Struttura dei dischi

Ivan Scagnetto

Università di Udine — Facoltà di Scienze MM.FF.NN.

A.A. 2011-2012

Struttura dei dischi

- La gestione dei dischi riveste particolare importanza.
- I dischi sono indirizzati come dei grandi array monodimensionali di blocchi logici, dove il blocco logico è la più piccola unità di trasferimento con il controller.
- L'array monodimensionale è mappato sui settori del disco in modo sequenziale.
  - Settore 0 = primo settore della prima traccia del cilindro più esterno.
  - la mappatura procede in ordine sulla traccia, poi sulle rimanenti tracce dello stesso cilindro, poi attraverso i rimanenti cilindri dal più esterno verso il più interno.

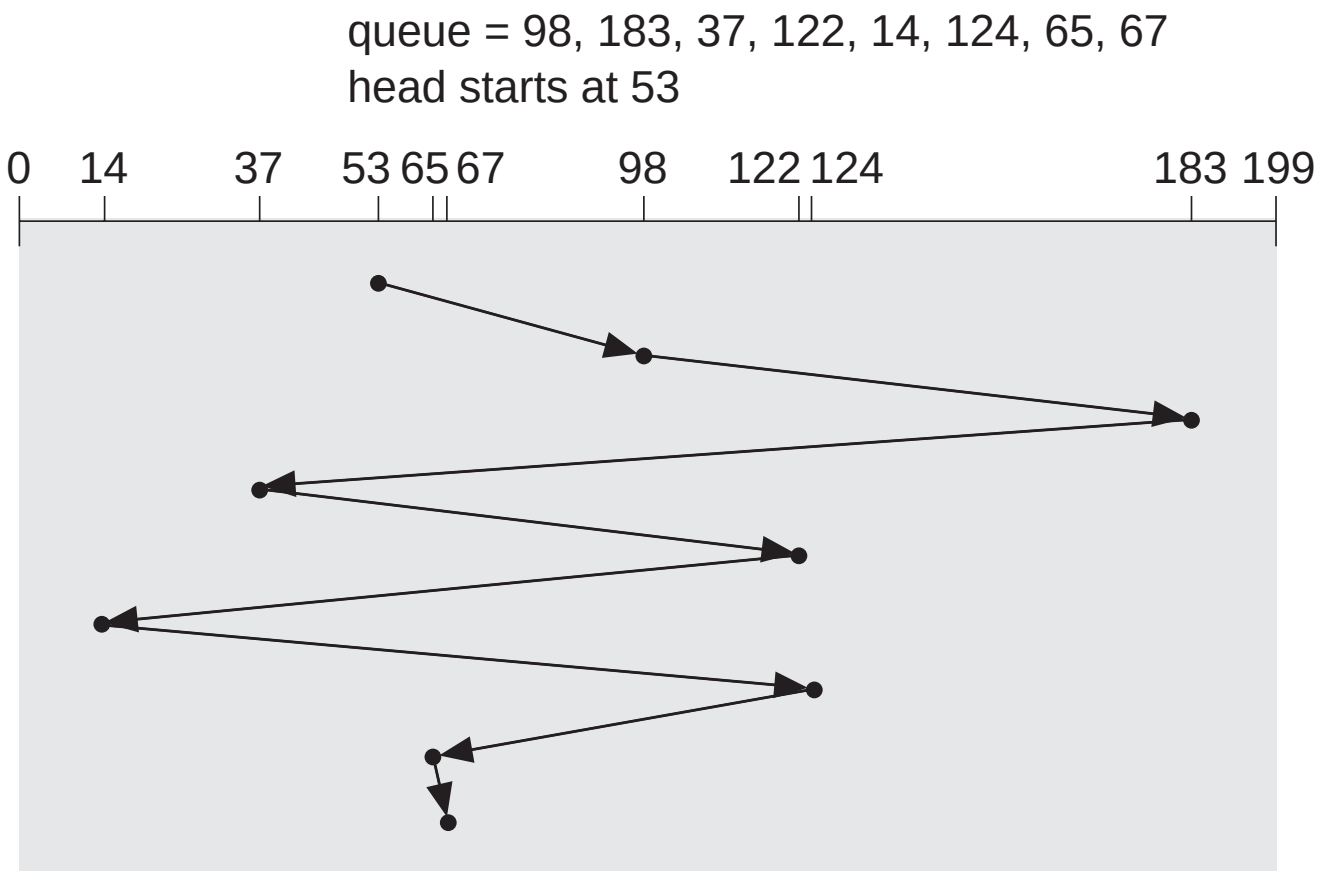
Schedulazione dei dischi

- Il sistema operativo è responsabile dell'uso efficiente dell'hardware.
  Per i dischi: bassi tempi di accesso e alta banda di utilizzo.
- Il tempo di accesso ha 2 componenti principali, dati dall'hardware:
  - *Seek time* = il tempo (medio) per spostare le testine sul cilindro contenente il settore richiesto.
  - *Latenza rotazionale* = il tempo aggiuntivo necessario affinché il settore richiesto passi sotto la testina.
- Tenere traccia della posizione angolare dei dischi è difficile, mentre si sa bene su quale cilindro si trova la testina.
- Obiettivo: mimimizzare il tempo speso in seek.
- Tempo di seek $\approx$ distanza di seek; quindi: mimimizzare la distanza di seek.
- Banda di disco = il numero totale di byte trasferiti, diviso il tempo totale dalla prima richiesta di servizio e il completamento dell'ultimo trasferimento.

Schedulazione dei dischi (Cont.)

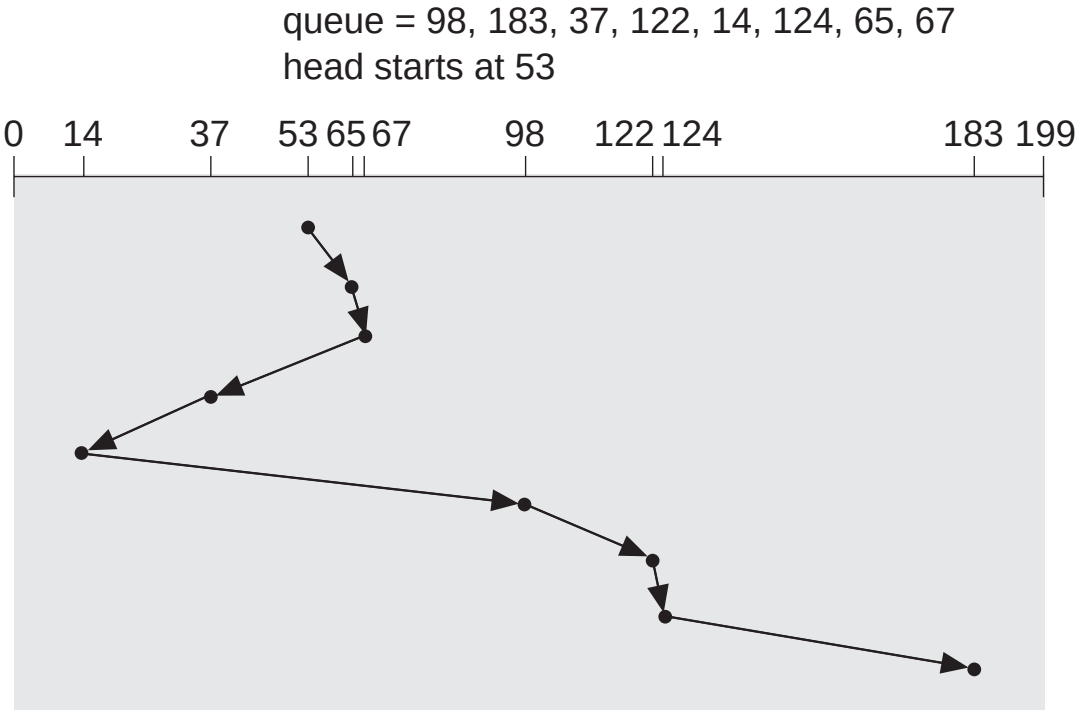
- Ci sono molti algoritmi per schedulare le richieste di I/O di disco.
- Al solito, una trattazione formale esula dal corso.
- Illustreremo con una coda di richieste d'esempio, su un range di cilindri 0–199: `98, 183, 37, 122, 14, 124, 65, 67`
- Supponiamo che la posizione attuale della testina sia 53.

Sull'esempio: distanza totale di 640 cilindri



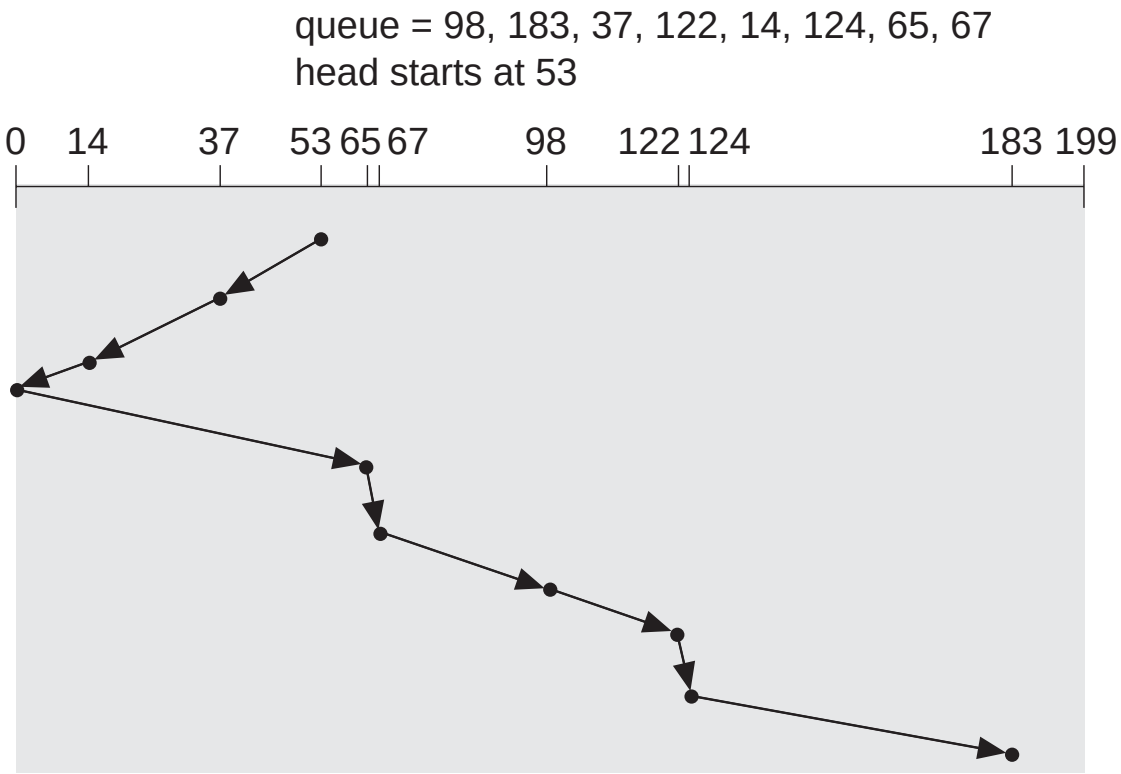
Shortest Seek Time First (SSTF)

- Si seleziona la richiesta con il minor tempo di seek dalla posizione corrente.
- SSTF è una forma di scheduling SJF; può causare *starvation*.
- Sulla nostra coda di esempio:
  distanza totale di 236 cilindri (36,875% di FCFS).



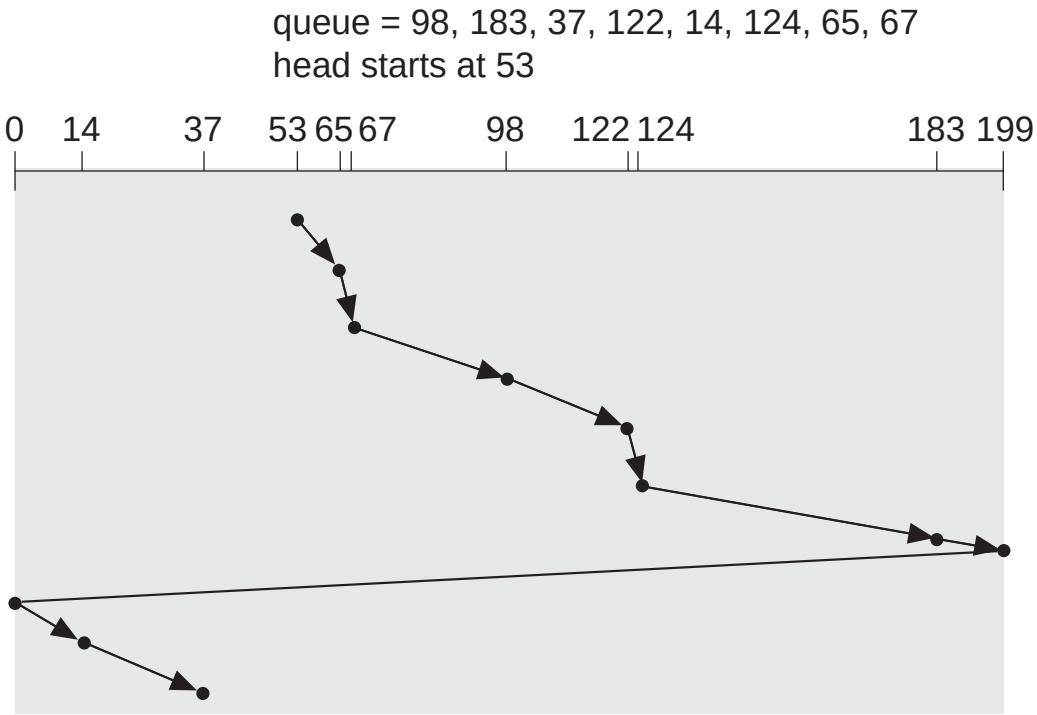
SCAN (o "dell'ascensore")

- Il braccio scandisce l'intera superficie del disco, da un estremo all'altro, servendo le richieste man mano. Agli estremi si inverte la direzione.
- Sulla nostra coda di esempio: distanza totale di 236 cilindri.



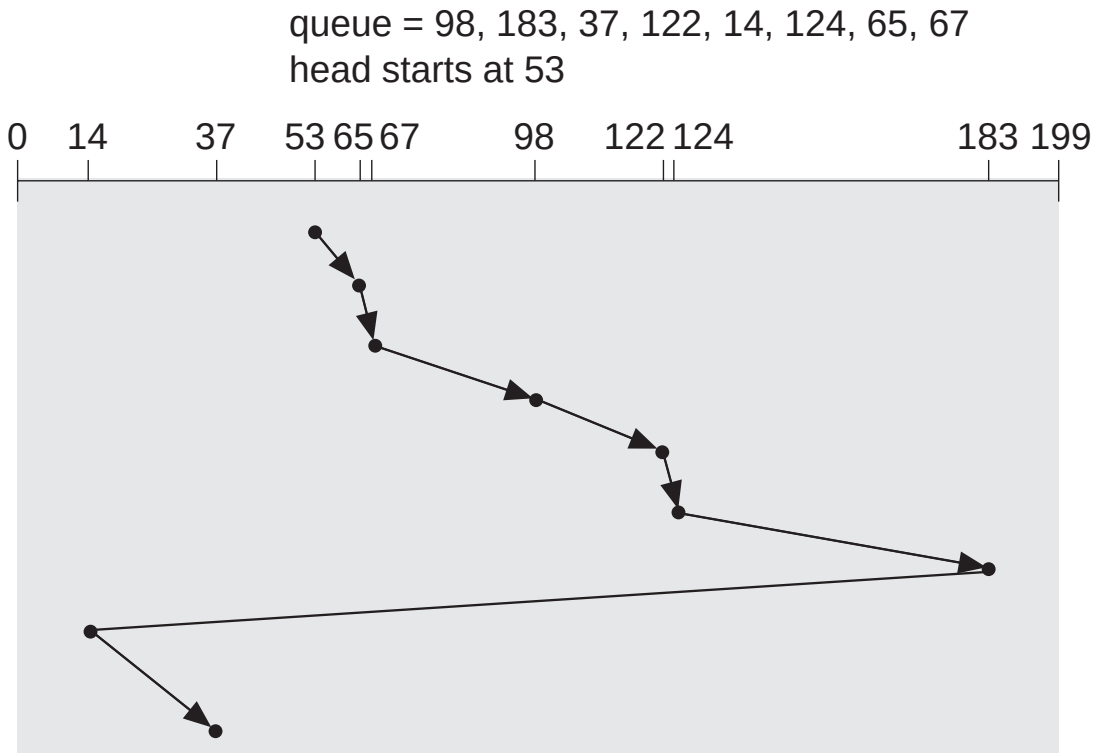
C-SCAN

- Garantisce un tempo di attesa più uniforme e equo di SCAN
- Tratta i cilindri come in lista circolare, scandita in rotazione dalla testina si muove da un estremo all'altro del disco. Quando arriva alla fine, ritorna immediatamente all'inizio del disco senza servire niente durante il rientro.



C-LOOK

- Miglioramento del C-SCAN (esiste anche il semplice LOOK)
- Il braccio si sposta solo fino alla richiesta attualmente più estrema, non fino alla fine del disco, e poi inverte direzione immediatamente.



Quale algoritmo per lo scheduling dei dischi?

- SSTF è molto comune e semplice da implementare, e abbastanza efficiente
- SCAN e C-SCAN sono migliori per i sistemi con un grande carico di I/O con i dischi (si evita starvation)
- Le performance dipendono dal numero e tipi di richieste
- Le richieste ai dischi dipendono molto da come vengono allocati i file, ossia da come è implementato il file system.
- L'algoritmo di scheduling dei dischi dovrebbe essere un modulo separato dal resto del kernel, facilmente rimpiazzabile se necessario.
- Sia SSTF che LOOK (e varianti circolari) sono scelte ragionevoli come algoritmi di default.

Gestione dell'area di swap

- L'area di swap è parte di disco usata dal gestore della memoria come estensione della memoria principale.
- Può essere ricavata dal file system normale o (meglio) in una partizione separata.
- Gestione dell'area di swap
  - 4.3BSD: alloca lo spazio appena parte il processo per i segmenti text e data. Per lo stack, lo spazio viene allocato man mano che cresce.
  - Solaris 2: si alloca una pagina sullo stack solo quando si deve fare un page-out, non alla creazione della pagina virtuale.
  - Windows 2000: Viene allocato spazio sul file di swap per ogni pagina virtuale non corrispondente a nessun file sul file system (es: DLL).

Affidabilità e performance dei dischi

- Aumenta la differenza di velocità tra applicazioni e dischi.
- Le memorie cache non sempre sono efficaci (es. transazioni, dati provenienti da esperimenti)
- Suddividere il carico tra più dischi che cooperano per offrire l'immagine di un disco unitario virtuale più efficiente.
- Problema di affidabilità:

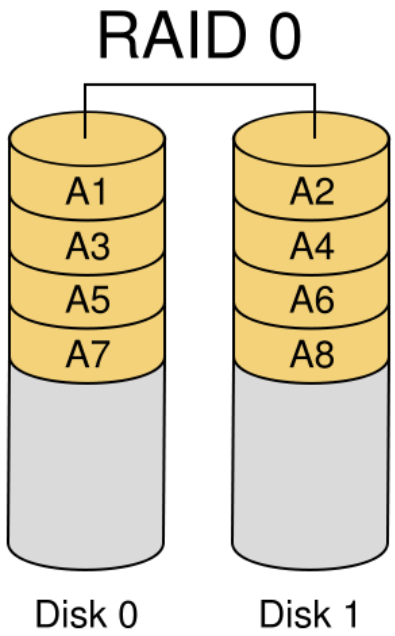
$$MTBF_{array} = \frac{MTBF_{disco}}{\#dischi}$$

RAID

- RAID = Redundant Array of Inexpensive/Independent Disks: implementa affidabilità del sistema memorizzando informazione ridondante.
- La ridondanza viene gestita dal controller (RAID hardware) — o molto spesso dal driver (RAID software).
- Diversi livelli (organizzazioni), a seconda del tipo di ridondanza

  0: striping: i dati vengono "affettati" e parallelizzati. Altissima performance, non c'è ridondanza.
  1: Mirroring o shadowing: duplicato di interi dischi. Eccellente resistenza ai crash, basse performance in scrittura.
  5: Block interleaved parity: come lo striping, ma un disco a turno per ogni stripe viene dedicato a contenere l'informazione di parità del resto della stripe. Alta resistenza, discrete performance (in caso di aggiornamento di un settore, bisogna ricalcolare la parità).

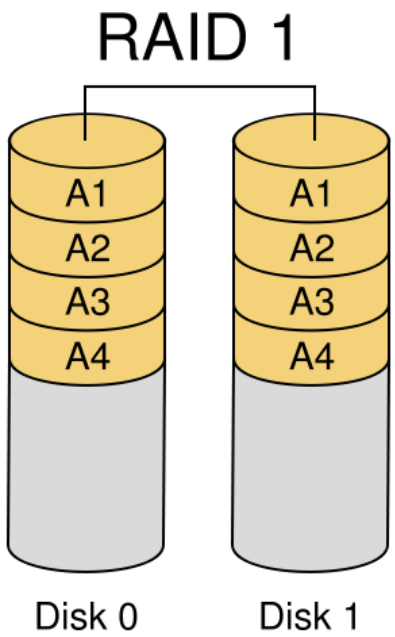
RAID 0 (minimo numero di dischi: 2)



Caratteristiche:

- In RAID 0 i dati vengono suddivisi in blocchi (stripe) ed ognuno di questi viene memorizzato in un disco dell'array a disposizione secondo lo schema indicato in figura.
- La performance dell'I/O aumenta notevolmente dato che il carico di lavoro viene suddiviso fra più dischi.
- Non c'è ridondanza nei dati.

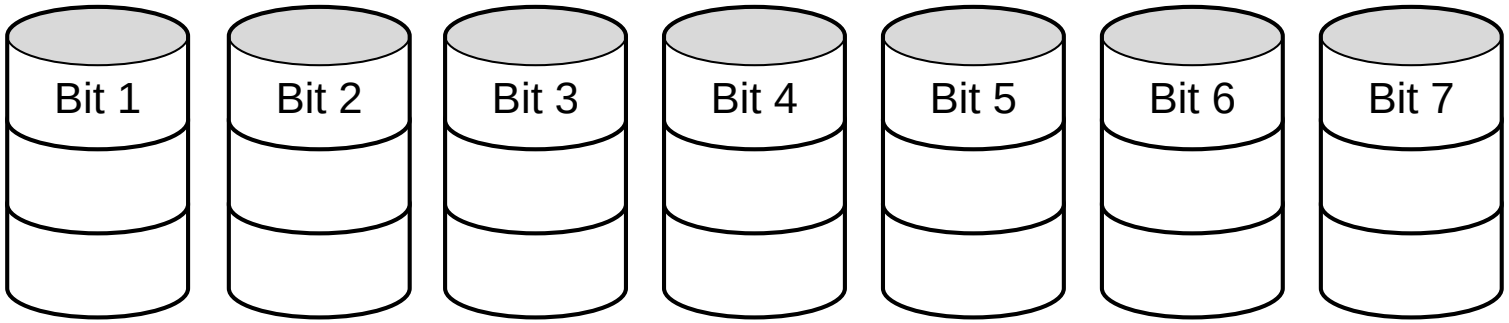
RAID 1 (minimo numero di dischi: 2)



Caratteristiche:

- I dati vengono "duplicati" su coppie di dischi (mirror).
- La performance della lettura dei dati raddoppia, mentre quella della scrittura rimane uguale al caso di un singolo disco.
- Nel caso di guasto ad un disco, si può utilizzare immediatamente l'altro (senza bisogno di procedure di ricostruzione dei dati).
- Aspetto negativo: spreco di spazio.

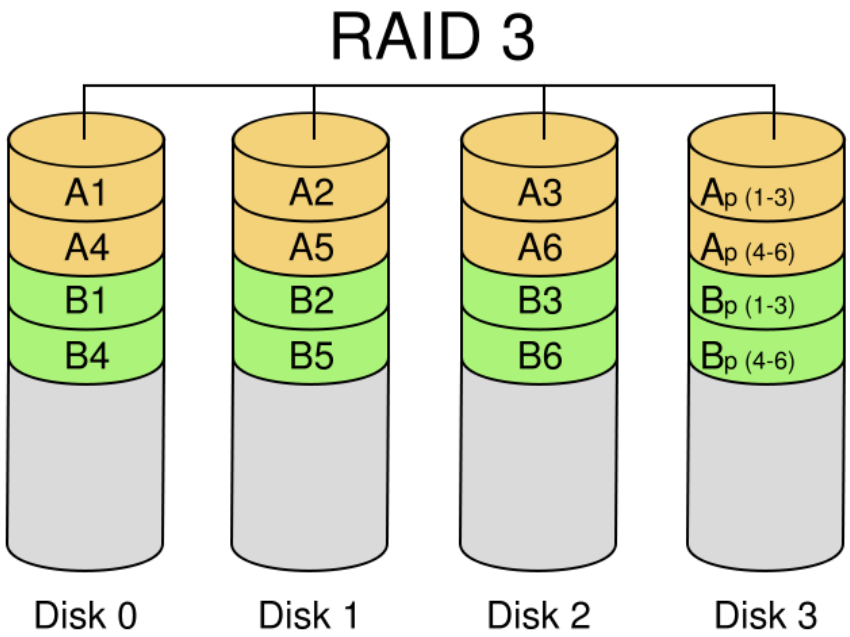
RAID 2 (Hamming Code ECC: 7 dischi)



Caratteristiche:

- Questo livello era nato per dischi che non possedevano meccanismi propri per gestire gli errori di lettura/scrittura.
- Opera a livello di bit utilizzando i codici di Hamming (7,4): 4 bit di dati ognuno su uno dei 4 dischi dati e 3 bit di parità ognuno su un disco di parità.
- Può correggere errori dovuti all'inversione di un singolo bit e rilevare errori dovuti all'inversione di due bit.

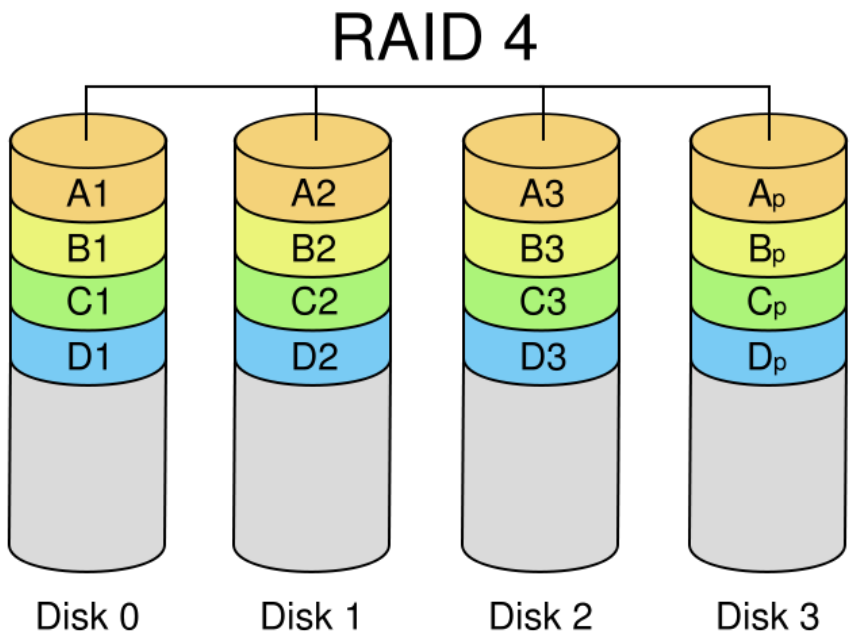
RAID 3 (minimo 3 dischi)



Caratteristiche:

- I dati sono organizzati a livello di bit o byte: come ECC utilizza XOR (i byte di parità sono registrati su un disco apposito).
- Prestazioni molto elevate in lettura/scrittura.
- Ottima tolleranza ai guasti.
- In generale si può evadere una singola richiesta per volta.

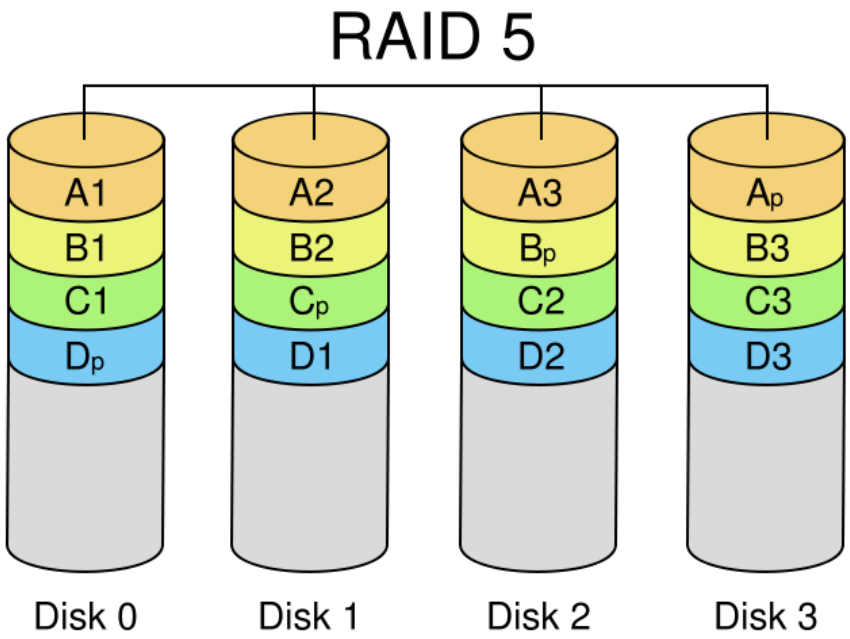
RAID 4 (minimo 3 dischi)



Caratteristiche:

- I dati sono suddivisi in blocchi: ogni blocco viene memorizzato su un disco nell'array secondo lo schema in figura (i blocchi di parità sono registrati su un disco apposito).
- Prestazioni molto elevate in lettura.
- Ottima tolleranza ai guasti.
- Il disco di parità è un "collo di bottiglia".

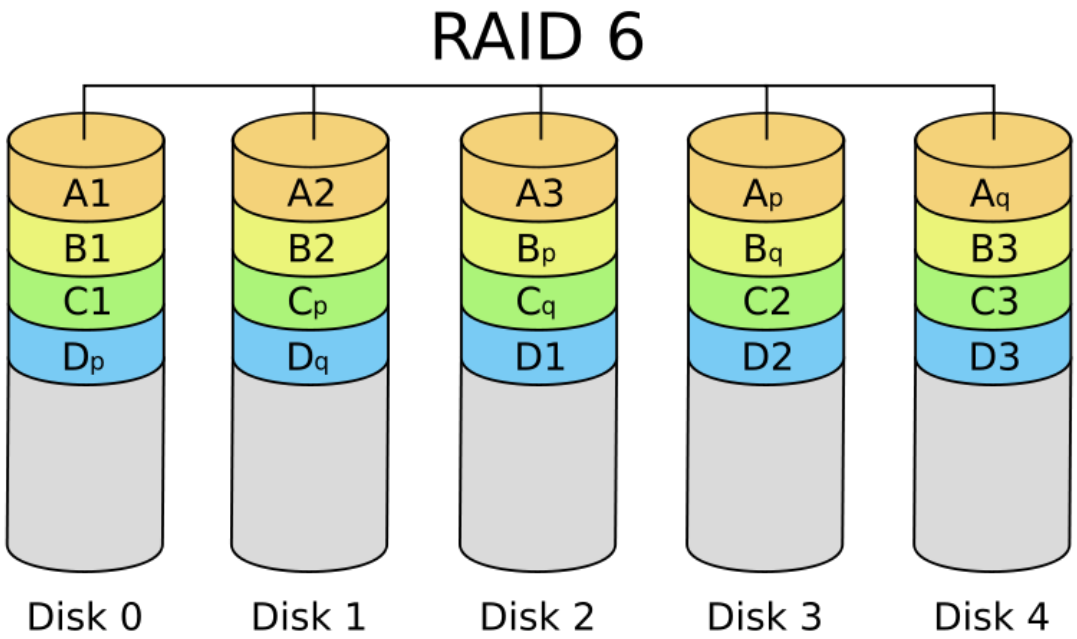
RAID 5 (minimo 3 dischi)



Caratteristiche:

- Il principio di funzionamento è quello del RAID 4, ma i blocchi di parità sono memorizzati in modo distribuito sui dischi dell'array.
- Prestazioni migliorate in scrittura.
- Il sistema è complesso e costoso da realizzare.

RAID 6 (minimo 4 dischi)



Caratteristiche:

- Il principio di funzionamento è quello del RAID 5, ma vengono utilizzati due tipi di controllo d'errori.
- Grande ridondanza e sicurezza dei dati.
- In caso di guasto la ricostruzione dei dati è molto lenta a causa del doppio controllo ECC.

RAID "composti" da più livelli

È possibile combinare fra loro diversi livelli RAID:

- RAID 10: un insieme di dischi in mirror (RAID 1) vengono suddivisi in "strisce" in un secondo insieme di dischi (RAID 0).
- RAID 0+1: le immagini di un insieme di dischi a livello 0 vengono replicate in un insieme di dischi a livello 1.
- ...

Solid State Drive (SSD)

- Da qualche tempo si stanno diffondendo i cosiddetti dischi a stato solido.
- Non hanno parti meccaniche in movimento, essendo basati su sistemi di memoria a stato solido (e.g., memoria flash basata su porte NAND).
- L'accesso è molto più veloce (i.e., 0,5-0,05 ms),
- Tuttavia permettono un accesso ai dati simile a quello dei dischi "tradizionali".
- Ovviamente per questa categoria di dispositivi si pongono problematiche molto differenti:
  - gli algoritmi tradizionali di scheduling delle richieste non hanno più senso;
  - insorgono problemi di write amplification con conseguente pericolo di usura precoce;
  - i S.O. devono fornire del supporto esplicito per una corretta gestione degli SSD (e.g., supporto dell'istruzione TRIM).

Solid State Drive (SSD): write amplification

- L'unità minima leggibile/scrivibile di un SSD è la pagina (e.g., 4KB).
- Un disco nuovo permette di scrivere i nuovi dati a livello di singole pagine.
- Tuttavia per riscrivere una pagina, bisogna prima cancellare l'intero blocco che la contiene (e.g., 256 KB):
  - se nel blocco da cancellare, ci sono altre pagine valide (da preservare), queste ultime vanno spostate in un altro blocco;
  - si parla quindi di write amplification:

$$\frac{\text{dati scritti sulla memoria flash}}{\text{dati inviati dal S.O.}}$$

  - sorge la necessità di un garbage collector per mantenere alte le performance.
- L'istruzione TRIM consente di informare il controller dell'unità SSD sulle pagine non più in uso da parte del filesystem.
- L'uso regolare di TRIM permette di ottimizzare il lavoro del garbage collector.