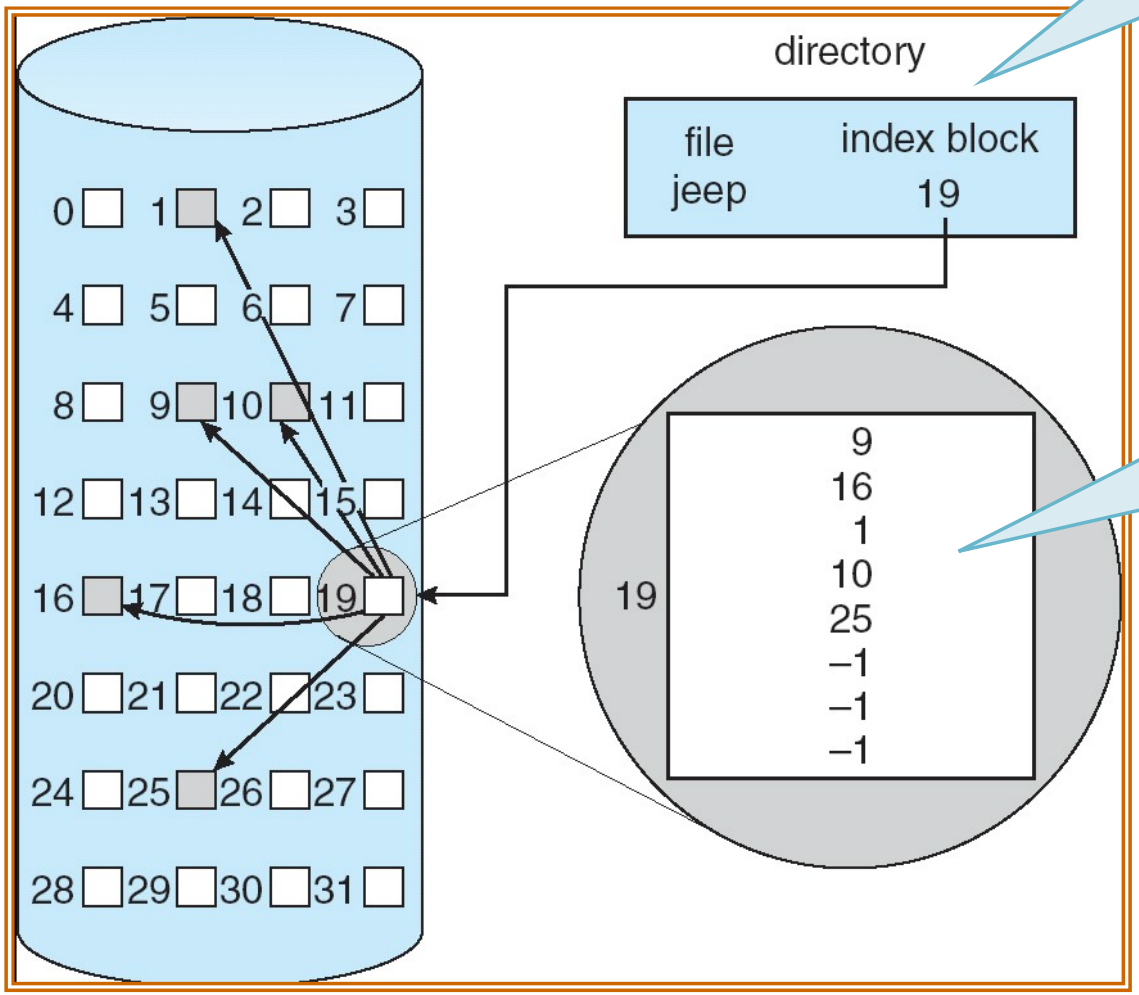


Allocazione indicizzata

- Per permettere un accesso diretto efficiente è possibile inglobare tutti i puntatori in una tabella di puntatori
  - Tale tabella di puntatori è solitamente denominata **blocco indice** o index-node (**i-node**)
- Ogni file ha la sua tabella, ovvero un vettore di indirizzi dei blocchi in cui il file è contenuto
  - L'i-esimo elemento del vettore individua l'i-esimo blocco del file

Allocazione indicizzata

Il direttorio contiene il solo puntatore al blocco indice



Non è una FAT perchè i puntatori sono tutti in sequenza (**non** si ha una **lista** di puntatori)

Allocazione indicizzata

- Rispetto all'allocazione concatenata occorre sempre allocare un blocco indice
  - Blocchi indice di dimensione ridotta permettono di non sprecare troppo spazio
  - Blocchi indice di dimensione elevata aumentano in numero di riferimenti inseribili nel blocco indice
    - In ogni caso occorre gestire la situazioni il cui il blocco indice **non** è sufficiente a contenere tutti i puntatori ai blocchi del file
    - Esistono diversi schemi
      - A blocchi indice concatenati
      - A blocchi indice a più livelli
      - **Combinato**

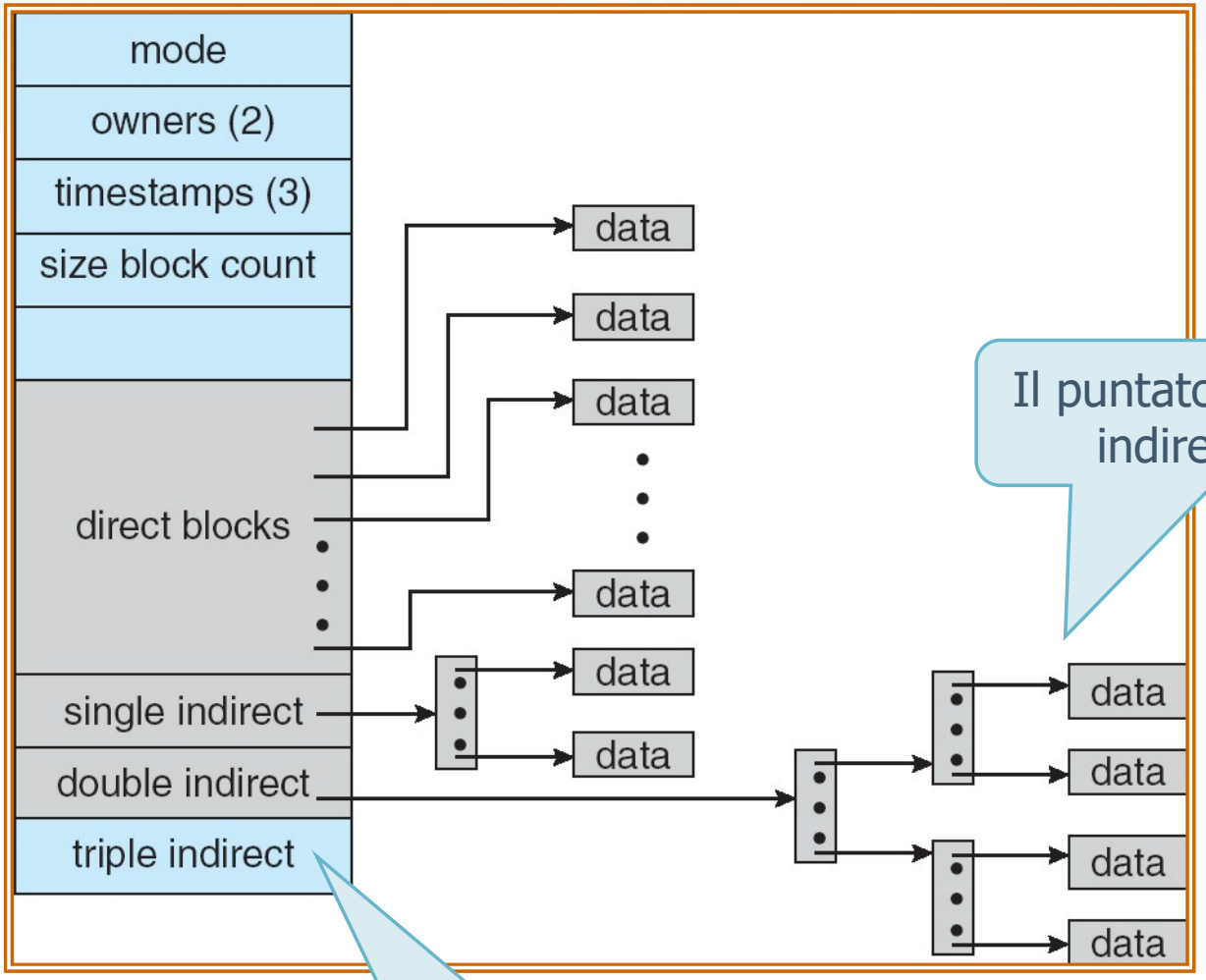
Schema UNIX/Linux

Allocazione indicizzata: schema combinato

- Lo schema combinato è utilizzato nei sistemi UNIX/Linux
- A ogni file è associato un blocco detto **i-node**
- Ogni **i-node** contiene diverse informazioni tra cui 15 puntatori ai blocchi dati del file
  - I primi 12 puntatori sono puntatori diretti, ovvero puntano a blocchi dei file
  - I puntatori 13, 14 e 15 sono puntatori indiretti, con livello di indirizzamento crescente
    - Il blocco individuato non contiene i dati ma i puntatori (i puntatori ai puntatori) [i puntatori ai puntatori ai puntatori] a blocchi di dati del file

Allocazione indicizzata: schema combinato

Ricordare comandi "ls –la" e "ls –i"



Il puntatore 14 è di tipo indiretto doppio

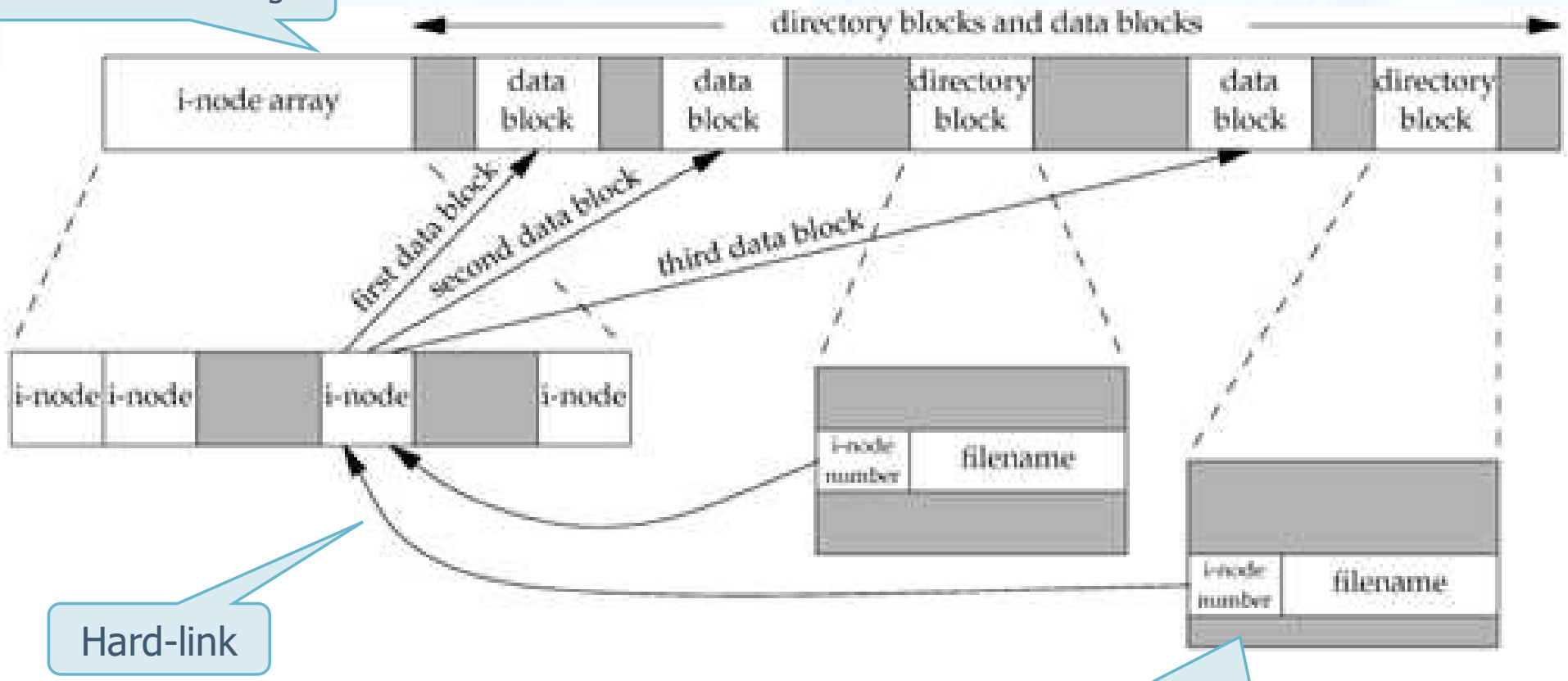
Il puntatore 13 è di tipo indiretto singolo

Il puntatore 15 è di tipo indiretto triplo

Con puntatori a 64 bit si memorizzano file sino a 2^{60} byte (exabyte)

Allocazione indicizzata: schema combinato

Da Stevens & Rago

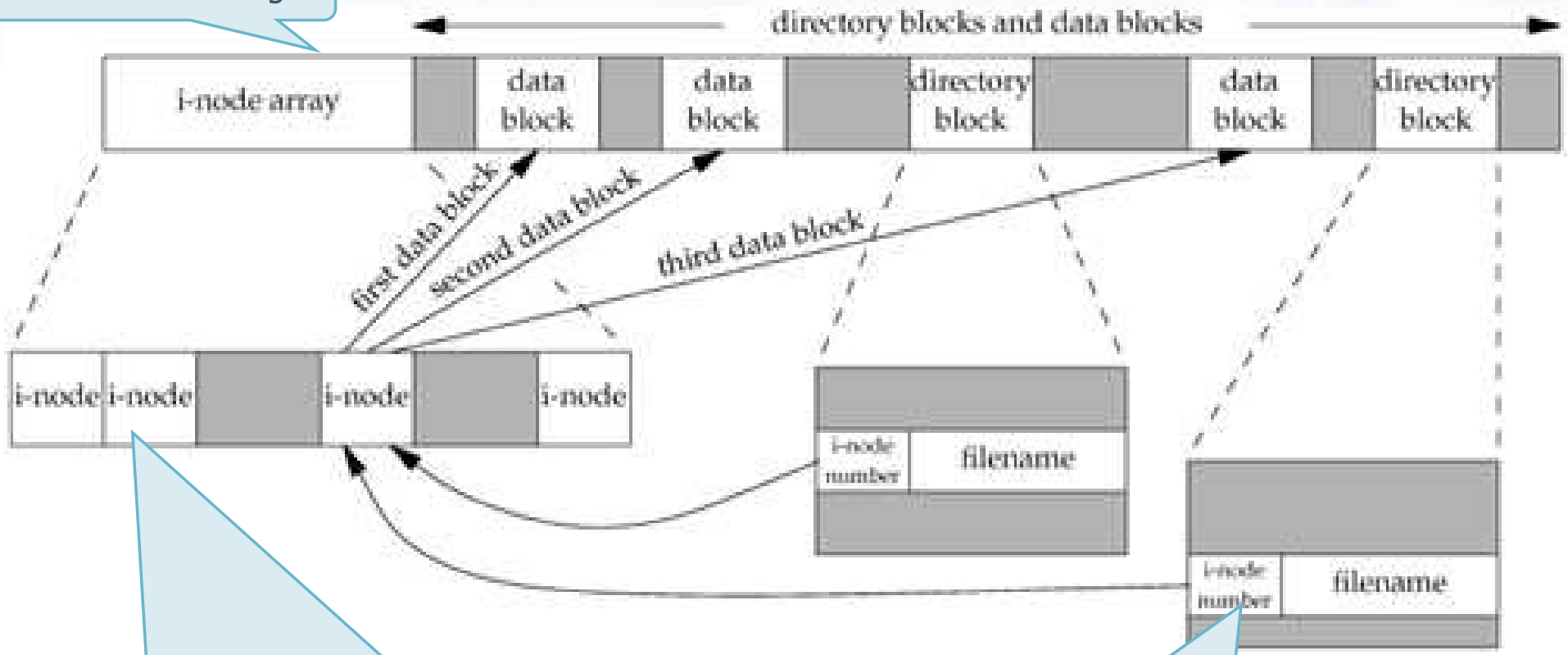


Hard-link

Un direttorio è una tabella che associa a ogni nome di file un **i-node number**
Il link da un direttorio al rispettivo i-node è detto **hard-link**
Lo stesso i-node number può essere individuato da più link

Allocazione indicizzata: schema combinato

Da Stevens & Rago



Record di lunghezza fissa che contiene la maggior parte di informazioni relative ai file (i.e., ne identifica i blocchi che lo compongono)
Contiene un contatore che individua il numero di puntatori (link)
Numerati a partire da 1; alcuni sono riservati al SO

L'i-node number corrisponde all'indice di una tabella in cui ogni elemento è un i-node e contiene le informazioni di un file

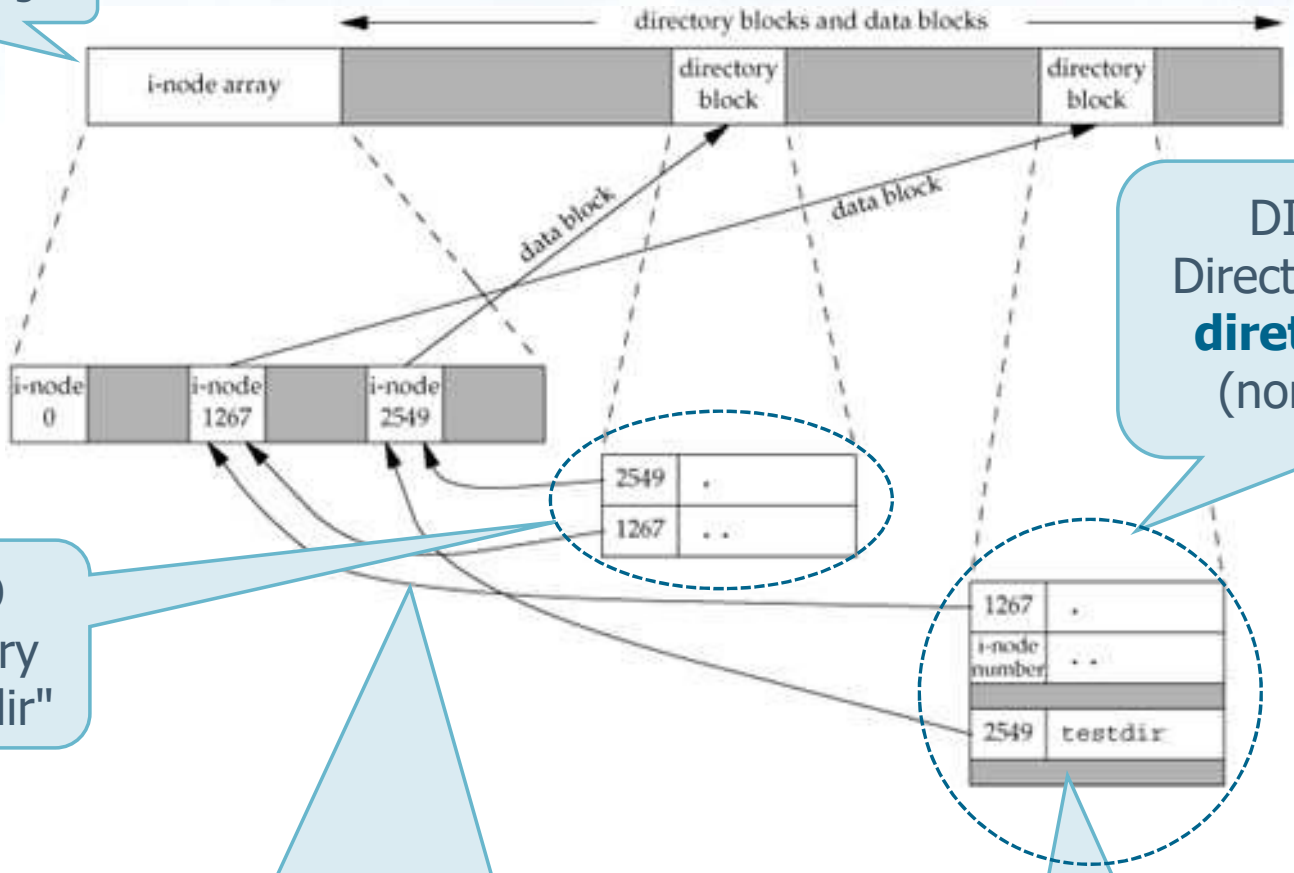
Allocazione indicizzata: schema combinato

Link in Windows
https://it.wikipedia.org/wiki/Collegamento_(Windows)

- Hard link (link effettivo o fisico)
  - Directory entry che punta a un i-node
  - Non esistono hard link
    - Verso direttori (evita filesystem a grafo ciclico)
    - Verso file su altri file system
  - Un file è fisicamente rimosso solo quando tutti i suoi hard link sono stati rimossi
- Soft link (link simbolico)
  - Il blocco dati individuato dall'i-node punta a un blocco che contiene il path name del file
  - Sostanzialmente è un file che come unico blocco dati ha il nome di un altro file

File system UNIX: Un esempio

Da Stevens & Rago



DIR PADRE
Directory entry del **direttorio** 1267 (nome ignoto)

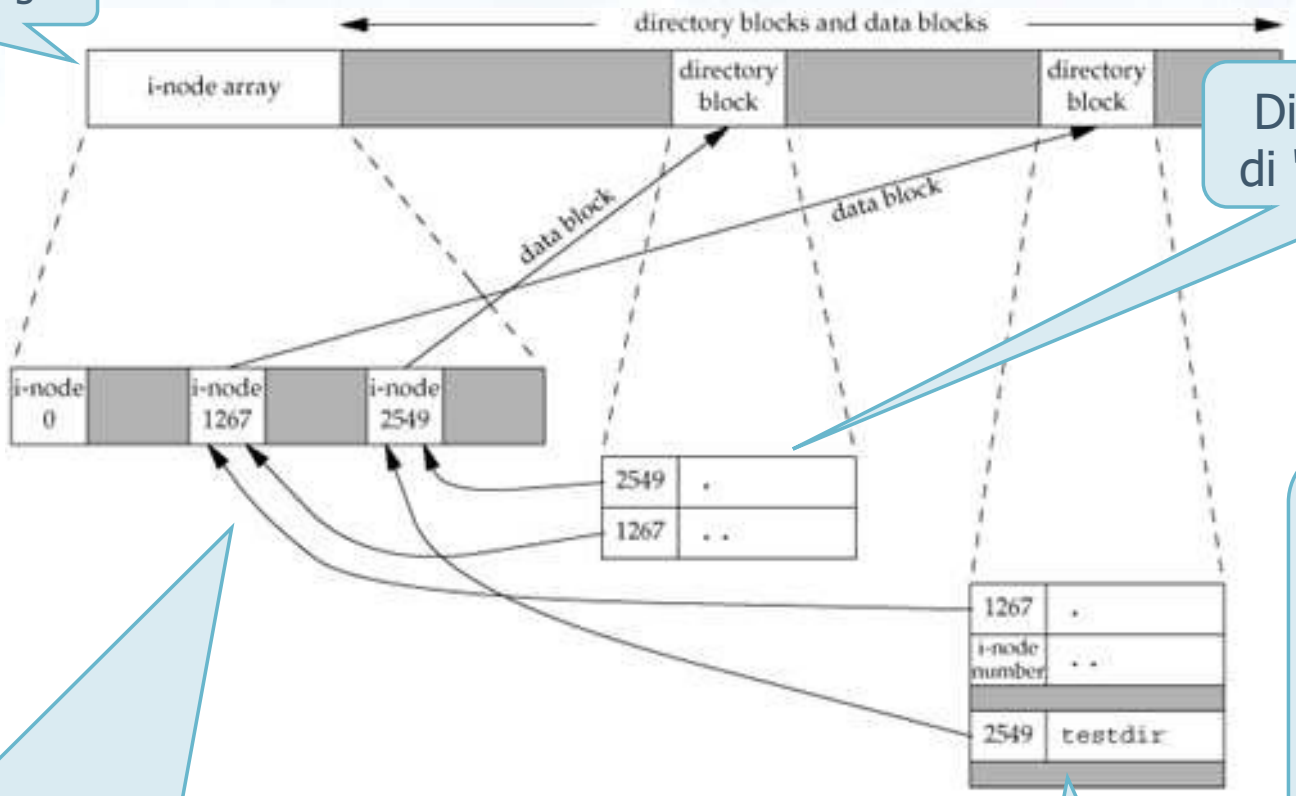
DIR FIGLIO
Directory entry di "2549 testdir"

L'i-node 2549 è un sotto-direttorio foglia
Il suo hard link count è **pari** a **2**
Uno deriva dal direttorio padre ("testdir")
Uno deriva da se stesso ("testdir/.")

L'i-node "2549 testdir" è un sotto-direttorio **foglia** di 1267

File system UNIX: Un esempio

Da Stevens & Rago



Directory entry di "2549 testdir"

Ogni sotto-direttorio incrementa il numero di link del padre di 1 !

L'i-node 1267 è un direttorio con un sotto-direttorio
Il suo link count è pari **almeno** a 3
Uno deriva dal direttorio padre (non indicato)
Uno deriva da se stesso (".")
Uno deriva dal direttorio figlio ("./testdir/..")

L'i-node "2549 testdir" è un sotto-direttorio **foglia** di 1267

File system moderni

- I file system odierni più utilizzati
  - FAT
  - NTFS
  - Ext

Gigabyte GB 10^9
Terabyte TB 10^{12}
Petabyte PB 10^{15}
Exabyte EB 10^{18}
Zettabyte ZB 10^{21}
Yottabyte YB 10^{24}

	Fat 32	Ex Fat	NTFS	Ext 4
Dimensione massima disco	2 TB	64 ZB	2 TB (aumentabile aumentando i cluster)	1 EB
Dimensione massima file	4 GB	16 ZB	Quanto il disco	16 TB
Utililzzo principale	Chiavetta USB	Chiavetta USB	Disco interno Windows	Disco interno Linux

File system moderni

- FAT

Windows
https://it.wikipedia.org/wiki/File_Allocation_Table

FAT12 per floppy disk

  - FAT16 (o semplicemente FAT, 1987)
    - Prima versione, non supporta file più grandi di 2GByte e un disco di dimensioni massime di 32GBytes

VFAT (Virtual FAT) supporta nomi dei file lunghi

  - FAT32
    - Evoluzione del FAT16, con cluster da 32 bit, aumenta il supporto per file e dischi di dimensioni maggiori
  - exFAT (extended FAT o FAT64, 2006)
    - Aumenta nuovamente il supporto per file e dischi di dimensioni maggiori, progettato per essere leggero per le unità flash

File system moderni

- NTFS

Windows
https://it.wikipedia.org/wiki/NTFS

  - Rispetto a FAT aumenta le dimensioni supportate
  - Come i filesystem Ext più recenti supporta il **journaling** e la **crittografia** del disco

Preserva l'integrità del file system da cadute di tensione tramite il concetto di transazione

  - Non è veloce come FAT o Ext ma è la scelta standard per dischi fissi Windows
  - MAC e Linux lo supportano con driver specifici in scrittura

File system moderni

Minix → Linux
https://it.wikipedia.org/wiki/Ext4

- Ext
  - Ext (1992)
    - La principale mancanza di ext è quella di gestire un unico time stamp per file, a differenza dei 3 time stamp odierni (creazione, ultima modifica, ultimo accesso)
  - Ext2 (1993)
    - Estensione delle dimensioni
    - **Non** garantisce il **journaling**
      - In caso di spegnimento del computer durante la fase di scrittura, magari dovuto a un calo di corrente, il file system viene corrotto, rendendo impossibile l'accesso ai file sul disco

File system moderni

- Ext3 (2001)
  - Risolve il problema della corruzione del file system
    - Durante la scrittura di un file, questo viene prima scritto sul disco, poi, se la scrittura è andata a buon fine, viene registrato sul file system
    - Se la scrittura viene interrotta senza essere conclusa, il file system rimane inalterato, senza accorgersi di nulla
- Ext4 (2006)
  - Aumenta il supporto per la dimensione sempre maggiore dei dischi e il miglioramento delle prestazioni, aumentando anche le performance in lettura e scrittura in termini di velocità
  - Retro-compatibile con ext3