



Capitolo 10:

Memoria secondaria

- Struttura dei dispositivi di memorizzazione
- Struttura dei dischi
- Connessione dei dischi
- Scheduling del disco
- Gestione dell'unità a disco
- Gestione dell'area di avvicendamento
- Strutture RAID
- Realizzazione della memoria stabile





Struttura dei dischi

- I moderni dischi dal punto di vista dell'indirizzamento si considerano come un grande vettore monodimensionale di **blocchi logici**,
 - ◆ dove un blocco logico è la minima unità di trasferimento.
- La dimensione di un blocco logico è di solito 512 byte, ma alcuni dischi si possono **formattare a basso livello** allo scopo di ottenere una diversa dimensione dei blocchi logici (ad es. 1024 byte).
- Il vettore monodimensionale di blocchi logici corrisponde in modo sequenziale ai settori del disco:
 - ◆ Il settore 0 è il primo settore della prima traccia sul cilindro più esterno.
 - ◆ La corrispondenza prosegue ordinatamente lungo la prima traccia, quindi lungo le rimanenti tracce del primo cilindro, e così via, di cilindro in cilindro, dall'esterno verso l'interno.



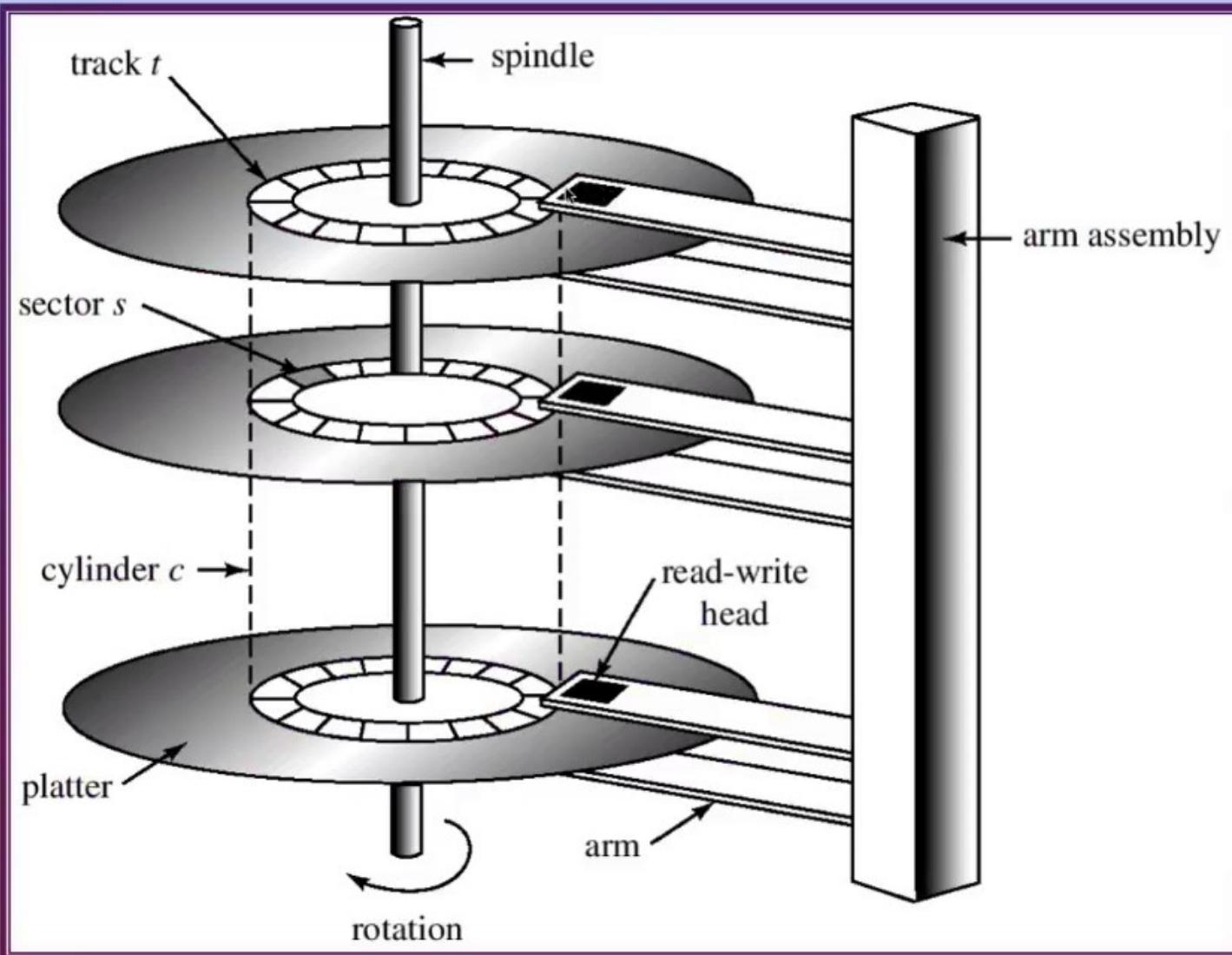


Dischi magnetici e a stato solido

- I **piatti** dei dischi magnetici hanno forma piana e rotonda
- le due superfici sono ricoperte di materiale magnetico su cui si memorizzano le informazioni.
- Una **testina** è sospesa su ciascuna superficie di ogni piatto.
- Le testine sono attaccate al **braccio** del disco che le muove in blocco.
- La superficie di un piatto è divisa logicamente in **tracce** circolari a loro volta divise in **settori**.
- L'insieme delle tracce corrispondenti ad una posizione del braccio costituisce un **cilindro**.
- Un disco a stato solido (SSD) è una memoria non volatile.
- Ci sono molte varianti di questa tecnologia, ad esempio le tecnologie di memoria flash oppure le DRAM dotate di batteria.
- Sono a volte più veloci dei dischi magnetici ma hanno ancora costi più alti.



Schema funzionale di un disco





Scheduling del disco

- La velocità di un disco nel trasferire i dati è un fattore di notevole importanza per tutto il sistema di elaborazione.
 - Il sistema operativo è responsabile per l'utilizzo efficiente dell'hardware.
 - Per quanto riguarda il disco questo significa l'avere
 - ◆ un **tempo di accesso** veloce ed
 - ◆ una grande **ampiezza di banda**
 - ✓ cioè il numero totale di byte trasferiti diviso il tempo totale intercorso tra la prima richiesta ed il completamento dell'ultimo trasferimento.
 - Il sistema operativo può migliorare il tempo medio di servizio del disco pianificando le richieste di accesso al disco stesso attraverso un processo di scheduling.
 - Il tempo di accesso ha due componenti principali
 - ◆ **tempo di ricerca**: il tempo necessario affinché il braccio dell'unità a disco sposti le testine fino al cilindro contenente il settore desiderato.
 - ◆ **latenza di rotazione**: tempo aggiuntivo necessario perché il disco ruoti finché il settore desiderato si trovi sotto la testina.
 - Tramite lo scheduling del disco si possono migliorare entrambe.
- 



Scheduling del disco

- Ogni volta che si devono compiere operazioni di I/O con un'unità a disco un processo effettua una system call.
- La richiesta contiene diverse informazioni:
 - ◆ se l'operazione sia di immissione o di emissione di dati,
 - ◆ l'indirizzo nel disco rispetto al quale eseguire il trasferimento,
 - ◆ l'indirizzo di memoria rispetto al quale eseguire il trasferimento,
 - ◆ il numero di byte da trasferire.
- Se l'unità a disco è libera la richiesta si può immediatamente soddisfare, altrimenti le nuove richieste si aggiungono alla coda di richieste inevase rispetto a quell'unità.
- La coda relativa ad un'unità a disco può essere spesso lunga,
 - ◆ quindi S.O. dovrà scegliere quale fra le richieste inevase converrà servire prima.



Scheduling in ordine di arrivo (FCFS)

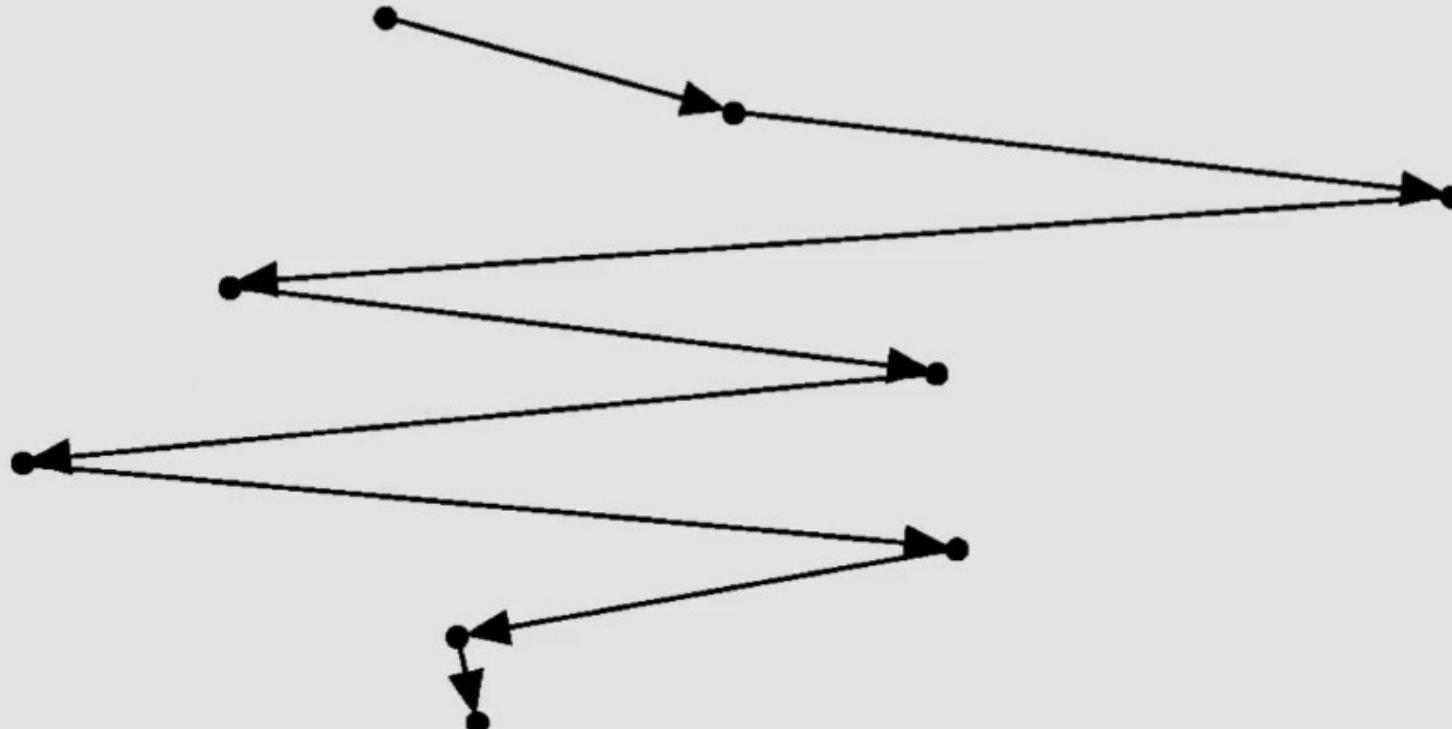
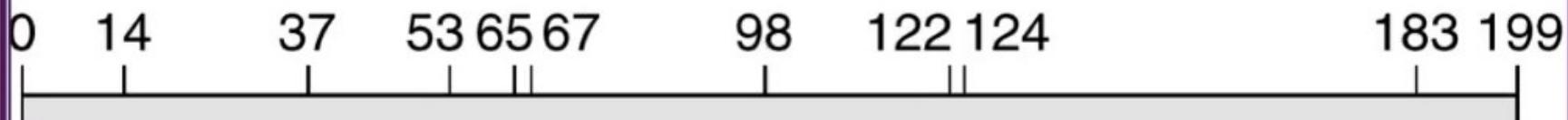
- La forma più semplice di scheduling è l'algoritmo di servizio secondo l'ordine di arrivo (FCFS).
- Nell'algoritmo di scheduling FCFS, la testina esegue le operazioni con lo stesso ordine di arrivo delle richieste.
- FCFS è un algoritmo semplice ma non è ottimale in termini di velocità del servizio.
- E' facile osservare infatti che le stesse richieste date in ordine diverso possono determinare tempi di servizio diversi.
- Tuttavia questo algoritmo viene utilizzato da quasi tutti i sistemi che hanno un carico basso:
 - ◆ in quanto i pochi benefici ottenuti dall'utilizzo di altri algoritmi, su un sistema a basso carico, non riescono a compensare i costi sostenuti per realizzarli.
- Si consideri ad esempio una coda di richieste per l'unità a disco che dia una lista di cilindri sui quali individuare i blocchi richiesti nel seguente ordine: **98, 183, 37, 122, 14, 124, 65, 67**.
- La testina è inizialmente posta al cilindro 53.



Scheduling FCFS

queue = 98, 183, 37, 122, 14, 124, 65, 67

head starts at 53





Scheduling per brevità (SSTF)

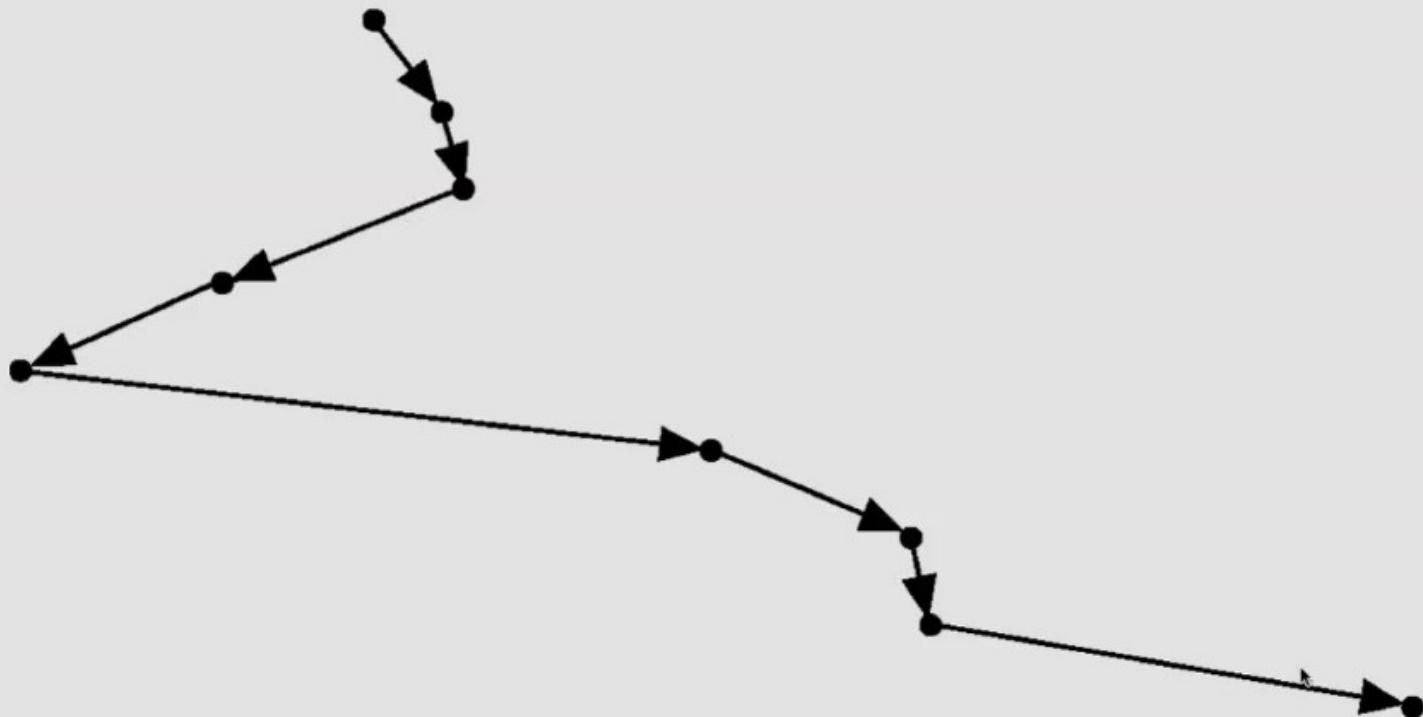
- L'algoritmo di scheduling per brevità (Shortest Seek Time First), a partire dalla posizione corrente seleziona la richiesta con tempo di ricerca minimo.
- In pratica la testina si sposta sempre sulla traccia più vicina, prima di allontanarsi per servire un'altra richiesta.
- Lo scheduling SSTF è essenzialmente una forma di scheduling SJF (Shortest-job-first), ed al pari di questo può portare a situazioni di starvation.
- Infatti poiché le richieste possono arrivare in qualunque momento, la richiesta di una traccia lontana dalla posizione iniziale potrebbe rimanere per un tempo indefinito in attesa.
- Pur avendo prestazioni migliori del FCFS, l'algoritmo SSTF non è ottimale in termini di velocità del servizio.



SSTF

queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53

0 14 37 53 65 67 98 122 124 183 199





Scheduling per scansione (SCAN)

- Secondo l'algoritmo di scheduling per scansione il braccio dell'unità a disco parte da un estremo del disco e si sposta mantenendo la stessa direzione,
 - ◆ servendo le richieste mentre attraversa i cilindri, fino a che non giunge all'altro estremo del disco.
- A questo punto il braccio inverte la marcia e la procedura continua.
- Le testine attraversano continuamente il disco nelle due direzioni.
- L'algoritmo SCAN viene talvolta chiamato algoritmo dell'ascensore,
 - ◆ perché il braccio del disco si comporta come un ascensore che serve prima tutte le richieste in salita e poi tutte quelle in discesa.
- Se nella coda delle richieste di I/O giunge una richiesta di lettura per una traccia molto vicina alla testina (rispetto al suo verso corrente), questa viene servita quasi immediatamente.
- Invece la richiesta di una traccia che si trova dietro la testina deve attendere fino a che la stessa non arrivi all'estremità del disco, inverta la direzione di spostamento e ritorni.





Scheduling per scansione circolare (C-SCAN)

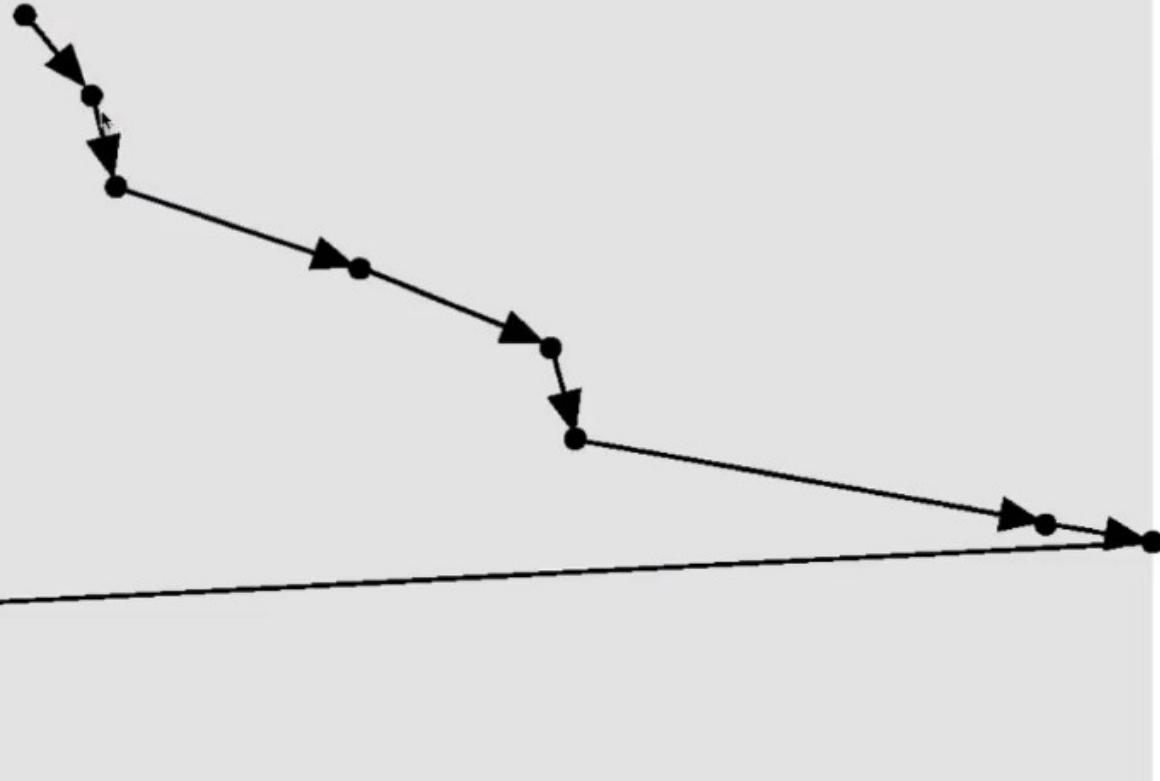
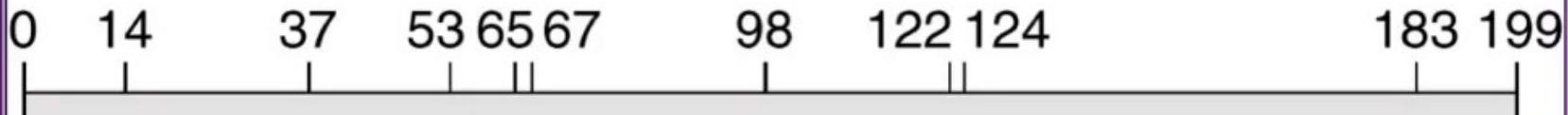
- L'algoritmo di scheduling per scansione circolare è una variante dell'algoritmo di scheduling SCAN progettato per fornire un tempo di attesa più uniforme.
- Ipotizzando una distribuzione uniforme per le richieste relative alle varie tracce, si consideri la densità di richieste che si presenta quando la testina raggiunge un'estremità e inverte la direzione di spostamento.
- A questo punto, dietro la testina sono presenti poche richieste, in quanto queste tracce sono state servite di recente.
- La maggiore densità di richieste è presente all'altra estremità del disco.
- Per ovviare a questa situazione quando la testina raggiunge l'estremità del disco, viene fatta tornare immediatamente all'inizio dello stesso senza servire le richieste che si trovano sul percorso di ritorno.
- Fondamentalmente, lo scheduling C-SCAN tratta il disco come una lista circolare, cioè come se il primo e l'ultimo cilindro fossero adiacenti.



C-SCAN

queue = 98, 183, 37, 122, 14, 124, 65, 67

head starts at 53





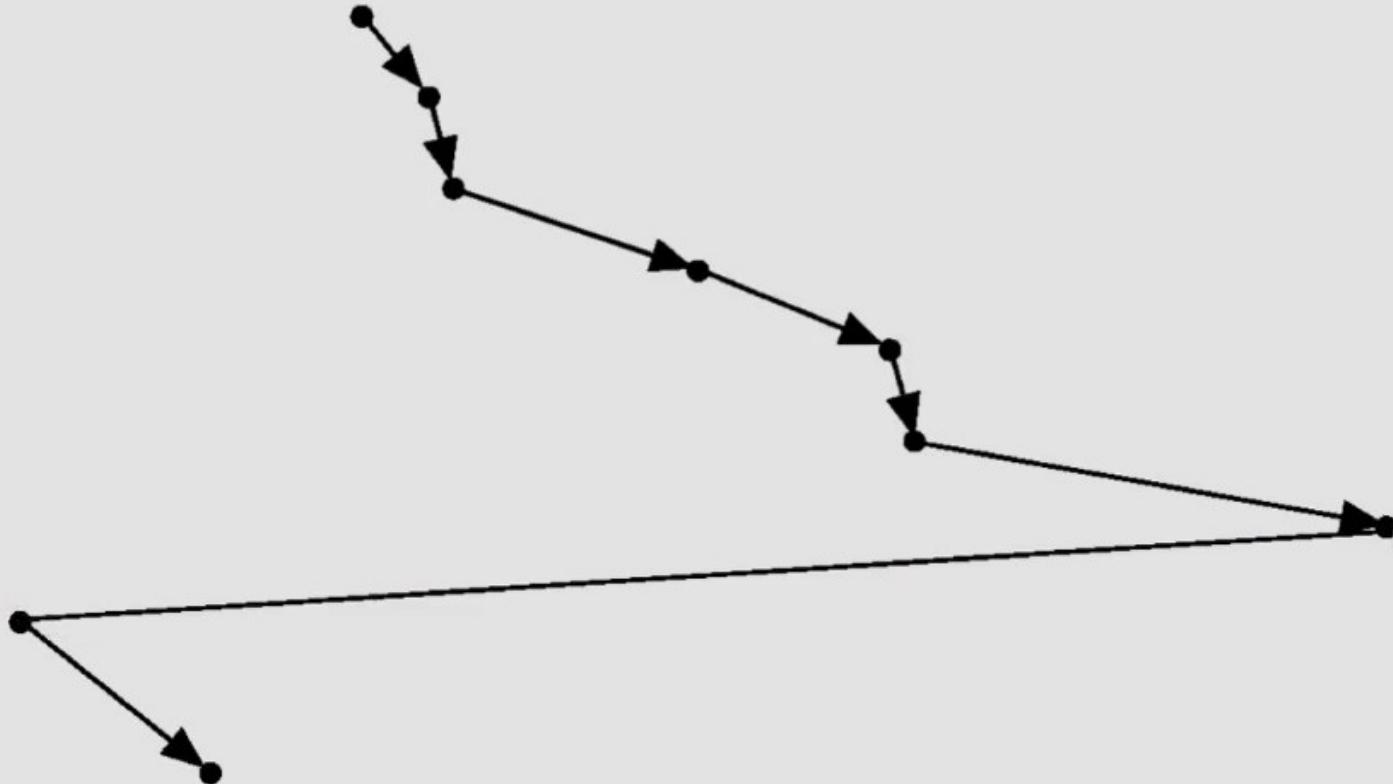
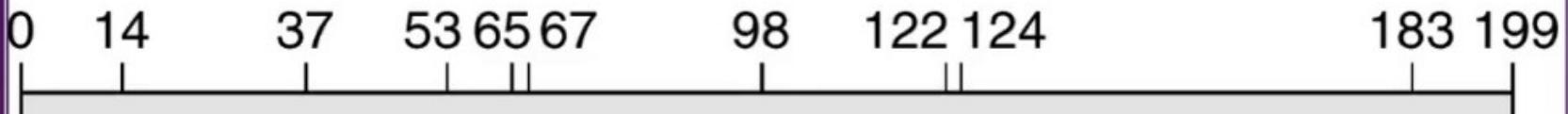
Scheduling LOOK

- Gli algoritmi di scheduling LOOK e C-LOOK rappresentano un ulteriore passo in avanti rispetto agli algoritmi SCAN e C-SCAN.
 - Infatti:
 - ◆ gli algoritmi del tipo SCAN effettuano sempre delle scansioni complete del disco,
 - ◆ gli algoritmi di tipo LOOK valutano ad ogni passo se vi sono altre richieste da servire in quella direzione, altrimenti viene immediatamente invertito il verso della testina .
- 

C-LOOK

queue = 98, 183, 37, 122, 14, 124, 65, 67

head starts at 53





Scelta di un algoritmo di scheduling

- Uno degli algoritmi più utilizzati è l'SSTF, poiché aumenta le prestazioni rispetto all'FCFS.
- SCAN e C-SCAN danno migliori prestazioni in sistemi che sfruttano molto le unità a disco, perché conducono con minor probabilità a situazioni di attesa indefinita.
- Definito l'insieme dei processi da schedulare, è possibile disegnare un algoritmo ad hoc ottimale,
 - ◆ ma il risparmio che ne deriva non e' sufficiente a compensare il calcolo necessario per ottenere l'ottimizzazione.
- E' chiaro che la prestazione dipende fortemente dal numero e dai tipi di richieste.
- Se c'è di solito solo una richiesta in sospeso, tutti gli algoritmi si equivalgono (anche FCFS).





Gestione dell'unità a disco

- Il processo di organizzazione della superficie del disco in tracce e settori che possano essere letti dal controllore è chiamato **formattazione di basso livello o formattazione fisica**,
 - ◆ quasi tutti gli hard disk oggi vengono pre-formattati dalle case produttrici.
- La formattazione di basso livello riempie il disco con una specifica struttura di dati per ogni settore.
- In quasi tutti i sistemi, inclusi PC e Mac, i settori contengono in genere un'intestazione, un'area per i dati (in genere 512 bytes) ed una coda.
- L'intestazione e la coda contengono le informazioni usate dal controllore del disco,
 - ◆ ad es. il numero di settore ed un codice per la correzione degli errori.
- Per utilizzare un disco, S.O. deve:
 - ◆ **partizionare** il disco (in una o più partizioni, suddividendolo in gruppi di cilindri)
 - ◆ **formattarlo logicamente** (cioè creare il file system).





Blocco di avviamento

- Affinché un calcolatore possa entrare in funzione, è necessario che esegua un programma iniziale.
- Di solito il **programma di avviamento** iniziale è molto semplice:
 - ◆ inizializza il sistema in tutti i suoi aspetti, dai registri della CPU ai controllori dei dispositivi e al contenuto della memoria centrale, quindi avvia il S.O..
- Per far ciò il programma di avviamento trova il nucleo del sistema operativo sui dischi, lo carica nella memoria e salta ad un indirizzo iniziale per avviare l'esecuzione del sistema operativo.
- In genere il programma di avviamento è memorizzato in una memoria di sola lettura (ROM).
- Pero' cambiare programma di avviamento significherebbe dover cambiare la ROM.
- A causa di questo inconveniente molti sistemi memorizzano nella ROM un piccolo **caricatore di avviamento** (bootstrap loader),
 - ◆ il cui solo scopo è caricare da disco il programma di avviamento completo, registrato in una partizione del **disco di avviamento** (o disco di sistema).
- In MS-DOS il programma di avviamento è breve: un blocco di 512 byte.





Blocchi difettosi

- Sebbene un ammontare straordinario di cura e sforzo da parte delle case produttrici vada nel costruire piatti per hard disk drives,
 - ◆ non è economicamente fattibile fabbricare platters al 100% liberi da errori.
- Perciò, tutti i moderni drives adottano nel controller una strategia per il management degli errori per assicurare spazio libero da **blocchi difettosi**.
- Alla fine del processo di manifattura, l'intera superficie del disco è analizzata per evidenziare eventuali difetti
 - ◆ e in questa fase il controllore del disco immagazzina una mappa delle loro ubicazioni.
- Quando il sistema operativo richiede che le informazioni debbano essere scritte in uno dei settori danneggiati,
 - ◆ il controllore del disco trasparentemente lo mappa ad uno dei settori di riserva inutilizzati (**accantonamento di settori**).
- Il controllore del disco aggiorna continuamente la mappa dei blocchi difettosi, per evitare che su nuovi settori danneggiati vengano memorizzate informazioni.





Gestione dell'area di avvicendamento

- La gestione dell'area di avvicendamento è un altro compito a basso livello del sistema operativo.
- La memoria virtuale usa lo spazio dei dischi come estensione della memoria centrale.
- I sistemi che adottano l'avvicendamento dei processi nella memoria possono usare l'area di avvicendamento (area di swap) per mantenere l'intera immagine del processo, inclusi i segmenti dei dati e del codice.
- I sistemi a paginazione possono semplicemente memorizzarvi pagine non contenute nella memoria centrale.
- Vi sono due possibili collocazioni per uno spazio di swap:
 - ◆ all'interno del file system
 - ◆ su una partizione indipendente del disco





Strutture RAID

- La presenza di più dischi rende possibile l'aumento della frequenza con cui i dati si possono leggere o scrivere, e permette di migliorare l'affidabilità della memoria secondaria.
 - Esistono tecniche per l'organizzazione dei dischi note con il nome comune di **batterie ridondanti di dischi** (Redundant Array of Independent Disks - RAID).
 - Il RAID è un metodo di organizzazione dei dischi in un array che appare al calcolatore come una singola unità di memorizzazione.
 - In passato strutture RAID composte da piccoli dischi economici erano viste come un'alternativa economicamente vantaggiosa rispetto a costosi dischi di grande capacità.
 - Oggi si impiegano invece per la loro maggiore affidabilità e velocità di trasferimento dei dati.
 - Quindi la I in RAID era originariamente letta come "Inexpensive".
- 



RAID: affidabilità tramite la ridondanza

- La possibilità che uno dei dischi in un insieme di n dischi si guasti è molto più alta della possibilità che uno specifico disco isolato presenti un guasto.
- Se si memorizzasse una sola copia dei dati, allora ogni guasto di un disco comporterebbe la perdita di una notevole quantità di dati.
- La soluzione al problema dell'affidabilità stà nell'introdurre una certa **ridondanza**,
 - ◆ cioè nel memorizzare informazioni che non sono normalmente necessarie ma che si possono usare in caso di guasto per ricostruire le informazioni perse.
- Ad esempio nella **copiatura speculare** (mirroring) ogni disco logico consiste di due dischi fisici e ogni scrittura si effettua su entrambi i dischi.





RAID: prestazioni tramite il parallelismo

- Con la copiatura speculare, se si può accedere in parallelo a più dischi, la frequenza con la quale si possono gestire le richieste di lettura raddoppia.
- Attraverso l'uso di più dischi è possibile anche migliorare la capacità di trasferimento distribuendo i dati in sezioni su più dischi.
- Ogni blocco di dati è frazionato in diversi sottoblocchi memorizzati in parallelo su dischi differenti.
- Nella sua forma più semplice questa distribuzione (**sezionamento dei dati**) consiste nel distribuire i bit di ciascun byte su più dischi (**sezionamento al livello dei bit**).
- Se distribuiamo i blocchi di un file su più dischi avremo il **sezionamento al livello dei blocchi**.
- Il fallimento di un singolo disco può però condurre a una grande perdita di dati.





Livelli RAID



- Sono stati proposti numerosi schemi per fornire ridondanza usando l'idea del sezionamento combinata con i bit di parità.
- Questi schemi realizzano diversi compromessi tra costi e prestazioni e sono stati classificati in più **livelli RAID**.
- **Raid 0:** batterie di dischi con ***sezionamento al livello dei blocchi***, ma senza ridondanza.
- **Raid 1: *copiatura speculare*,**
 - ◆ ovvero tutti i dati sono memorizzati in due copie su due diversi dischi.
- **Raid 2: *organizzazione con codici per la correzione degli errori*.**
 - ◆ I dati sono memorizzati sul disco in gruppi di bit. I bit del pattern dei dati sono memorizzati in parallelo su tutti i dischi dell'array.
 - ◆ Codici per la correzione degli errori sono generati e scritti sui dischi di ECC per identificare e correggere errori.



Livelli RAID (II)

■ Raid 3: organizzazione con bit di parità intercalati.

- ◆ Giacché i controllori dei dischi possono rilevare se un settore è stato letto correttamente, un unico bit di parità si può usare sia per individuare gli errori che per correggerli.
- ◆ Se uno dei settori è danneggiato, per ogni bit del settore è possibile determinare il suo valore considerando tutti gli altri bit dell'ottetto memorizzati sugli altri dischi + il bit di parità.

■ Raid 4: organizzazione con blocchi di parità intercalati.

- ◆ Si impiega il sezionamento dei blocchi (RAID 0) e si tiene un blocco di parità in un disco separato.
- ◆ Se uno dei dischi si guasta il blocco di parità viene utilizzato insieme agli altri blocchi corrispondenti per ripristinare il blocco perso.

■ Raid 5: organizzazione con blocchi intercalati a parità distribuita.

- ◆ Differisce dal livello 4 perché invece di memorizzare i dati in n dischi e la parità in un disco separato, i dati e la parità sono distribuiti tra gli $n+1$ dischi.





Livelli RAID (III)

■ Raid 6: *schema di ridondanza P+Q*.

- ◆ Molto simile al 5 ma memorizza ulteriori informazioni ridondanti per gestire guasti contemporanei di più dischi. Invece della parità si utilizzano i **codici di Reed-Solomon**.

■ Raid 0 + 1:

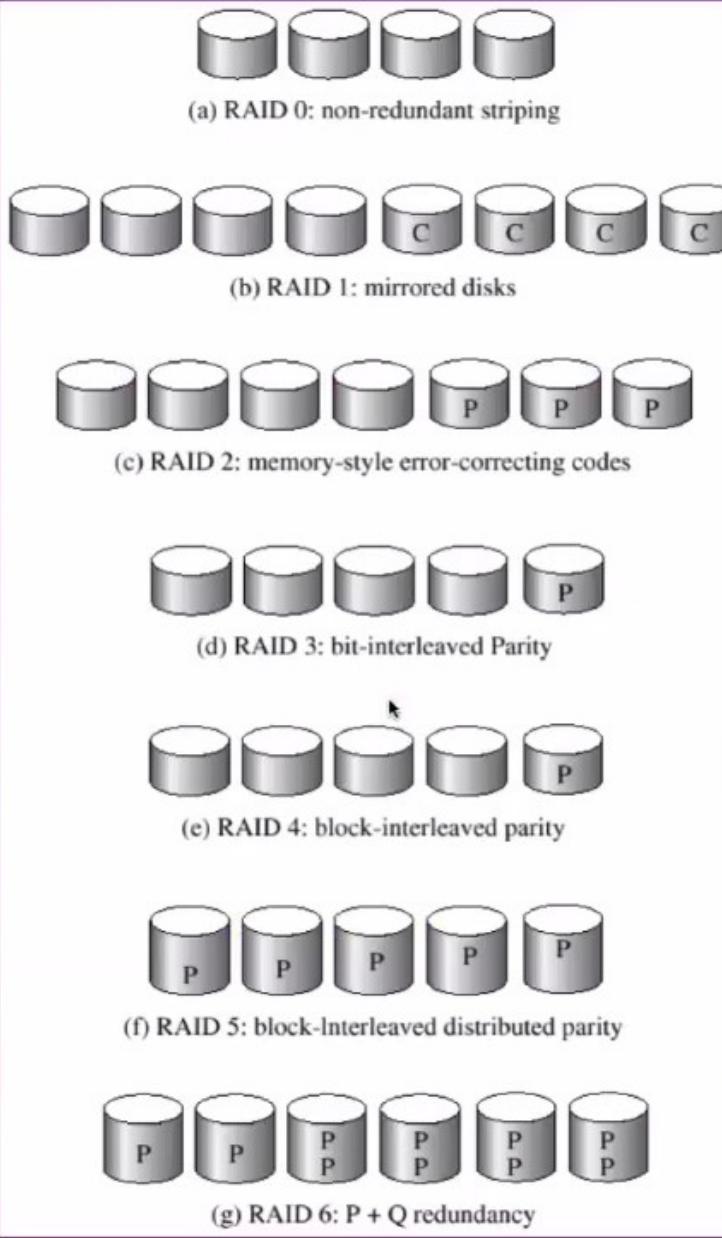
- ◆ è una combinazione dei livelli 0 e 1.
- ◆ Si sezionano al livello dei blocchi i dati in un insieme di dischi e poi si duplica (indipendentemente ed eventualmente in modo diverso) ogni sezione con la tecnica della copiatura speculare.

■ Raid 1 + 0:

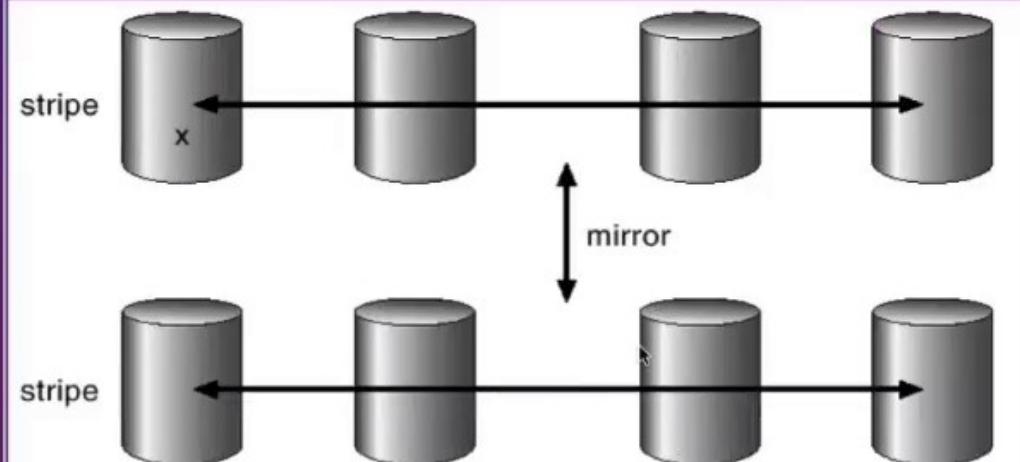
- ◆ si fa prima la copiatura speculare dei dischi a coppie e poi il sezionamento su queste coppie.
- Teoricamente in 0+1 se si guasta un disco l'intera sezione dati diventa inaccessibile, mentre in 1+0 si può utilizzare lo stesso disco rimanente per entrambe le sezioni.



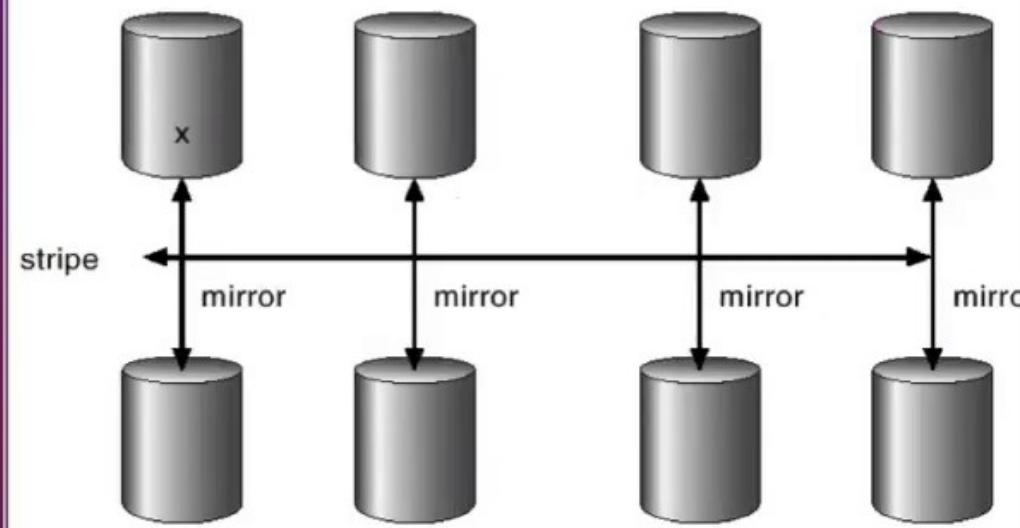
Schematizzazione dei livelli RAID



RAID (0 + 1) e (1 + 0)



a) RAID 0 + 1 with a single disk failure



b) RAID 1 + 0 with a single disk failure



Scelta di un livello RAID

- Se un disco si guasta il tempo per ricostruire i dati in esso memorizzati può essere rilevante e può variare secondo il livello RAID impiegato.
- La ricostruzione più semplice si ha al livello 1, poiché i dati possono essere semplicemente copiati dal duplicato del disco.
- Per gli altri livelli è necessario accedere a tutti i dischi della batteria.
- Il RAID di livello 0 si usa nelle applicazioni ad alte prestazioni in cui le perdite di dati non sono critiche.
- Il RAID di livello 1 si usa nelle applicazioni che richiedono un'alta affidabilità ed un rapido ripristino.
- I RAID 0+1 e 1+0 si usano dove entrambe prestazioni ed affidabilità sono importanti, per esempio per piccole basi di dati.
- Per la memorizzazione di grandi quantità di dati, in genere si preferisce impiegare il RAID di livello 5, non essendo il livello 6 sempre disponibile.
- I concetti relativi ai sistemi RAID sono stati generalizzati ad altri dispositivi,
 - ◆ ad es. batterie di nastri o diffusione di dati in sistemi wireless.

