

## 5 – Memoria secondaria ottimizzazione delle prestazioni nei dischi

### Sommario

- Introduzione
- Evoluzione dei dispositivi di memoria secondaria
- Caratteristiche dei dischi a testina mobile
  - Arrays Ridondanti di dischi Indipendenti (RAID)
- Strategie di scheduling del disco
  - First-Come-First-Served (FCFS)
  - Shortest-Seek-Time-First (SSTF)
  - SCAN e varianti: C-SCAN, FSCAN e N-Step SCAN
  - LOOK e C-LOOK
- Ottimizzazione rotazionale
  - Scheduling SLTF
  - Scheduling SPTF e SATF
- Considerazioni sul sistema
- Cache e Buffering del disco
- Gestione degli errori
- Software per I/O
- Altre tecniche di miglioramento delle prestazioni

S. Balsamo – Università Ca' Foscari Venezia – SO.5.0

0

## Obbiettivi

- Realizzazione delle operazioni di input/output su disco
- Come si completa input/output
- Importanza dell'ottimizzazione delle prestazioni
- Ottimizzare la ricerca (seek) e la rotazione
- Strategie di scheduling del disco
- caching e buffering
- Altre tecniche per migliorar le prestazioni del disco
- Principali schemi per realizzare Array Ridondanti di Dischi Indipendenti (RAID)

S. Balsamo – Università Ca' Foscari Venezia – SO.5.1

1

## Introduzione

- La memoria secondaria è spesso uno collo di bottiglia
  - Dispositivi di memoria permanente, economico, riscrivibile, di lunga durata
    - Nastri – inadeguati se è richiesto un accesso rapido alle locazioni
    - Dischi – ad accesso 'casuale' (diretto)
  - Evoluzione di costo/prestazioni
  - Vincoli meccanici
- Altri dispositivi I/O: tastiera, mouse, monitor
- I miglioramenti delle prestazioni di memoria secondaria aumentano in modo significativo le prestazioni dell'intero sistema
- Soluzioni possono essere basate sia su software e sia su hardware

S. Balsamo – Università Ca' Foscari Venezia – SO.5.2

2

## Introduzione

- Dispositivi di I/O
  - a blocchi – di dimensione fissa
    - es. dischi, penne USB
  - a caratteri
    - es. stampanti, interfacce di rete, mouse
- Alcune altre categorie
  - es. clock, touch screen
- Diverse velocità dei dispositivi
- Controllore del dispositivo o adattatore (componente elettronica)
- Dispositivo (componente meccanica)

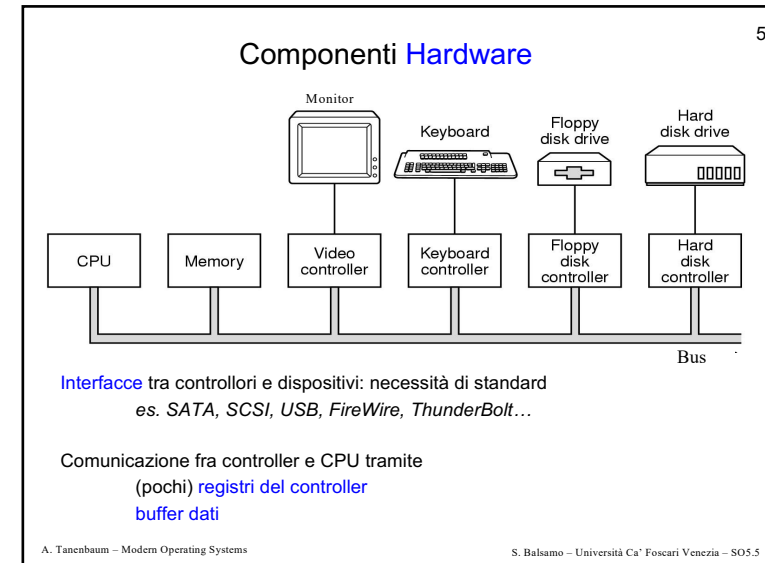
S. Balsamo – Università Ca' Foscari Venezia – SO.5.3

3

Dispositivo	Velocità di trasferimento dei dati
Tastiera	10 byte/s
Mouse	00 byte/s
Modem a 56 K	7 KB/s
Scanner a 300 dpi	1 MB/s
Videocamera digitale	3,5 MB/s
Disco Blu-ray 4x	18 MB/s
802.11n Wireless	37,5 MB/s
USB 2.0	60 MB/s
FireWire 800	100 MB/s
Gigabit Ethernet	125 MB/s
Disco fisso SATA 3	600 MB/s
USB 3.0	625 MB/s
Bus SCSI Ultra 5	640 MB/s
Bus PCIe 3.0 single lane	985 MB/s
Bus Thunderbolt 2	2,5 GB/s
Rete SONET OC-768	5 GB/s

A. Tanenbaum - Modern Operating Systems S. Balsamo - Università Ca' Foscari Venezia - SO.5.4

4



5

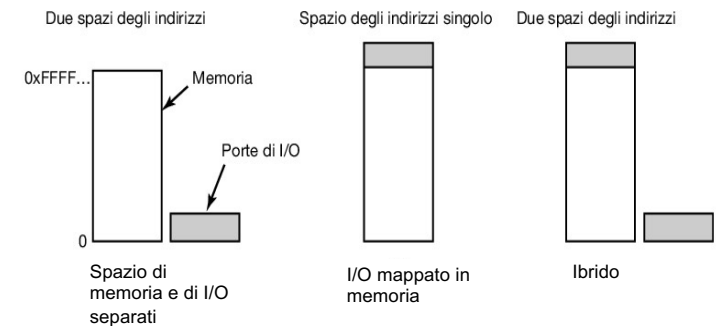
## Comunicazione con I/O

- **Comunicazione** fra CPU e dispositivi di I/O
  - **registri** di controllo ai quali è assegnata una **porta di I/O** (8 o 16 bit)
    - **spazio** delle porte: insieme delle porte
    - Protezione
  - **I/O mappato in memoria**
    - ad ogni registro è assegnato uno unico indirizzo di memoria, al quale non è assegnata memoria
  - **ibrido**
    - Due spazi separati, un buffer dai dati dei dispositivi IO mappati in memoria e porte IO separate per i registri di controllo

S. Balsamo - Università Ca' Foscari Venezia - SO.5.6

6

## Comunicazione con I/O



A. Tanenbaum - Modern Operating Systems

S. Balsamo - Università Ca' Foscari Venezia - SO.5.7

7

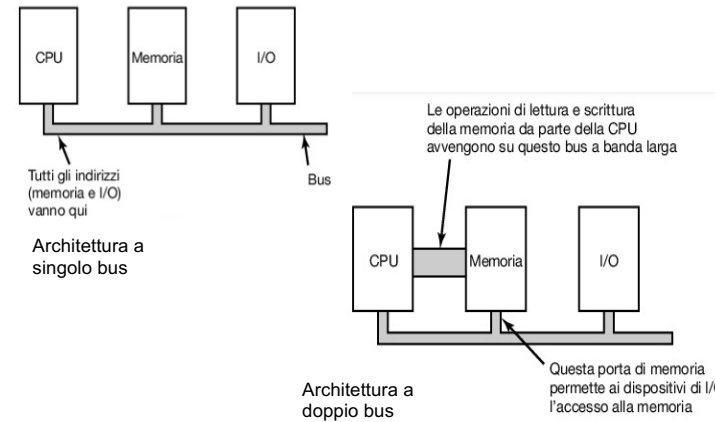
## I/O mappato in memoria

- Vantaggi
  - Il driver può essere scritto in **linguaggio** ad alto livello (es. C) e non in assembly – i registri sono solo variabili in memoria facilmente modificabili
  - **Protezione** semplice – controllo degli indirizzi
  - Le istruzioni possono riferirsi ai registri di controllo direttamente e semplificare la progettazione
- Svantaggi
  - Uso della **cache** non è possibile, va disabilitata selettivamente, azione potenzialmente complessa
  - Con bus separati i dispositivi di I/O potrebbero non poter vedere indirizzi di memoria spediti sul bus della memoria – alcune soluzioni

S. Balsamo – Università Ca' Foscari Venezia – SO.5.8

8

## I/O mappato in memoria



A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.5.9

9

## DMA

DMA – **Direct Memory Access**

**Controllore** DMA, accesso diretto alla memoria  
Accede al bus indipendentemente dalla CPU

Ha **molte** registri

inclusi registri di memoria,  
di conteggio di byte  
di controllo

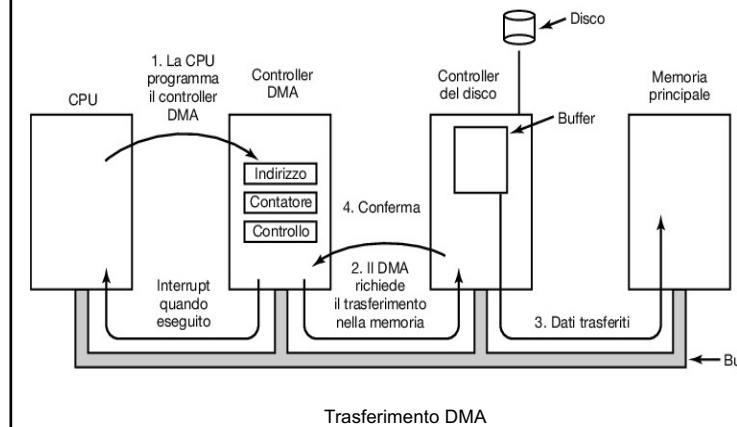
i registri di controllo contengono indicazioni delle porte I/O  
direzione di trasferimento  
dimensione dell'unità di trasferimento  
numero di byte da trasferire alla volta

Possibili trasferimenti multipli, con più registri di controllo, uno per canale e ogni trasferimento regolato da un controller di dispositivo

S. Balsamo – Università Ca' Foscari Venezia – SO.5.10

10

## DMA

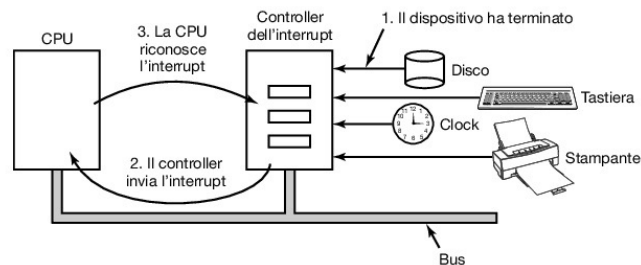


A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.5.11

11

### Gestione degli Interrupt



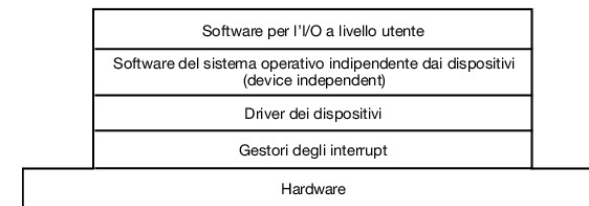
1. Il dispositivo causa l'Interrupt segnalato sulla linea del bus assegnatagli
2. Rilevato dal **controller dell'Interrupt** può essere trattato o momentaneamente ignorato.  
Il controller assegna linee di indirizzo al dispositivo e manda il segnale alla CPU
3. La CPU tratta l'Interrupt e usa le linee di indirizzo come indice della tabella (**vettore di interrupt**) e preleva il nuovo PC
4. Inizia la **procedura** per il **trattamento** dell'Interrupt

A. Tanenbaum - Modern Operating Systems

S. Balsamo - Università Ca' Foscari Venezia - SO.5.12

12

### Livelli di software di I/O



#### Gerarchia e livelli

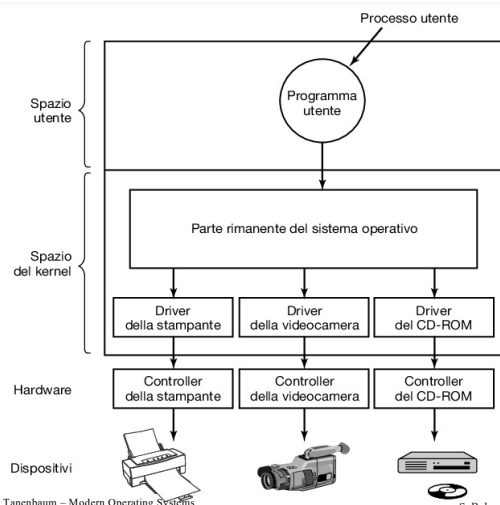
- I gestori di interrupt di regola non sono visibili all'utente
- **Driver** del dispositivo: codice di controllo - solitamente nel nucleo dispositivi a blocchi / a caratteri
- **Software** indipendente dai dispositivi fornisce l'interfaccia al
- **Software** di livello utente

A. Tanenbaum - Modern Operating Systems

S. Balsamo - Università Ca' Foscari Venezia - SO.5.13

13

### Driver dei dispositivi di I/O



A. Tanenbaum - Modern Operating Systems

S. Balsamo - Università Ca' Foscari Venezia - SO.5.14

14

### Software **indipendente** dai dispositivi di I/O

Interfacciamento uniforme dei driver dei dispositivi

Buffering

Segnalazione degli errori

Allocazione e rilascio dei dispositivi dedicati

Dimensione dei blocchi indipendente dai dispositivi

Alcune funzioni

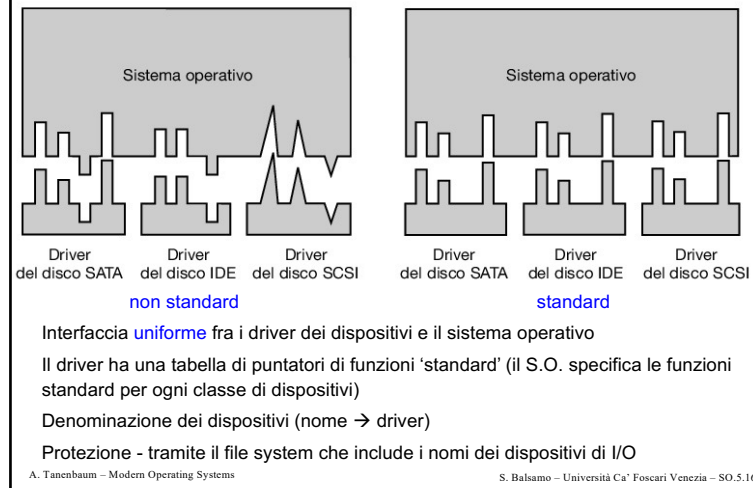
**Interfaccia uniforme** a livello utente

A. Tanenbaum - Modern Operating Systems

S. Balsamo - Università Ca' Foscari Venezia - SO.5.15

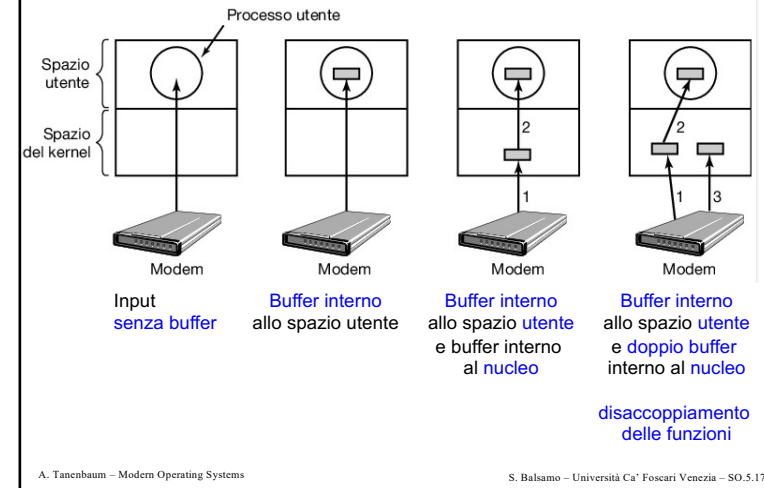
15

### Interfacce standard dei driver dispositivi di I/O



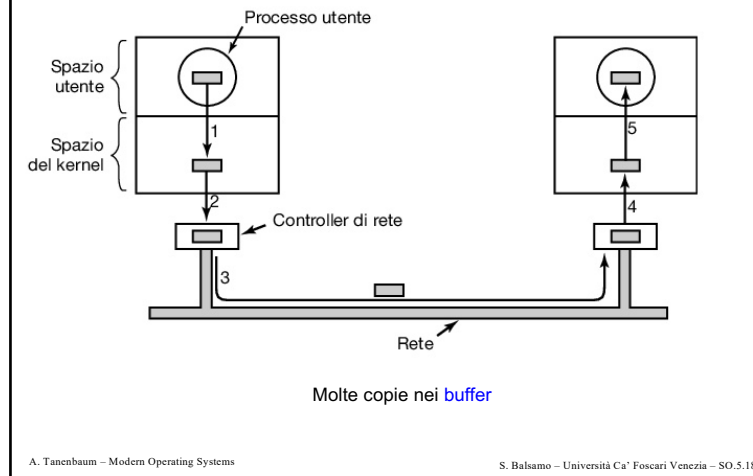
16

### Buffering per dispositivi di I/O



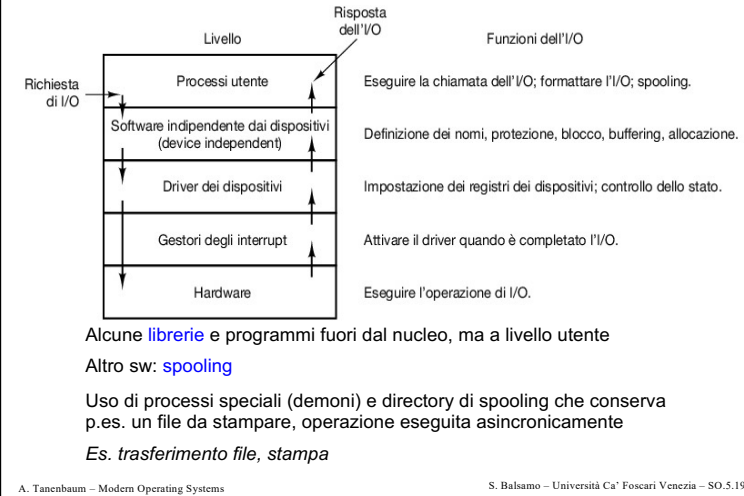
17

### Buffering per dispositivi di I/O e su rete



18

### Software di I/O nello spazio utente



19

## Obbiettivi del software di I/O

- Progettazione del software **indipendente** dal dispositivo  
definizione **uniforme** dei nomi (di file, di dispositivi,...)
- **Affidabilità**, correzione degli errori gestiti preferibilmente hardware
- **Tipi** di trasferimento - **comunicazione** dati CPU-I/O con operazioni  
sincrone (bloccanti) o  
asincrone (con interrupt)
- Gestione dei **buffer** nel trasferimento dei dati  
prestazioni, (es. operazioni real-time)
- **Condivisione**  
dispositivi condivisi da più utenti/processi  
dispositivi dedicati

A. Tanenbaum - Modern Operating Systems

S. Balsamo - Università Ca' Foscari Venezia - SO.5.20

20

## Tipi di software per I/O

Metodi di gestione software dell'I/O

- I/O **programmato**  
delega alla CPU il controllo dell'operazione  
**busy waiting** della CPU  
semplice, ma potenzialmente inefficiente
- I/O guidato dall'**interrupt**  
la CPU può eseguire altri processi mentre un processo è  
bloccato  
uso di interrupt  
maggior utilizzazione della CPU, ma molti interrupt
- I/O su **DMA**  
il controllore DMA, indipendente dalla CPU, interagisce con il  
dispositivo  
uso di hardware speciale, ma maggior concorrenza e utilizzo  
CPU

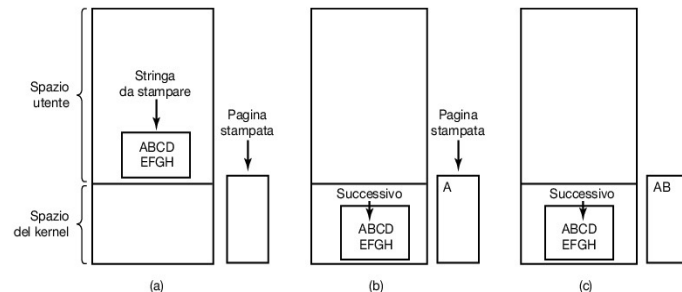
A. Tanenbaum - Modern Operating Systems

S. Balsamo - Università Ca' Foscari Venezia - SO.5.21

21

## Tipi di software per I/O

Esempio di I/O **programmato**



- (a) Buffer nello spazio utente - chiamata di sistema e copia nello spazio kernel  
(b) Controllo della stampa un carattere per volta - uso del registro dati della stampante  
(c) Avanzamento dopo il controllo che la stampante sia disponibile

A. Tanenbaum - Modern Operating Systems

S. Balsamo - Università Ca' Foscari Venezia - SO.5.22

22

## Tipi di software per I/O

Esempio di I/O **programmato**

Stampa di una stringa  
p buffer nel nucleo  
un carattere per volta tramite il buffer  
CPU controlla

```
copy_from_user(buffer, p, count);          /* p è il buffer del kernel*/
For (i = 0; i < count; i++){                /* ripeti per tutti i caratteri*/
    while ("printer_status_reg != READY");  /* ripeti finché lo stato della stampante
                                                non è READY*/

    *printer_data_register = p[i];          /* output di un carattere*/
}
return_to_user( );
```

A. Tanenbaum - Modern Operating Systems

S. Balsamo - Università Ca' Foscari Venezia - SO.5.23

23

## Tipi di software per I/O

Esempio di I/O guidato dall'interrupt

Stampa di una stringa  
p buffer nel nucleo  
un carattere per volta via via che arrivano  
gestione dell'interrupt

```
copy_from_user(buffer, p, count);
enable_interrupts();
while ("printer_status_reg != READY");
*printer_data_register = p[0];
scheduler();
```

Codice eseguito al tempo di  
chiamata di sistema per la stampa

```
If count == 0 {
    unblock_user();
} else {
    *printer_data_register = p[0];
    count = count - 1;
    i = i + 1;
}
acknowledge-interrupt();
return_from_interrupt( );
```

Procedura di gestione dell'interrupt per la stampa

A. Tanenbaum - Modern Operating Systems

S. Balsamo - Università Ca' Foscari Venezia - SO.5.24

24

## Tipi di software per I/O

Esempio di I/O su DMA

Stampa di una stringa tramite DMA  
gestione tramite accesso diretto alla stampante

```
copy_from_user(buffer, p, count);
setup_DMA_controller();
scheduler();
```

Codice eseguito al tempo di  
chiamata di sistema per la stampa

```
acknowledge-interrupt();
unblock_user( );
return_from_interrupt( );
```

Procedura di gestione dell'interrupt per la stampa

A. Tanenbaum - Modern Operating Systems

S. Balsamo - Università Ca' Foscari Venezia - SO.5.25

25

## Elaborazione di un interrupt per I/O

Alcuni passi del S.O. per trattare un interrupt I/O

- Salvataggio dei registri non ancora salvati dall'hardware (es PSW)
- Caricamento contesto per la procedura di gestione dell'interrupt
- Impostazione stack
- Avviso al controllore degli interrupt (o riabilitazione interrupt)
- Copia dei registri salvati nella tabella dei processi
- Esecuzione della procedura di gestione dell'interrupt, che recupera le informazioni dai registri del controllore del dispositivo
- Scelta del prossimo processo da eseguire
- Impostazione del contesto della MMU per il prossimo processo, eventualmente anche della TLB
- Caricamento dei nuovi registri del processo, compreso PSW
- Avvio dell'esecuzione del nuovo processo

A. Tanenbaum - Modern Operating Systems

S. Balsamo - Università Ca' Foscari Venezia - SO.5.26

26

## Evoluzione dei sistemi di memoria secondaria

- La maggior parte dei dispositivi di memorizzazione secondaria si basano su supporti magnetici
  - Accesso ai dati con una testina di lettura-scrittura
  - I primi tecnologie utilizzavano memoria sequenziale
    - Informazioni accessibili in modo ordinato uno per volta
    - Inefficiente per applicazioni ad accesso diretto
- Memorizzazione ad accesso casuale
  - Anche detto memoria ad accesso diretto
  - Accesso ai record in qualsiasi ordine

S. Balsamo - Università Ca' Foscari Venezia - SO.5.27

27

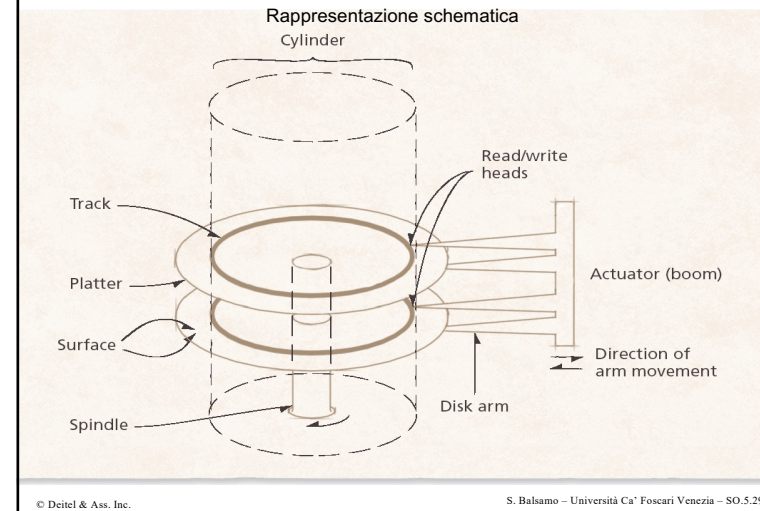
## Caratteristiche dei dischi a tesina mobile

- Struttura **fisica** di unità disco
  - Insieme di **dischi** (piatti) magnetici
    - Che **ruotano** su un perno (rotore)
    - Alta velocità
    - Composto da **tracce**, che a loro volta contengono **settori**
    - **Cilindri**: formati da gruppi verticali di tracce
    - Testina di lettura-scrittura molto vicina (micron)
    - **Braccio mobile** collegato ad un attuatore (*boom*)
    - Movimento della testina fra i cilindri
    - Ricerca del cilindro (*seek*)

S. Balsamo – Università Ca' Foscari Venezia – SO.5.28

28

## Caratteristiche dei dischi a tesina mobile



© Dcitel &amp; Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.5.29

29

## Caratteristiche dei dischi a tesina mobile

Parametro	Floppy disk IBM 360 KB	Disco fisso WD 3000 HLFS
Numero dei cilindri	40	36.481
Tracce per cilindro	2	255
Settori per traccia	9	63 (media)
Settori per disco	720	586.072.368
Byte per settore	512	512
Capacità del disco	360 KB	300 GB
Tempo di ricerca (cilindri adiacenti)	6 ms	0,7 ms
Tempo di ricerca (situazione media)	77 ms	4,3 ms
Tempo di rotazione	200 ms	6 ms
Tempo di stop/avvio del motore	250 ms	1 ms
Tempo per trasferire 1 settore	22 ms	1 μs

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.5.30

30

## Caratteristiche dei dischi a tesina mobile

- Indici di prestazione
  1. **Tempo di ricerca (*seek*)**
    - Tempo per la testina di lettura-scrittura per spostarsi nuovo cilindro
  2. **Latenza rotazionale**
    - Tempo di ritardo dovuto alla rotazione, perché i dati ruotino dalla posizione attuale alla testina di lettura-scrittura
  3. **Tempo di trasmissione**
    - Tempo di **trasferimento** perché tutti i dati cercati ruotino sotto la testina di lettura-scrittura

S. Balsamo – Università Ca' Foscari Venezia – SO.5.31

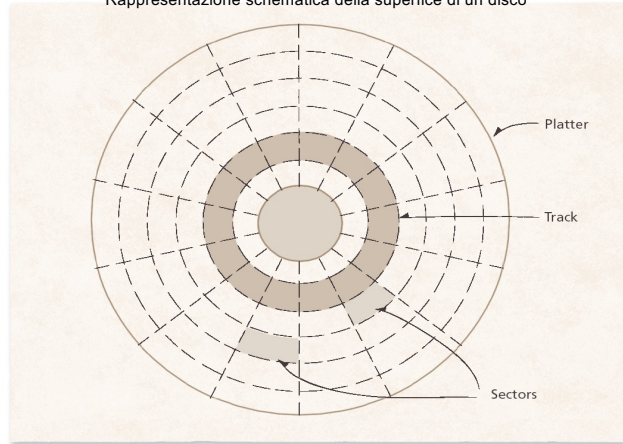
31



### Caratteristiche dei dischi a tesina mobile

Dischi dividono le tracce in molti settori, ognuno tipicamente contiene 512 byte

Rappresentazione schematica della superficie di un disco

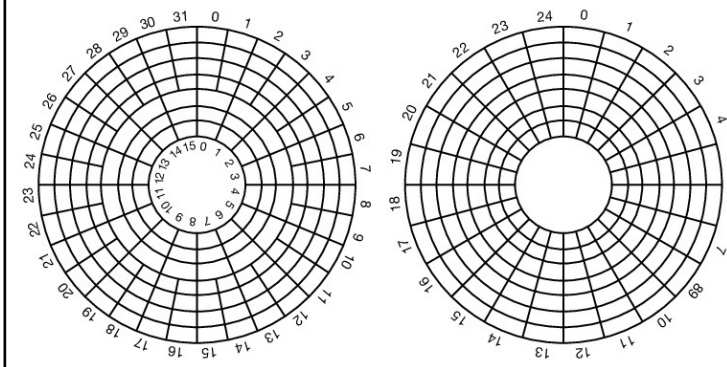


© Deitel & Ass, Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.5.32

32

### Caratteristiche dei dischi a tesina mobile



Numero di settori per cilindro – **geometria fisica** e **geometria virtuale**

I dischi moderni supportano un sistema di **indirizzamento logico** dei blocchi con numerazione dei settori senza considerare la geometria del disco

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.5.33

33

### RAID – Redundant array of inexpensive disks

Uso di meccanismi di

ridondanza per aumentare l'**affidabilità**

parallelismo per migliorare le **prestazioni** dei dischi

RAID è **visto** dal sistema come **una** sola grande unità disco

La progettazione include

la gestione di un **insieme di dischi** a cui si accede in parallelo

la gestione di molte **copie** di parti di dati

la **distribuzione** delle parti dei dati su più dischi

**Distribuzione**

**Partizione**

Insiemi (pila) di dischi SCSI o SATA

Compatibilità dei driver

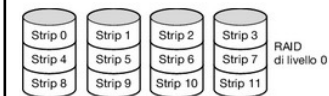
Diversi schemi di organizzazione dei RAID, detti livelli

A. Tanenbaum – Modern Operating Systems

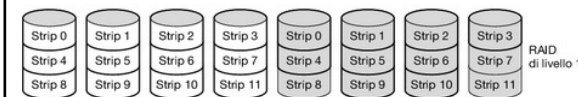
S. Balsamo – Università Ca' Foscari Venezia – SO.5.34

34

### RAID – Redundant array of inexpensive disks



**Striping**  
ciascuna di k settori

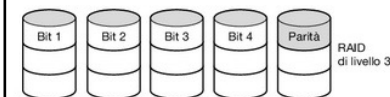


**Parallelismo**  
**Prestazioni**



**Ridondanza** per la  
tolleranza i guasti

**Parità** a parole o a  
strip  
**Sincronizzazione** dei  
dischi



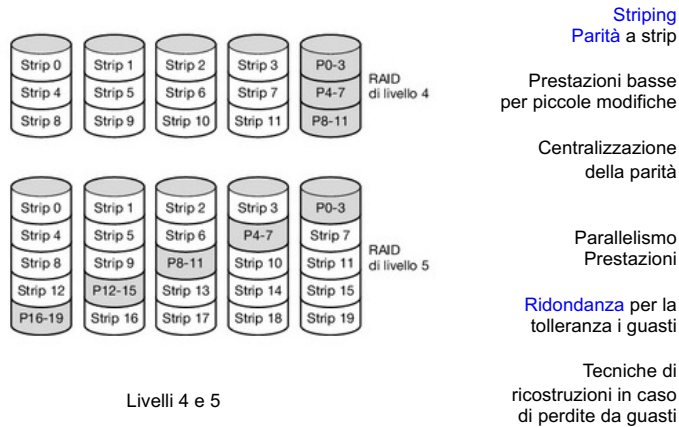
**Livelli 0,1,2,3**

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.5.35

35

## RAID – Redundant array of inexpensive disks



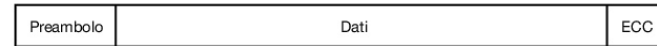
A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.5.36

36

## Formattazione dei dischi

**Formattazione a basso livello** dei piatti del disco, via sw  
Tracce concentriche con i settori



Settore di un disco

ECC informazioni ridondanti per recupero di errori, es. 16 byte

Lo spazio nel disco formattato si riduce es. circa del 20%

**Cylinder Skew**

per ogni traccia il settore 0 è spostato rispetto alla traccia precedente per migliorare le prestazioni

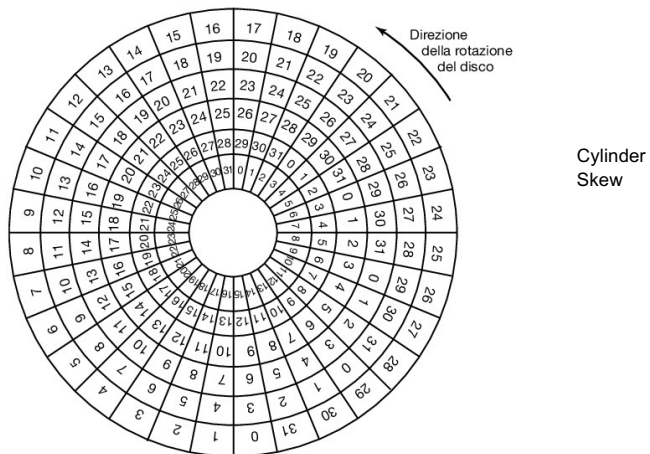
**Partizionamento del disco:** tabella delle partizioni e dimensione di ogni partizione**Formattazione ad alto livello** di ogni partizioneDefinisce: il blocco di avvio, gestione dello spazio libero, directory principale (*root*) e il file system (vuoto)

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.5.37

37

## Formattazione dei dischi

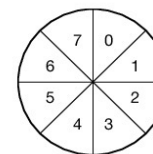


A. Tanenbaum – Modern Operating Systems

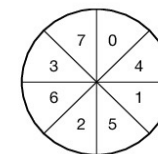
S. Balsamo – Università Ca' Foscari Venezia – SO.5.38

38

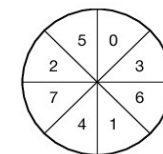
## Formattazione dei dischi



Nessun interleaving



Interleaving singolo



Interleaving doppio

Lettura di due settori consecutivi

Permette al controller di copiare un settore dal buffer in memoria principale

- Interleaving singolo

- Interleaving doppio (per copie lente)

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.5.39

39

## Necessità dello Scheduling del disco

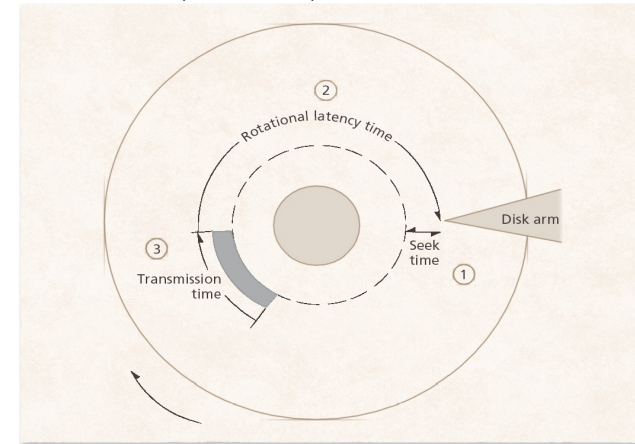
- Scheduling **First-come-first-served** (FCFS) : notevoli svantaggi
  - La ricerca di dati con posizioni distribuite in modo casuale produce lunghi tempi di attesa e scarso throughput
  - Sotto carico pesante, il sistema può comportarsi in modo critico (*thrashing*)
- Le richieste devono essere **servite in ordine logico** per **minimizzare i ritardi**
  - Servire le richieste che richiedono il **minimo** movimento meccanico
- I primi algoritmi di scheduling del disco si sono concentrati sulla **minimizzazione del tempo di seek**, la componente del tempo di accesso al disco con maggior latenza
- I moderni sistemi ottimizzano **anche** il **tempo di rotazione**

S. Balsamo – Università Ca' Foscari Venezia – SO.5.40

40

## Necessità dello Scheduling del disco

Componenti del tempo di accesso ad un disco



© Deitel &amp; Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.5.41

41

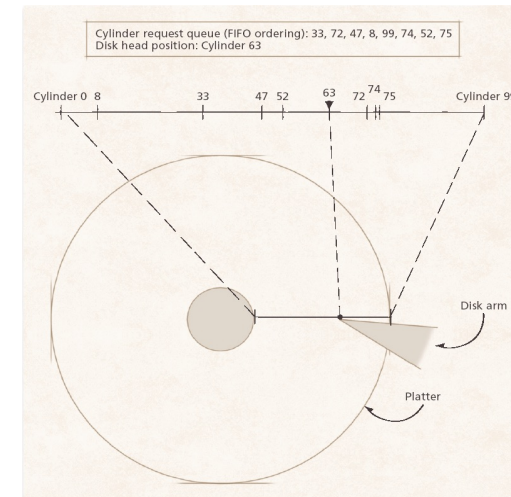
## Strategie di Scheduling del disco

- Tre criteri per misurare le strategie
  - **Throughput**
    - Numero di richieste servite per unità di tempo
  - **Tempo medio di risposta**
    - Tempo medio di attesa che la richiesta sia servita e di servizio
  - **Varianza** del tempo di risposta
    - Misura della prevedibilità del tempo di risposta
- Obiettivi generali
  - **Massimizzare** il **throughput**
  - **Minimizzare** il **tempo di risposta** e la **varianza** di tempi di risposta

S. Balsamo – Università Ca' Foscari Venezia – SO.5.42

42

## Strategie di Scheduling del disco



© Deitel &amp; Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.5.43

Modello  
di richieste  
di accesso  
ad un disco

43

### Scheduling del disco **First-Come-First-Served** (FCFS)

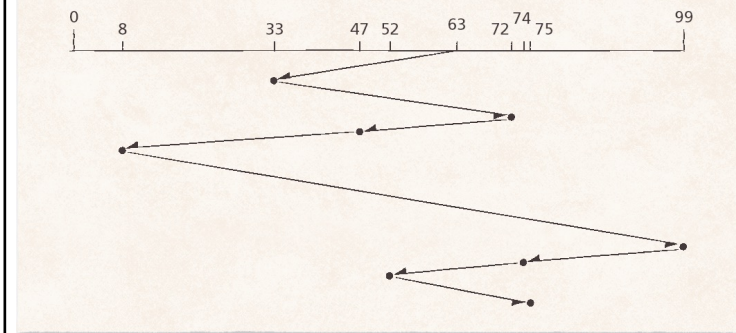
- Richieste servite in ordine di arrivo
  - Vantaggi
    - Equo
    - Previene l'attesa infinita
    - basso overhead
  - Svantaggi
    - Possibile throughput estremamente basso
      - FCFS in genere porta ad un modello di ricerca di seek casuale perché non riordina le richieste per minimizzare il ritardo di servizio

S. Balsamo – Università Ca' Foscari Venezia – SO.5.44

44

### Scheduling del disco **First-Come-First-Served** (FCFS)

Operazioni di seek con la strategia FCFS



Coda di richieste: 33, 72, 47, 8, 99, 74, 52, 75 – Cilindro di partenza: 63  
 Tempo medio di seek: tempo totale 294, tempo medio 36,75

© Deitel &amp; Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.5.45

45

### Shortest-Seek-Time-First

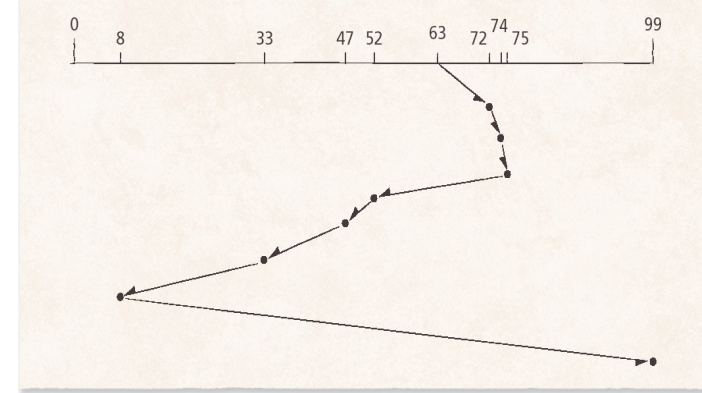
- **SSTF**: richiesta di servizio più vicina alla testina di lettura-scrittura
  - vantaggi
    - throughput maggiore e tempi di risposta inferiori rispetto a FCFS
    - soluzione ragionevole per i sistemi di elaborazione batch
  - svantaggi
    - Non garantisce equità
    - Possibilità di **attesa infinita**
    - Alta varianza dei tempi di risposta
    - Il tempo di risposta generalmente inaccettabile per sistemi interattivi

S. Balsamo – Università Ca' Foscari Venezia – SO.5.46

46

### Shortest-Seek-Time-First

Operazioni di seek con la strategia SSTF



Coda di richieste: 33, 72, 47, 8, 99, 74, 52, 75 – Cilindro di partenza: 63  
 Tempo medio di seek: tempo totale 170, tempo medio 21,25

© Deitel &amp; Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.5.47

47



## Scheduling del disco SCAN

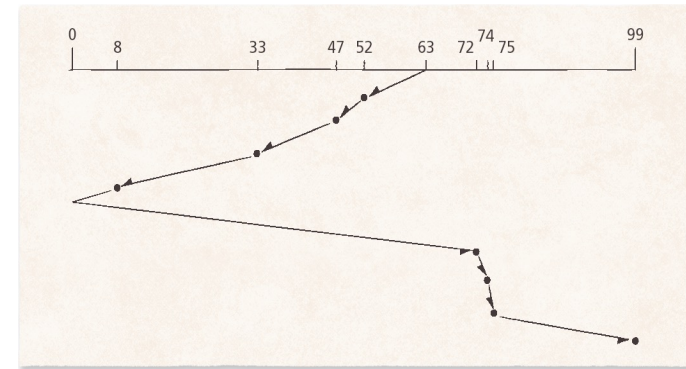
- SCAN: tempo più breve di *seek* in una direzione preferita
  - Non cambia direzione fino a quando non si è raggiunto il limite del disco
  - Algoritmo dell'*ascensore*
  - Caratteristiche simili a SSTF
  - Attesa infinita ancora possibile
  - Non equo, le tracce centrali favorite
  - Migliora la *varianza dei tempi di risposta*

S. Balsamo – Università Ca' Foscari Venezia – SO.5.48

48

## Scheduling del disco SCAN

Operazioni di seek con la strategia SCAN



Coda di richieste: 33, 72, 47, 8, 99, 74, 52, 75 – Cilindro di partenza: 63  
 Tempo medio di seek: tempo totale 165, tempo medio 20,25

© Deitel &amp; Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.5.49

49

## Scheduling del disco C-SCAN

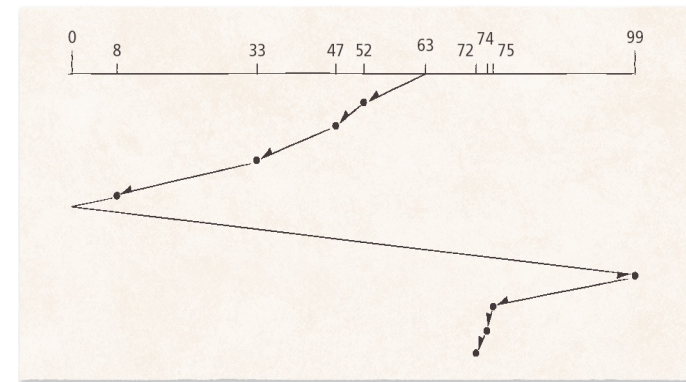
- C-SCAN: (Circolare):  
 simile a SCAN, ma alla fine di una scansione verso l'interno,  
 il braccio del disco salta (senza servire richieste) al cilindro più esterno  
  
 si muove sempre nella stessa direzione per servire le richieste
  - Ulteriore *riduzione della varianza dei tempi di risposta*, a scapito del throughput e del tempo medio di risposta

S. Balsamo – Università Ca' Foscari Venezia – SO.5.50

50

## Scheduling del disco C-SCAN

Operazioni di seek con la strategia C-SCAN



Coda di richieste: 33, 72, 47, 8, 99, 74, 52, 75 – Cilindro di partenza: 63  
 Tempo medio di seek: tempo totale 180, tempo medio 23,625

© Deitel &amp; Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.5.51

51

## Scheduling del disco FSSCAN e N-Step SCAN

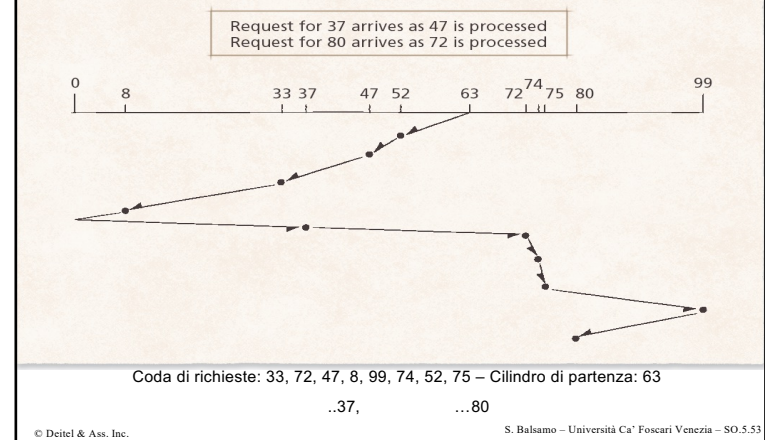
- Gruppi batch di richieste in
- **FSCAN**  
"congela" periodicamente la coda di richieste al disco e serve solo le richieste in coda in quel momento
- **N-Step SCAN**:  
serve solo le prime  $n$  richieste nella coda in quel momento
  - Entrambe le strategie prevengono l'attesa infinita
  - Entrambe **riducono la varianza dei tempi di risposta** rispetto a SCAN

S. Balsamo – Università Ca' Foscari Venezia – SO.5.52

52

## Scheduling del disco FSSCAN e N-Step SCAN

Operazioni di seek con la strategia FSCAN

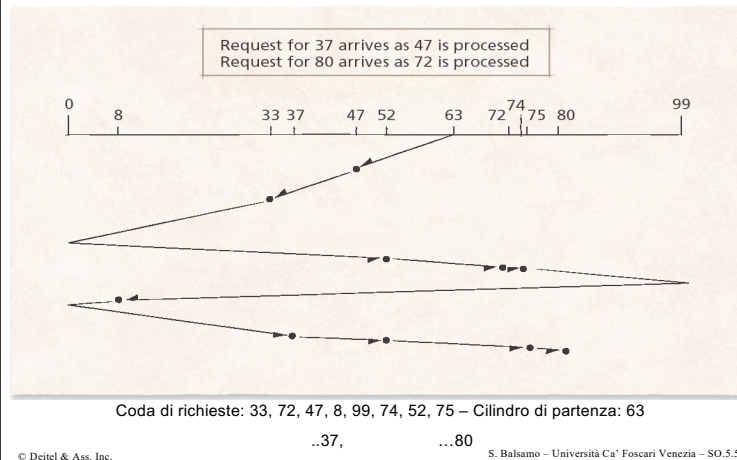


© Deitel &amp; Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.5.53

53

## Scheduling del disco FSSCAN e N-Step SCAN

Operazioni di seek con la strategia N-Step SCAN ( $n=3$ )

© Deitel &amp; Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.5.54

54

## Scheduling del disco LOOK e C-LOOK

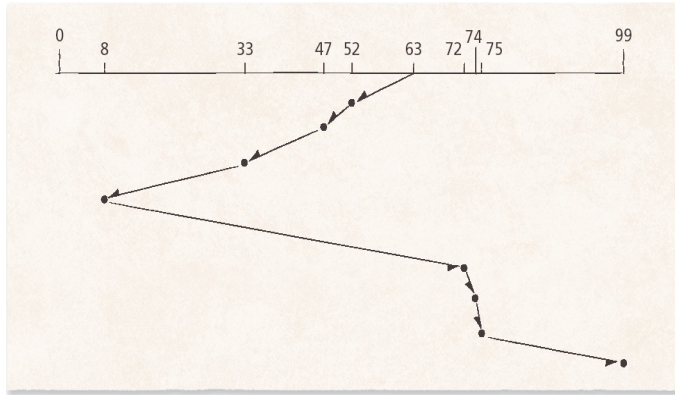
- **LOOK**: Migliora lo scheduling SCAN
  - (*look ahead*) continua fino al termine dell'attraversamento attuale per servire richieste, se non ci sono cambia direzione
    - Muove il braccio del disco verso il bordo esterno del disco se non ci sono richieste pendenti per tali regioni
    - Migliora l'efficienza evitando operazioni inutili di ricerca
    - Throughput elevato
- **C-LOOK** migliora lo scheduling C-SCAN
  - Combinazione di LOOK e C-SCAN
  - Quando non ci sono richieste nell'attraversamento verso l'interno si sposta verso le richieste posizionate più all'esterno senza servirsene altre in mezzo e inizia un nuovo attraversamento
  - Minor varianza dei tempi di risposta di LOOK, a scapito della throughput

S. Balsamo – Università Ca' Foscari Venezia – SO.5.55

55

## Scheduling del disco LOOK e C-LOOK

Operazioni di seek con la strategia LOOK



Coda di richieste: 33, 72, 47, 8, 99, 74, 52, 75 – Cilindro di partenza: 63

© Deitel & Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.5.56

56

## Scheduling del disco – sintesi e confronto

Strategia	Descrizione
<b>FCFS</b>	Serve le richieste in ordine di arrivo
<b>SSTF</b>	Serve per prime le richieste con minor distanza di <i>seek</i>
<b>SCAN</b>	La testina si sposta avanti e indietro e serve secondo SSTF
<b>C-SCAN</b>	La testina si sposta avanti e serve secondo SSTF in quella direzione, arrivata all'interno salta a quella più esterna e ripete
<b>FSCAN</b>	Come SCAN eccetto le nuove rinviate al successivo attraversamento
<b>SCAN n-STEPS</b>	Come FSCAN ma serve solo <i>n</i> richieste per attraversamento. Evita l'attesa infinita
<b>LOOK</b>	Come SCAN, ma la testina cambia direzione quando raggiunge l'ultima richiesta nella direzione preferenziale
<b>C-LOOK</b>	Come C-SCAN, ma la testina si ferma quando raggiunge l'ultima richiesta nella direzione preferenziale, serve la richiesta al cilindro più vicino al lato opposto del disco

57

## Ottimizzazione rotazionale

- Il tempo di ricerca (*seek*) precedentemente determinava i problemi di prestazioni
- Se i **tempi di ricerca** e la **latenza rotazionale** sono dello **stesso ordine di grandezza**
  - strategie sviluppate di recente tentano di ottimizzare le prestazioni del disco **riducendo la latenza rotazionale**
  - Importante **quando** si accede a **piccoli porzioni di dati distribuiti casualmente** in tutto il disco

S. Balsamo – Università Ca' Foscari Venezia – SO.5.58

58

## Scheduling SLTF

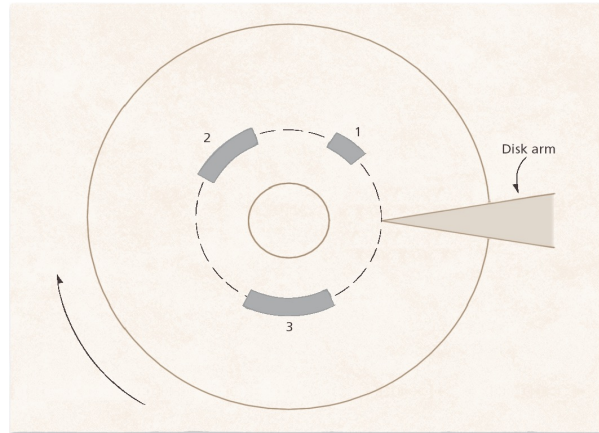
- Shortest-latency-time-first** scheduling
  - In un dato cilindro, serve le richieste con **la minima latenza di rotazione**
  - Facile da implementare
  - Accodamento dei settori
  - Raggiunge prestazioni quasi ottimali per la latenza rotazionale

S. Balsamo – Università Ca' Foscari Venezia – SO.5.59

59

## Scheduling SLTF

SLTF scheduling: le richieste sono servite nell'ordine indicato senza considerare l'ordine di arrivo



© Deitel & Ass. Inc.

S. Balsamo - Università Ca' Foscari Venezia - SO.5.60

60

## Scheduling SPTF e SATF

- Shortest-positioning-time-first scheduling
  - Tempo di **posizionamento**: somma di **tempo di ricerca** e la **latenza di rotazione**
  - SPTF serve per prima la richiesta con il minimo tempo di posizionamento
  - Buone prestazioni
  - Può causare attesa infinita (cilindri più sui bordi)

S. Balsamo - Università Ca' Foscari Venezia - SO.5.61

61

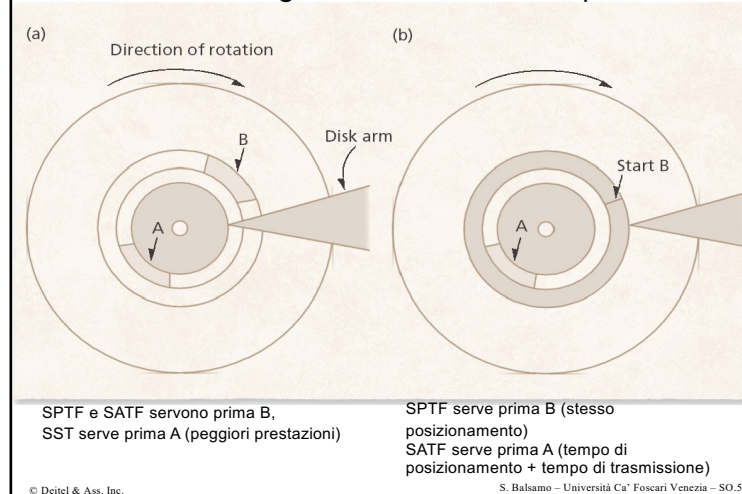
## Scheduling SPTF e SATF

- Shortest-access-time-first scheduling
  - Variante di SPTF
  - Tempo di **accesso**: tempo di **posizionamento** più il tempo di **trasmissione**
  - Throughput elevato
    - Anche in questo caso, può causare attesa infinita
- Sia SPTF e SATF possono implementare LOOK (*ahead*) per migliorare le prestazioni
- Svantaggio
  - Sia SPTF e SATF richiedono la conoscenza delle caratteristiche di prestazioni del disco che potrebbero non essere immediatamente disponibili (tempi di *seek*, latenza, posizioni dei settori)
  - Possibile mascheramento al S.O. per controllo degli errori, correzione dei dati e riassegnazione trasparente dei settori danneggiati

S. Balsamo - Università Ca' Foscari Venezia - SO.5.62

62

## Scheduling SPTF e SATF - esempio



© Deitel & Ass. Inc.

S. Balsamo - Università Ca' Foscari Venezia - SO.5.63

63



## Considerazioni di sistema

- Lo scheduling del disco è spesso utile, ma non sempre
  - Non aiuta sensibilmente nei sistemi processor-bound
  - Tipologie di **carico**: beneficio all'aumentare della multiprogrammazione e casualità
  - Richieste al disco in sequenze imprevedibili
    - Esempi: archiviazione in reti locali, uso di database e web server, molti utenti e richieste piccole (Online Transactions Processing)
  - Per **distribuzioni non uniformi delle richieste**, l'overhead dello scheduling può ridurre le prestazioni
    - Esempio: archiviazione di file posizionati su cilindri adiacenti
  - Le tecniche di **organizzazione dei file** a volte contrastano algoritmi di scheduling
  - Geometria reale e **geometria virtuale** possono vanificare i vantaggi degli algoritmi di scheduling

S. Balsamo – Università Ca' Foscari Venezia – SO.5.64

64

## Caching e Buffering

- Buffer Cache**: memorizzazione di una **copia dei dati su disco in memoria più veloce**
  - Situato in **memoria principale**, onboard cache, o sul controller del disco
  - Tempi di accesso molto minori dell'accesso al disco
  - Può essere usato come un **buffer** per ritardare la scrittura dei dati finché disco è sotto carico leggero
- Potenziale incoerenza
  - Il contenuto della memoria principale potrebbe essere perso per mancanza di corrente o guasto del sistema

S. Balsamo – Università Ca' Foscari Venezia – SO.5.65

65

## Caching e Buffering

Gestione della possibile incoerenza

- Cache write-back** (scrivi alla fine)
  - i dati non sono scritti su disco immediatamente, ma accorpati
  - Migliora le prestazioni
  - Periodicamente flushed verso il disco
- Write-through caching** (scrivi subito)
  - Scrive contemporaneamente su disco e cache
  - Riduce le prestazioni rispetto al write-back, ma garantisce la coerenza

Possibile **cache preventivo** di più settori da parte del controller

Nota: **cache del controller** indipendente dalla cache del sistema operativo

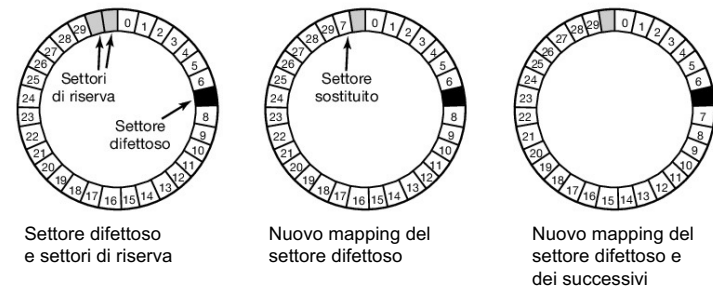
per blocchi non richiesti ma letti per convenienza della posizione

S. Balsamo – Università Ca' Foscari Venezia – SO.5.66

66

## Gestione degli errori

Errori su una **traccia**, di **setto**



Traccia di un disco con un settore con errore  
Ridefinizione del mapping dei settori. Uso di tabelle interne, una per traccia.

Errori su un **cilindro**, di posizionamento

Possibile ricalibratura

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.5.67

67

## Gestione degli errori

Problemi di affidabilità - uso di RAID

Problemi: **crash** e **errori durante la lettura** con corruzione dei dati

**Memoria stabile**: sottosistema disco che o scrive correttamente o non esegue niente.

Assumendo di disporre di una coppia di dischi identici D1 e D2 e corrispondenti.

- Operazione dei **scrittura stabile**  
scrive il blocco in D1, se non è corretto ripete  $n$  volte  
se fallisce  $n$  volte mappa il blocco con uno di riserva e ripete  
eventualmente ripete finché non completa  
copia sul blocco corrispondente di D2  
(scrittura corretta senza **crash** della CPU)
- Operazione dei **lettura stabile**  
legge il blocco da D1, se non è corretto ripete  $n$  volte  
se fallisce  $n$  volte legge da D2  
(probabilità di doppio errore trascurabile)
- Ripristino da **crash**  
scansione e confronto dei blocchi da D1 e D2  
se uguali e validi è completato  
se uno è errato viene riscritto con l'altro corrispondente  
se validi ma diversi D1 sovrascrive D2

A. Tanenbaum - Modern Operating Systems

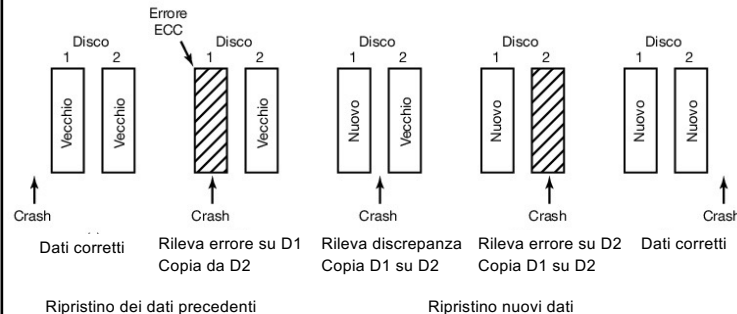
S. Balsamo - Università Ca' Foscari Venezia - SO.5.68

68

## Gestione degli errori

Senza crash sono presenti due copie valide

In caso di **crash della CPU** diverse possibili situazioni per il ripristino



A. Tanenbaum - Modern Operating Systems

S. Balsamo - Università Ca' Foscari Venezia - SO.5.69

69

## Altre tecniche di ottimizzazione di prestazioni del disco

- Frammentazione di file e record a seguito di modifiche (aggiunte/rimozioni)
- **Deframmentazione** (riorganizzazione del disco)
  - Applicati **periodicamente**
  - Inserire i dati in relazione in **settori contigui**
  - Diminuisce il numero di operazioni di ricerca richiesto
  - Usare allocazione adiacente a spazio libero per dati frequenti o in espansione
  - Il partizionamento può aiutare a ridurre la frammentazione (file memorizzati in partizioni di disco)
- **Compressione**
  - I dati consumano meno spazio su disco
  - Migliora i tempi di trasferimento e di accesso
  - Maggiore overhead del tempo di esecuzione per la compressione / decompressione

S. Balsamo - Università Ca' Foscari Venezia - SO.5.70

70

## Altre tecniche di ottimizzazione di prestazioni del disco

- **Copie multiple** di dati richiesti più frequentemente
  - Diverse posizioni del disco
  - Accesso alla **copia più vicina** alla testina di lettura-scrittura
  - Minor tempo di ricerca e di rotazione
  - Può comportare **overhead** significativi di memoria
  - Adatta per dati in sola lettura o rare modifiche (congruenza delle copie)
- Accorpamento di record (**blocking**)
  - Leggere / scrivere più record come **unico blocco** di dati
  - Riduce i tempi
- **Anticipazione del braccio del disco**
  - Quando inattivo, sposta il braccio del disco nella **posizione** dove è **maggiore** la **probabilità** del **prossimo accesso** ai dati, o al centro
  - Minor tempo di attesa specie in caso di località nella zona prevista
  - Se il braccio del disco predice in modo non corretto il prossimo accesso al disco, le prestazioni possono subire forti degradazioni
  - Meno efficace con la multiprogrammazione

S. Balsamo - Università Ca' Foscari Venezia - SO.5.71

71

## Software per I/O

Input: da **tastiera** e da **mouse**

Output: verso **finestre** di testo, **interfacce grafiche (GUI)**

Il driver della tastiera fornisce un numero  
 il driver converte in caratteri (*orientata a carattere*)  
 usa una tabella ASCII  
 li passa al programma  
 oppure gestisce una riga (*orientata a riga*) (*in POSIX modalità canonica*)  
 la passa al programma  
 caratteri speciali (comandi di controllo e gestione I/O)

Eccezioni, adattamenti necessari per altre lingue  
 molti sistemi operativi forniscono keymap o codici caricabili

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.5.72

72

## Software per I/O

Carattere	Nome POSIX	Commento
CTRL-H	ERASE	Cancella un carattere prima del cursore
CTRL-U	KILL	Cancella l'intera linea digitata
CTRL-V	LNEXT	Interpreta letteralmente il carattere successivo
CTRL-S	STOP	Ferma l'output
CTRL-Q	START	Avvia l'output
DEL	INTR	Interrompe il processo (SIGINT)
CTRL-\	QUIT	Forza il core dump (SIGQUIT)
CTRL-D	EOF	Fine del file
CTRL-M	CR	A capo (non modificabile)
CTRL-J	NL	Nuova riga (non modificabile)

Caratteri speciali in modalità canonica POSIX

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.5.73

73

## Software per I/O

Output: verso **finestre** di testo, **interfacce grafiche (GUI)**

Finestra di testo  
 blocco di caratteri (es. una linea)  
**editori** di schermo più complessi – comandi per gestire il cursore  
 sequenze di escape – *termcap* pacchetto sw uniforme  
 standard ANSI

Sequenza di escape	Significato
ESC[nA	Muovi verso l'alto di n linee
ESC[nB	Muovi verso il basso di n linee
ESC[nC	Muovi a destra di n spazi
ESC[nD	Muovi a sinistra di n spazi
ESC[m;nH	Muovi il cursore a (m, n)
ESC[sJ	Cancella lo schermo a partire dal cursore (0 fino alla fine, 1 dall'inizio, 2 tutto)
ESC[sK	Cancella la linea a partire dal cursore (0 fino alla fine, 1 dall'inizio, 2 tutto)
ESC[nL	Inserisci n linee a partire dal cursore
ESC[nM	Cancella n linee a partire dal cursore
ESC[nP	Cancella n caratteri a partire dal cursore
ESC[n@	Inserisci n caratteri a partire dal cursore
ESC[nm	Abilita l'interpretazione n (0 = normale, 4 = grassetto, 5 = lampeggiante, 7 = in negativo)
ESCM	Fa scorrere lo schermo indietro se il cursore è alla prima linea

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.5.74

74

## Software per I/O

### Sistema X Window (X)

interfaccia per sistemi Unix  
 sviluppato al M.I.T., portabile, eseguito nello **spazio utente**  
 sistema sw cliente-servente  
 possono essere eseguiti sulla stessa macchina  
 o su macchine diverse

guidato da **eventi**

input: tastiera e mouse (X client)

output: schermo (X server)

es. su Linux ambienti desktop Gnome e KDE eseguiti su X

Sistema X non è una GUI completa

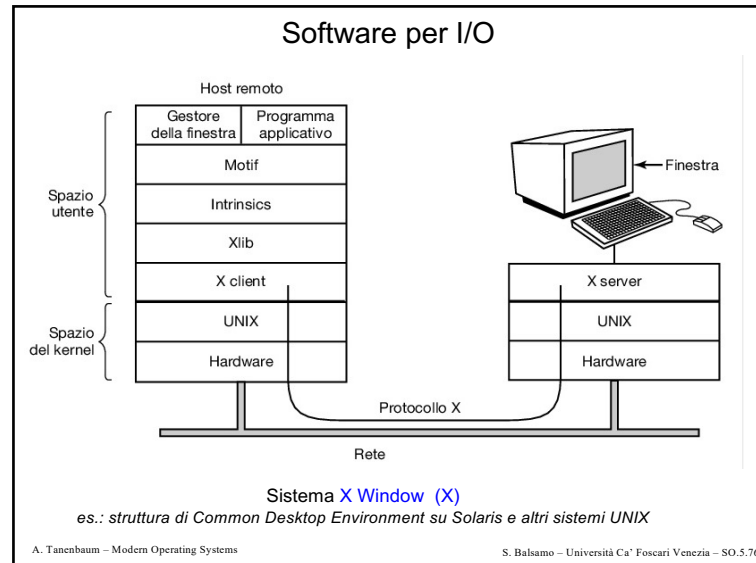
Struttura a livelli

Xlib	libreria X: procedure per accedere alle
funzioni di X	
protocollo X	dialogo fra X client e X server
Intrinsics	strato con strumenti per accedere e usare Xlib
	es. <i>gestione dei testi, barre di scorrimento, etc – widget</i>
Motif	rende uniforme l'accesso alle funzioni

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.5.75

75



76

### Software per I/O

La gestione delle finestre non è parte del sistema **X Window Manager** cliente separato dall'X client  
 gestione dello schermo  
 interagisce con X client

Schema adottato nei s.o. **Unix**, Linux, nelle diverse varianti  
 Interfaccia standard

Nel s.o. **Windows** i sistemi a finestra e le GUI sono uniti  
 collocati nel livello **nucleo**  
 più complessa la manutenzione e portabilità

A. Tanenbaum – Modern Operating Systems      S. Balsamo – Università Ca' Foscari Venezia – SO.5.77

77

### Software per I/O - GUI

**Graphical User Interface - GUI**

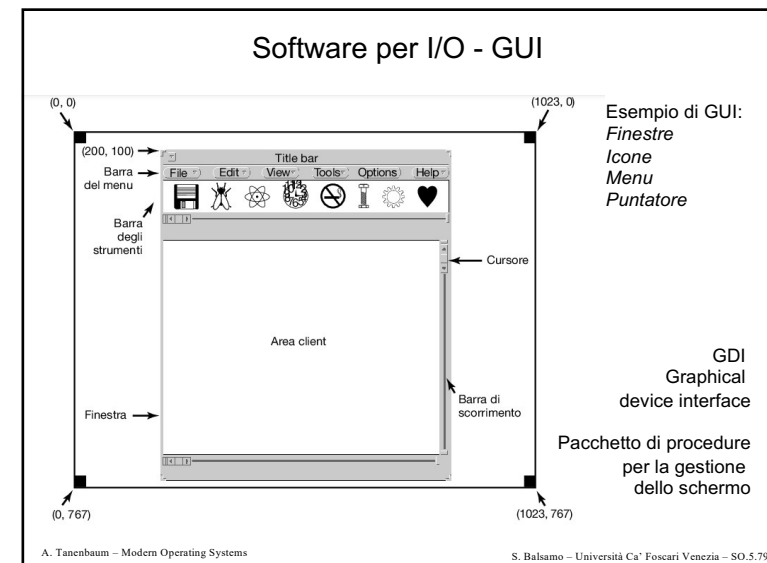
Sviluppato alla Università di Stanford, poi da Xerox,  
 poi adottato dalla Apple  
 Lisa – sistema Macintosh  
 Microsoft poi ne usa alcuni elementi di interfaccia e sviluppa il sistema Windows

Elementi del GUI: WIMP    Windows – Icons – Menu – Pointer

Sviluppato      a livello utente    (*Unix*)  
                   a livello nucleo    (*Windows*)

A. Tanenbaum – Modern Operating Systems      S. Balsamo – Università Ca' Foscari Venezia – SO.5.78

78



79