Sistemi Operativi 1

AA 2018/2019

Gestione della
memoria secondaria

UNIVERSITÀ DEGLI STUDI
DI TRENTO

Sommario

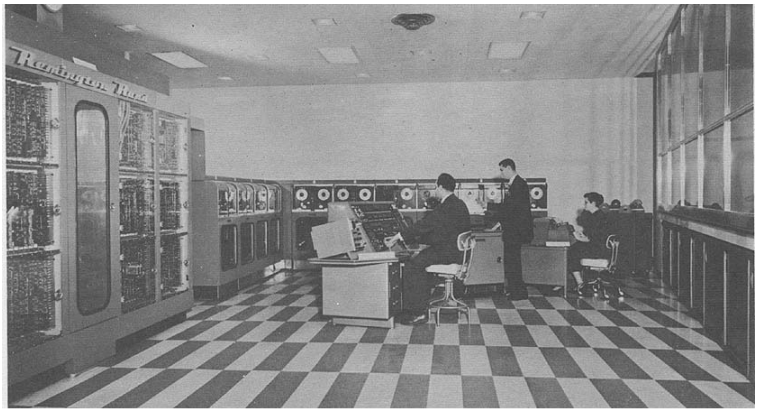
- Tipologia del supporto
  - Nastri magnetici
  - Dischi magnetici
  - Dispositivi a stato solido
- Scheduling degli accessi a disco
- Gestione del disco
  - Formattazione
  - Blocchi difettosi
  - Area di swap

TIPOLOGIE DEL SUPPORTO

Nastri magnetici

- Sottile striscia in materiale plastico, rivestita di un materiale magnetizzabile.

- Usati per memorizzare dati digitali per la prima volta nel 1951 su Eckert-Mauchly UNIVAC I



Nastri magnetici

- Massima capienza:
  - Circa 5TB nel 2011
- Accesso sequenziale
  - Molto più lenti della memoria principale e dei dischi magnetici in termini di tempo di accesso
  - Riposizionamento della testina di lettura richiede decine di secondi
    - 3 ordini di grandezza superiore rispetto a dischi magnetici
- Rimpiazzati dai dischi magnetici/memorie a stato solido
- Ormai usati solo per backup

Dischi magnetici

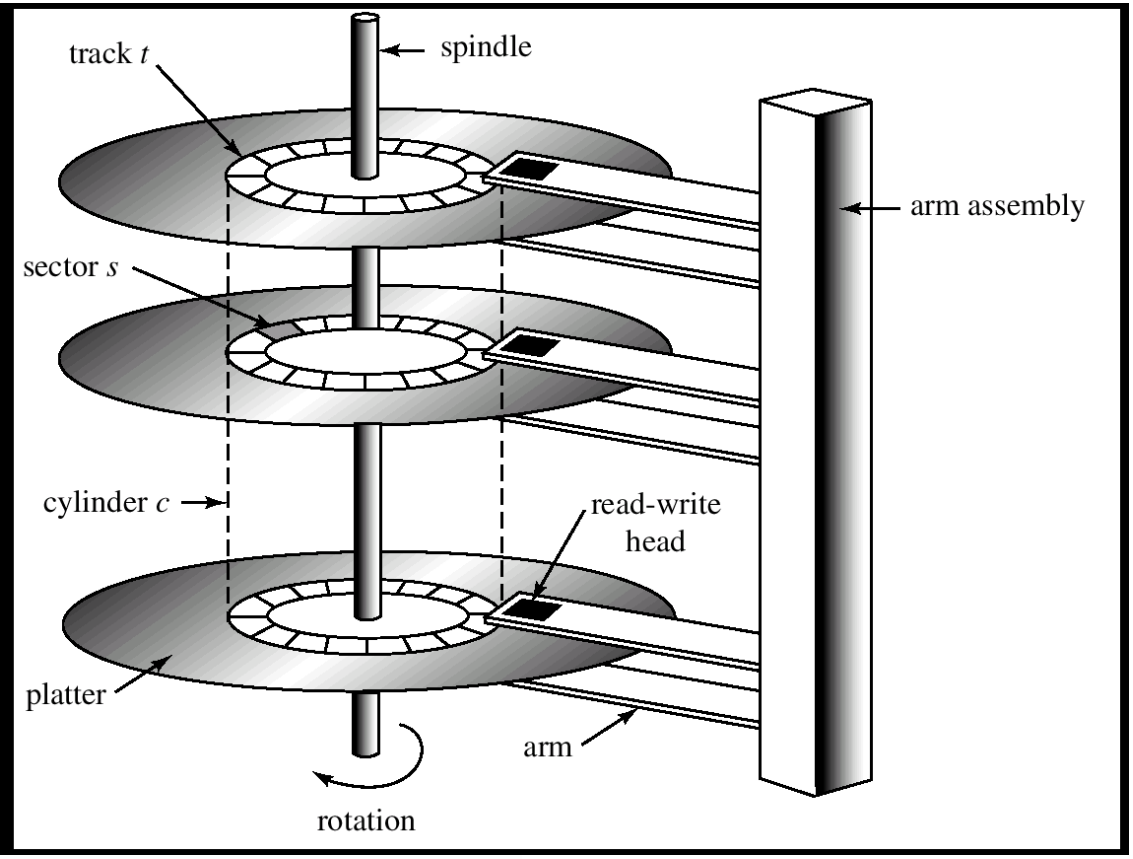
- Piatti d'alluminio (o di altro materiale) ricoperti di materiale ferromagnetico
- Massima capienza
  - 4TB nel 2011
- Fattore di forma (diametro)
  - sempre più piccolo (consente velocità di rotazione maggiori)
  - 3.5 pollici per i sistemi desktop e fino a 1 pollice per i mobili

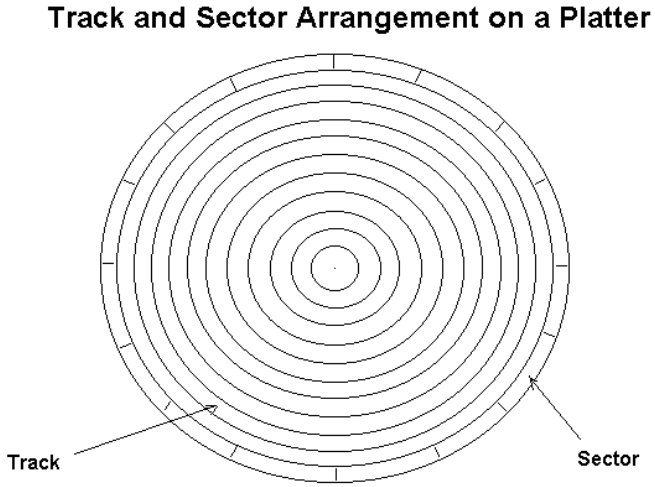
Dischi magnetici

- Lettura/scrittura tramite testina sospesa sulla superficie magnetica
  - scrittura:
    - passaggio di corrente positiva o negativa attraverso la testina magnetizza la superficie
  - lettura:
    - passaggio sopra un'area magnetizzata induce una corrente positiva o negativa nella testina.

Struttura di un disco

- Vista fisica
  - Superfici (platter), cilindri (cylinder), tracce (track), settori (sector)





Settore

- Unità più piccola di informazione che può essere letta/ scritta su disco
- Dimensione variabile tra 32 byte e 4 KB (solitamente 512 byte)
- Generalmente raggruppati dal punto di vista logico in cluster per motivi di efficienza
    - Dimensione dipende dalla grandezza del disco e dal sistema operativo (da 2KB a 32KB)
    - Es.: Disco fisso da 60GB → cluster da 4KB su WinXP e Linux
  - Un file occupa sempre almeno un cluster!
- Per accedere a un settore bisogna specificare la superficie, la traccia e il settore stesso

Tempo di accesso al disco

- Tempo di accesso = Seek time + Latency time + Transfer time
- Seek time
  - Tempo necessario a spostare la testina sulla traccia
- Latency time (latenza)
  - Il tempo necessario a posizionare il settore desiderato sotto la testina
  - Dipende dalla velocità di rotazione
- Transfer time
  - Il tempo necessario al settore per passare sotto la testina (lettura vera e propria)

Tempo di accesso al disco - valori

- Seek time
  - Tipicamente da 3ms (server driver) fino a 15ms (mobile driver)
  - Mediamente 9ms per un disco su desktop
- Latenza
  - in media mezza rotazione del disco per ogni accesso = 0,5*(60/ velocità di rotazione) = 0,5/(vel. di rot. in secondi)
  - Per um disco da 7200 rpm si ha 4.16ms
- Transfer time
  - dimensione blocco/ velocità di trasferimento
  - Per un disco da 7200 rpm si ha:
    - "disk-to-buffer" 1030 Mbits/sec.
    - "buffer-to-computer" 300 MB/sec.

Tempo di accesso al disco - esempio

- Velocità di trasferimento = 40MB/s
- Velocità di rotazione = 10000 rpm = 166 rps
- Rotazione media = 1/2 traccia
- Dimensione blocco = 512 Byte
- Seek time = 5ms

- $T_{accesso}$ = 5 ms + 0.5/166 + 0.5KB / 40 MB = 5 ms + 3ms + 0.0000125 = 8.0000125 ms

Considerazione

- Seek time dominante!
- Quindi, dato il grande numero di processi nel sistema che accedono al disco è necessario minimizzare il seek time tramite algoritmi di scheduling per ordinare gli accessi al disco in modo da:
  - minimizzare il tempo di accesso totale
    - Ridurre seek time significa ridurre lo spostamento della testina
  - massimizzare la banda = n° di byte trasferiti / tempo
    - Misura la velocità effettiva

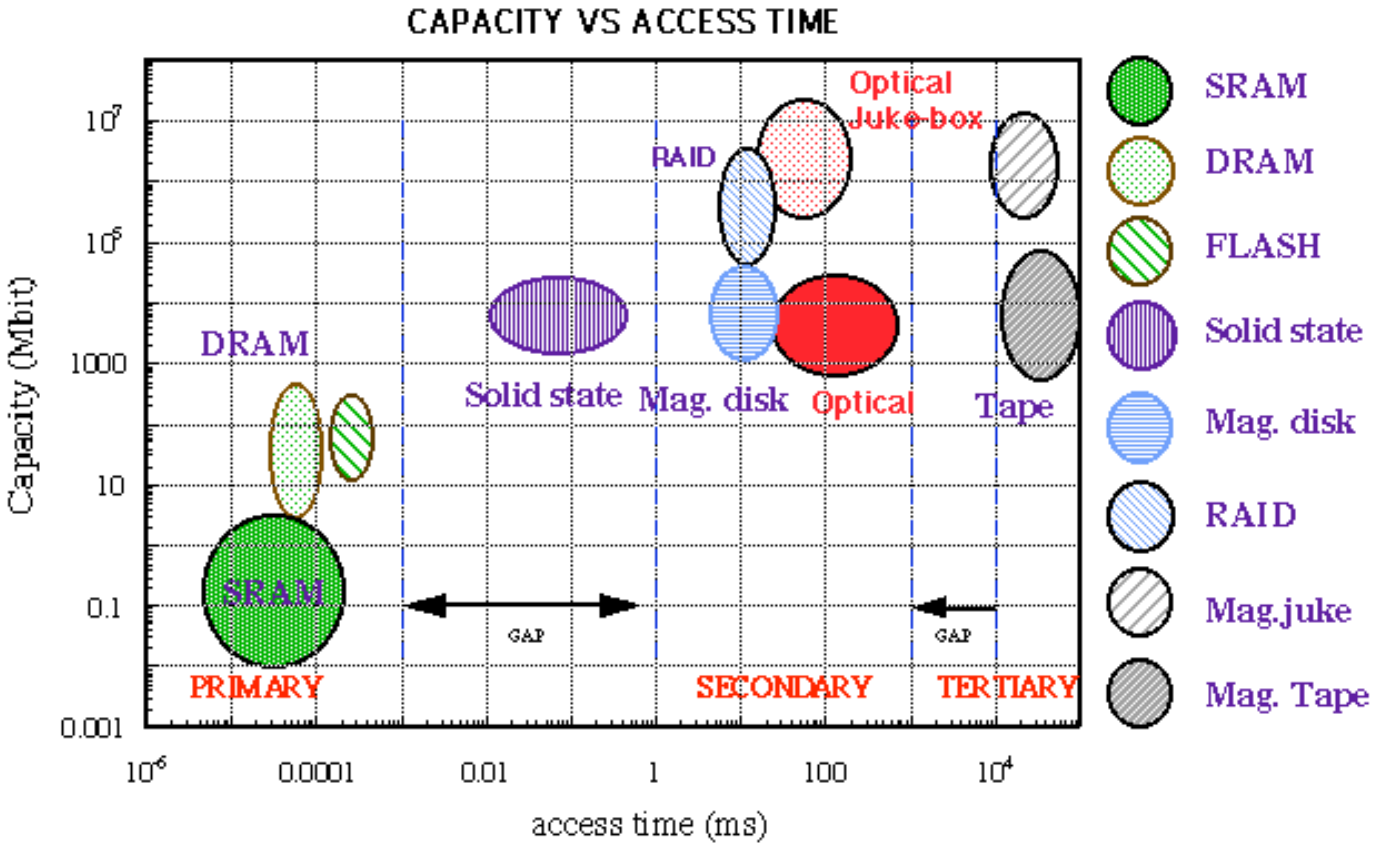
Dispositivi a stato solido

- Utilizza chip (NVRAM o memorie flash NAND) per memorizzare i dati in modo non volatile
- Usano la stessa interfaccia dei dischi fissi e quindi possono rimpiazzarli facilmente

- Capacità:
  - Fino a 2TB, ma generalmente non più di 256GB a causa del costo

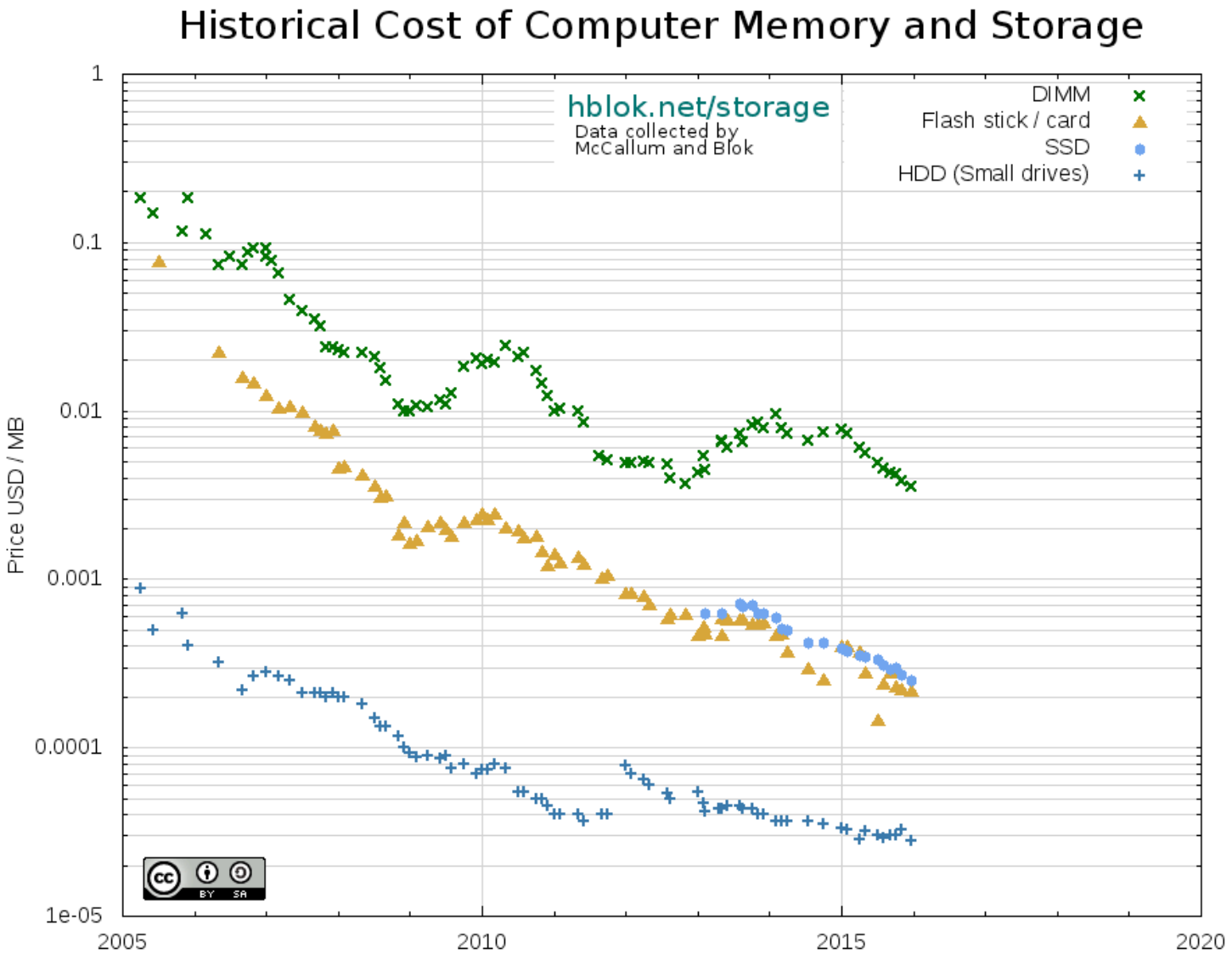
Flash memory

- Performance
  - Letture simili a DRAM (~ns)
  - Scritture simili a dischi magnetici (~ms). La scrittura e' un'operazione complessa.



Dispositivi a stato solido

- Rispetto ai dischi fissi sono:
  - meno soggette a danni
    - ma memorie flash hanno un limite sul numero di write (1-5 milioni)
  - più silenziosi (non si muovono)
  - più efficienti in termini di tempo di accesso
  - non necessitano di essere deframmentate
  - più costose

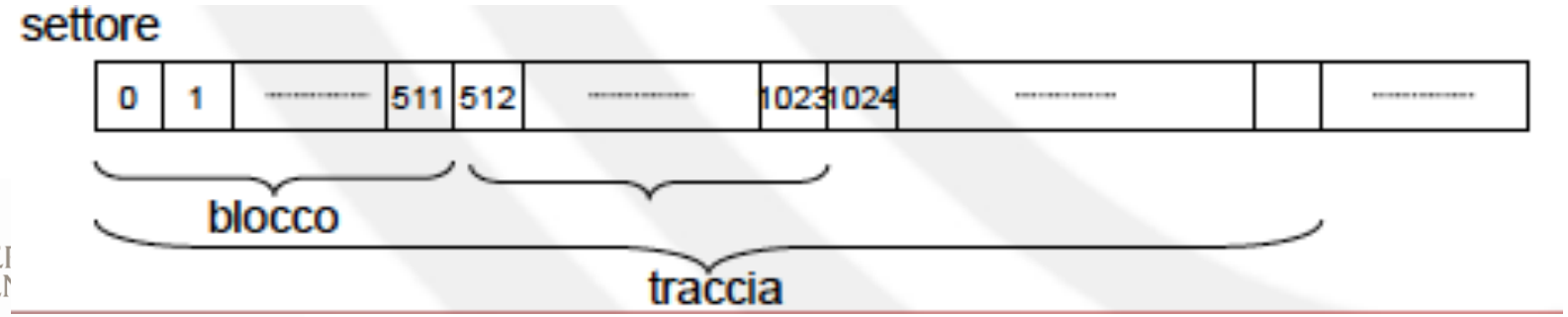

Historical Cost of Computer Memory and Storage
hblok.net/storage
Data collected by
McCallum and Blok
DIMM
Flash stick / card
SSD
HDD (Small drives)
Price USD / MB
1
0.1
0.01
0.001
0.0001
1e-05
2005
2010
2015
2020

SCHEDULING DEGLI ACCESSI A DISCO

UNIVERSITÀ DEGLI STUDI
DI TRENTO

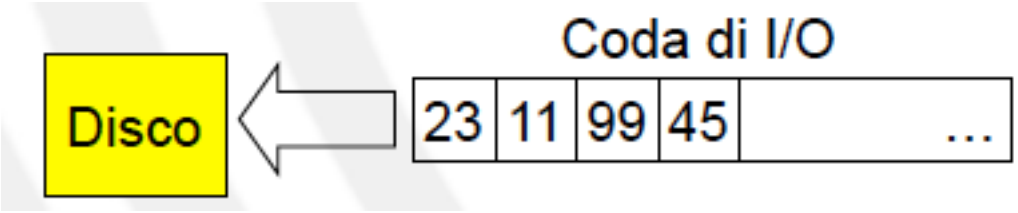
Struttura di un disco

- Vista logica
  - Vettore unidimensionale di blocchi logici
  - Blocco (cluster) = unità minima di trasferimento
- Il vettore è mappato sequenzialmente sui settori del disco
  - Settore 0 = primo settore della prima traccia del cilindro più esterno
  - La numerazione procede gerarchicamente per settori, tracce, piatti (nell'ordine)



Disk scheduling

- Processo che necessita I/O esegue system call
- S.O. usa la vista logica del disco
  - Sequenza di accessi = sequenza di indici del vettore



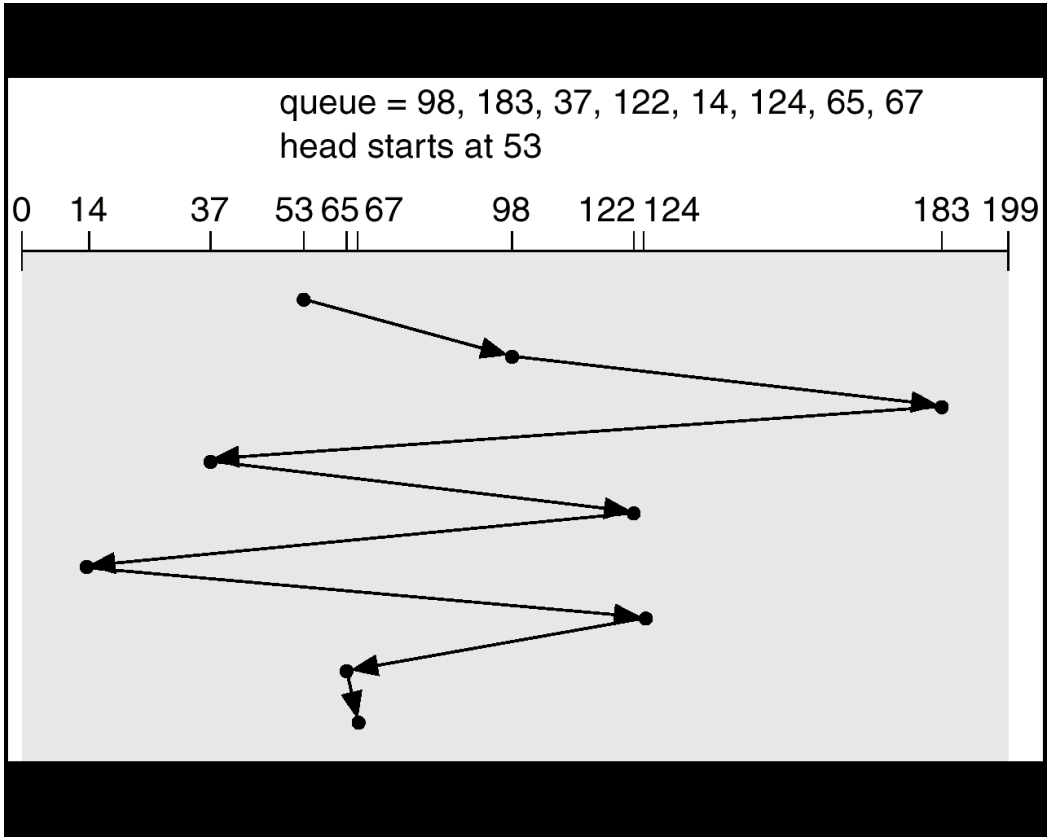
  - Altre informazioni contenute nelle richieste di I/O
    - Tipo di accesso (R/W)
    - Indirizzo di memoria destinazione
    - Quantità di dati da trasferire

Algoritmi di disk scheduling

- Tradizionale compromesso costo/efficacia
- Valutati tramite una sequenza di accessi di esempio
  - Esempio di riferimento:
    - Inizialmente testina sulla traccia 53
    - Accessi: 98, 183, 37, 122, 14, 124, 65, 67
    - Intervallo valori ammissibili = [0,199]

FCFS

- Richieste processate nell'ordine d'arrivo
- Esempio:
  - Spostamento totale testina = 640 tracce

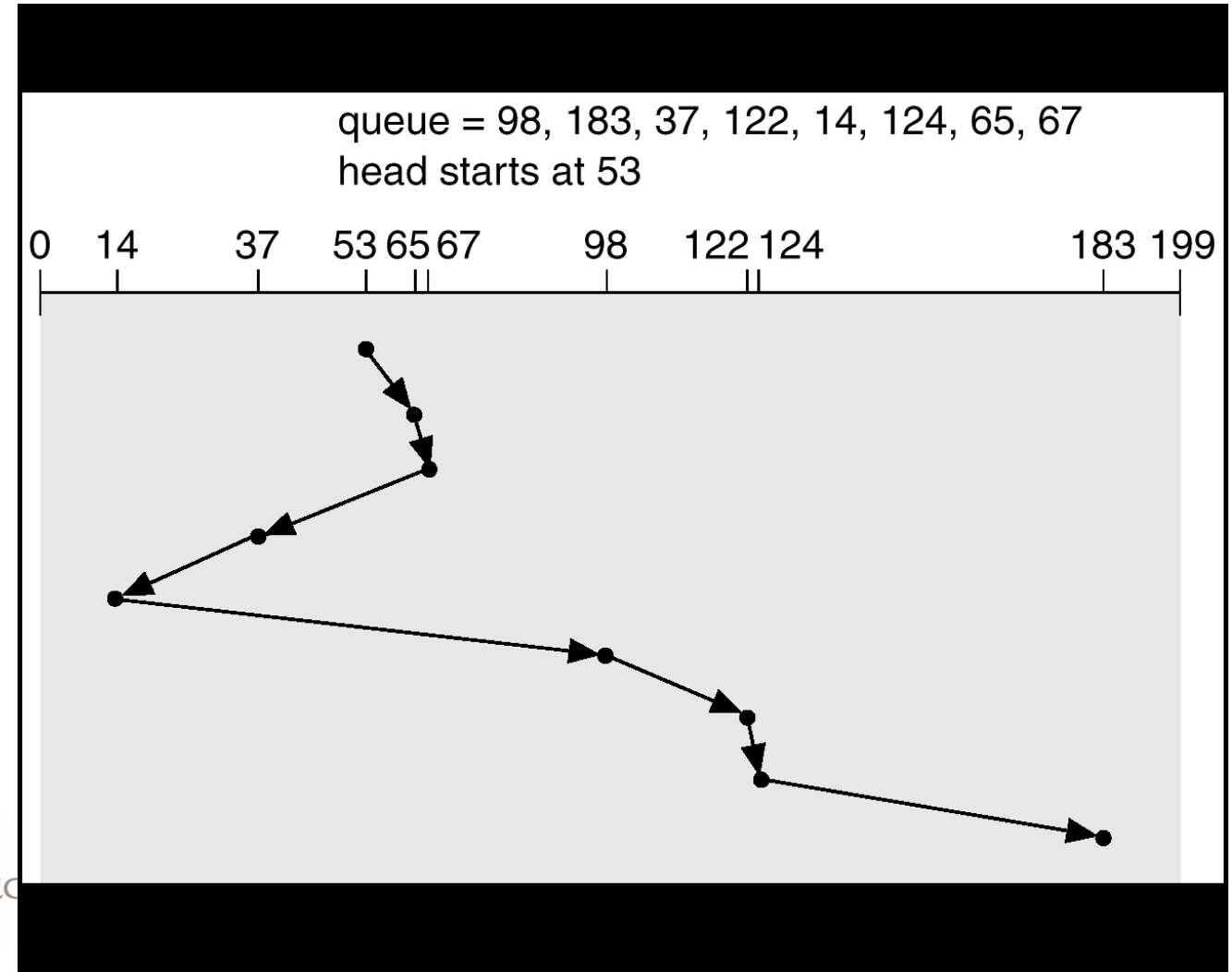


SSTF

- SSTF = Shortest Seek Time First
- Scelta più efficiente:
  - selezionare la richiesta con il minimo spostamento rispetto alla posizione attuale della testina
- Simile a SJF
  - Possibilità di starvation di alcune richieste
- Migliora FCFS, ma non è ottimo!

SSTF - esempio

- Spostamento totale testina = 236 tracce

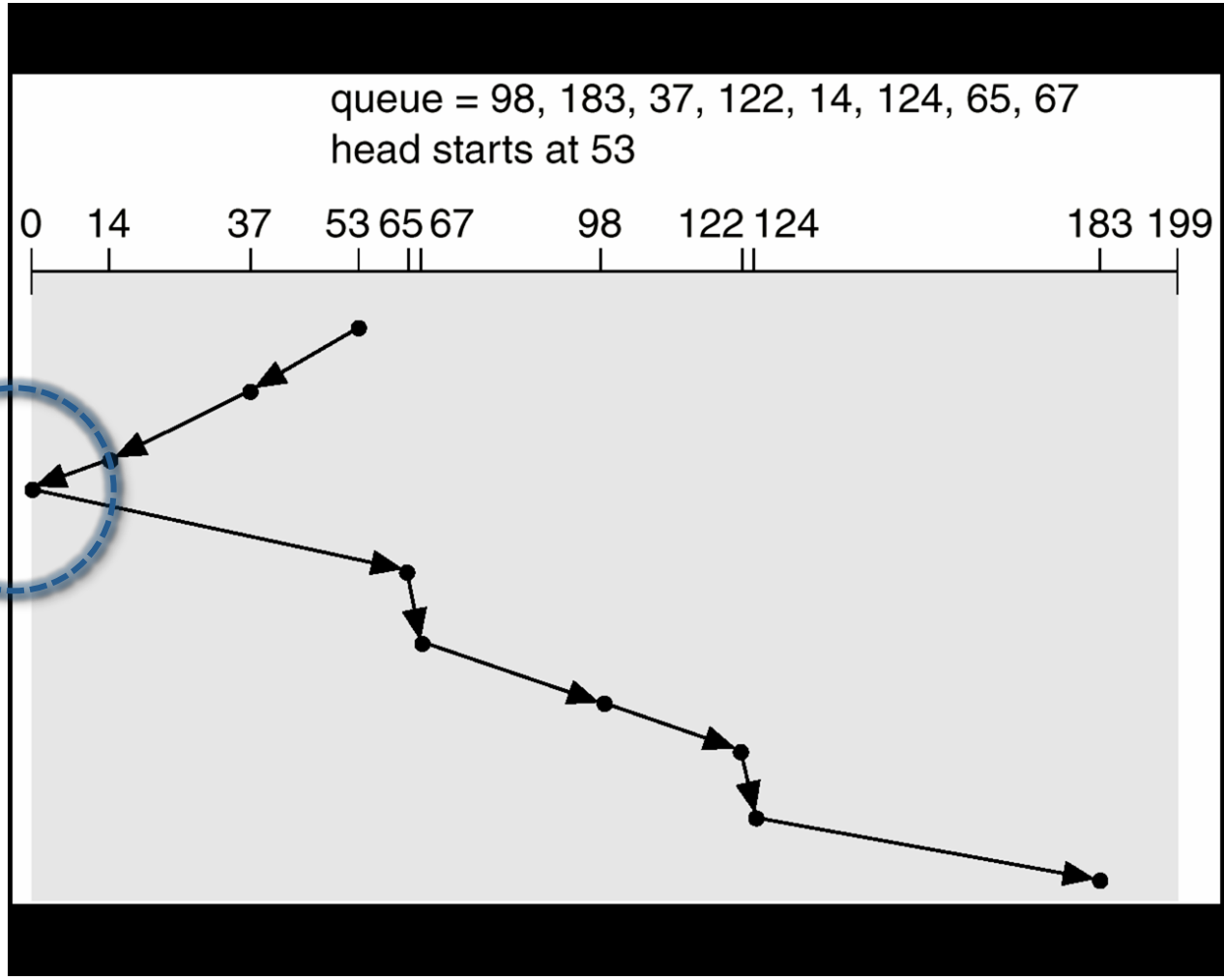


SCAN

- L'osservazione della natura dinamica delle richieste porta all'algoritmo SCAN
  - La testina parte da un'estremità del disco
  - Si sposta verso l'altra estremità, servendo tutte le richieste correnti
  - Arrivato all'altra estremità, riparte nella direzione opposta, servendo le nuove richieste
- Detto anche algoritmo dell'ascensore

SCAN - esempio

- Spostamento totale testina = 236 tracce
  - Direzione di spostamento decrescente

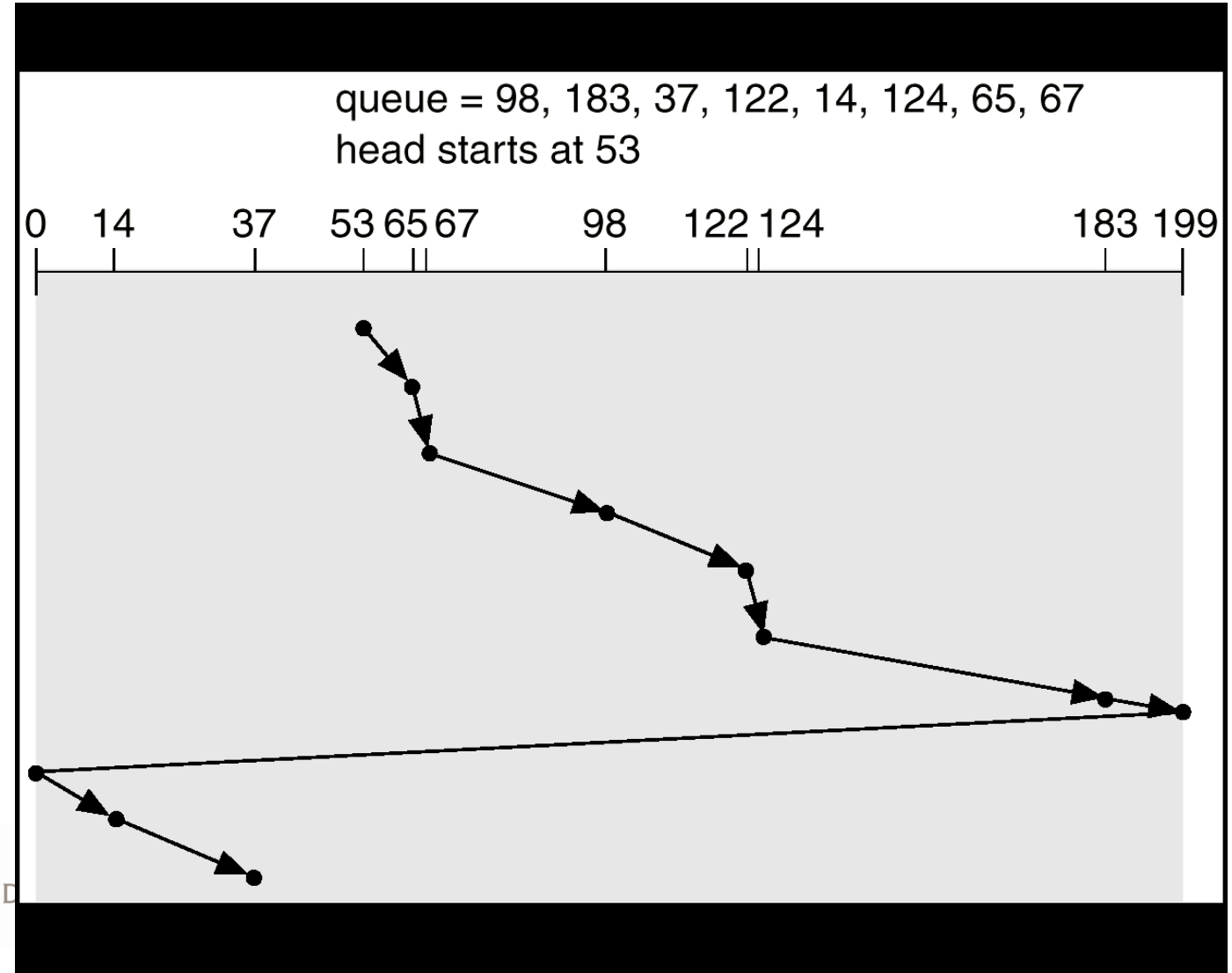


Varianti dello SCAN - CSCAN

- SCAN Circolare
  - Come SCAN, ma quando la testina arriva ad una estremità, riparte immediatamente da 0 senza servire altre richieste
  - Disco è visto come lista circolare
  - Tempo di attesa più uniforme rispetto a SCAN
  - Alla fine di una scansione ci saranno + richieste all'altro estremo
    - Esempio spalatore di neve

CSCAN - esempio

- Spostamento = 382

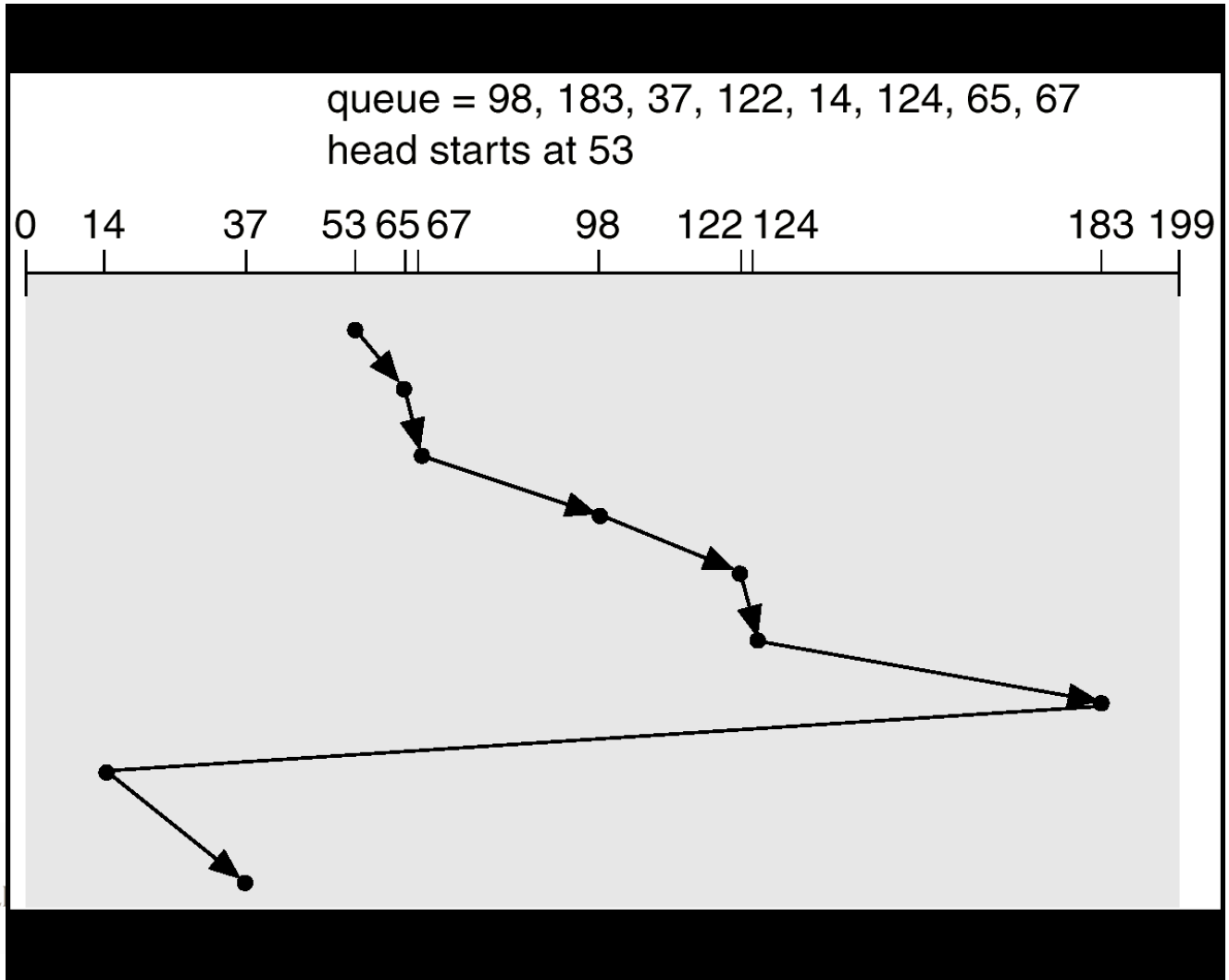


Varianti dello CSCAN – (C-)LOOK

- La testina non arriva sino all' estremità del disco
- Cambia direzione (LOOK) o riparte dalla prima traccia (C-LOOK) non appena non ci sono più richieste in quella direzione

C-LOOK - esempio

- Spostamento = 322



Varianti dello SCAN – N-step SCAN

- Per evitare che la testina rimanga sempre nella stessa zona
  - La coda delle richieste viene partizionata in più code di dimensione massima N
  - Quando una coda viene processata per il servizio (scan in una direzione), gli accessi in arrivo riempiono altre code
    - Dopo che una coda è stata riempita non è possibile riordinare le richieste
  - Le code "sature" vengono servite nello scan successivo
    - N grande = degenera in SCAN
    - N = 1 degenera in FCFS
- FSCAN
  - Simile, ma con due sole code

Last-In-First-Out (LIFO)

- In certi casi, può essere utile schedulare gli accessi in base all'ordine inverso di arrivo
- Idea
  - Utile nel caso di accessi con elevata località
  - Es: serie di accessi "vicini"
- Possibilità di starvation!

Analisi degli algoritmi

- Nessuno di quelli visti è ottimo!
  - Algoritmo ottimo poco efficiente
    - Risparmio ottenibile non compensa complessità di calcolo
- Fattori che influenzano l'analisi
  - Distribuzione degli accessi (numero e dimensioni)
    - Molti medio-piccoli vs. pochi grandi
    - Più l'accesso è "grande" più il peso relativo del seek time diminuisce!
  - Organizzazione delle informazioni su disco (file system)
    - Accesso alle directory essenziale: tipicamente posizionate lontane dalle estremità del disco
- In generale
  - – SCAN/C-SCAN migliori per sistemi con molti accessi a disco
- Disk-scheduling spesso implementato come modulo indipendente dal S.O., con diverse scelte di algoritmo

GESTIONE DEL DISCO

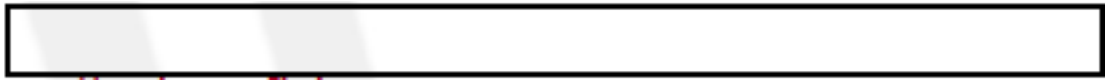
UNIVERSITÀ DEGLI STUDI
DI TRENTO

Formattazione dei dischi

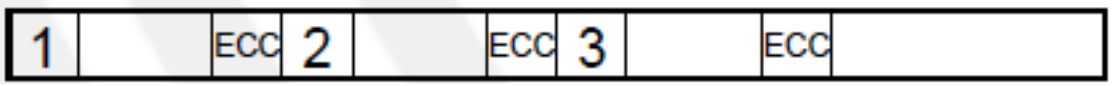
- Formattazione di basso livello o fisica
  - Divisione del disco in settori che il controllore può leggere o scrivere
- Per usare un disco come contenitore di file, il S.O. deve memorizzare le proprie strutture sul disco
  - Partizionamento del disco in uno o più gruppi di cilindri (partizioni)
  - Formattazione logica per creare un file system
  - Programma di boot x inizializzare il sistema
    - Programma di bootstrap memorizzato nella ROM (bootstrap loader) carica il bootstrap dal disco (dai blocchi di boot)
      - Carica driver dei dispositivi
      - Lancia avvio del sistema operativo

Formattazione dei dischi

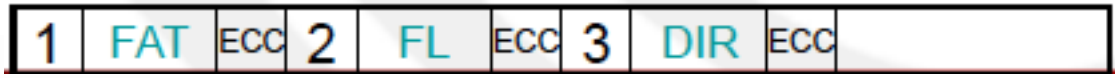
- Vista concettuale:
  - Disco “nudo”



  - Formattazione fisica:
    - Suddivisione in settori
    - Identificazione dei settori
    - Aggiunta di spazio per correzione di errori (ECC)



  - Formattazione logica:
    - File system
    - Lista spazio occupato (FAT) e libero (Free List – FL)
    - Directory vuote



Gestione dei blocchi difettosi

- L'ECC (Error Correction Code) serve a capire (in lettura o scrittura) se un settore contiene dati corretti o meno
- Schema tipico (es.: lettura)
  - Il controllore legge un settore (incluso ECC)
  - Il controllore calcola l'ECC per i dati appena letti
  - Se il risultato è diverso → ERRORE (bad block)
- L'errore va segnalato in qualche modo
- Come gestire i bad block?
  - Off-line
  - On-line

Gestione dei blocchi difettosi

- Gestione off-line dei bad block
  - Durante la formattazione logica si individuano i bad block e si mettono in una lista
    - Rimozione
    - Marcatura della entry nella FAT
  - Successivamente si può eseguire una opportuna utility per "isolare" bad block (es. CHKDSK del DOS)
  - Tipico di controllori IDE

Gestione blocchi difettosi

- Gestione on-line dei bad block
  - Il controllore mappa il bad block su un blocco buono
    - Non può essere un blocco già usato!
  - Necessario disporre di blocchi “di scorta” opportunamente riservati (sector sparing)
  - Quando il S.O. richiede il bad block, il controllore mappa in modo trasparente l’accesso al bad block sul blocco di scorta
  - Problema: potrebbe inficiare le ottimizzazioni fornite dallo scheduling!
    - Soluzione: si allocano spare block in ogni cilindro!

Gestione dello spazio di swap

- Usato dalla memoria virtuale come estensione della RAM
- Opzioni
  - Ricavato dal normale file system
    - Possibile usare comuni primitive di accesso a file
    - Inefficiente
      - Costo addizionale nell'attraversamento delle strutture dati per le directory e i file
  - In una partizione separata
    - Non contiene informazioni/strutture relative al file system
    - Usa uno speciale gestore dell'area di swap (swap daemon)
    - Soluzione più tipica!

Gestione dello spazio di swap

- Gestione dello spazio di swap
  - Allocato quando il processo viene creato (4.3 BSD)
    - Assegnato spazio per pagine di istruzioni e di dati
    - Kernel usa due mappe di swap per tracciare l'uso dello spazio di swap
  - Allocato quando una pagina viene forzata fuori dalla memoria fisica (Solaris 2)