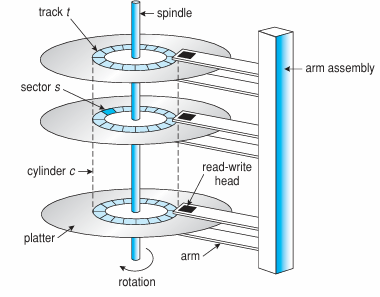
# CH11 -DISCHI

Struttura di memoria di massa:

* Da 10GB a TB
* HDD hanno capienza superiore rispetto alle NMV memorie non volatili

## **HDD**

dispositivi in cui:

* Informazione immagazinata grazie a dispositivi magnetici
  + Bit si basano su bipoli magnetici
    - Isolando, su ogni piccola area, un insieme di bipoli magnetici orientandoli usando delle testine calamite 

Dispositivo che ruota e una traccia a centri concentrici:

* Abbiamo dei dischi con delle piccole aree disposte su una circonferenza → **tracce**
  + bit messi in fila in diversi settori che formano una circonferenza
* Le testine sono in grado di modificare o leggere un campo magentico
* Tracce devono essere sottili
  + traccia (Fila di bit su una faccia)

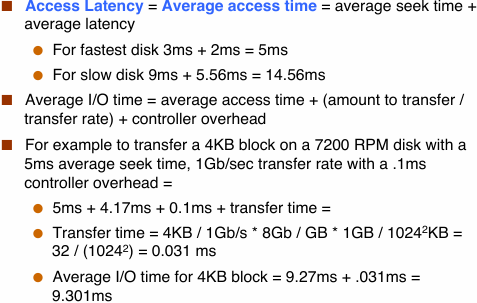
**Funzionamento**:

* Bisogna capire dove scrivere un bit/byte:
  + **Identificare Cilindro** (fila di bit su più facce)
    - Bisogna spostare la testina ad una determinata distaza dal centro → cilindro
  + **Identificare settore**
    - Bisogna svolgere un movimento angolare per raggiungere il settore

Nota su **performance HDD**:

* Ruota da 60 a 250 volte al secondo → non molto veloce in realtà
* **Velocità di trasferimento** → quanti bit/byte vanno dal disco/verso il disco quando si legge o scrive
  + ~ 1GB/sec
* **Tempo di posizionamento** = seak time+latenza rotazionale
  + **Seek time** → tempo pre raggiungere il cilindro interessato
    - Dai 3ms ai 12ms
      * Varia in base a quello che si sta facendo e alla situazione attuale
      * seekTime medio calcolato come 1/3 delle tracce
  + **Immagine che contiene testo, Carattere, schermata

    Descrizione generata automaticamenteLatenza rotazionale** → tempo per raggiungere il settore, ruotando
    - dipende dal settore in cui mi trovo e dove devo andare
      * se si gira in una sola direzione:
        + caso peggiore → tempo per effettuare un giro
        + caso medio → mezzo giro
* **Hard crash** → se la testina per qualche problema meccanico, non è alla distanza dovuta dal piatto e lo colpisce → buttare

****  
  
  
  
  
4.17=(1/(7200/60))/2  
→ conversione in Byte

**Nota**: frame → memoria fiscia

**Pagine** → memoria logica

**Blocco** → ora

**NMV**: non volatili basate su silicio → non perdono informazioni quando si spengono (SSD)

## **SSD**

memorie DRAM con tecniche derivanti dal ROM dopo una serie di upgrade.

* Più affidabili degli hard disk
* Costano di più
* Potrebbero avere vita inferirore
* Meno capacità
* Molto più veloci

Leggere è più facile

Per scrivere, devo prima cancellare; inoltre c’è un limite sulla riscrittura (Es: max 100mila cancellazioni)

* Alcune pagine rischiano di non essere più valide perché hanno raggiunto un certo numero di riscritture
  + Bisogna poter marcare delle pagine come valide e altre come non valide → diminuisce lo spazio disponibile
    - Serve una tabella
    - Bisogna avere un garbage collector
    - Si alloca un po’ più di memoria in modo tale che se uno dei pezzi diventa non valido, ne ho ancora un po’
    - Supponnendo che serva leggere e scrivere molte volte il blocco 2500 → si può fare **wearLeveling** → ogni volta che riscrivi, scrivo da un’altra parte e faccio un mapping in modo tale da non dover scrivere sempre sullo stesso

Immagine che contiene testo, schermata, Rettangolo, Carattere

Descrizione generata automaticamente

Memorie non valitili NVM:

* non hanno problemi di scheduling.
* Il difetto delle NVM è che le operazioni di schittura costano di più → possono rallenatare.

# Memorie volatili:

è possibile, in certi contesti, che servi una memoria di massa volatile → salvare solo con dispositivo acceso e poi perdere tutto. Questo perchè ci sono informazioni che sono viste dal sw come dei file che servono solo da accensione a spegnimento. → RAM

* Molto veloci
* Molto piccole

## STRUTTURA DEL DISCO:

dal punto di vista logico , dischi sono organizzati come:

* **Vettore di blocchi**

Formattazione di basso livello → crea una tabella logica con corrispondenza tra blocchi logici e dove stanno fisicamente:

* Mappato sui settori (dove settore 0: primo settore sulla prima traccia dell’esterno)
* Traduzione dovrebbe essere semplice tranne in caso di bad sector
  + La miglior cosa sarebbe avere velocità di lettura costante

### HDD Scheduling:

* Dischi non utilizzati necessariamente per una sola richiesta alla volta ma essendoci concorrenza, più processi e parti kernel che hanno bisogno del disco nello stesso momento, qualcuno dovrà decidere **chi passa prima e chi passa dopo** con l’obbiettivo di:
  + Minimizzare il tempo di seek per minimizzare il tempo perso
* **SO mantiene una coda delle richieste**
  + C’è solo un processo → servito tutto
  + Ci sono più richieste → algoritmo di ottimizzazione (gestito a bordo del disco)

**Immagine che contiene testo, Carattere, schermata, bianco

Descrizione generata automaticamente**

### **Algoritmi:**

* **FCFS**: primo arrivato, primo servito

Immagine che contiene linea, schermata, Diagramma, diagramma

Descrizione generata automaticamente

Tragittto toale di 640 cilindri

* Immagine che contiene linea, testo, diagramma, Diagramma

  Descrizione generata automaticamente**SCAN**: testina parte ad una delle estremità del disco e va verso l’altra raccogliendo tutti quelli che vanno in quella direzione

Spostamenti totali di 208 cilindri

Immagine che contiene linea, testo, schermata, Diagramma

Descrizione generata automaticamente

* **CSCAN**: più uniforme percè vanno in un’unica direzione
  + Discesa veloce senza vedere nulla
  + Salita lenta fornendo i servizi richiesti
* **SSTF** → vado a quello più vicino a me

Starvation → **impossibilità perpetua, da parte di un processo pronto all'esecuzione, di ottenere le risorse sia hardware sia software di cui necessita per essere eseguito.**

## Gestione errori

È possibile che ci siano informazioni che si guastino o falliscano per qualche motivo (avevi sctitto un valore e ne leggi un altro o un valore non stabile), dunque:

**DETECTION**: determinare se c’è stato un problema:

* **Bit di parità** 
  + Se sbagliano un numero pari di bit → non se ne accorge
* **Checksum** 
  + Usa aritmetica modulare per calcolare, salvare e comparare valori di parole di lunghezza fissa

**Codice per correzione:**

* Oltre a detection, possono fare correzione sfruttando rindondanza
  + Es: bit di parità sia per riga che per colonna
    - Se becchi incrocio RIGA-COLONNA errato → cambia valore

## Storage Device Management:

Gestione del disco:

* Disco va **formattato**: nel passaggio di un disco **da**  semplice sequenza di blocchi **a** file system organizzato, ci sono molte posizione intermedie
  + **Liv fisico** (Low-level): dividere il disco **in settori** che possono essere trattati come parti da leggere o scrivere in modo individuale
    - Ad ogni settore si possono aggiungewre delle informazioni per rilevamento, gestione, correzione dell’errore ( ~kb)
  + **Liv logico**

Per usare un disco per contenere file, il sistema operativo ha bisogno di **memorizzare la sua struttura dati:**

* Partizionando il disco in uno o più gruppi di cilindri, ognuno trattato come un **disco logico**
  + Su ognuno di questi o su gruppi, si potrà fare una formattazione logica → **filesystem**
* Esiste la possibilità di raggruppare i blocchi in cluster in grado di aumentare l’efficienza

Una delle partizioni è la **root partition**, contenente il S.O.  
Altre partizioni possono ospitare altri s.o., parti di s.o o dati.  
Le partizioni possono essere:

* Mounted/agganciate ad un S.O. in esecuzione alla fase di bootstrap
* Agganciate successivamente in modo manuale/automatico

Quando si fa **mount**, c’è una fase di verifica di consistenza → dati siano corretti.

Nella **partizione di boot**, c’è un blocco di bout: blocco che va copiato direttamente in RAM per far partire il blocco di bootstrap:

* Immagine che contiene testo, schermata, diagramma, Carattere

  Descrizione generata automaticamenteCarica da disco il kernel/BootLoader che carica il pezzo di disco da kernel

**BootLoader**: invece che mettere direttamente tutto il kernel, si fa partire un pezzo di codcie che carica il kernel in memoria RAM.

C’è una parte della RAM, detta **ROM** (contiene il BIOS), che fa in modo che parta l’operazione alla partenza: il **BIOS legge e carica in RAM il MBR** dove c’è un pezzo di codice detto bootLoader, partition table → infomraizoni su come far partitre il disco e edove trovare le informazioni da caricare in ram per far partire tutto.

Un disco è formattato fisicamente e logicamente ed è anche partizionato.

**SwapSpaceManagement**

*Utilizzato per spostare interi processi (swapping) o pagine (paging) dalla DRAM alla memoria secondaria quando la DRAM non è abbastanza grande per tutti i processi.*

Pezzo di memoria secondaria (+ lenti di DRAM), può essere vista come un vettore di blocchi (e non filesystem)

* Ho bisogno semplicemente di vedere un vettore di blocchi per vedere le strutture dati attraverso una swap map che indica quali sono i blocchi attivi e quali no
  + Cosa c’è e come viene usato quel blocco
* Immagine che contiene schermata, testo, linea, diagramma

  Descrizione generata automaticamenteDunque, lo swap non richiede file system

Ci sono delle organizzazioni di dischi che coinvolgono pesantemente la rete → **Storage Attachment:**

* **Host** attached →a bordo del computer che si utilizza
  + attaccabili in vari modi, BUS o altro
* Quando si parla di **Network** Attach Storage → in rete si vedono proprio dei dischi (D,C,F)
  + Attraverso una LAN, noi siamo in aula ma vediamo un disco fornito da un dipartimento
* Su **cloud**, non si vedono dei dischi ma dei servizi → livello di itnerfaccia più vicino all’applicazione
  + C’è client, fornitore di servizi e rete ma ci sono una serie di itnerfacce e API che forniscono varie cose

RAID: organizzazione particolare di dischi che mischia la ridondanza e la molteplicità dei dischi con le strategie viste per gestione errori

* Invece di mettere un disco, metto 2/4/8 dischi → miglioro affidabilità in base a ridondanza
  + **Striping**: non veramente ridondante
    - Se hai un’informazione e questa informazione vale 100, dividila in 4 info d 25 e mettila in 4 posti diversi → diminuisce la probabilità di perdere tutto
      * Se si guasta un disco, perdi ¼ dell’informazione
  + **Mirroring**: ridondanza piena
    - Duplica l’informazione
      * Memorizzo su un disco
      * Memorizzo sulla sua copia
    - Garantisce che, quando un disco si guasta, non perdi l’informazione perché c’è una sua copia

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamenteMTTF/probabilità che fallisca il secondo

Immagine che contiene testo, schermata, design

Descrizione generata automaticamente