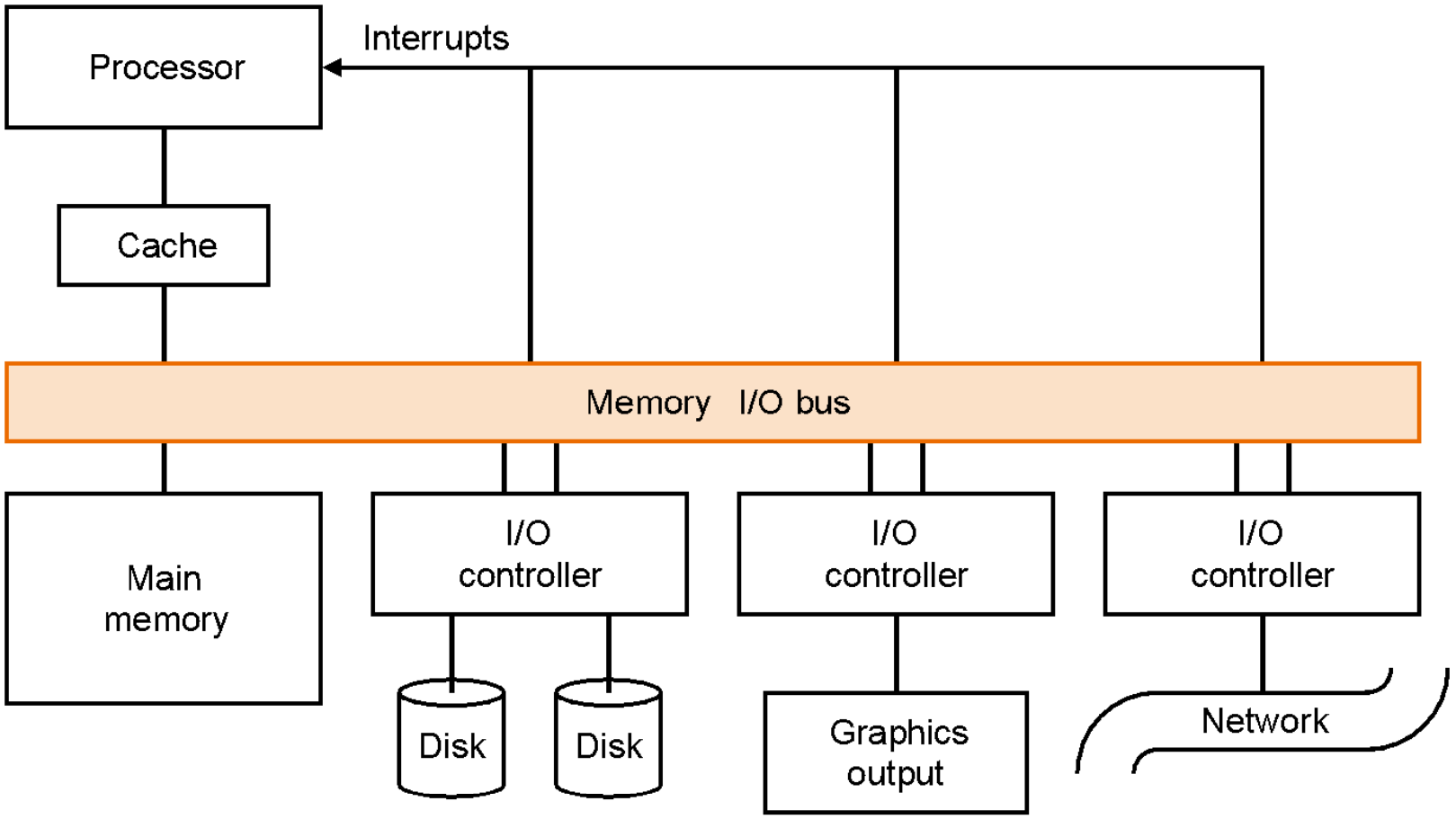
Dispositivi di I/O

Lucidi fatti in collaborazione con l’Ing. Valeria Cardellini

Possibile organizzazione di un calcolatore



Dispositivi di I/O

- Un dispositivo di I/O è costituito da due componenti:
  - Il dispositivo fisico *effettivo* (disco, stampante, mouse, video, …)
  - Il *device controller* (o interfaccia) che gestisce tutte le operazioni che il dispositivo è in grado di svolgere
    - Permette di uniformare la connessione tra il dispositivo ed il resto del sistema
- Il device controller è collegato attraverso il ***bus** di sistema* con CPU e memoria principale
- Il device controller è un sottosistema specializzato nel controllo dei dispositivi di I/O
  - Fornisce eventuali registri dove possono essere appoggiati i dati del trasferimento ed i comandi al dispositivo

Eterogeneità dei dispositivi di I/O

- Hanno caratteristiche molto diverse tra loro, classificabili in base a 3 dimensioni
  - **Comportamento**
    - Input/output o memorizzazione di dati
  - **Controparte (partner)**
    - Uomo o macchina
  - **Tasso di trasferimento dati**
    - Dal dispositivo in memoria e viceversa

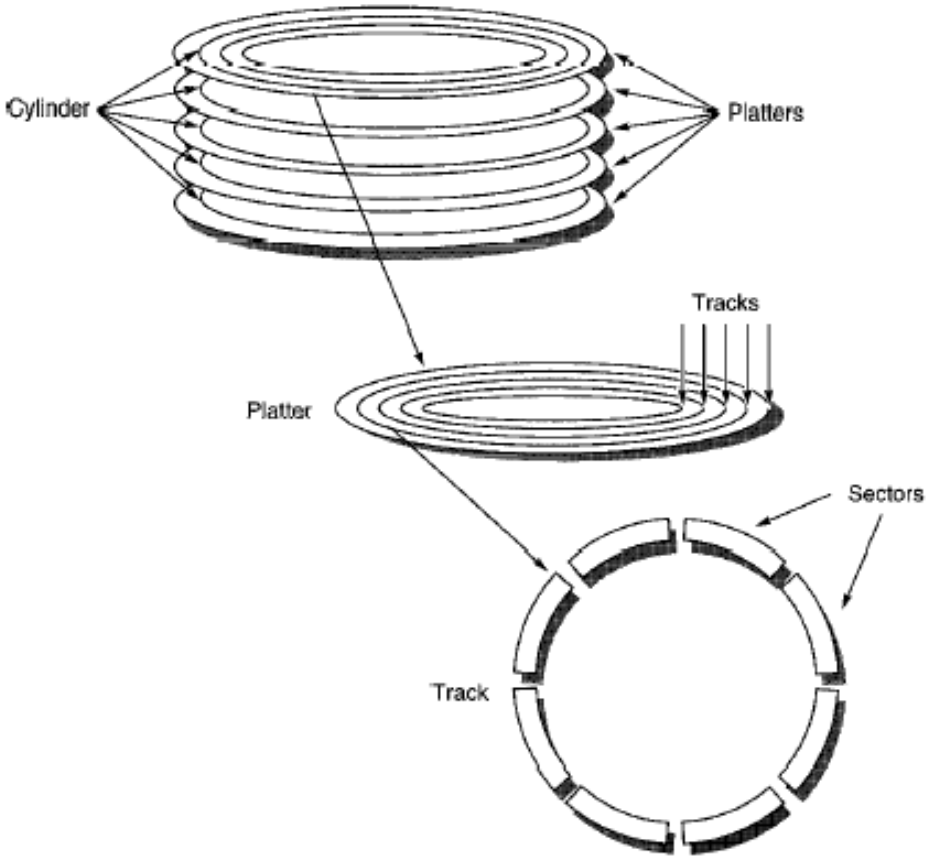
Dispositivo	Funzione	Partner	Velocità (Mb/sec)
Tastiera	input	umano	0,0001
Mouse	input	umano	0,0038
Stampante laser	output	umano	3,2
Network/ wireless LAN	input o output	macchina	11-54
Network/LAN	input o output	macchina	100-1000
Disco ottico	memoriz-zazione	macchina	80
Disco magnetico	memoriz-zazione	macchina	240-2560
Scheda grafica	output	umano	800-8000

Disco magnetico

- Costituito da un insieme di *piatti* rotanti (da 1 a 15)
  - Piatti rivestiti di una superficie magnetica
- Esiste una testina (bobina) per ogni faccia del piatto
  - Generalmente piatti a doppia faccia
- Le testine di facce diverse sono collegate tra di loro e si muovono contemporaneamente in modo solidale
- Velocità di rotazione costante (ad es. 10000 RPM)
- La superficie del disco è suddivisa in anelli concentrici (*tracce*)
- Registrazione seriale su tracce concentriche
  - 1000-5000 tracce
  - Tracce adiacenti separate da spazi

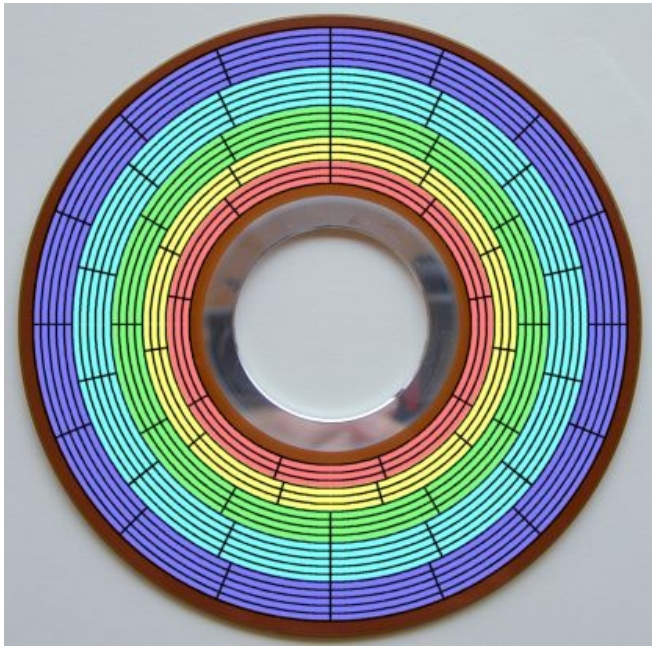
Disco magnetico (2)

- Ciascuna traccia è divisa in *settori*
  - Settore: la più piccola unità che può essere trasferita (scritta o letta)
  - Centinaia di settori per traccia, generalmente di lunghezza fissa (es., 512 B)
  - Il settore contiene un ID del settore, i dati e un codice di correzione di errore: la *capacità formattata* scende del 15%
- Tracce sovrapposte su piatti diversi formano un *cilindro*



Organizzazione dei dati sul disco

- Nei dischi più vecchi
  - Ogni traccia conteneva lo stesso numero di settori
  - Le tracce esterne (più lunghe) memorizzavano informazioni con densità minore
- Nei dischi recenti
  - Per aumentare le prestazioni, si utilizzano maggiormente le tracce esterne: *zoned bit recording* (o multiple zone recording)
  - Tracce raggruppate in *zone* sulla base della loro distanza dal centro
    - Una zona contiene lo stesso numero di settori per traccia
  - Più settori per traccia nelle zone esterne rispetto a quelle interne
  - Densità di registrazione (quasi) costante



Lettura/scrittura di un disco

- Processo composto da 3 fasi:
  - Posizionamento della testina sul cilindro desiderato (*tempo di seek*)
    - Da 3 a 14 ms (può diminuire del 75% se si usano delle ottimizzazioni)
    - Dischi di diametro piccolo permettono di ridurre il tempo di posizionamento
  - Attesa che il settore desiderato ruoti sotto la testina di lettura/scrittura (*tempo di rotazione*)
    - In media è il tempo per ½ rotazione
    - Tempo di rotazione medio = 0.5/numero di giri al secondo
      Es.: 7200 RPM → Tempo di rotazione medio = 0.5/(7200/60) = 4.2 ms
  - Operazione di lettura o scrittura di un settore (*tempo di trasferimento*)
    - Da 30 a 80 MB/sec (fino a 320 MB/sec se il controllore del disco ha una cache built-in)
- In più: tempo per le operazioni del disk controller (*tempo per il controller*)

Prestazioni dei dischi magnetici

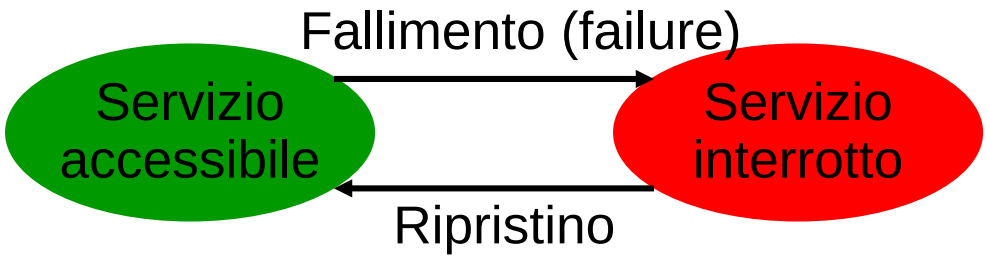
- Calcolo del tempo medio necessario a leggere o scrivere un settore di 512 byte sapendo che:
  - Il disco ruota a 10000 RPM
  - Il tempo medio di seek è 6 ms
  - Il transfer rate è di 50 MB/sec
  - L'overhead del controller è di 0.2 ms

  Tempo di seek + tempo medio di rotazione + tempo medio di trasferimento + overhead del controller =

  = 6 ms + (0.5/(10000/60))·1000 ms + 0.5 KB/(50 MB/sec) + 0.2 ms = (6.0 + 3.0 + 0.01 + 0.2) ms =

  = 9.2 ms

Affidabilità e disponibilità



- Fallimento (*failure*): il comportamento del servizio non è conforme alle specifiche
  - Il fallimento è causato da un errore (*error*) – i.e. porzione di stato scorretto
  - La causa di un errore è un guasto (*fault*)
    - tipo: hw, sw o operativo
    - durata: transiente, permanente, intermittente
    - visibilità esterna: fail-stop, bizantino

Affidabilità e disponibilità (2)

- **Affidabilità - reliability:** probabilità che il sistema funzioni secondo le specifiche di progetto continuamente dall'istante in cui viene attivato all'istante di "osservazione" – R(t)
- **Disponibilità (availability) all'istante t:** probabilità che il sistema funzioni secondo le specifiche di progetto quando gli si chiede un servizio – A(t)
- **Disponibilità (a regime permanente) - availability:** disponibilità quando t -> infinito

Affidabilità e disponibilità (3)

- Tempo medio di fallimento (mean time to failure o **MTTF**)
  - Tempo medio che intercorre tra l'istante in cui il servizio è ripristinato ed il fallimento successivo
  - E' un indice dell'affidabilità (*reliability*) del servizio
- Tempo medio di riparazione (mean time to repair o **MTTR**):
  - Tempo medio necessario per ripristinare il servizio

Affidabilità e disponibilità (4)

- Tempo medio tra due fallimenti (mean time between failures o **MTBF**)
  - Tempo medio tra due fallimenti consecutivi
    MTBF = MTTF + MTTR

- Disponibilità (*availability*) a regime permanente:

$$\text{Availability} = \frac{\text{MTTF}}{\text{MTTF} + \text{MTTR}}$$

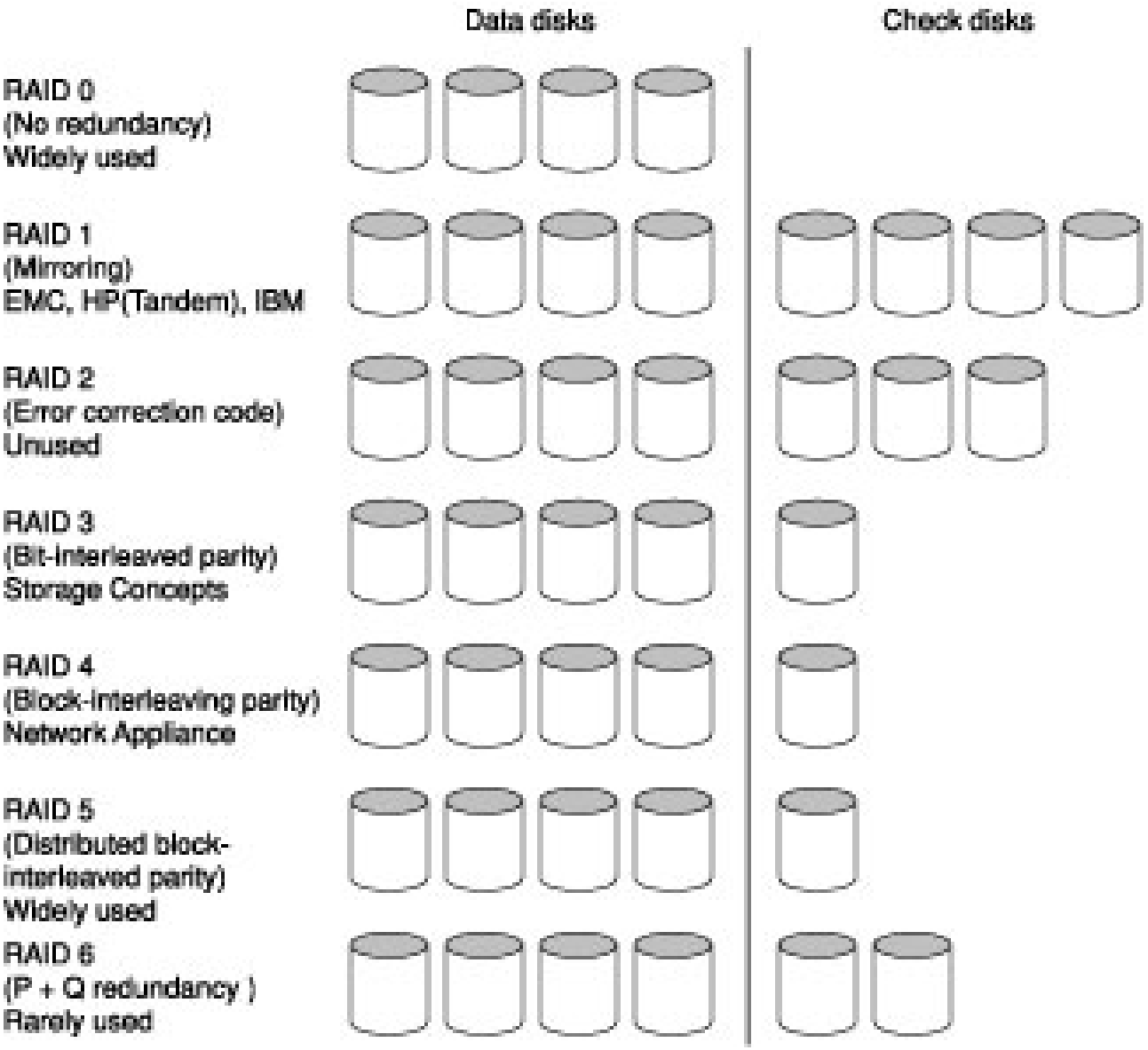
- Per aumentare il MTTF
  - Evitare i guasti (p.e. uso di componenti più costosi)
  - Tollerare i guasti
    - *Tolleranza ai guasti*: capacità del servizio di non subire fallimenti anche in presenza di guasti
    - Occorre introdurre *ridondanze (spaziale, temporale)*
  - Predire i guasti (evitare di usare sistemi con componenti prossimi al guasto) – manutenzione preventiva

RAID

- Le prestazioni dei dischi crescono più lentamente di quelle dei processori
  - Accesso ai dischi migliorato di 5/10 volte in 20 anni
- Idea di Patterson et al. nel 1987: usare in parallelo più dischi per aumentare le prestazioni dei dischi
- *Problema*: un array di dischi (senza ridondanza dei dati) è inaffidabile!

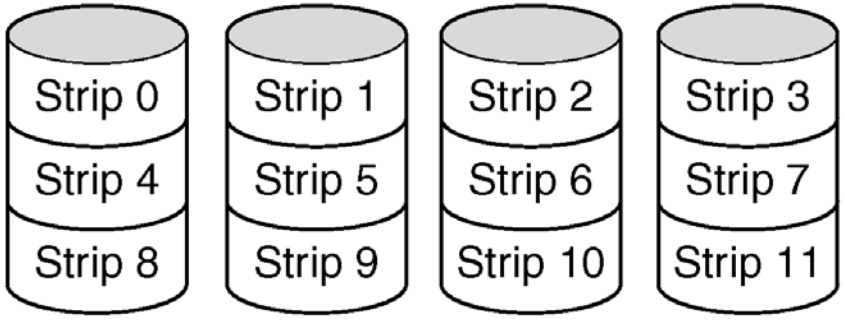
  Affidabilità di un array da N dischi = Affidabilità di 1 disco/N
- *Soluzione*: definire un'organizzazione dei dati memorizzati sui dischi in modo da ottenere un'elevata affidabilità (*tolleranza ai guasti*) replicando i dati sui vari dischi dell'array
- RAID: **Redundant Array of Inexpensive (Independent) Disks**
  - Insieme di dischi a basso costo ma coordinati in azioni comuni per ottenere diversi livelli di tolleranza ai guasti

Livelli RAID



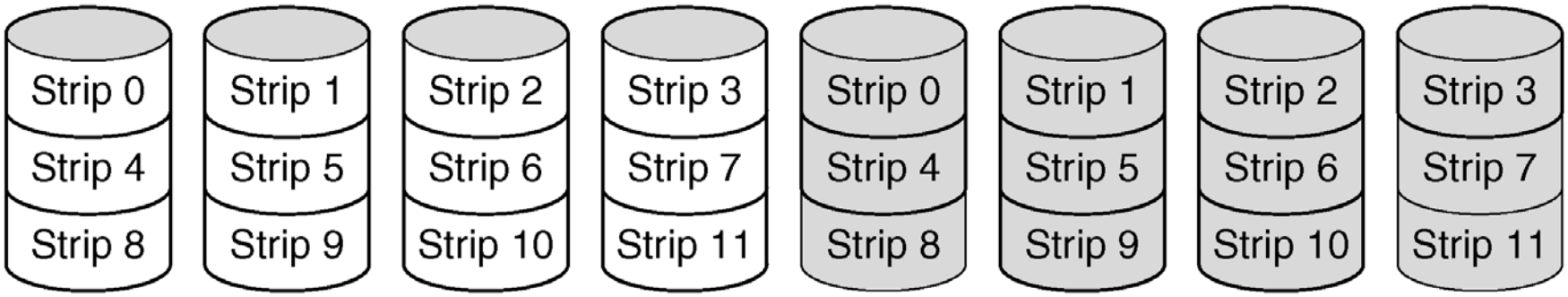
RAID 0

- Nessuna ridondanza dei dati
- Solo *striping* dei dati
  - Striping: allocazione di blocchi logicamente sequenziali (memorizzanti p.e. lo stesso file, che quindi è suddiviso in più blocchi) su dischi diversi per aumentare le prestazioni rispetto a quelle di un singolo disco
  - Lettura e scrittura in parallelo di *stripe* (strisce) su dischi diversi
- Non è un vero RAID perché non c'è nessuna ridondanza
- E' la migliore soluzione in scrittura, perchè non ci sono overhead per la gestione della ridondanza, ma non in lettura



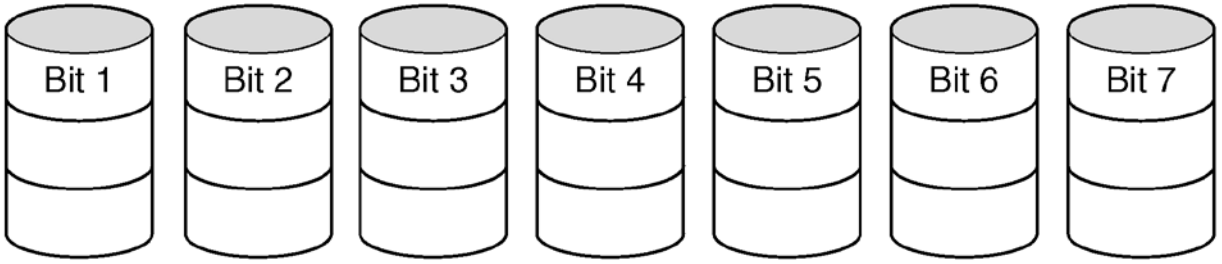
RAID 1

- Mirroring (o shadowing)
- Ciascun disco è completamente replicato su un disco ridondante (mirror), avendo così sempre una copia
  - Usa il doppio dei dischi rispetto a RAID 0
- Ottime prestazioni in lettura
  - Molte possibilità di migliorare le prestazioni (es.: leggere dal disco con il minimo tempo di seek, leggere due file contemporanemanete su dischi “gemelli”)
- Una scrittura logica richiede due scritture fisiche
- E’ la soluzione RAID più costosa



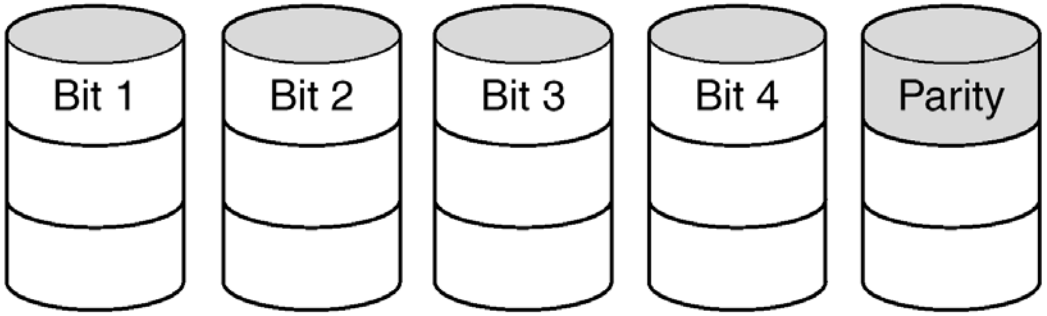
RAID 2

- Rivelazione e correzione degli errori (*codice di Hamming*)
- *Striping a livello di parola o di byte* (in RAID 0 e 1 strip di settori)
  - Es. in figura: 4 bit (*nibble*) più 3 bit (codice di Hamming a 7 bit)
- Svantaggio: rotazione dei dischi *sincronizzata*
- Resiste a guasti semplici
- Ad ogni scrittura bisogna aggiornare i dischi di "parità"
  anche per la modifica di un singolo bit di informazione
- Forte *overhead* per pochi dischi (in figura +75%), ha senso con molti dischi, ad esempio:
  - Parola da 32 bit+(6+1) bit di parità $\Rightarrow$ 39 dischi
  - Overhead del 22% (=7/32)
- In disuso



RAID 3

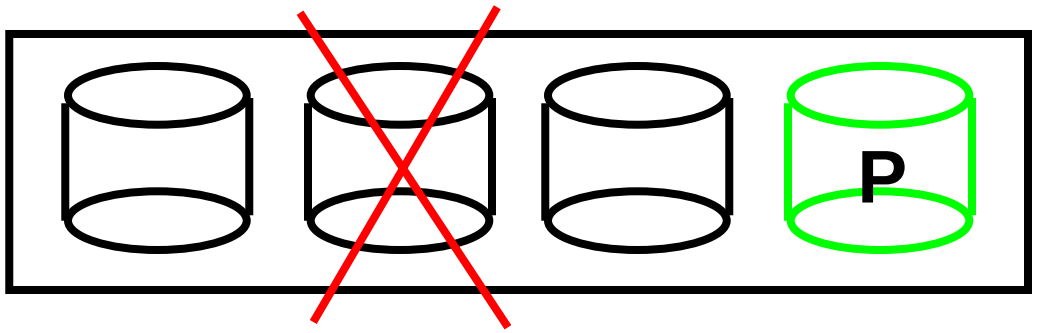
- Un bit di parità orizzontale ed uno verticale
- Resiste ad un guasto (transiente o permanente) alla volta
- Overhead abbastanza contenuto
- Solo un'operazione su disco per volta
  - Ciascuna operazione coinvolge tutti i dischi
- Soluzione diffusa per applicazioni che operano su grandi quantità di dati in lettura (come nei video games o nelle fruizioni multimediali), disco di parità collo di bottiglia in caso di scrittura



RAID 3: esempio

Record logico

10010011
11001101
10010011
. . .



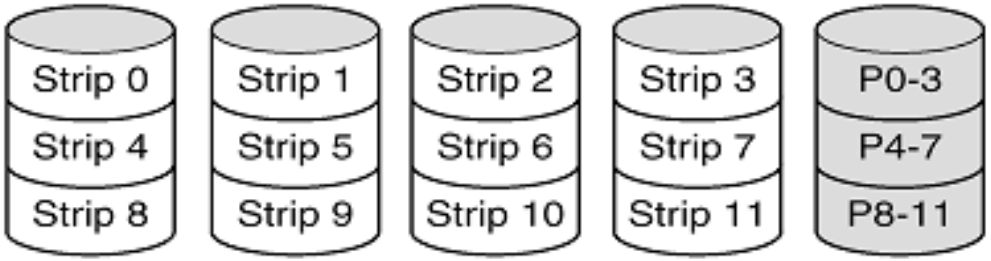
Record fisici

1	1	1	1
0	1	0	1
0	0	0	0
1	0	1	0
0	1	0	1
0	1	0	1
1	0	1	0
1	1	1	1
0	**1**	**0**	**1**

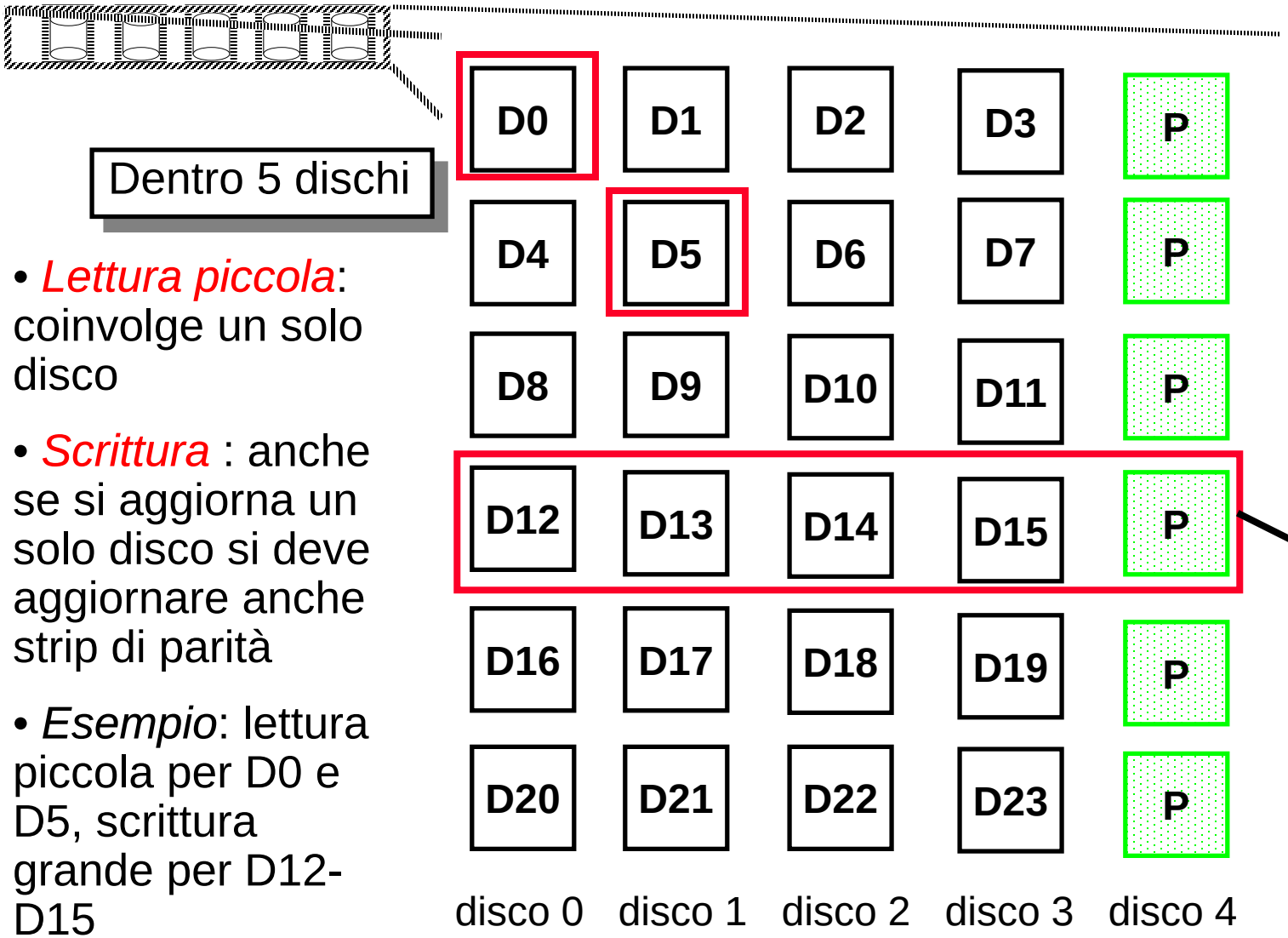
- P contiene il bit di parità dei bit (strip) memorizzati negli altri dischi
- Se un disco fallisce (in modo transiente o permanente), utilizzando P, i bit di parità verticale e i bit degli altri dischi si recupera l'informazione mancante
- Overhead accettabile (un terzo nell'esempio, in genere 1/(n-1) se n sono i dischi utilizzati)

RAID 4

- Evoluzione di Raid 3 con striping a blocchi (come RAID 0)
  - la stripe nell'ultimo disco contiene i bit di parità dell'insieme di bit omologhi di tutte le altre stripe
- No rotazione sincronizzata (come in RAID 2 e 3)
- Resiste a guasti singoli (transienti e permanenti)
- Consente letture indipendenti sui diversi dischi
  - Se si legge una quantità di dati contenuta in una sola strip
- Il disco di parità è il collo di bottiglia



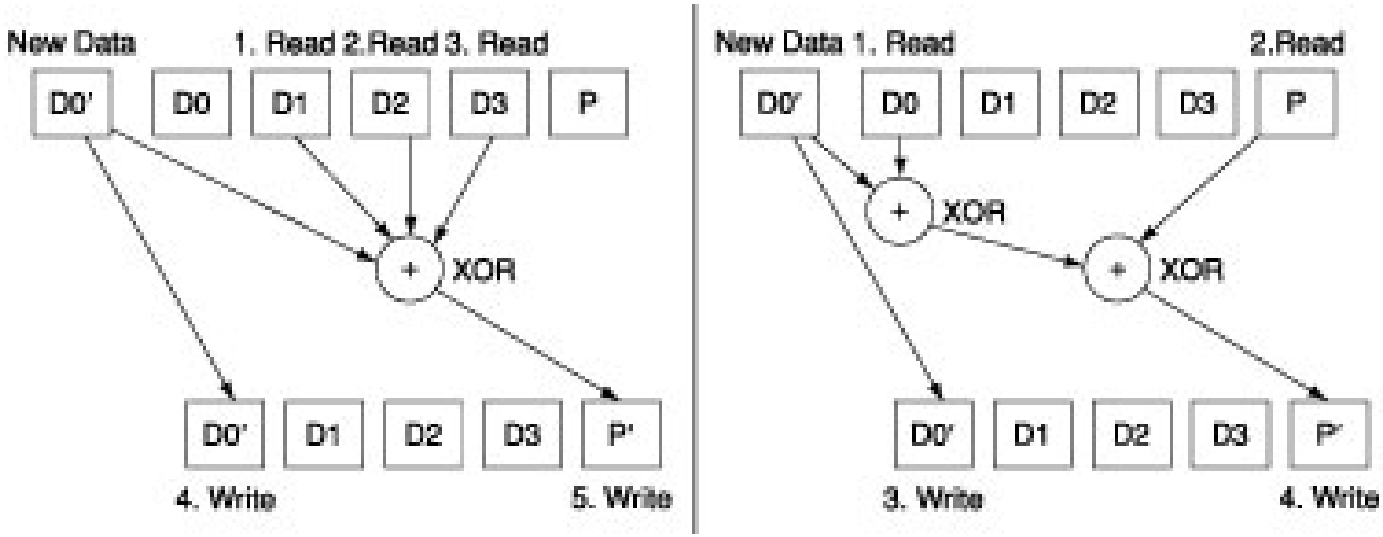
RAID 4: lettura e scrittura



- *Lettura piccola*: coinvolge un solo disco
- *Scrittura* : anche se si aggiorna un solo disco si deve aggiornare anche strip di parità
- *Esempio*: lettura piccola per D0 e D5, scrittura grande per D12-D15

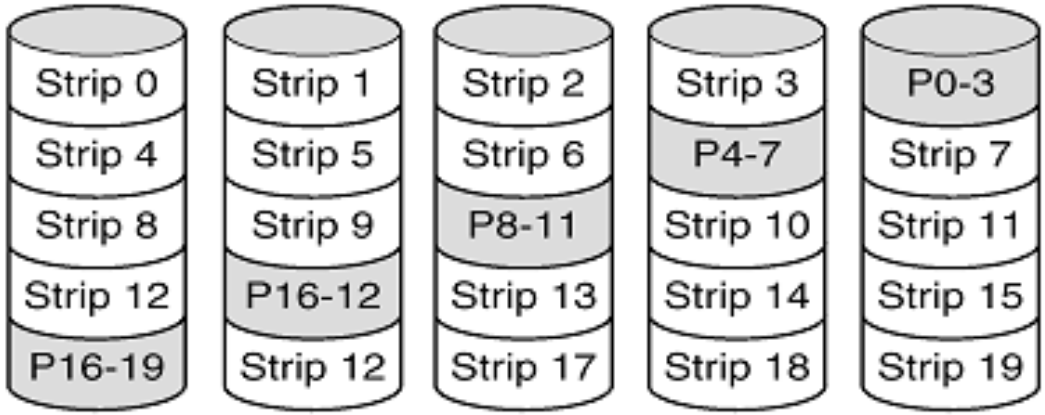
Scrittura in RAID 3 e RAID 4

- Esempio di scrittura piccola in RAID 4:
  - *Opzione 1*: si leggono i dati sugli altri dischi, si calcola la nuova parità P' e la si scrive sul disco di parità (come per RAID 3)
    - Es.: 1 scrittura logica = 3 letture fisiche + 2 scritture fisiche
  - *Opzione 2*: poiché il disco di parità ha la vecchia parità, si confronta il vecchio dato D0 con il nuovo D0', si aggiunge la differenza a P, e si scrive P' sul disco di parità
    - Es.: 1 scrittura logica = 2 letture fisiche + 2 scritture fisiche



RAID 5

- Blocchi di parità distribuita
- Le stripe di parità sono distribuite su più dischi in modalità round-robin (circolare)
- Si evita il collo di bottiglia del disco di parità in RAID 4
- La scrittura piccola è gestita come in RAID 4



RAID 5: scrittura

- Sono possibili scritture indipendenti in virtù della parità interallacciata
- Esempio: la scrittura di D0 e D5 usa i dischi 0, 1, 3, 4

disco 0	disco 1	disco 2	disco 3	disco 4
D0	D1	D2	D3	P
D4	D5	D6	P	D7
D8	D9	P	D10	D11
D12	P	D13	D14	D15
P	D16	D17	D18	D19
D20	D21	D22	D23	P

RAID 6

- Ridondanza P+Q (si aumenta la distanza di Hamming)
- Anziché la parità, si usa uno schema che consente di ripristinare anche un secondo guasto
  - la singola parità consente di recuperare un solo guasto
- Overhead di memorizzazione doppio rispetto a RAID 5