

Sistemi Operativi

Docente: Ugo Erra
ugoerr+so@dia.unisa.it



13° LEZIONE MEMORIA SECONDARIA E TERZIARIA

*CORSO DI LAUREA TRIENNALE IN INFORMATICA
UNIVERSITA' DEGLI STUDI DELLA BASILICATA*



Sommario della lezione



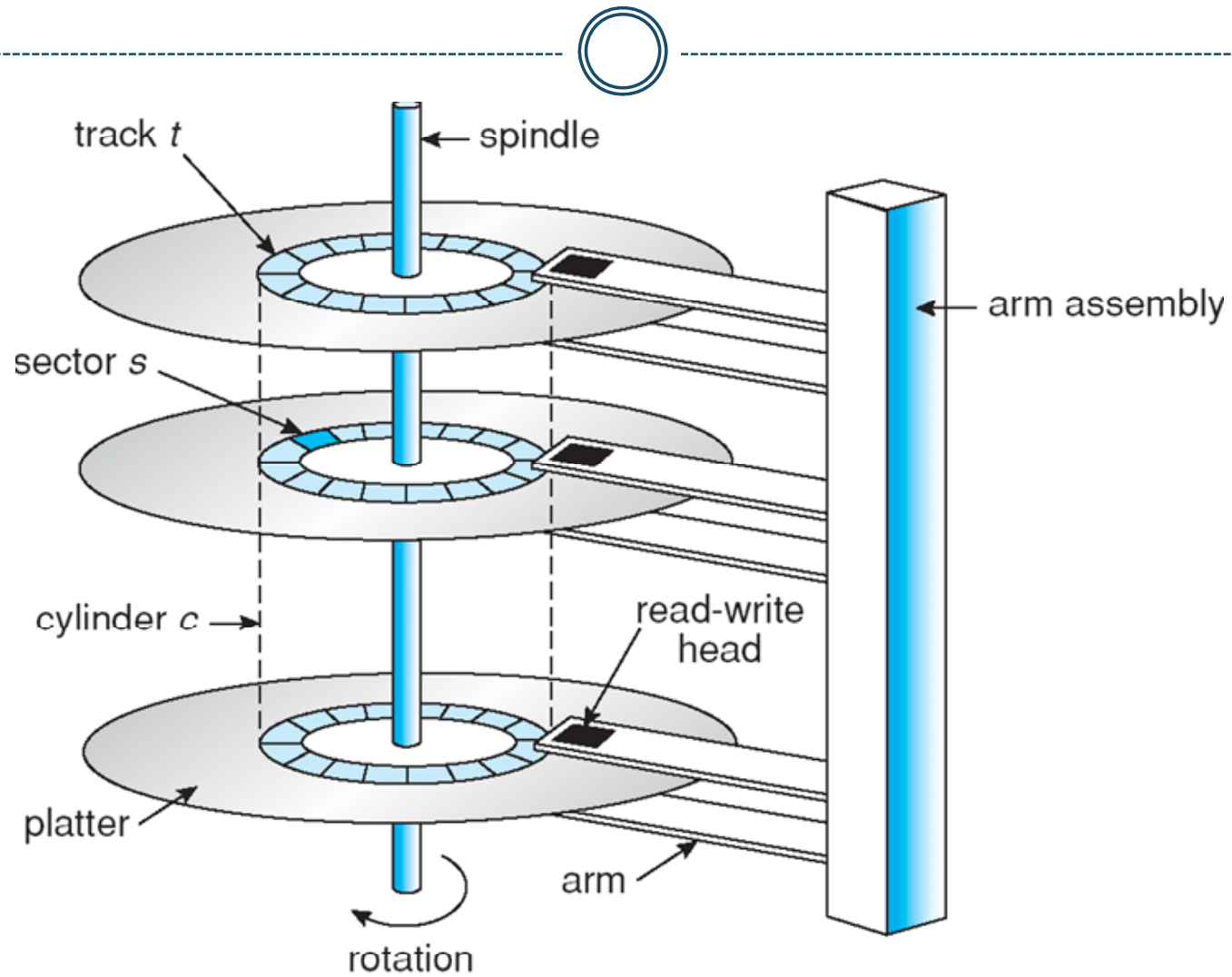
- **Struttura dei dispositivi di memorizzazione**
 - Dischi magnetici
- **Scheduling del disco**
- **Gestione dell'unità a disco**
- **Strutture RAID**

Dischi magnetici

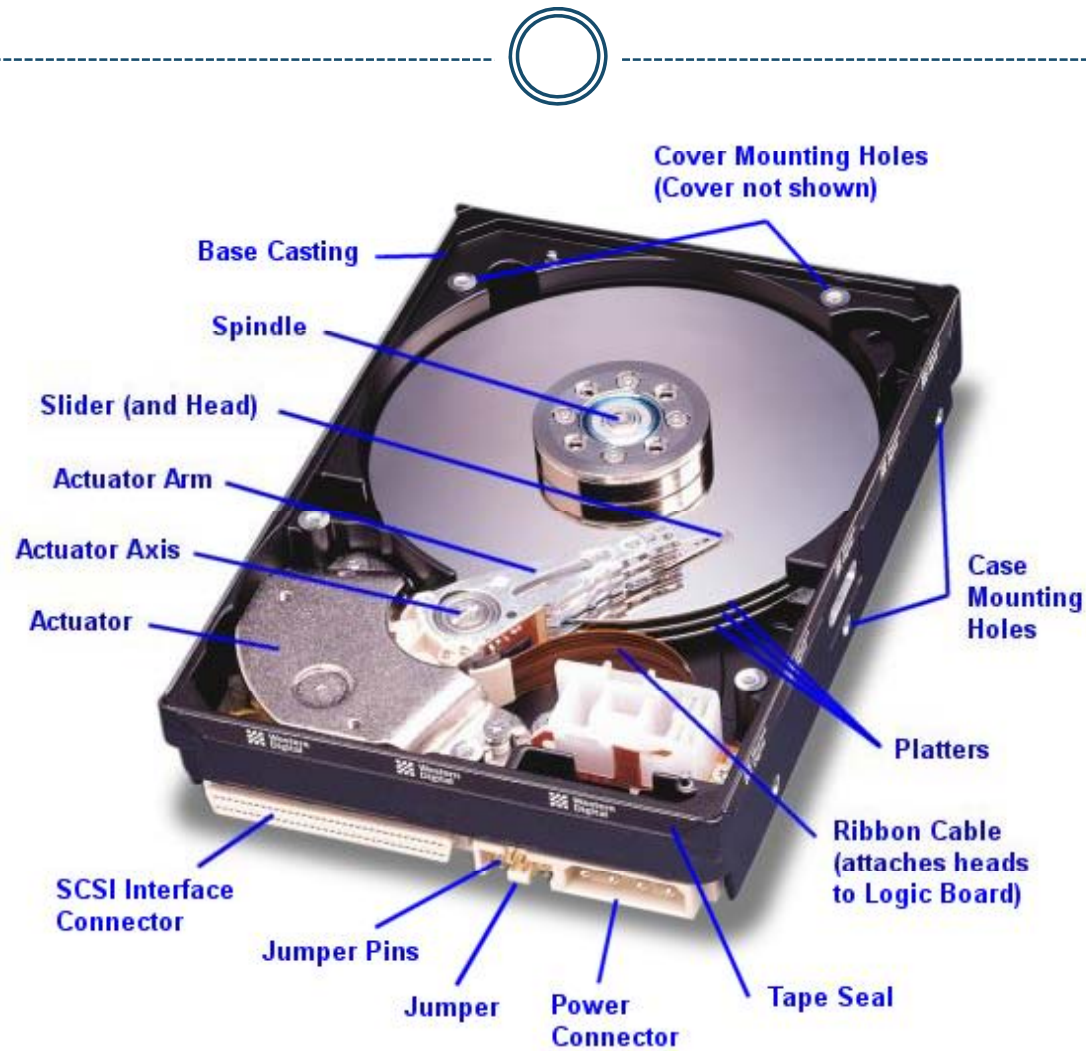


- Un disco fisso è composto da una serie di **piatti** sovrapposti
- Ogni piatto è suddiviso in **tracce** circolari concentriche
- Ogni traccia è suddivisa in una serie di **settori**
- L'insieme delle tracce nella stessa posizione sui diversi piatti prende il nome di **cilindro**
- Un braccio mobile supporta una testina di lettura e scrittura per ogni piatto

Schema funzionale di un disco



Schema funzionale di un disco



Struttura dei dischi - 1



- Un disco fisso può essere visto come un array unidimensionale di blocchi logici di 512 byte che corrisponde alla più piccola unità di trasferimento
 - Ma potrebbe essere anche formattato a basso livello per ridimensionare la dimensione dei blocchi
- L'array monodimensionale corrisponde in modo sequenziale ai blocchi del disco
 - Ogni settore contiene un blocco logico
 - Il settore 0 è il primo settore della traccia più esterna del primo piatto (di solito il più in alto della pila)
 - Consecutivamente si numerano gli altri settori della traccia, i settori delle tracce più interne, e si procede con la numerazione allo stesso modo nei piatti inferiori

Struttura dei dischi - 2



- Per tradurre un numero di blocco in un punto preciso del disco è necessario determinare
 - Il numero di cilindro
 - Il numero di piatto all'interno del cilindro (n. di cilindro e n. di piatto identificano insieme una ben precisa traccia)
 - Il numero di settore nella traccia
- La conversione può risultare complicata per via dei difetti di fabbricazione e delle diverse lunghezze delle tracce

Struttura dei dischi - 3



- Solitamente quando i dischi presentano settori difettosi dei meccanismi interni li nascondono attraverso la mappatura dei blocchi logici sui settori fisici sul disco
- Per ragioni geometriche il numero dei settori per traccia non è costante
 - Una traccia più è lontana dal centro del disco, più è lunga, e quindi maggiore è il numero di settori che può contenere
 - La densità dei bit aumenta sulle tracce esterne fino al 40% in più di una traccia interna
 - La velocità di rotazione aumenta mano a mano che le testine accedono ai settori interni per mantenere costante la quantità di dati trasferiti

Tempo di accesso al disco



- Il Sistema Operativo è responsabile dell'utilizzo efficiente dei dischi fissi installati minimizzando i tempi di accesso e massimizzando la quantità di dati trasferiti
- Il tempo di accesso ad un settore (e quindi blocco) del disco dipende da due componenti principali
 - **Seek time (tempo di posizionamento)**: il tempo impiegato per muovere le testine sul cilindro desiderato
 - **Rotational latency (latenza rotazionale)**: il tempo richiesto perché il disco ruotando porti il settore interessato sotto la testina magnetica

Scheduling del disco - 1



- Se l'unità a disco è disponibile allora una richiesta di lettura/scrittura può essere soddisfatta immediatamente
- Se l'unità a disco è impegnato a servire una richiesta ogni nuova richiesta viene inserita in una coda di richieste pendenti
- Il Sistema Operativo ha il compito di decidere quale richiesta deve essere servita cercando di migliorare il tempo di accesso e l'ampiezza di banda
- Gli algoritmi di scheduling del disco determinano la prossima richiesta da soddisfare

Scheduling del disco - 2



- La latenza rotazionale non potrà essere influenzata dal Sistema Operativo ma solo dalle specifiche hardware
 - In media è pari a $\frac{1}{2}$ del tempo necessario a compiere una rotazione completa
- Il Sistema Operativo può influire sul seek time medio complessivo cercando di ordinare le richieste in modo tale che le testine si spostino il meno possibile

Algoritmi di scheduling

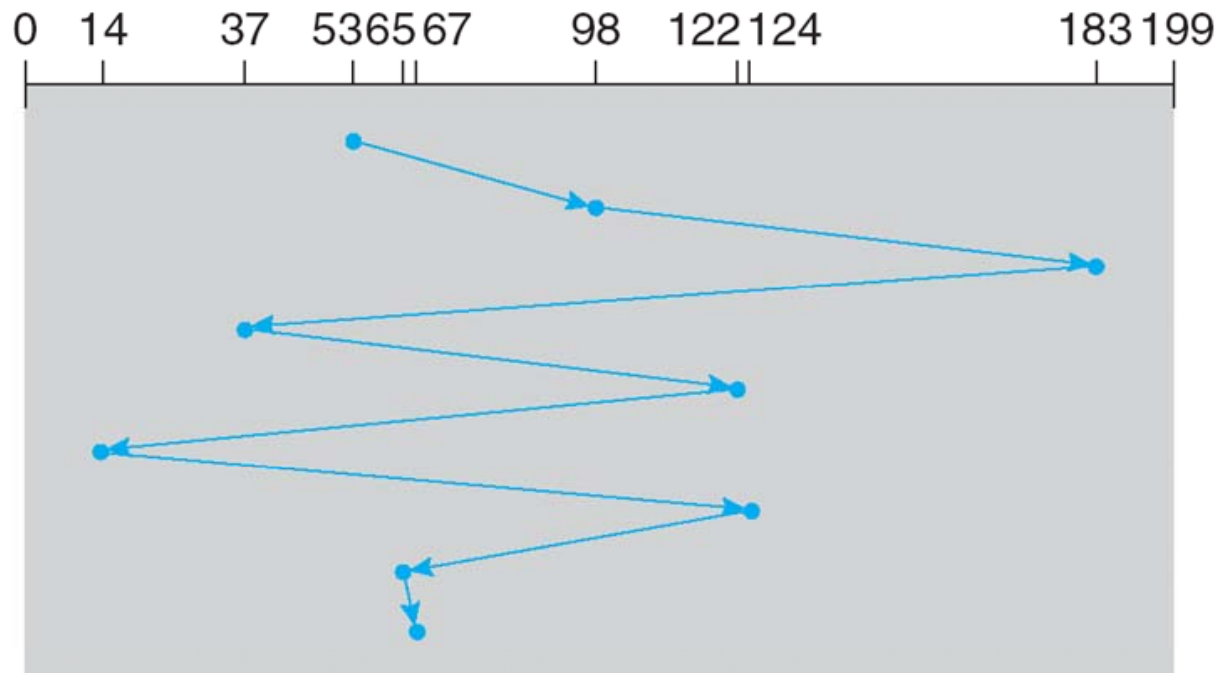


- Algoritmi di scheduling per le richieste di I/O del disco
 - Scheduling in ordine d'arrivo – FCFS
 - Scheduling per brevità – SSTF
 - Scheduling per scansione - SCAN
 - Scheduling per scansione circolare – C-SCAN
- Utilizziamo come esempio la seguente coda di richieste
98, 183, 37, 122, 14, 124, 65, 67
e supponiamo che inizialmente la testina sia posizionata
sul cilindro 53

Scheduling in ordine d'arrivo - FCFS



queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53



- Lo spostamento totale della testina è di 640 cilindri
- Notiamo che lo spostamento “122 – 14 – 124” è un salto evitabile...non era meglio fare “122 – 124 – 14” ?

Scheduling per brevità - SSTF



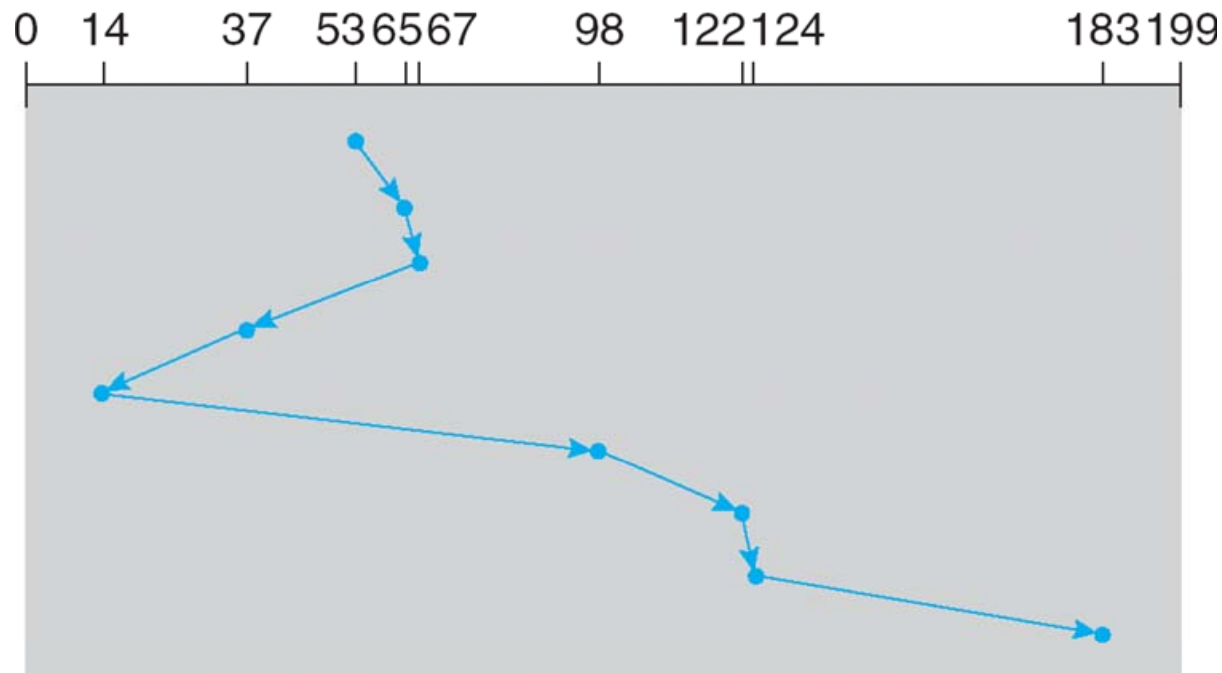
- Nello scheduling per brevità scegliamo la richiesta secondo il più breve tempo di ricerca ovvero con il tempo minimo rispetto alla posizione della testina
- SSTF è una variante dello scheduling SJF
 - Può causare starvation su alcune richieste
- Le prestazioni sono migliori rispetto al scheduling FCFS ma non necessariamente ottimali
 - Le richieste sono in coda quindi non è necessario fare previsioni sul futuro

Scheduling per brevità - SSTF



queue = 98, 183, 37, 122, 14, 124, 65, 67

head starts at 53



- Lo spostamento totale della testina è di 236 cilindri
- La sequenza "... 53, 37, 14, 65, 67, 98, ..." da uno spostamento di 208

Scheduling per scansione - SCAN



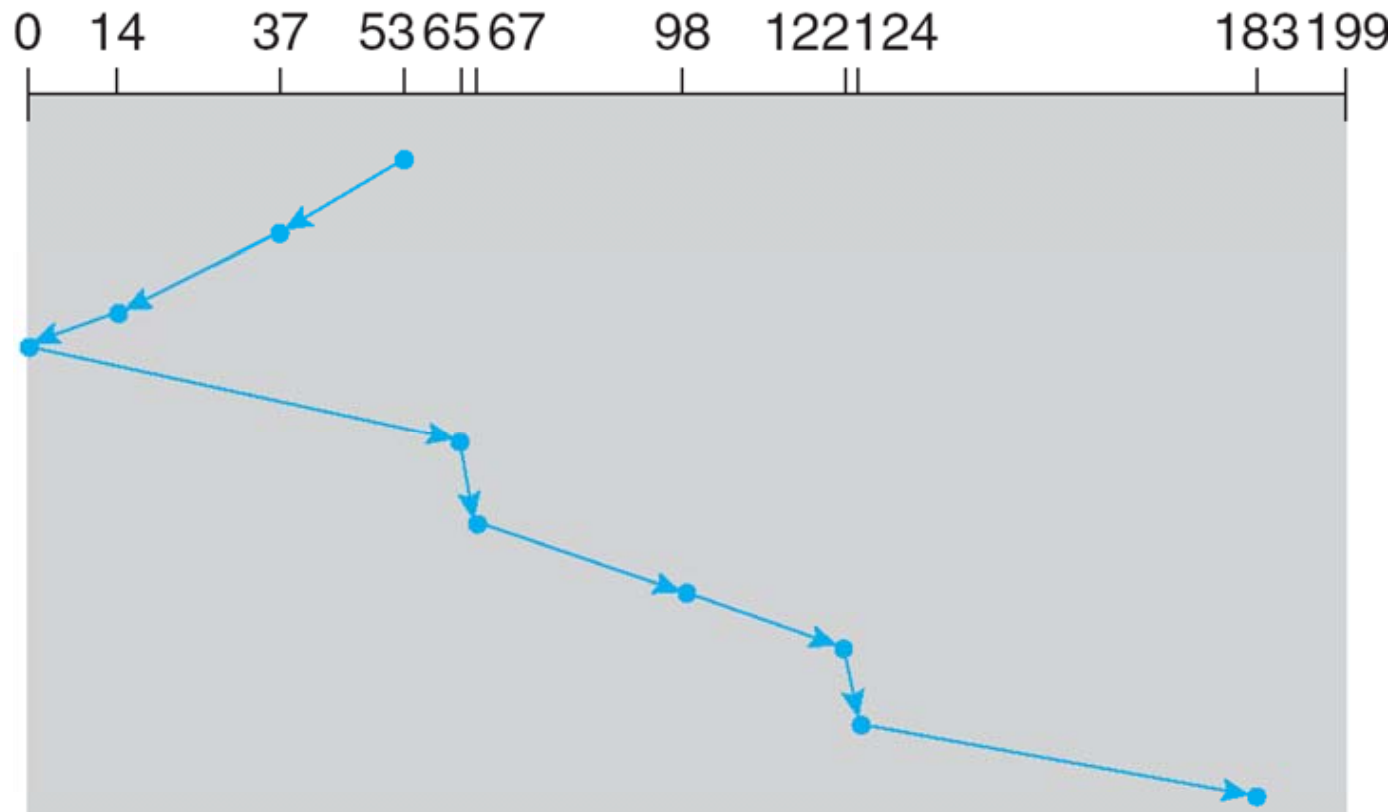
- La testina si muove da un estremo all'altro del disco, servendo le richieste
- Quando raggiunge l'estremità del disco torna indietro servendo le richieste che incontra
- L'algoritmo è chiamato anche **algoritmo dell'ascensore**

Scheduling per scansione - SCAN



queue = 98, 183, 37, 122, 14, 124, 65, 67

head starts at 53



Scheduling del disco per scansione circolare – C-SCAN

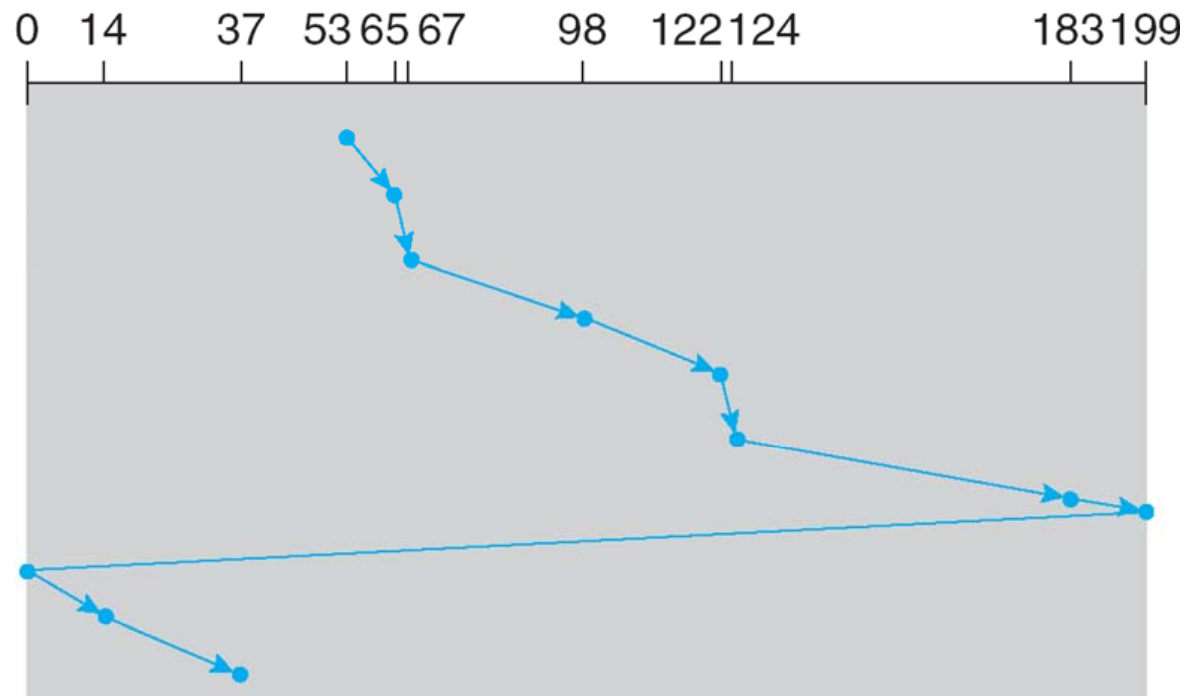


- La testina si muove da un estremo all'altro del disco, servendo le richieste
- Quando raggiunge l'estremità del disco, torna immediatamente all'inizio senza servire richieste
 - Tratta i cilindri come una lista circolare
- Fornisce un tempo di attesa, per le varie richieste, più uniforme di altri algoritmi, anche se non riesce a garantire un tempo medio di attesa minimo

Scheduling del disco per scansione circolare – C-SCAN



queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53



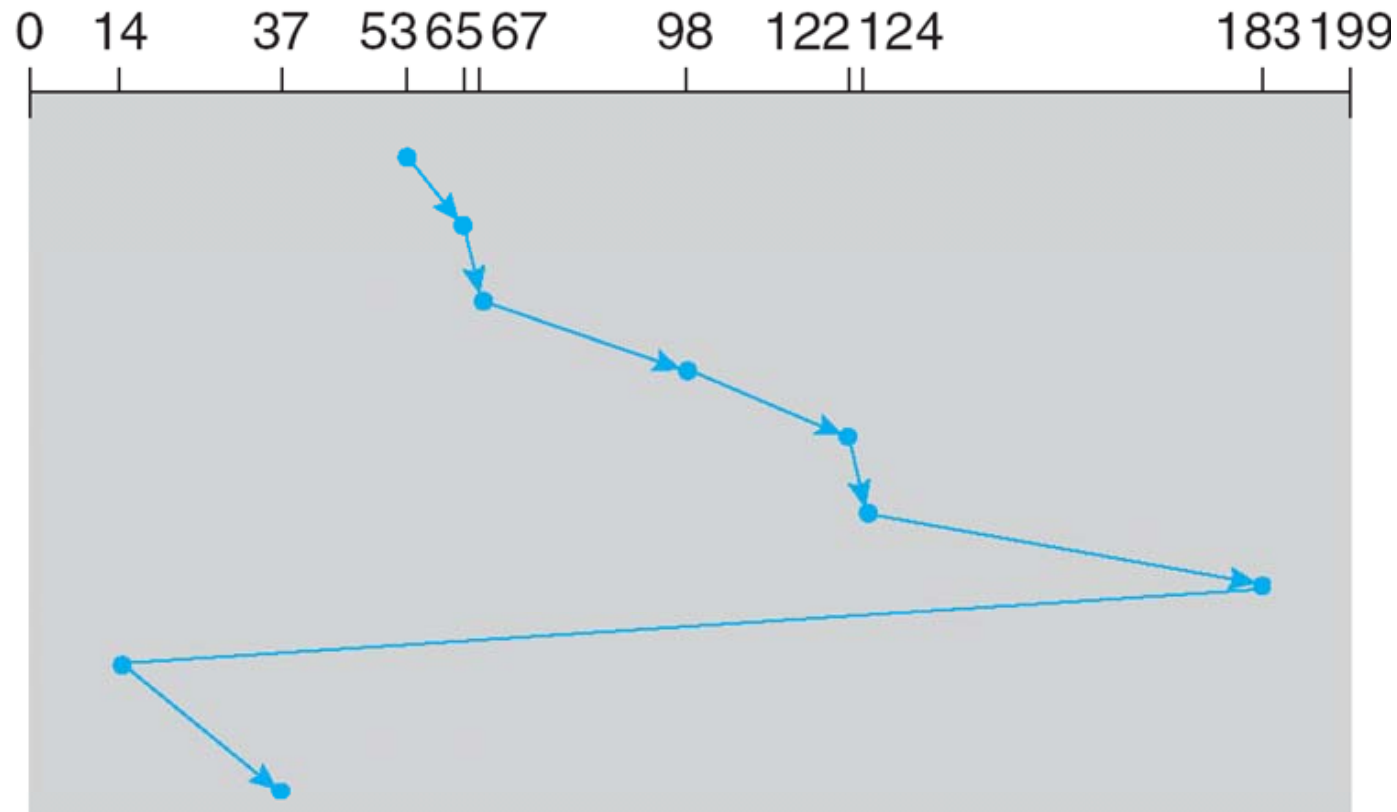
- Lo spostamento totale della testina è di 183 (+ 200) cilindri

Scheduling LOOK



queue 98, 183, 37, 122, 14, 124, 65, 67

head starts at 53



Considerazioni sugli algoritmi di scheduling



- SSTF è il più comune e sembra essere quello più naturale
- SCAN e C-SCAN hanno le migliori prestazioni nei sistemi con un alto carico di richieste da soddisfare
- In generale le performance dipendono dal numero e dal tipo di richieste
 - Anche il tipo di file system può influire sulle performance dello scheduling
- L'algoritmo di scheduling del disco dovrebbe essere implementato come un modulo separata del Sistema Operativo in modo da poterlo rimpiazzare con un algoritmo differente se necessario
- Sia SSTF che LOOK sono una scelta ragionevole come algoritmo di default

Gestione dell'unità a disco



- Il Sistema Operativo è ancora responsabile
 - Formattazione del disco
 - Blocco d'avviamento
 - Blocchi difettosi

Formattazione del disco - 1



- Un disco prima di poter essere usato deve subire un processo di **formattazione a basso livello**
- La formattazione a basso livello normalmente è affidata al costruttore e permette di:
 - Associare ad ogni settore il suo numero
 - Prevedere uno spazio per inserire un codice di correzione degli errori, usato in ogni operazione di I/O su quel settore
- Durante la formattazione a basso livello è anche possibile scegliere la dimensione dei blocchi fisici
 - il valore standard è di 512 byte per settore

Formattazione del disco - 2



- Dopo la formattazione a basso livello il Sistema Operativo è in grado di effettuare una formattazione logica necessaria per creare e gestire il file system
- Il Sistema Operativo crea una lista di blocchi liberi (secondo lo schema adottato), e una directory iniziale
- Sull'HD vengono poi riservate le aree che dovranno essere gestite direttamente dal SO:
 - Il boot block (che può anche essere vuoto)
 - L'area che contiene gli attributi dei file (ad esempio, gli inode Unix o la MFT di Windows XP)

Blocco d'avviamento



- Il **blocco d'avviamento** (o *boot block*) contiene il codice necessario per far partire il Sistema Operativo
- All'accensione, un piccolo programma contenuto in ROM istruisce il disk controller in modo da trasferire il contenuto del Boot Block in RAM
- Il controllo è trasferito al codice del boot block, che si occupa di far partire l'intero Sistema Operativo prelevando in codice dal disco stesso

Blocchi difettosi



- Le unità a disco sono portate a malfunzionamenti per via delle parti mobili
 - Alcuni dischi fissi in commercio possiedono già dei blocchi difettosi
- Solitamente il Sistema Operativo quando trova un blocco difettoso lo marca in modo da impedire che venga utilizzato in fase di allocazione
- I dischi fissi internamente adottano strategie automatiche per la gestione dei blocchi difettosi come l'**accantonamento dei settori**
 - Durante la formattazione a basso livello alcuni blocchi vengono riservati come settori di riserva in modo da utilizzarli quando un settore diventa difettoso
- In generale la gestione dei blocchi difettosi non è mai una operazione completamente automatica

Gestione dell'area di swap - 1



- Durante la formattazione logica del disco fisso, il Sistema Operativo riserva a se stesso uno spazio da usare come area di swap
- La soluzione più semplice sarebbe quella di utilizzare un file molto grande all'interno del file system
- Questa soluzione è però inefficiente, dato che occorre passare attraverso le strutture di gestione del FS per ogni accesso all'area di Swap
 - Vmware adotta questa soluzione per simulare un disco fisso attraverso un file

Gestione dell'area di swap - 2



- Solitamente si utilizza una partizione specifica del disco
- Questa partizione non viene trattata allo stesso modo del FS ma vengono usate strategie di allocazione diverse per migliorare al massimo la velocità d'uso
 - Ad esempio, l'allocazione dei blocchi per le pagine swappate può essere contigua, in modo da non dover gestire un meccanismo di ricerca dei blocchi liberi

Strutture RAID



- La disponibilità di dischi di grandi capacità e poco costosi permette di equipaggiare un sistema con molti dischi
- La presenza di più dischi aumenta però la probabilità di guasti
- L'uso di più dischi rende possibile migliorare l'affidabilità della memoria secondaria in caso di guasto
- Organizzando i dischi mediante **batterie ridondanti di dischi** (*redundant array of independent [inexpensive] disks* - RAID) è possibile ottenere un aumento dell'affidabilità e delle prestazioni

Miglioramento dell'affidabilità



- Il modo più semplice per aumentare l'affidabilità è attraverso la **copiatura speculare** (*mirroring* o *shadowing*)
- Ogni disco logico contiene due dischi fisici (il Sistema Operativo vede sempre un solo disco) ed ogni scrittura si esegue su entrambi
 - Se un disco si guasta i dati possono essere letti dall'altra disco
- Solitamente si presuppone che i guasti siano indipendenti sui dischi ma cali di tensione o disastri naturali possono danneggiare tutti i dischi
 - Anche una batteria di dischi proveniente dalla stesso fornitore potrebbe essere difettosa

Miglioramento delle prestazioni



- L'aumento delle prestazioni può essere ottenuto attraverso un accesso parallelo ai dischi
 - Con due dischi la velocità di lettura dal singolo disco è la stessa ma il numero di lettura per unità di tempo raddoppia
- Utilizzando il **sezionamento dei dati** (*data striping*) ogni bit di un byte è distribuito su più dischi
 - Se il sistema utilizza una batteria di otto dischi ogni bit viene scritto su un disco differente (**sezionamento a livello dei bit**)
 - Sono possibili anche sezionamento con un numero di dischi multipli di otto
- Nel **sezionamento a livello dei blocchi** i blocchi di un file si distribuiscono su più dischi
- La batteria dei dischi è sempre vista dal Sistema Operativo come un unico disco logico

Livelli RAID



- La copiatura speculare aumenta l'affidabilità ma è costosa mentre il sezionamento aumenta le prestazioni ma non è affidabile
- I **livelli RAID** permettono di ottenere il meglio delle due tecniche in base alle esigenze del proprio sistema di calcolo
- L'idea è di utilizzare più dischi alcuni dei quali possono essere scelti per mantenere una copia speculare ed altri per mantenere i bit di correzione

Livelli RAID



(a) RAID 0: non-redundant striping.



(b) RAID 1: mirrored disks.



(c) RAID 2: memory-style error-correcting codes.



(d) RAID 3: bit-interleaved parity.



(e) RAID 4: block-interleaved parity.



(f) RAID 5: block-interleaved distributed parity.



(g) RAID 6: P + Q redundancy.

- P indica i bit di correzione
- C indica una seconda copia dei dati