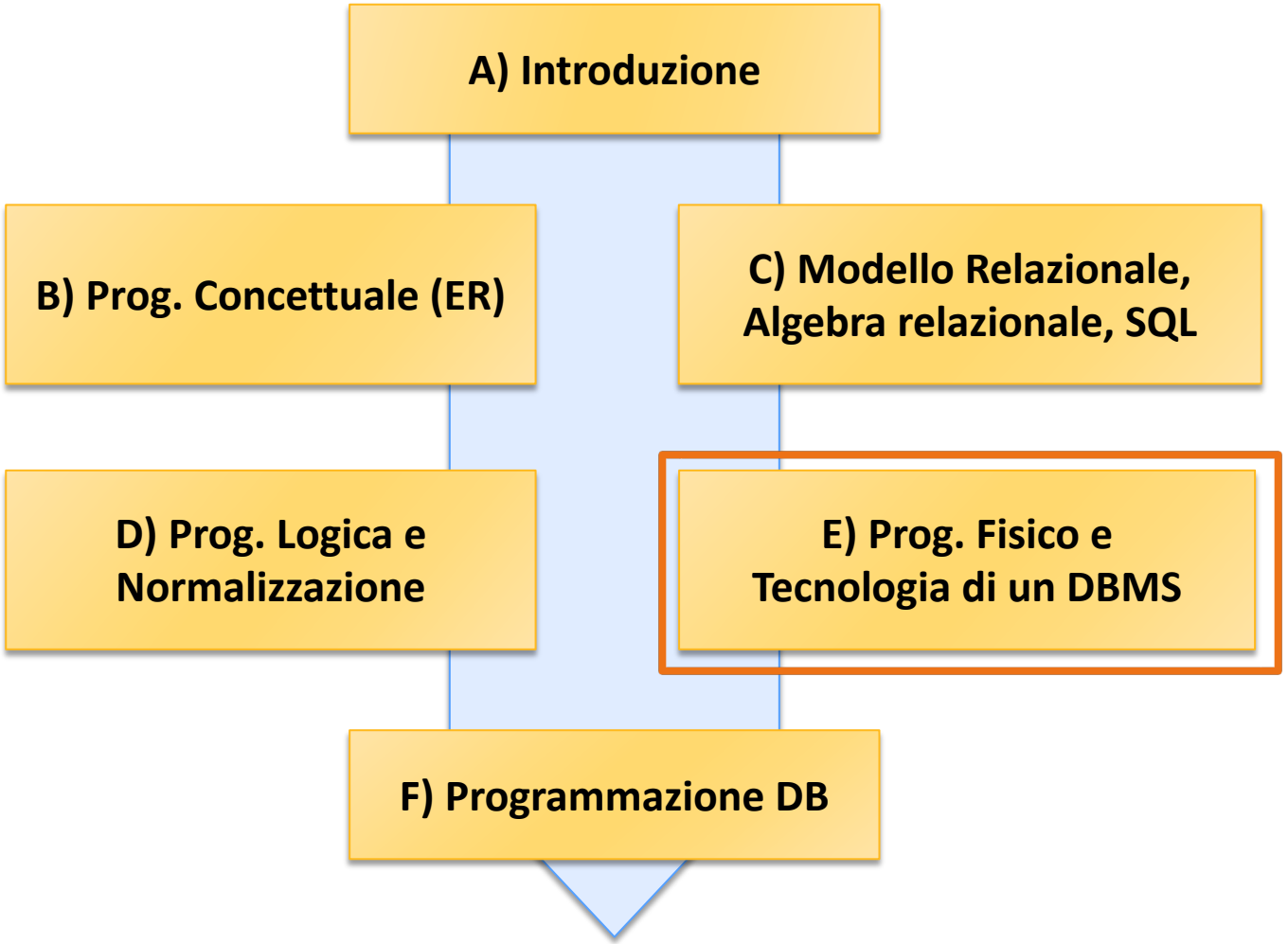
Basi di Dati

Metodi di accesso (Scansione sequenziale e ordinamento)

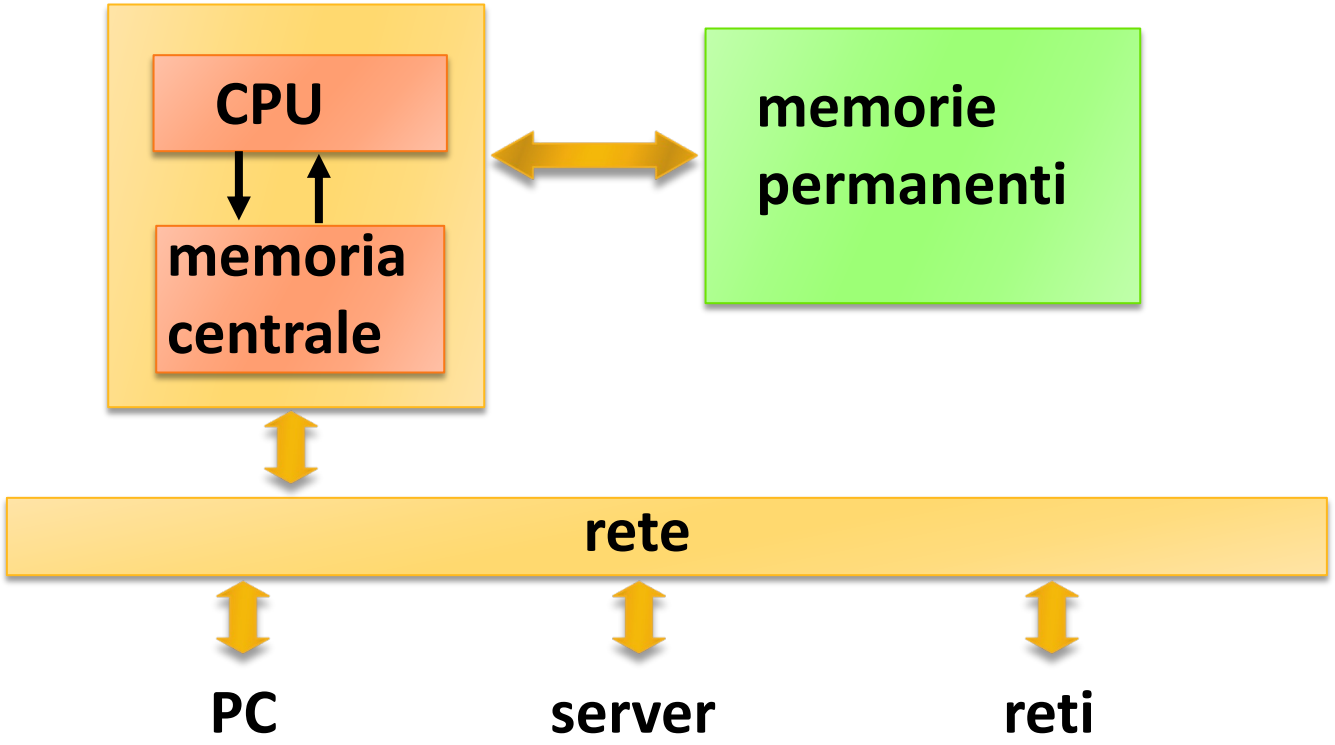
Basi di Dati – Dove ci troviamo?



Metodi di accesso

Introduzione
Data server e tempo di accesso a disco

Struttura di un data server



Qualità di un data server

- velocità della CPU
- capacità e velocità della memoria centrale (…o memoria di sevizio…)
- capacità e velocità delle memorie permanenti (…o memorie secondarie…)

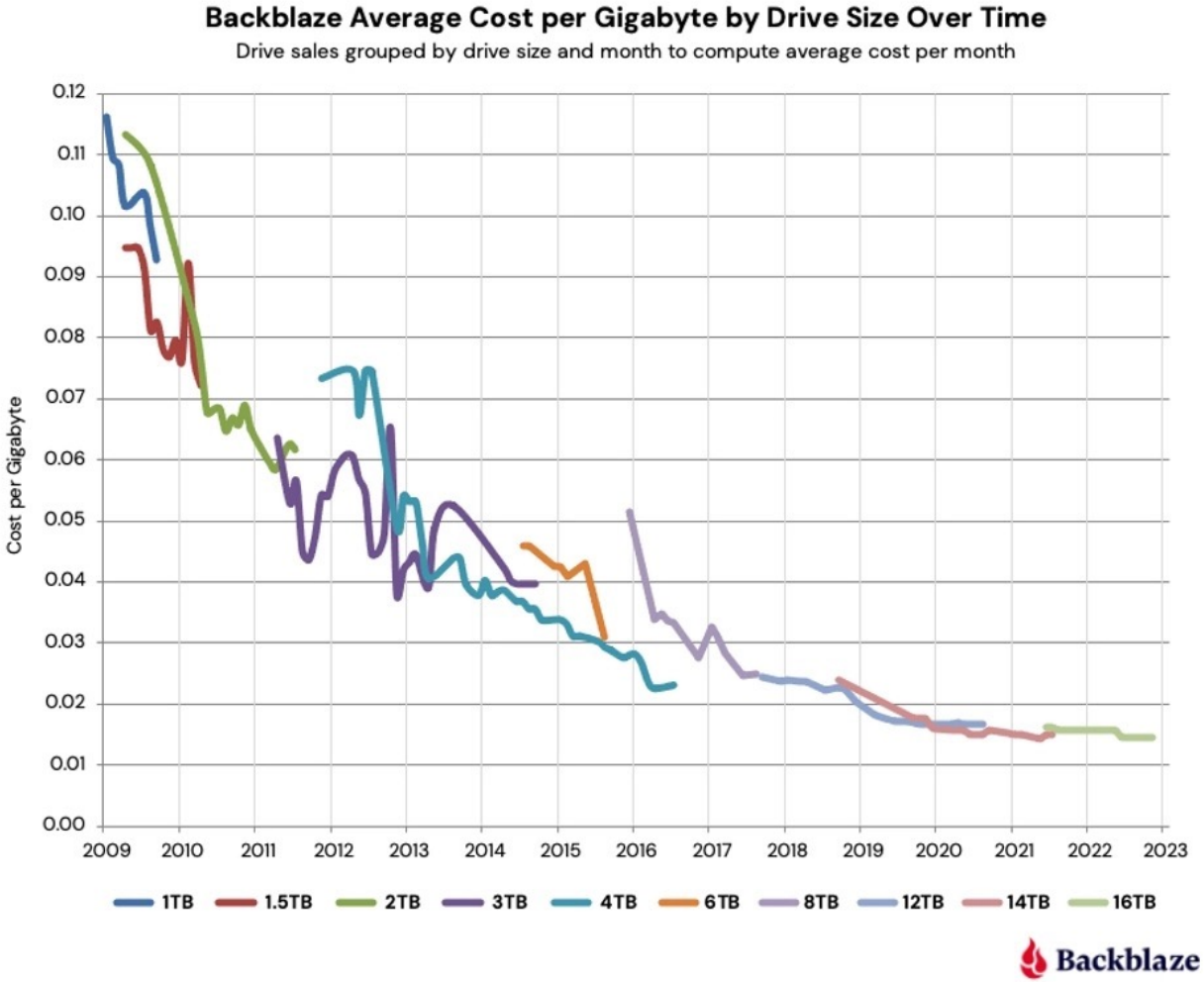
  si tende ad enfatizzare le prime due mentre la più importante è la terza!

  perché condiziona la velocità del servizio nelle applicazioni gestionali

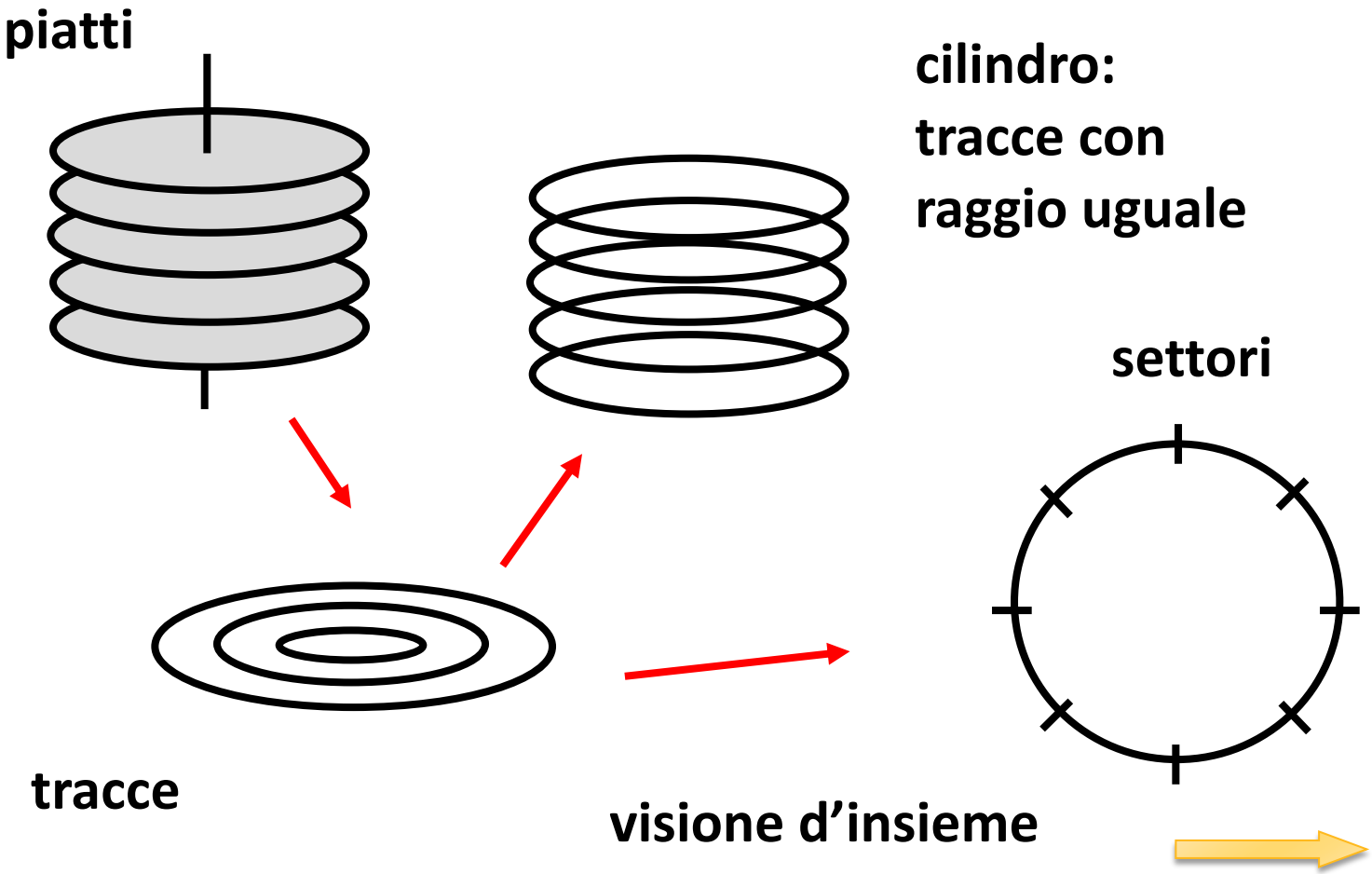
Tipi di memorie permanenti

- memorie magnetiche
  - dischi rigidi
    - disco singolo
    - RAID (dischi paralleli)
- memorie ottiche
  - CD-ROM, CD-R, CD-RW
  - DVD-ROM, DVD-R, DVD-RW
  - BD-ROM
- memorie elettroniche
  - memorie flash

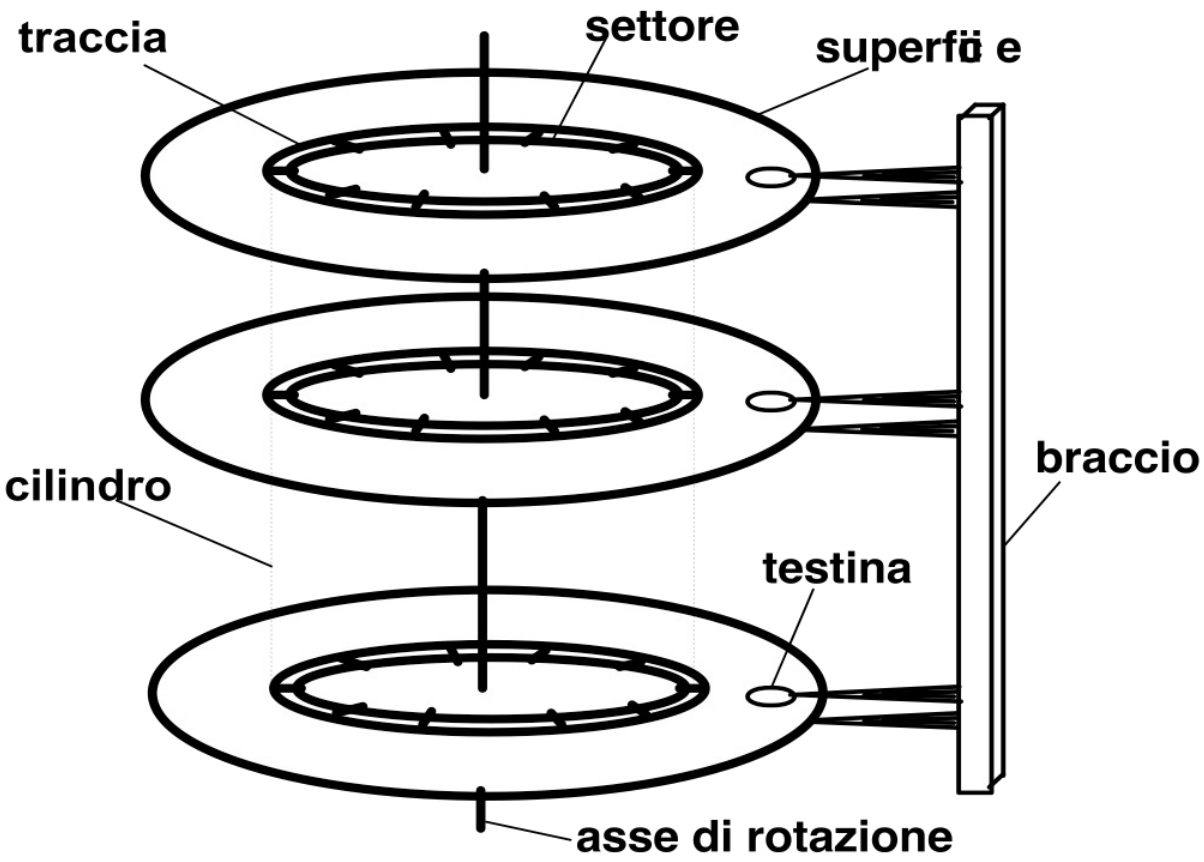
Hard Disk (HD) - costi



Memorie magnetiche – il disco

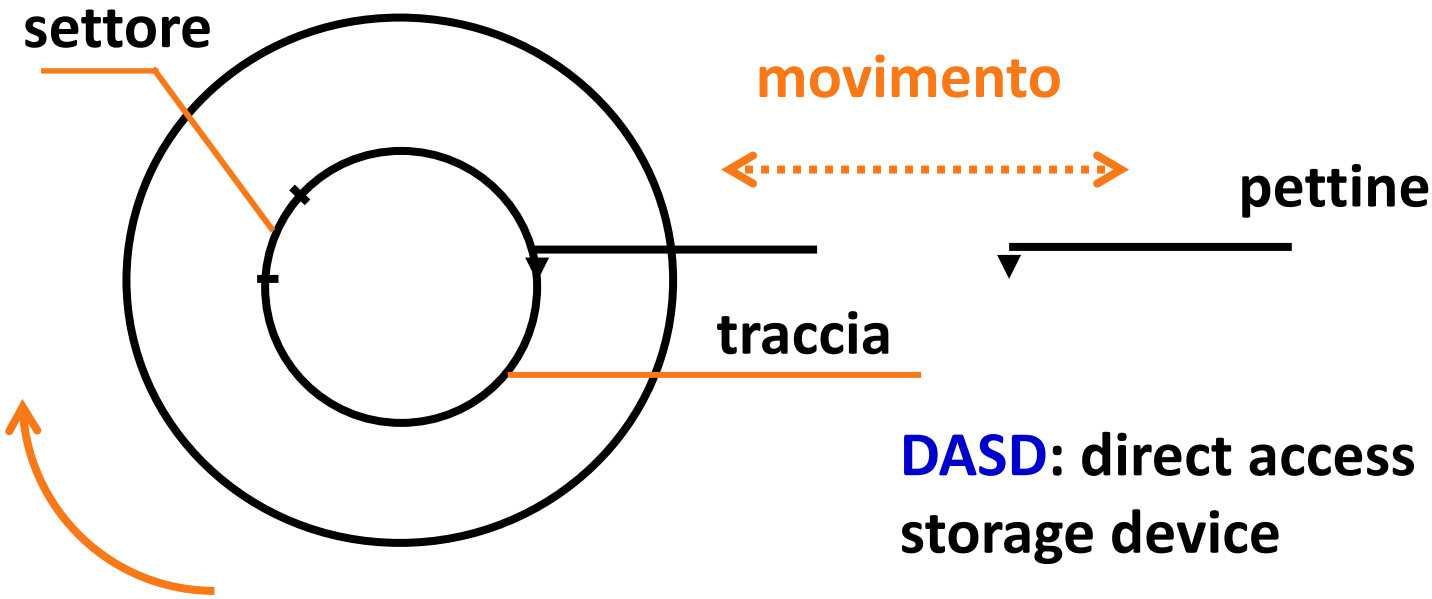


Memorie magnetiche – il disco



Meccanica del disco

Funzionamento: movimento del pettine, raggiungimento del cilindro richiesto, attivazione della testa relativa alla traccia, attesa del settore , lettura/scrittura



DASD: direct access storage device

Meccanica del disco

- Il settore è l'unità minima di trasferimento, i settori possono essere raggruppati in blocchi (pagine)
- l'indirizzo di un settore è :
  - num. cilindro, num. traccia, num. settore.
- il tempo di servizio è:
  - tempo di posizionamento (seek time) : Ts
  - tempo di latenza rotazionale : Tr
  - tempo di lettura (scrittura) : Tb
  - per la scrittura si usa anche il metodo read after write che ricontrolla dopo un giro
  - tempo impiegato dal controller (elettr.):Tc

Meccanica del disco

- il tempo di posizionamento (seek time) : Ts viene indicato dal costruttore come tempo medio di spostamento tra due possibili tracce, vengono anche indicati il Tmax ed il Tmin.
- il tempo di latenza rotazionale : Tr è mediamente la metà del tempo di rotazione
- il tempo di lettura (scrittura) : Tb dipende dalla dimensione del blocco
- il metodo read after write richiede un ulteriore 2×Tr
- il tempo impiegato dal controller (elettr.):Tc è generalmente indicato dal costruttore
- transfer rate misurato in MB/sec.

Meccanica del disco

- **Esempio:**

  Ts = 9ms, transfer rate = 30 MB/sec ,

  blocco = 4096 bytes, Tc = 1 ms.

  rotazione 7200 rpm

- **tempo di accesso:**

  Ts + Tr + Tb + Tc
  = 9 ms + 0.5 *60.000/ 7200rpm + 4KB / (30MB/sec) +1 ms
  = 9 + 4.15 + 0.1 + 1 = 14.3 ms

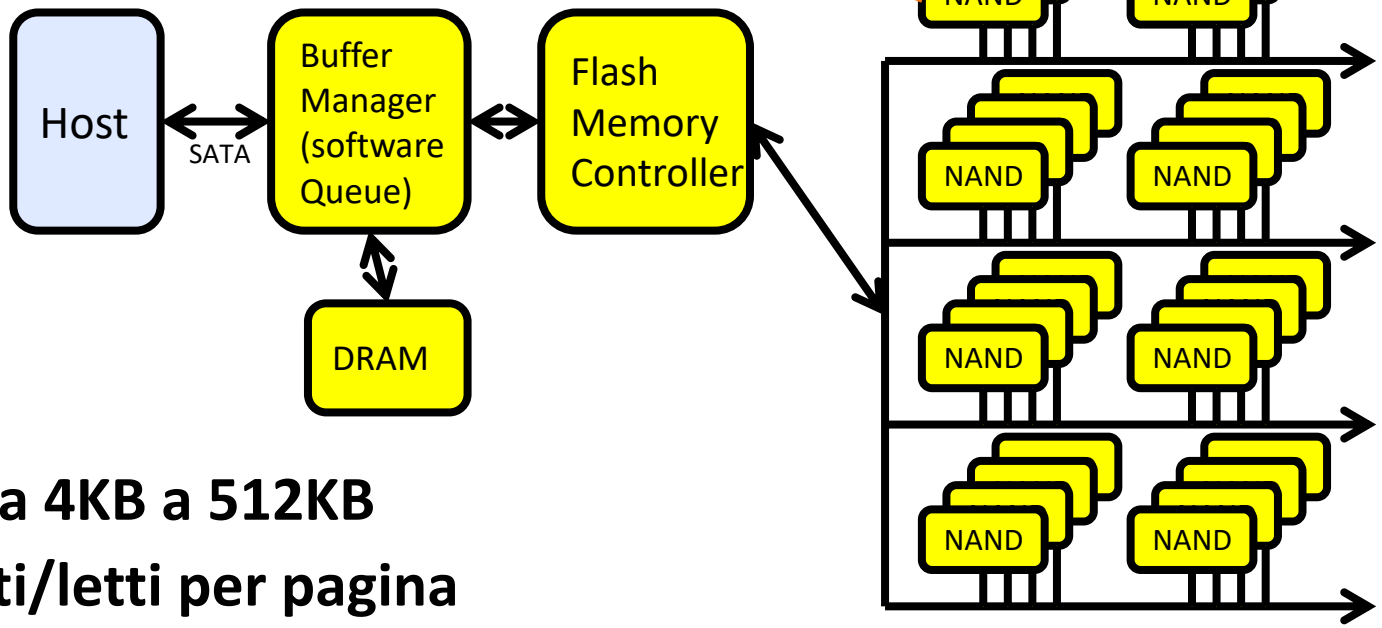
- **con read after write :**

  14.3 + 2×4.15 = 22.6 ms

l'ordine di grandezza è di molto superiore a quello delle operazioni elettroniche

Dischi a stato solido (SSD)

NAND ovvero"NOT AND": circuito integrato che utilizza le porte NAND per memorizzare i dati nelle celle di memoria(alternativa NOR)

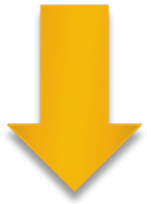


- **Pagine: da 4KB a 512KB**
- **Dati scritti/letti per pagina**
- **Scritture possibili solo su pagine vuote / cancellate**
- **Una cella di memoria si consuma dopo circa 100,000 cicli**

Dischi a stato solido (SSD) - Letture

- **Nessun tempo di seek o di latenza rotazionale**
- **Tempo di lettura (Tb): (pagina da 4KB)**
  - **~10 µs**
- **Tempo di accesso = Tq + Tc + Tb**

  (Tq: tempo di accodamento – queue)



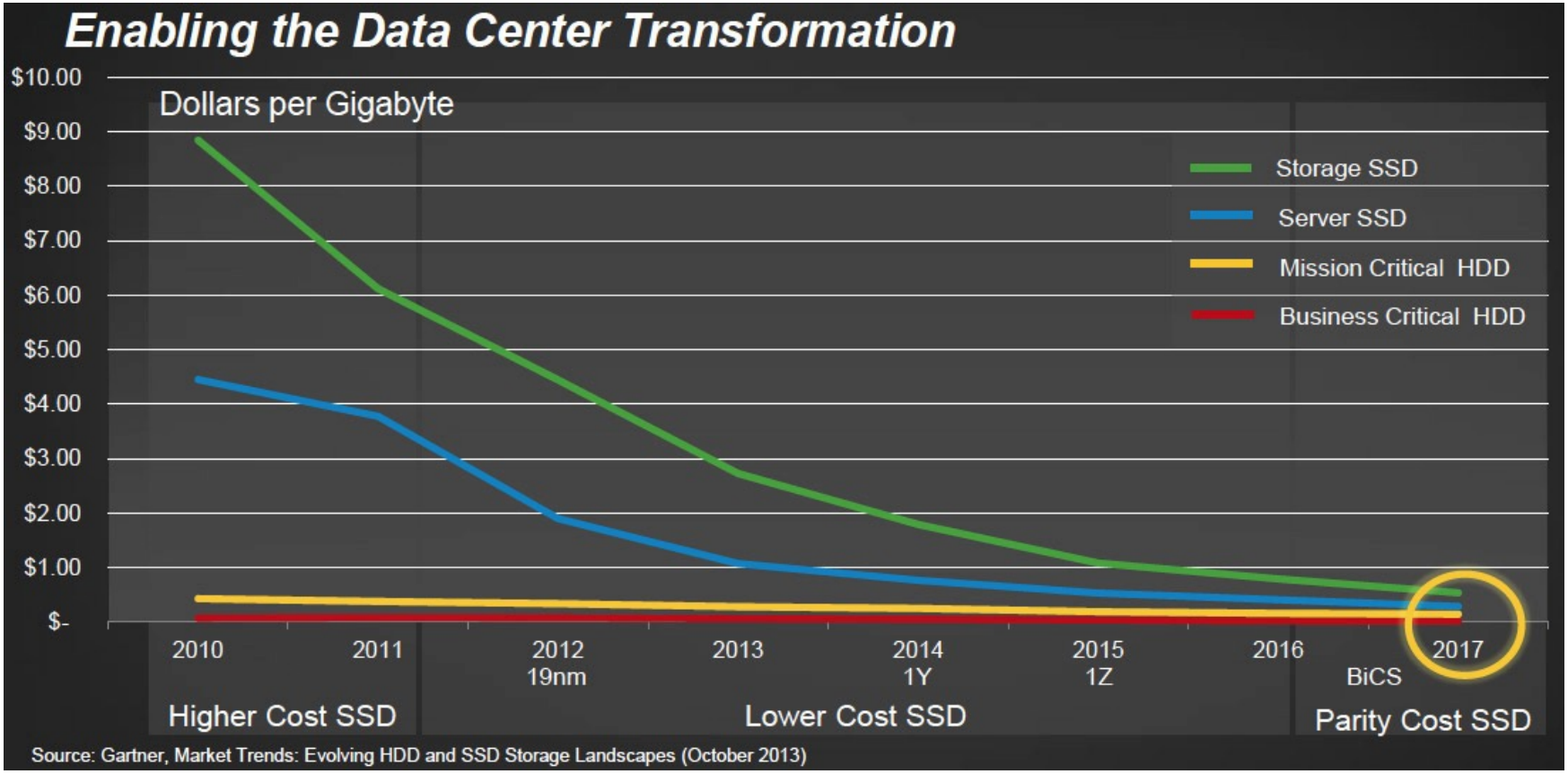
- **Tempo di lettura per una pagina da 4 KB:**
  - **~20 µs**

Dischi a stato solido (SSD) - Scritture

- **Scrivere dati è complesso! (~200µs – 1.7ms)**
- **La pagina deve essere cancellata prima di poter essere riscritta**
- **La cancellazione è possibile solo su insiemi di pagine (tipicamente da 32 a 128) (~1.5ms)**
- **Il controller riscrive e riorganizza molti più dati di quelli effettivamente aggiornati (write amplification)**
- **Regola del pollice:**

  scrittura = 10x lettura

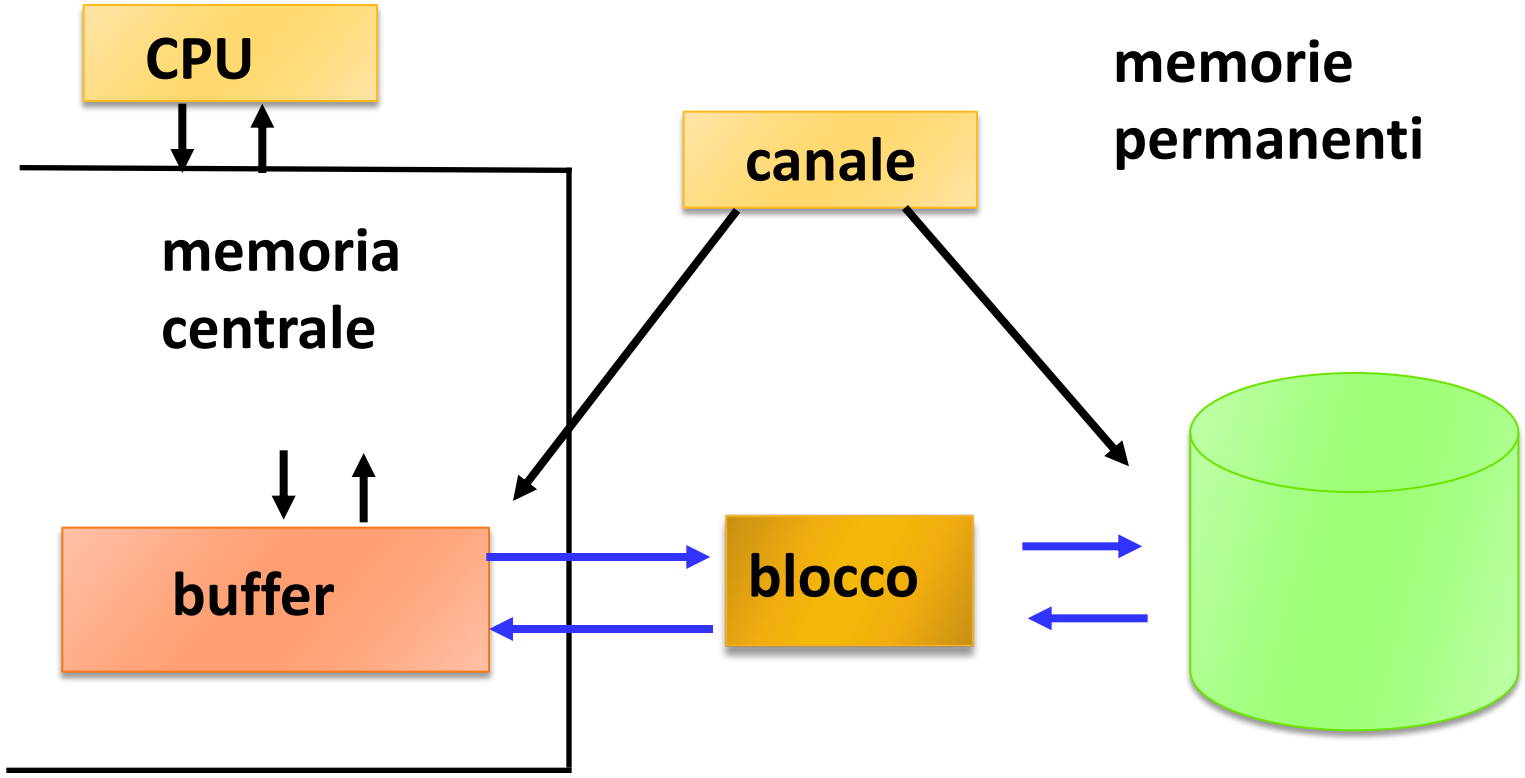
Dischi a stato solido (SSD) - costi



Metodi di accesso

Accesso sequenziale e ordinato

Struttura di un data server



blocco-min = settore

Struttura dei file

- **Anche se ci sono più livelli di indirizzamento e i file system ottimizzano la gestione del buffer e il trasferimento, le prestazioni generali per l'accesso ad un file vengono valutate come somma del numero di blocchi di dati che vengono scritti o letti da un'operazione**

Accesso sequenziale al file

Select * from cittadini where codice = ‘cf’

- **cittadini è una relazione di NT tuple contenuta in un file di NB blocchi**
- **la relazione non è in ordine di codice**
- **vengono visitati i blocchi fino a che si trova ‘cf’**

Accesso sequenziale al file

- Se il 'cf' non c'è il numero di accessi è = NB
- Se il 'cf' c'è, si assume che mediamente verranno acceduti metà dei blocchi:

$$NB/2$$

assumendo NB = 10000 e il tempo di accesso ad un blocco su disco = 20 ms (T_D), si ha:

$$T_A = NB \times T_D / 2 = 100 \text{ sec (sono molti)}$$

Ordinamento del file

- **Ordinare un file è utile non solo per la presentazione del contenuto (elenchi, listini anagrafi ecc.), ma anche per velocizzare la ricerca**
- **L'ordinamento di un file molto grande è un'operazione molto lenta che viene di regola effettuata con il metodo Sort/Merge (a M vie),**
  supponiamo di avere un file di NB blocchi che non può essere contenuto in memoria di lavoro
  il file viene ordinato in due fasi:
  1. **la fase di sort**
  2. **la fase di merge**

Ordinamento del file

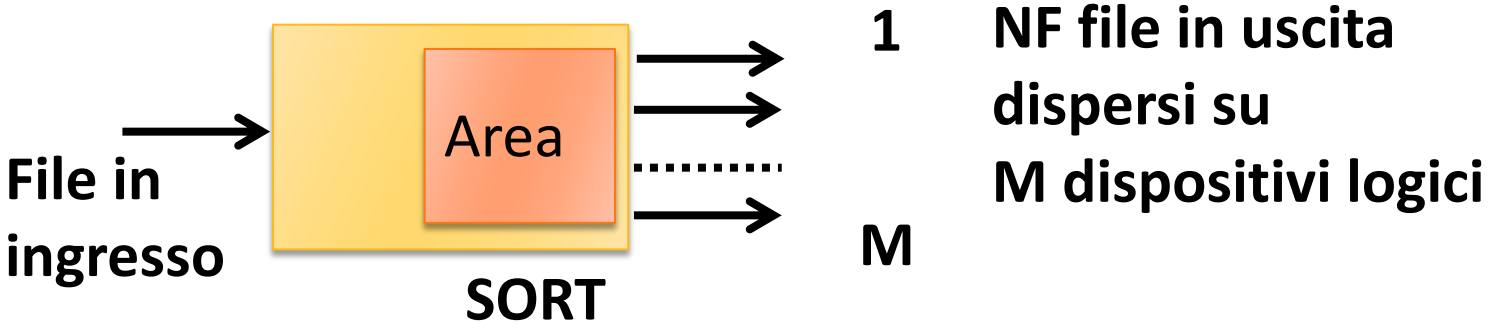
FASE DI SORT:

- **vengono portati in memoria NM blocchi (NM: disponibilità memoria centrale) per volta e le tuple ordinate con un algoritmo di sort (es. Quicksort),**
- **ogni gruppo di NM blocchi viene memorizzato in un file distinto (NF = ⌈NB/NM⌉ file)**

Ordinamento del file

FASE DI SORT:

Il primo passo è un SORT, dove i blocchi costituenti il file (NB) vengono raggruppati in NF file (a gruppi di NM) ed ordinati, e successivamente dispersi su M dispositivi logici.



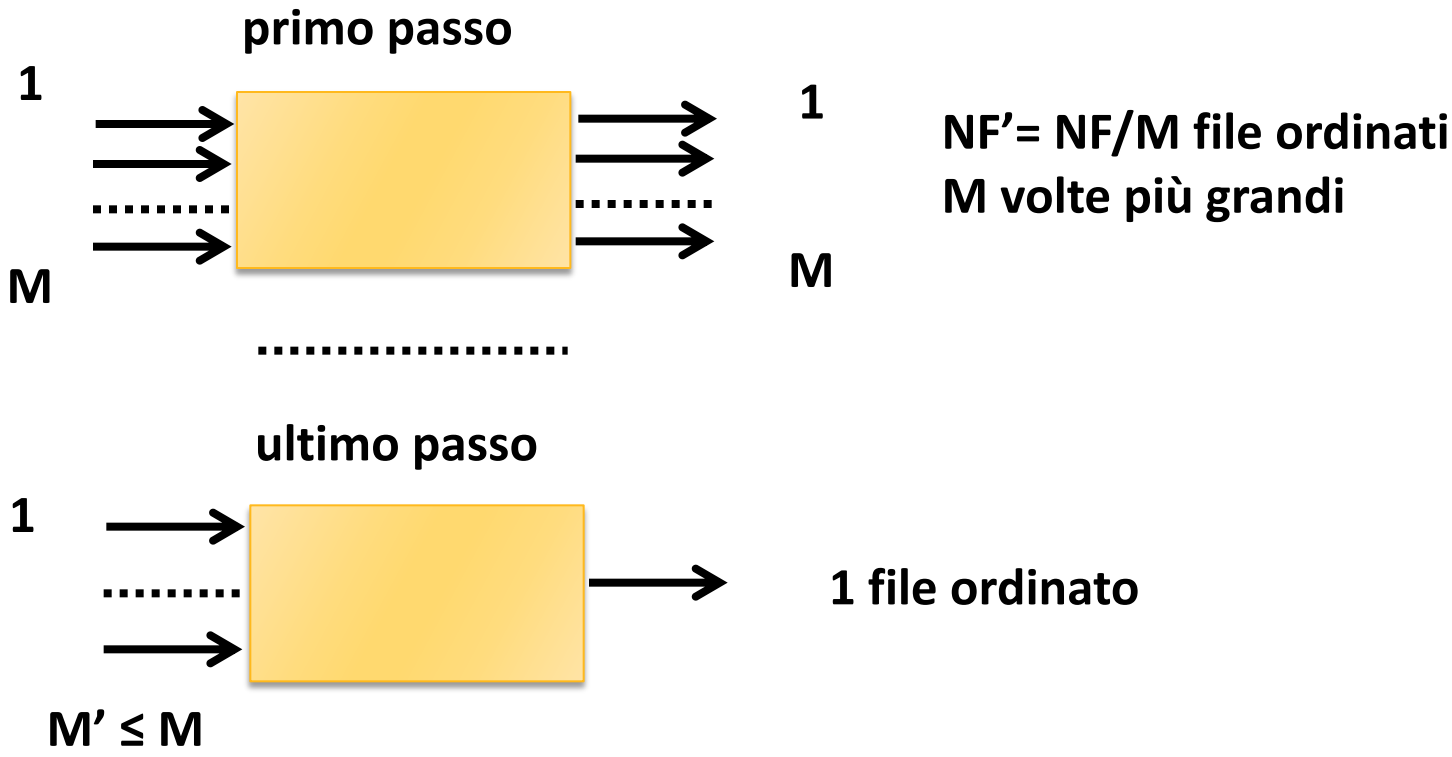
Ordinamento del file

FASE DI MERGE

- la fase è costituita da più passi:
  - passo: vengono portati in memoria gradualmente i blocchi di M (parametro del merge) file , si opera una fusione ordinata delle tuple contenute ottenendo un file M volte più grande (ordinato)
  - l'operazione si ripete fino ad esaurire gli NF file (ad ogni passo NF diminuisce)
- i passi si ripetono fondendo file sempre più grandi fino ad ottenere un unico file ordinato

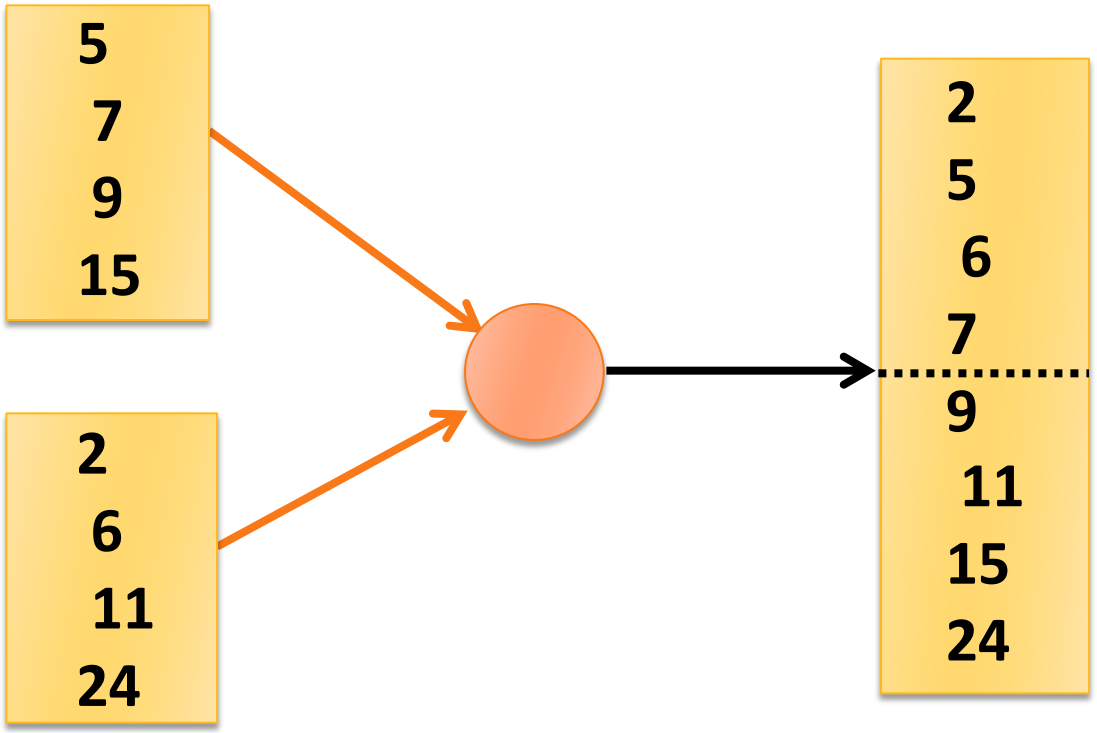
Ordinamento del file

FASE DI MERGE



Ordinamento del file

Esempio di fusione (merge) con M = 2



Ordinamento del file

- **Vediamo i passi di MERGE per un file di 81 blocchi con M = 3 e NM=3 , il sort produce 27 file di 3 blocchi:**

F4 (9 file di 3 blocchi)
F5 (9 file di 3 blocchi)
F6 (9 file di 3 blocchi)

passo 1

F1 (3 file di 9 blocchi)
F2 (3 file di 9 blocchi)
F3 (3 file di 9 blocchi)

passo 2

F4 (1 file di 27 blocchi)
F5 (1 file di 27 blocchi)
F6 (1 file di 27 blocchi)

passo 3

F1 (1 file di 81 blocchi)

Ordinamento del file

- I passi di merge PM sono stati $\text{Log}_3 (27) = 3$

  dove 27 è il numero di file che escono dal passo di sort
- In generale $PM = \text{Log}_M (NB/NM)$

  se NB/NM è una potenza di M il merge è bilanciato, altrimenti: $PM = \lceil \text{Log}_M (NB/NM) \rceil$
- Ad ogni passo di fusione la lunghezza di ogni file intermedio di uscita diventa:

  $$NM \times M,\ NM \times M^2,\ NM \times M^3, \ldots$$

  l'algoritmo termina alla k-esima fusione quando:

  $NM \times M^k \geq NB$

Ordinamento del file

- Ogni passo comporta una lettura e una scrittura per ogni blocco (NB): 2 x NB blocchi acceduti ad ogni passo
- Considerando che la fase di sort interno iniziale comporta un passo preventivo, otteniamo:

$$C_{sort} = 2 \times NB \times (PM + 1)$$

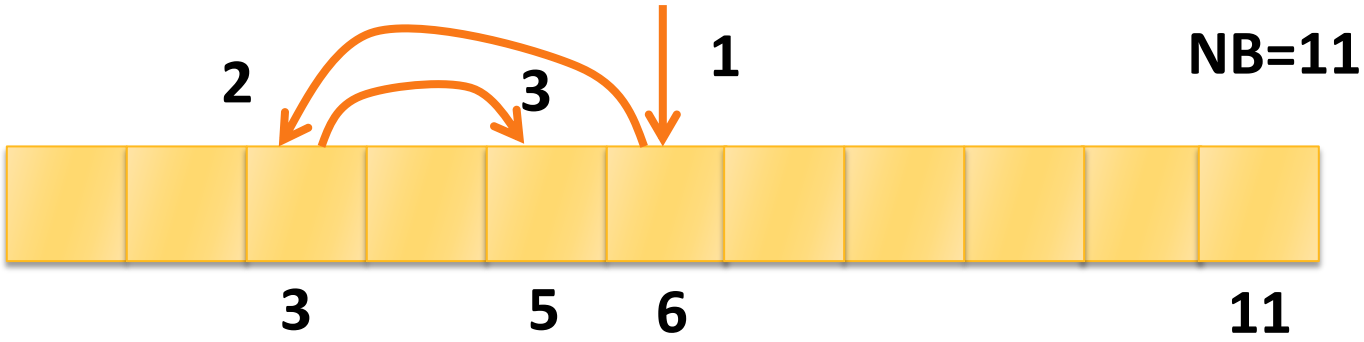
- Quindi:

$$C_{sort} = 2 \times NB \times (\lceil Log_M (NB/NM) \rceil + 1)$$

(sono comunque possibili ottimizzazioni)

Ricerca binaria

Sul file ordinato si può effettuare la ricerca binaria



la ricerca binaria ha un costo C_{bin} :

$C_{bin} = \lceil \text{Log}_2 \text{ NB} \rceil - 1$ (costo medio con successo)

$C_{bin} = \lfloor \text{Log}_2 \text{ NB} \rfloor + 1$ (caso peggiore, insuccesso)

con i dati dell'esercizio e NM = 16, M = 8 si ha

C_{sort} = 100000 e C_{bin} = 13÷15 << NB/2 = 5000

Calcolo del costo di lettura

Esempio di calcolo del costo di lettura di un record, quando i record sono *ordinati rispetto alla chiave di ricerca*.
Consideriamo come valori caratteristici del dispositivo quelli di un disco dove il costo della lettura di un blocco di dimensioni 512 byte è dato da:

$t_{read} = t_s + t_r + t_b$
con t_s = 16 ms; t_r = 8.3 ms;
$t_b = L_b/t$(data rate) = 512 byte / (3Mbyte/sec) = 0.17 ms
t_{read} = 24.47 ms
Se supponiamo di avere un archivio con
NB = 100000, L_b = 512,

Calcolo del costo di lettura

- con ricerca sequenziale abbiamo un numero di accessi medio = NB/2 e quindi un costo :
  - con blocchi non contigui
  - NL = NB/2 x t_{read} = 50000 x 24.47ms = 1223.5 s = ≃ 20 min
  - con blocchi contigui sullo stesso cilindro e trascurando il tempo di cambio di cilindro
  - NL = NB/2 x t_{read} = 50000 x 8.47ms = ≃ 7 min
- con ricerca binaria abbiamo un numero di accessi medi = $\lceil \log_2 NB \rceil - 1 = 16$ accessi:

  NL = 16 x 24.47 = ≃ 392 ms cioè meno di un secondo!

Metodi di organizzazione

- In un file può esistere un solo ordinamento (su una sola colonna o su un gruppo)
- l'ordinamento è costoso da ottenere e da mantenere(a seguito di inserimenti di nuove tuple)
- l'ordinamento su un attributo favorisce solo alcune query e non ne favorisce altre
- l'ordinamento è un metodo di organizzazione 'primario', vedremo altre organizzazioni che possono essere utilizzate sia come primarie che come secondarie

Metodi di organizzazione

I tipi di organizzazione sono sostanzialmente due:

- **le organizzazioni ad INDICE che utilizzano file di supporto che riportano per ogni valore della chiave l'indirizzo nel file di dove è localizzata la tupla**
- **le organizzazioni HASH (indirizzo calcolato) che per allocare una tupla in un file sottopongono la chiave ad una trasformazione con una funzione detta funzione hash che trasforma il valore della chiave in un valore numerico che corrisponde all'indirizzo nel file**