

Administración de E/S - Discos

Explicación de práctica 6

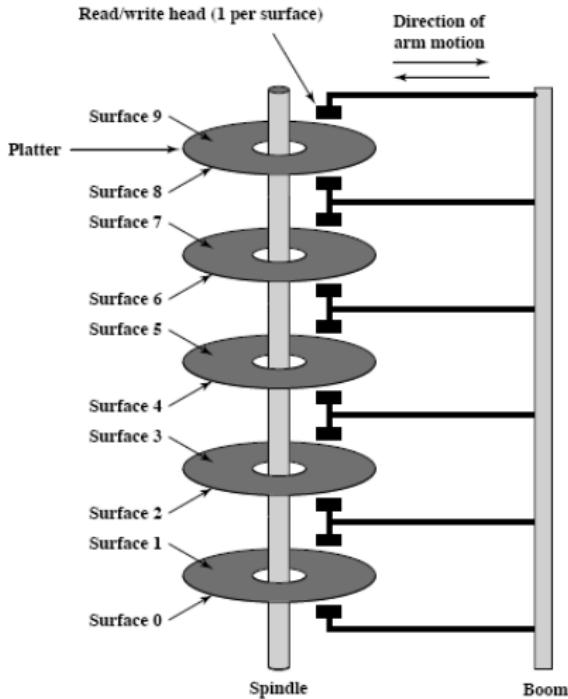
Introducción a los Sistemas Operativos

Facultad de Informática
Universidad Nacional de La Plata

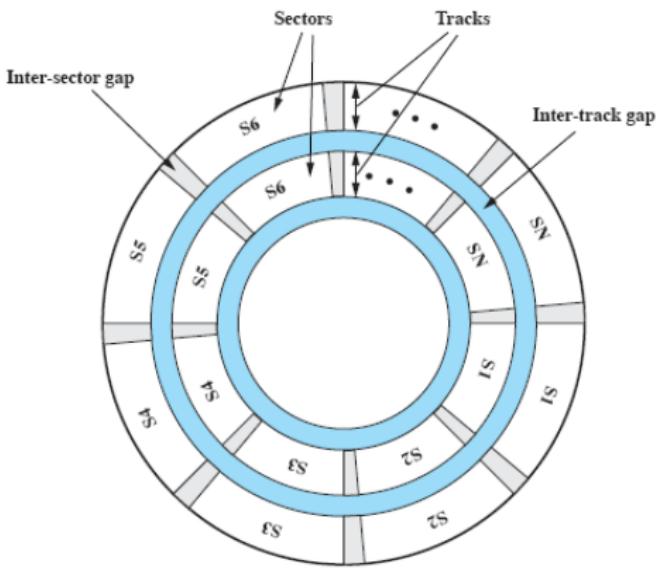
2023



Organización física de un HDD

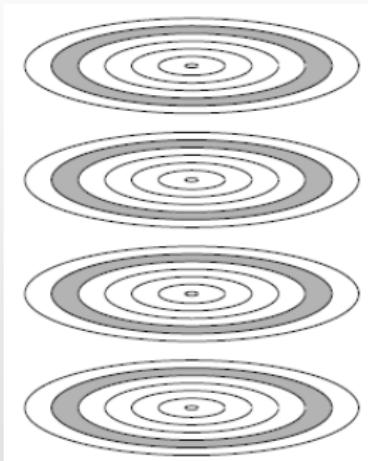


Organización física de un HDD (cont.)



Organización física de un HDD (cont.)

- Cilindro N: todas las n-esimas pistas de todas las caras



- La capacidad de un disco esta dada por el producto de:
 - Cantidad de caras: W
 - Cantidad de pistas: X
 - Cantidad de sectores por pista: Y
 - Tamaño de sector: Z

$$\text{capacidad} = W * X * Y * Z$$

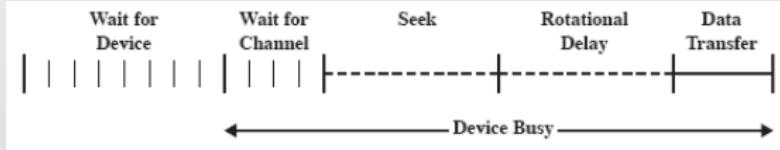


- Para realizar una *E/S*, por ejemplo un acceso a disco, se requiere de una llamada al sistema (*System Call*). En la misma se especifica:
 - Tipo de operación (*E* o *S*)
 - Dirección en disco para la transferencia (file descriptor que se obtuvo al abrir un archivo)
 - Dirección en memoria para la transferencia (de donde se lee o escribe)
 - Número de bytes a transferir
- Este requerimiento es pasado, por el *kernel*, al subsistema de *E/S* quien lo traduce en: (#Cara, #Cilindro, #Sector)

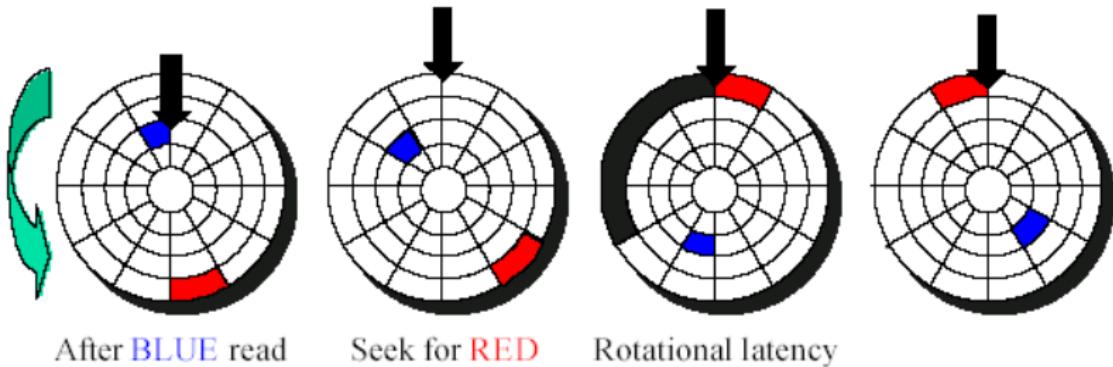


Tiempo de acceso a un HDD

- El tiempo de acceso esta dado por:
 - **Seek time** (posicionamiento): tiempo que tarda en posicionarse la cabeza en el cilindro
 - **Latency time** (latencia): tiempo que sucede desde que la cabeza se posiciona en el cilindro hasta que el sector en cuestión pasa por debajo de la misma
 - **Transfer time** (transferencia): tiempo de transferencia del sector (bloque) del disco a la memoria



Tiempo de acceso a un HDD (cont.)



Tiempo de acceso a un HDD (cont.)

- *Latency time* → si este tiempo no se conoce, se considera que es igual a lo que tarda el disco en dar media vuelta
- Ejemplo. Disco de 5400 RPM →

5400 vueltas → 1' = 60" = 60000 ms

1/2 vuelta → x = 5,5 ms



- **Almacenamiento secuencial:**

$$\text{seek} + \text{latency} + (\text{tiempo_transferencia_bloque} * \#\text{bloques})$$

- **Almacenamiento aleatorio:**

$$(\text{seek} + \text{latency} + \text{tiempo_transferencia_bloque}) * \#\text{bloques}$$


- Prefijos: nos permiten representar números largos de manera más reducida
- Prefijos binarios:
 - Nos permiten crear múltiplos binarios (basados en potencias de 2)
 - Son similares, en concepto, aunque difieren en valor a los prefijos del *Sistema Internacional (SI)* basados en potencias de 10
 - En la práctica se adopta el sistema de prefijos binarios



Prefijos - Equivalencias

Unidades básicas de información (en bytes)				
Prefijos del Sistema Internacional			Prefijo binario	
Múltiplo - (Símbolo)	Estándar SI	Binari o	Múltiplo - (Símbolo)	Valor
kilobyte (kB)	10^3	2^{10}	kibibyte (KiB)	2^{10}
megabyte (MB)	10^6	2^{20}	mebibyte (MiB)	2^{20}
gigabyte (GB)	10^9	2^{30}	gibibyte (GiB)	2^{30}
terabyte (TB)	10^{12}	2^{40}	tebibyte (TiB)	2^{40}



Capacidad de un HDD - Ejemplo

- Supongamos un disco con 6 platos, 2 caras útiles, 1500 pistas por cara y 700 sectores por pista de 256 bytes cada uno
- Si queremos calcular la capacidad total del disco, hacemos:

$$\text{tamaño_disco} = \#\text{caras} * \#\text{pistas_cara} * \#\text{sectores_pista} * \\ \text{tamaño_sector}$$

$$(6 * 2) * 1500 * 700 * 256 \text{ bytes} = 3225600000 \text{ bytes} \\ = 3,00407 \text{ GiB(Gibibytes)}$$



Ocupación sobre un HDD - Ejemplo

- Supongamos un disco con 6 platos, 2 caras útiles, 1500 pistas por cara y 700 sectores por pista de 256 bytes cada uno
- Si queremos cuantas caras ocupará un archivo de 513 Mibytes almacenado de manera contigua a partir del primer sector de la primera pista de una cara determinada:
 - Calculamos la capacidad de 1 cara:
 $1500 * 700 * 256 \text{ bytes} = 268800000 \text{ bytes}$
 - Dividimos el tamaño del archivo por la capacidad de una cara:
 $513 \text{ MiB} = 537919488 \text{ bytes}$
 $537919488 / 268800000 = 2,00118 \rightarrow 3 \text{ caras}$



Tiempo de acceso a un HDD - Ejemplo

- Supongamos un disco con 6 platos, 2 caras útiles, 1500 pistas por cara y 700 sectores por pista de 256 bytes cada uno. El disco gira a 12600 RPM , tiene un tiempo de posicionamiento (seek) de 2 milisegundos y una velocidad de transferencia de 15 Mib/s (Mebibits por segundo)
- Si queremos saber cuantos milisegundos se tardarían en transferir un archivo **almacenado de manera contigua y aleatoria** de 4500 sectores



Tiempo de acceso a un HDD - Ejemplo (cont.)

- Calculamos los datos que faltan:

- Latencia:

$$12600 \text{ vueltas} \rightarrow 1' = 60 \text{ s} = 60000 \text{ ms}$$

$$0,5 \text{ vueltas} \rightarrow x = 2,3809 \text{ ms}$$

- Transferencia:

$$15 \text{ Mibits} \rightarrow 1 \text{ s} = 1000 \text{ ms}$$

$$256 \text{ bytes} \rightarrow x$$

Unificamos unidades:

$$15728640 \text{ bits} \rightarrow 1000 \text{ ms}$$

$$2048 \text{ bits} \rightarrow x = 0,1302 \text{ ms}$$



Tiempo de acceso a un HDD - Ejemplo (cont.)

- Datos obtenidos:
 - Seek time: 2 ms
 - Latency time: 2,3809 ms
 - Tiempo transferencia bloque: 0,1302 ms
 - #bloques: 4500 → eventualmente se tienen que calcular
- Resultados:
 - Almacenamiento secuencial:
 $\text{seek} + \text{latency} + \text{tiempo_transferencia_bloque} * \# \text{bloques}$
 $2 + 2,3809 + 0,1302 * 4500 = 590,2809 \text{ ms}$
 - Almacenamiento aleatorio:
 $(\text{seek} + \text{latency} + \text{tiempo_transferencia_bloque}) * \# \text{bloques}$
 $(2 + 2,3809 + 0,1302) * 4500 = 20299,95 \text{ ms}$



Planificación de requerimientos de un HDD

- Seek time → parámetro que más influye en el tiempo de acceso al disco
- El sistema operativo:
 - Es responsable de utilizar el hardware en forma eficiente. Para los discos, esto significa obtener el menor tiempo de atención de los requerimientos
 - Debe por lo tanto minimizar el tiempo de seek → implica menor distancia de recorrido por el brazo



Algoritmos de planificación en un HDD

- Objetivo: minimizar el movimiento de la cabeza
- Como: ordenando lógicamente los requerimientos pendientes (*que estan en la cola*) al disco, considerando el número de cilindro de cada requerimiento. En cualquier momento se pueden encolar nuevos movimientos
- La atención de requerimientos a pistas duplicadas se resuelven según el algoritmo de planificación:
 - **FCFS**: se atienden de manera separada (tantas veces como se requieran). Por ejemplo, si tengo {10, 40, 70, 10}, al 10 lo atiendo 2 veces
 - **SSTF/SCAN/LOOK/C-SCAN/C-LOOK**: se atienden de manera consecutiva



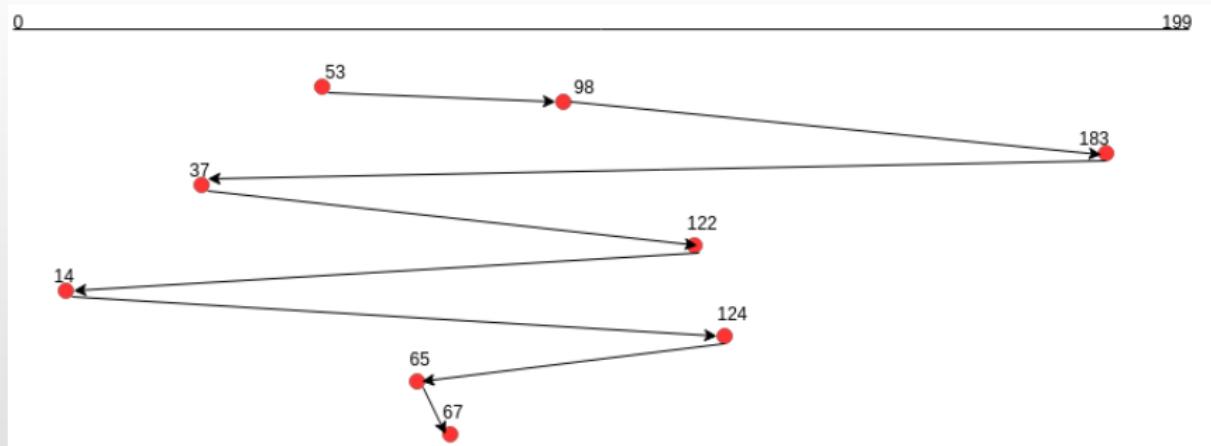
Algoritmos - Ejemplo de enunciado sin page faults

- Cantidad de pistas: 200 (0..199)
- Requerimientos en la cola: {98 , 183 , 37, 122, 14, 124, 65, 67}
- Viene de: pista 61
- Ubicación actual del cabezal: pista 53 → derecha-izquierda



First Come First Served

- **FCFS**: atiende los requerimientos por orden de llegada

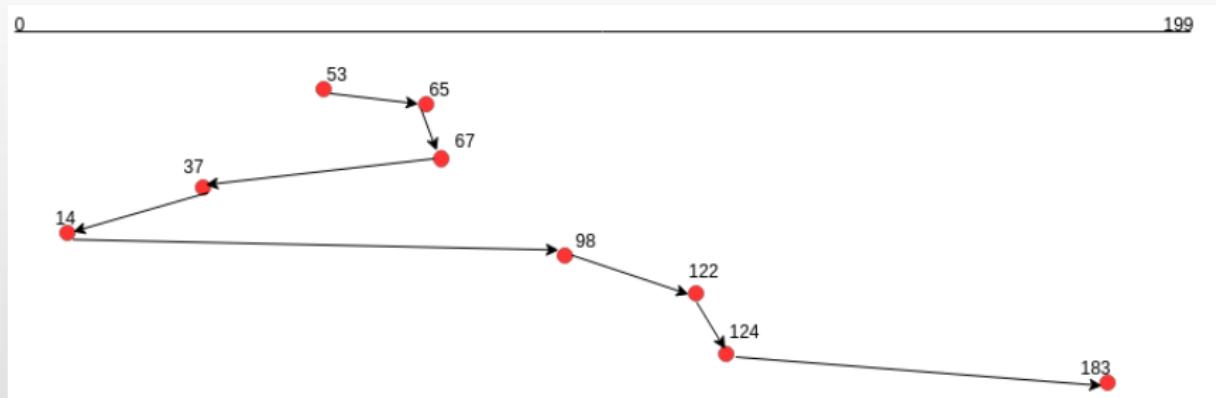


Movimientos: 640

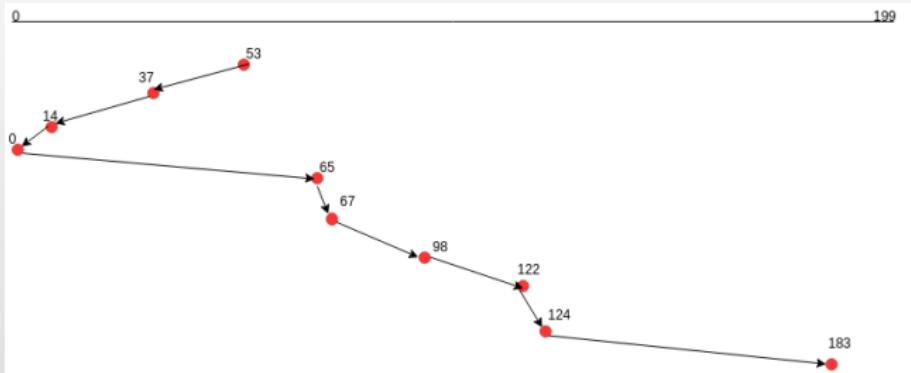


Sortest Seek Time First

- **SSTF**: selecciona el requerimiento que requiere el menor movimiento del cabezal



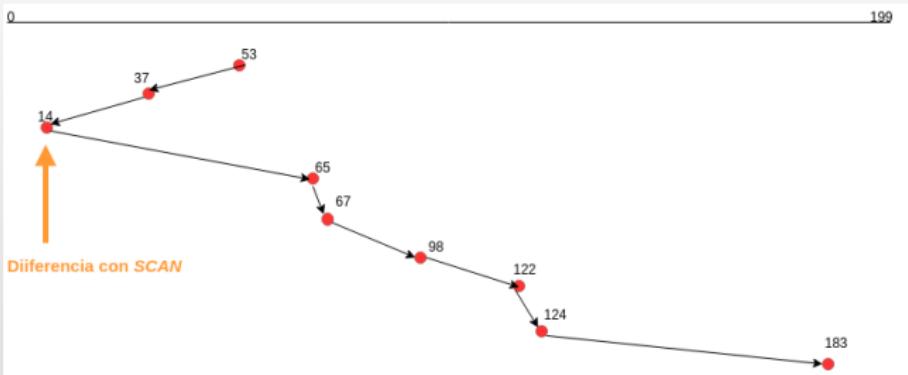
- **SCAN:** barre el disco en una dirección atendiendo los requerimientos pendientes en esa ruta hasta llegar a la última pista del disco y cambia la dirección. Es **importante** saber en que pista se **está** y de que pista se **viene** para determinar el sentido del cabezal



Movimientos: 236



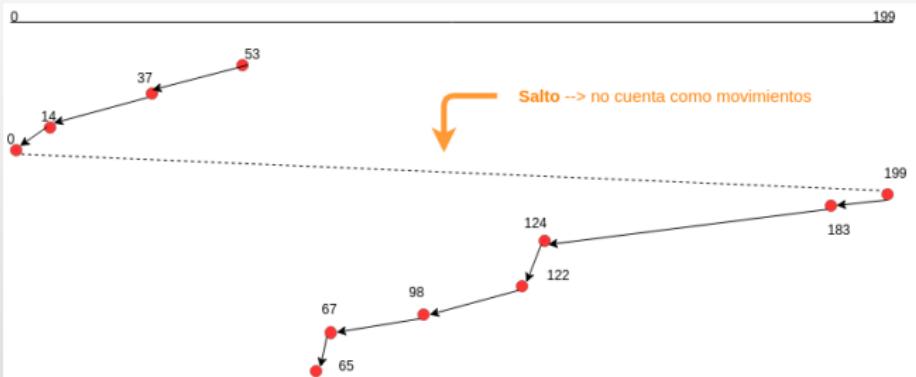
- **LOOK:** se comporta igual que el *SCAN* pero no llega hasta la última pista del disco sobre la dirección actual sino que llega hasta el último requerimiento de la dirección actual. Es **importante** saber en qué pista se **está** y de qué pista se **viene** para determinar el sentido del cabezal



Movimientos: 208

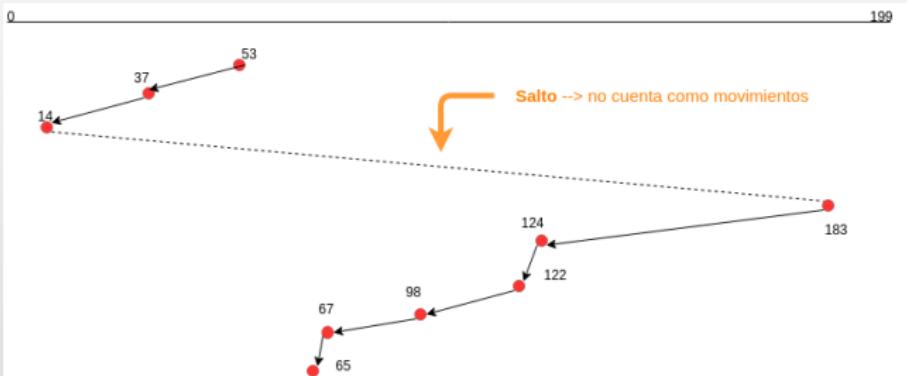


- **C-SCAN:** se comporta igual que el **SCAN** pero restringe la atención en un solo sentido. Al llegar a la última pista del disco en el sentido actual vuelve a la pista del otro extremo (**salto** → no se cuentan los movimientos) y sigue barriendo en el mismo sentido



Circular LOOK

- **C-LOOK:** se comporta igual que el *LOOK* pero restringe la atención en un solo sentido. Al llegar a la última pista de los requerimientos en el sentido actual vuelve a la primer pista más lejana del otro extremo (**salto** → no se cuentan los movimientos) y sigue barriendo en el mismo sentido



Movimientos: 157



- Existen requerimientos especiales que deben atenderse con urgencia. Los *fallos de página* indican simplemente que tienen mayor prioridad con respecto a los requerimientos convencionales, por lo tanto deben ser atendidos inmediatamente después del requerimiento que se está atendiendo actualmente
- La lógica de atención de múltiples *PF* se maneja según el algoritmo de planificación. Ejemplos:
 - **FCFS**: Si tengo {10, 40PF, 70PF, 10}, primero se atiende al 40PF y luego al 70PF
 - **SSTF**: si tengo {10, 40PF, 70PF, 10} y estoy en la pista 65, primero atiendo al 70PF y luego al 40PF
- En todos los algoritmos, los movimientos utilizados para atender estos requerimientos especiales deben ser contados



Algoritmos - Atención de PF (cont.)

- Una vez que no existan más requerimientos por *page faults* en la cola, se procede:
 - **FCFS**: en orden *FCFS*
 - **SSTF**: en orden *SSTF*
 - **SCAN**: con el sentido que determina la atención de los últimos dos requerimientos → puede cambiar de sentido
 - **C-SCAN**: con el sentido original → el sentido no cambia
 - **LOOK**: del mismo modo en que lo hace el *SCAN*
 - **C-LOOK**: del mismo modo en que lo hace el *C-SCAN*

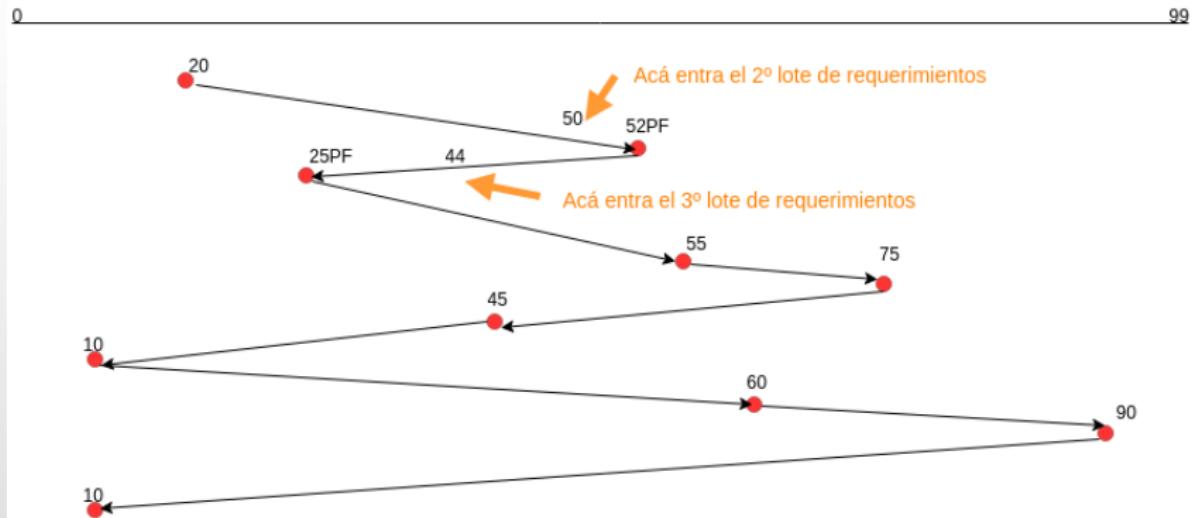


Algoritmos - Ejemplo de enunciado con page faults

- Cantidad de pistas: 100 (0..99)
- Requerimientos en la cola: {55 , 75 , 52^{PF} , 45, 10}. Luego de 30 movimientos $\{25^{PF}, 60\}$ y luego de 10 movimientos más (40 desde el comienzo de la planificación) entra {90, 10}
- Se viene de la pista 15
- Se está atendiendo la pista 20 → izquierda-derecha



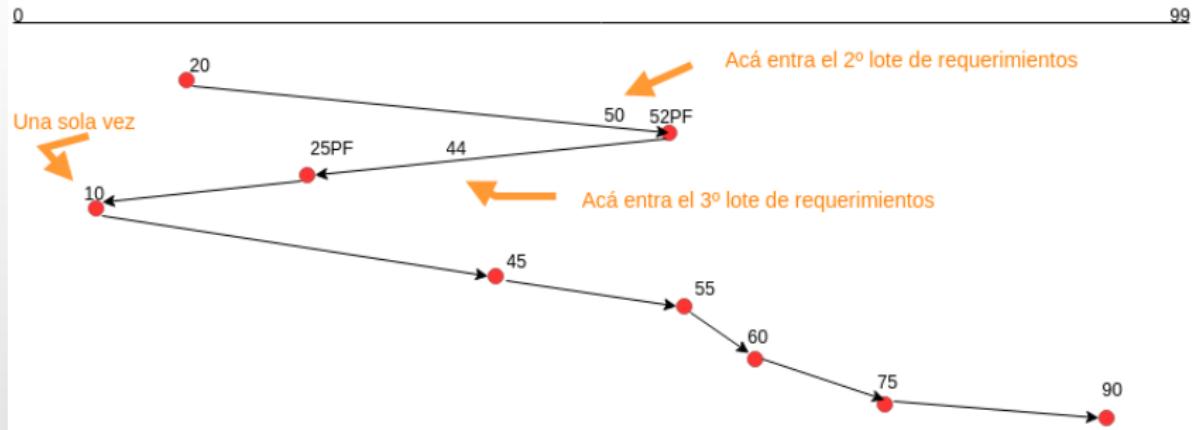
First Come First Served



Movimientos: 334

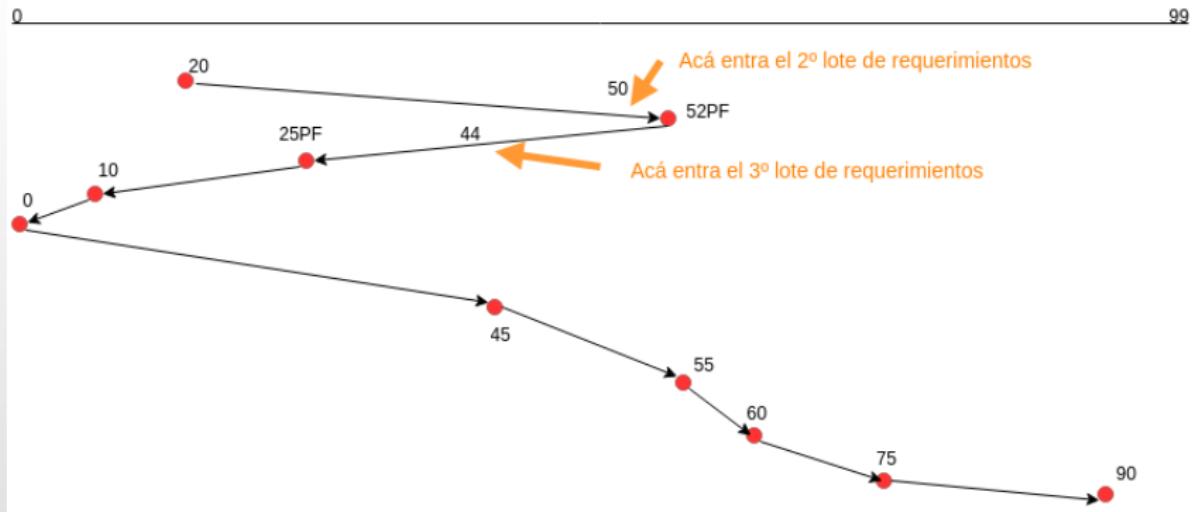


Sortest Seek Time First



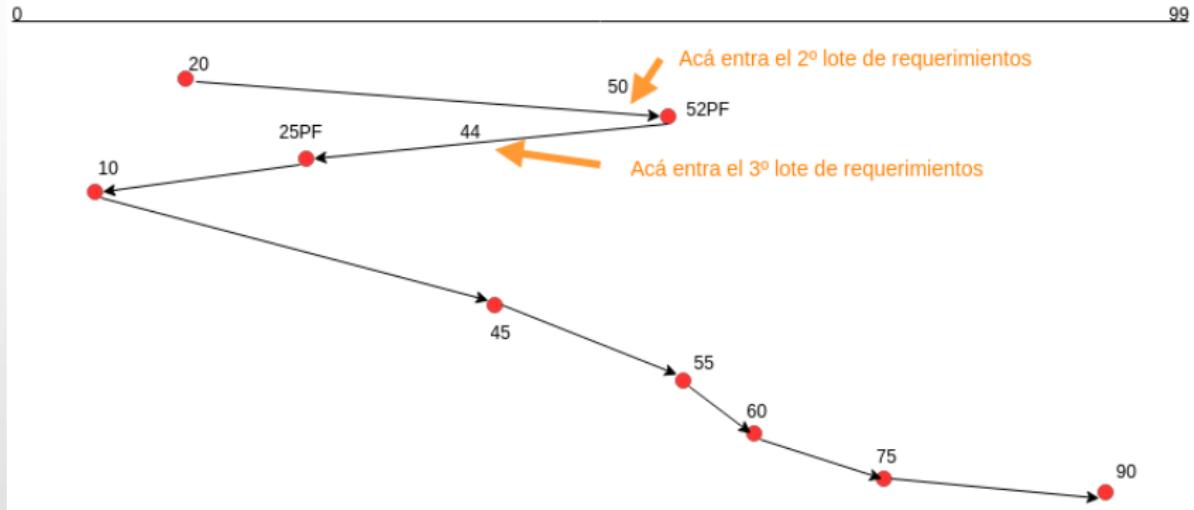
Movimientos: 154





Movimientos: 174

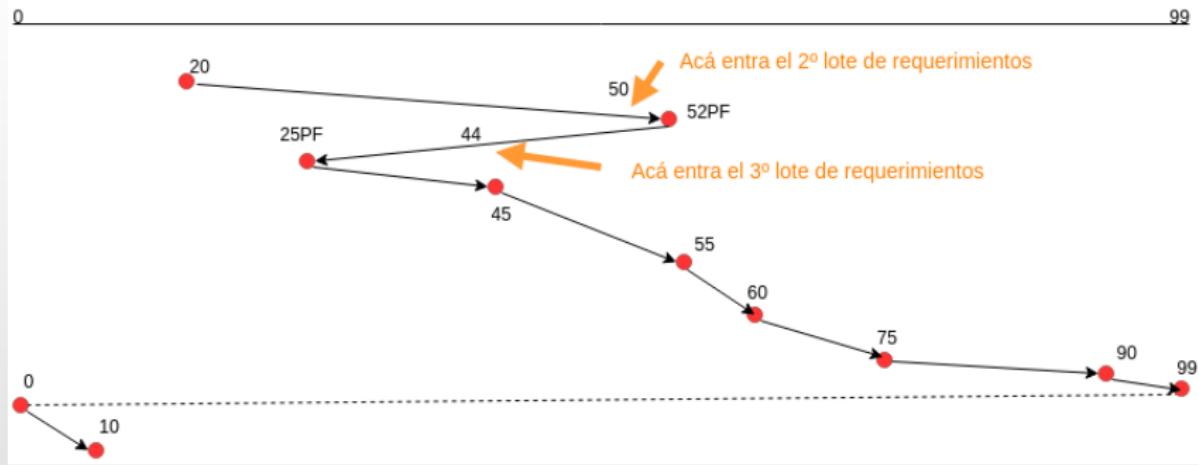




Movimientos: 154



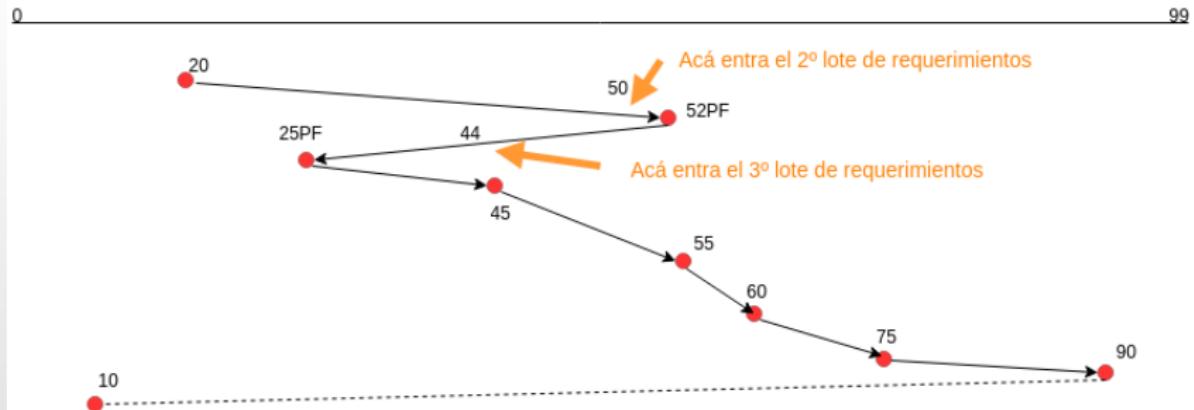
Circular SCAN



Movimientos: 143



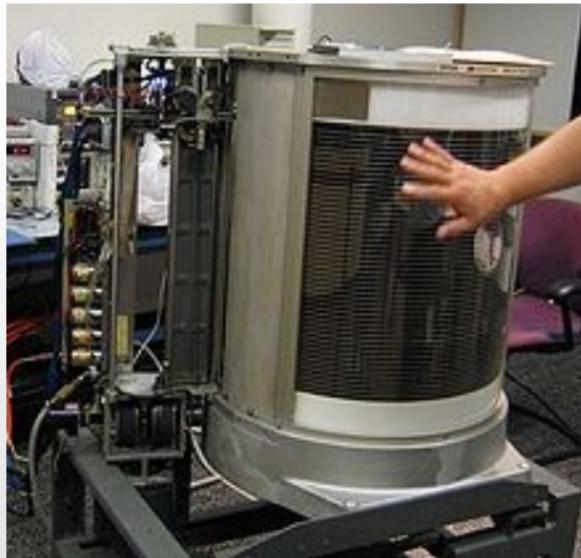
Circular LOOK



Movimientos: 124



Imágenes de discos en la historia



Imágenes de discos en la historia (cont.)



Imágenes de discos en la historia (cont.)



¿Preguntas?

