Sistemas Operativos

Planificación de Disco

Planificación de Disco

El tiempo de lectura/escritura de un sector del disco depende de tres factores:

- Tiempo de búsqueda del cilindro.
- Retardo rotacional.
- Tiempo de transferencia.

El tiempo de búsqueda del cilindro es el único que se puede optimizar desde el programa gestor del disco, los otros dependen de las características propias del disco.

Cuando un programa requiere una operación de E/S del disco envía la siguiente información.

- Tipo de operación. (entrada o salida)
- Dirección en el disco (unidad, cilindro, superficie, bloque)
- Dirección en memoria
- Cantidad de información a transferir (numero de bytes)

Planificación de Disco

Planificación FCFS: Primero en llegar primero en ser atendido.

- Este método es la forma mas sencilla de gestionar la búsqueda del sector.
- La solicitud se almacena en una memoria tipo FIFO, de manera que la primera petición que llega es la primera que se atiende.
- Es sencillo de programar y se puede considerar inherentemente justo.
- No ofrece el mejor tiempo de servicio.

Ejemplo:

Un disco con 200 pistas tiene las siguientes peticiones de pistas: 22, 124, 105, 181, 142, 36, 5, 59, 115.

La posición inicial de la cabeza lectura/escritura esta en la pista 95 y se moverá a la 22, luego a la 124 y asi sucesivamente, atendiendo por el orden de llegada todas las peticiones que se encuentran e la cola

Planificación de Disco

Planificación FCFS: Primero en llegar primero en ser atendido.

Proxima pista a la que se accede	22	124	105	181	142	36	5	59	115	
Numero de pistas que se atraviesan	73	102	19	76	39	106	31	54	56	LMB = 61.8

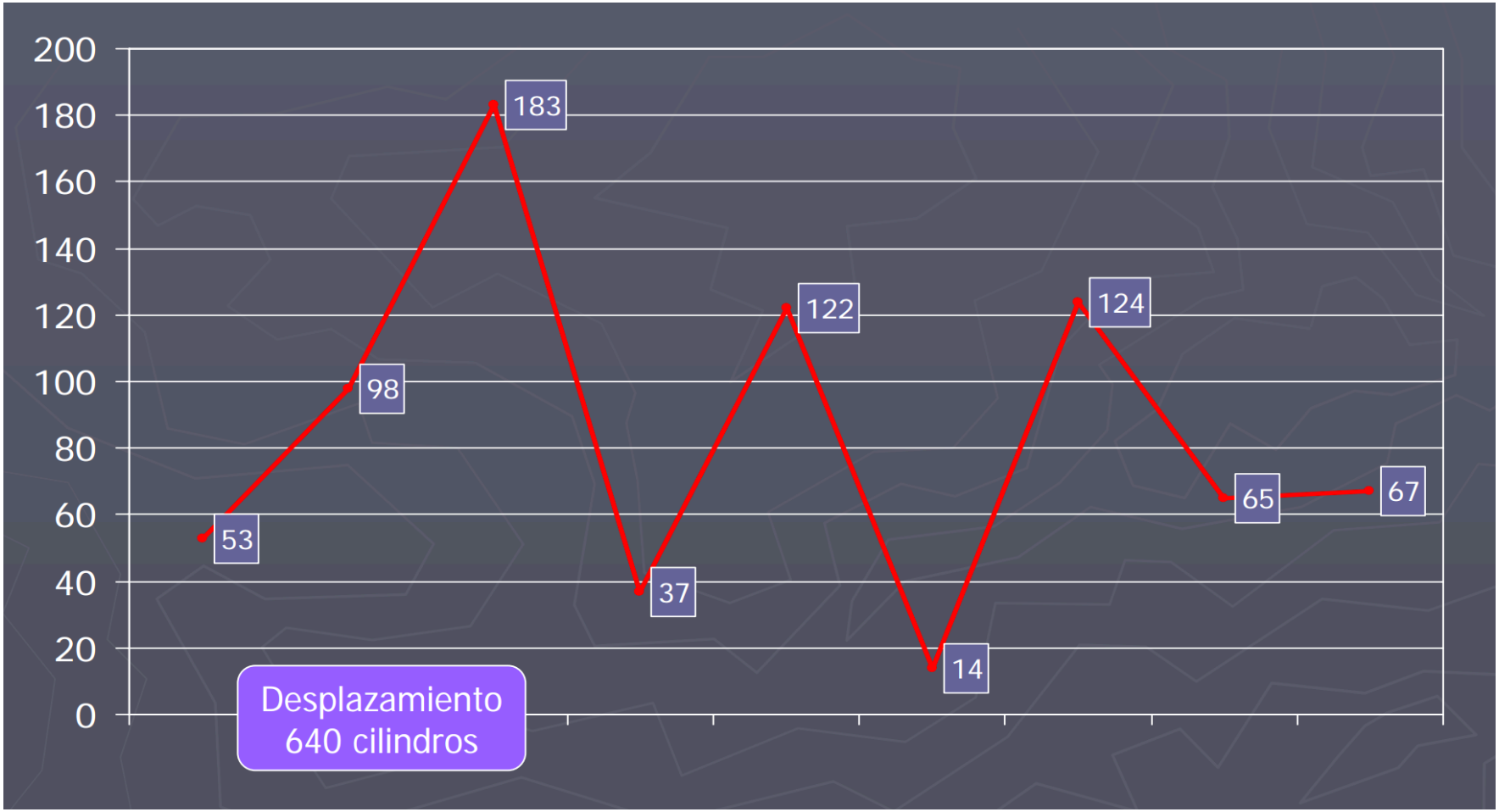
La longitud media de búsqueda (LMB) es de 61.8.

Un inconveniente de FCFS esta en los movimientos bruscos de vaivén a los que se ve sometida la cabeza lectura/escritura.

Si se pudieran atender juntas las peticiones de las pistas 22,36, 5 y 59 antes o después de las solicitudes de las pistas 124, 105, 181 y 115 disminuiría enormemente el movimiento de la cabeza y el tiempo para servir cada solicitud.

Planificación de Disco

Ejmplo planificación FCFS: Primero en llegar primero en ser atendido.



Planificación de Disco

Planificación Shortest Seek Time First - SSTF: Primero la de menor tiempo de posicionamiento.

- La estrategia SSTF consiste en atender la petición que requiere el menor movimiento de la cabeza de lectura/escritura desde su posición actual.
- De esta forma se elige la opción que incurre en el menor tiempo de búsqueda.
- Sin embargo no garantiza que el tiempo medio de búsqueda a lo largo de un numero sea mínimo.

En el ejemplo de la cola de solicitudes anterior, la pista mas próxima a la posición inicial (95) es la pista 105. Una vez situados en esta pista, la siguiente petición mas próxima es la de la 115 y así sucesivamente tal como se muestra en la siguiente tabla.

Proxima pista a la que se accede	105	115	124	142	181	59	36	22	5	
Numero de pistas que se atraviesan	10	10	9	18	39	122	23	14	17	LMB = 29.1

Planificación de Disco

Planificación SSTF: Primero la de menor tiempo de posicionamiento.

- Un problema potencial que se puede presentar con este algoritmo es el bloqueo indefinido de algunas peticiones.
- Conviene observar, que en un sistema real las solicitudes pueden llegar en cualquier momento.
- Suponga que se tiene en la cola dos peticiones, una para la pista 5 y otra para la 181. Si mientras se esta atendiendo a la petición de la pista 5 llega la de otra que esta próxima a ella, esta será la siguiente en servirse.
- Por lo que la solicitud de la 181 deberá esperar. Este argumento podría repetirse de forma indefinida con otras pistas cercanas entre si, lo que ocasionaría que la solicitud de la pista 181 espera indefinidamente.

Planificación de Disco

Planificación SSTF: Primero la de menor tiempo de posicionamiento.

Planificación de Disco

Planificación SCAN

- El algoritmo SCAN evita el bloqueo indefinido que se puede producir con la planificacion SSTF.
- La estrategia es ir recorriendo todas las pistas en una direccion y satisfaciendo todas las peticiones que se encuentra en el camino hasta que alcanza la ultima pista.
- En este punto se invierte el sentido del recorrido y la busqueda prosigue de la misma forma.
- Tambien se le conoce como el algoritmo del ascenso, por su analogia a como se atienden las llamadas de servicio para desplazarse de un piso a otro en un edificio.

Planificación de Disco

Planificación SCAN

- La tabla ilustra la estrategia SCAN para el ejemplo que se viene tratando (se ha puesto que el movimiento de la cabeza, desde la posicion inicial, era en la posicion de las pistas decrecientes).

Proxima pista a la que se accede	59	36	22	5	105	115	124	142	181	
Numero de pistas que se atraviesan	36	23	14	17	110	10	9	18	39	LMB = 30.6

- Si llega una peticion a la cola justo delante de la cabeza se atendera inmediatamente, mientras que si corresponde a una posicion que esta por detrás debera esperar a que se llegue a un extremo del disco y se invierta la direccion del movimiento.
- Proporciona ventajas a los procesos cuyas peticiones son a pistas que estan localizadas en los cilindros mas internos y externos del disco.

Planificación de Disco

Ejemplo planificación SCAN

Planificación de Disco

Planificación C-SCAN

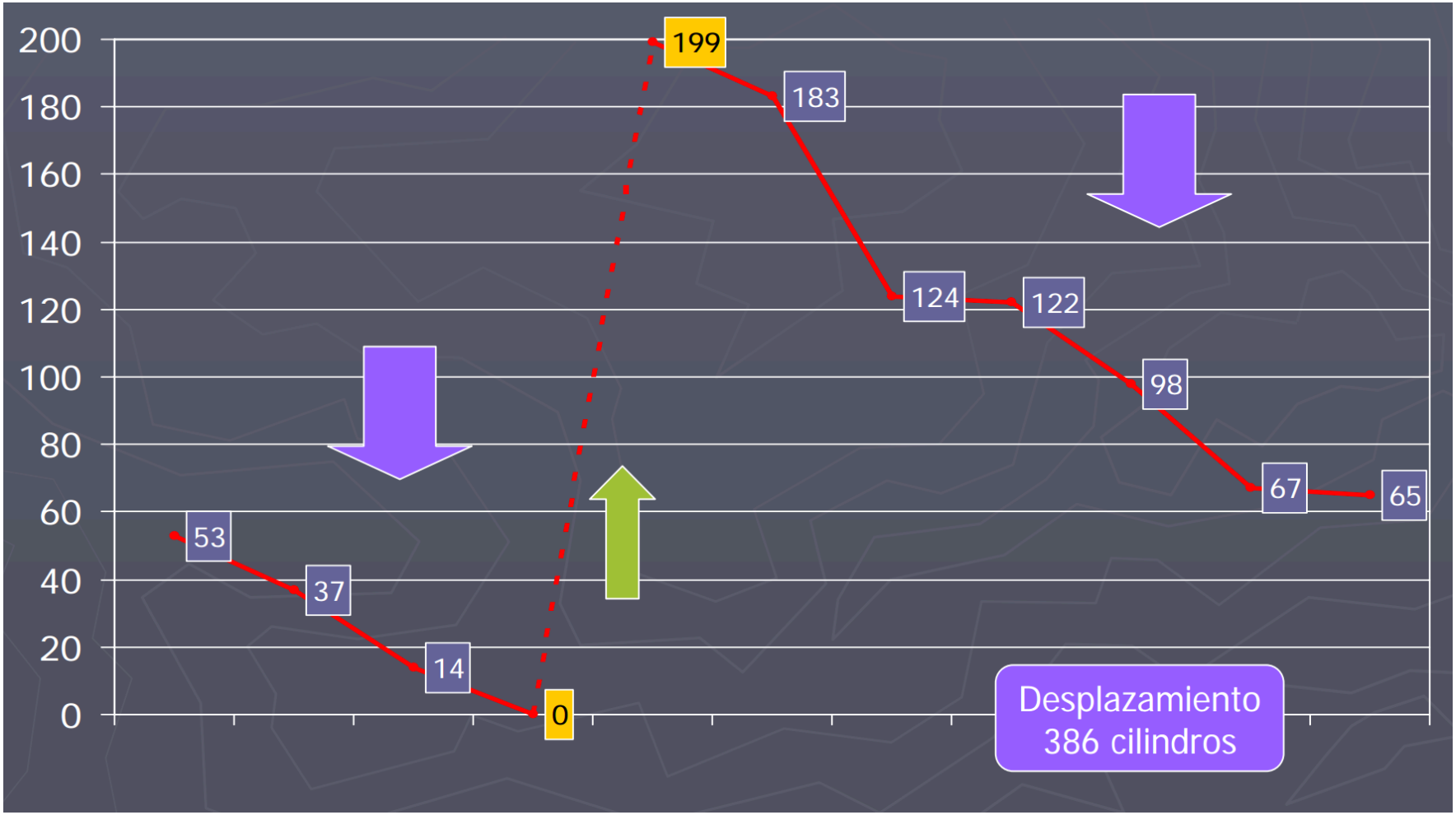
- Esta estrategia restringe el rastreo a una unica direccion (rastreo circular). Asi, cuando se ha visitado la ultima pista en una direccion, la cabeza vuelve al extremo opuesto del disco y comienza otra vez la exploracion.
- De esta forma se consigue reducir el retardo maximo que experimentan las nuevas peticiones.

Proxima pista a la que se accede	59	36	22	5	181	142	124	115	105	
Numero de pistas que se atraviesan	36	23	14	17	224	39	18	9	10	LMB = 43.3

- Basicamente, la planificacion C-SCAN considera al disco como si fuera circular, con la ultima pista adyacente a la primera.

Planificación de Disco

Ejemplo planificación C-SCAN



Planificación de Disco

Planificación LOOK y C-LOOK

- Las estrategias SCAN y C-SCAN mueven siempre la cabeza desde un extremo del disco a otro. En realidad ninguno de esos dos algoritmos se implementan de esa manera, sino que es usual que la cabeza se mueva hasta la ultima peticion de cada direccion.
- LOOK y C-LOOK miran hacia delante para ver si existe o no una solicitud antes de moverse en esa direccion.

Algoritmo Look

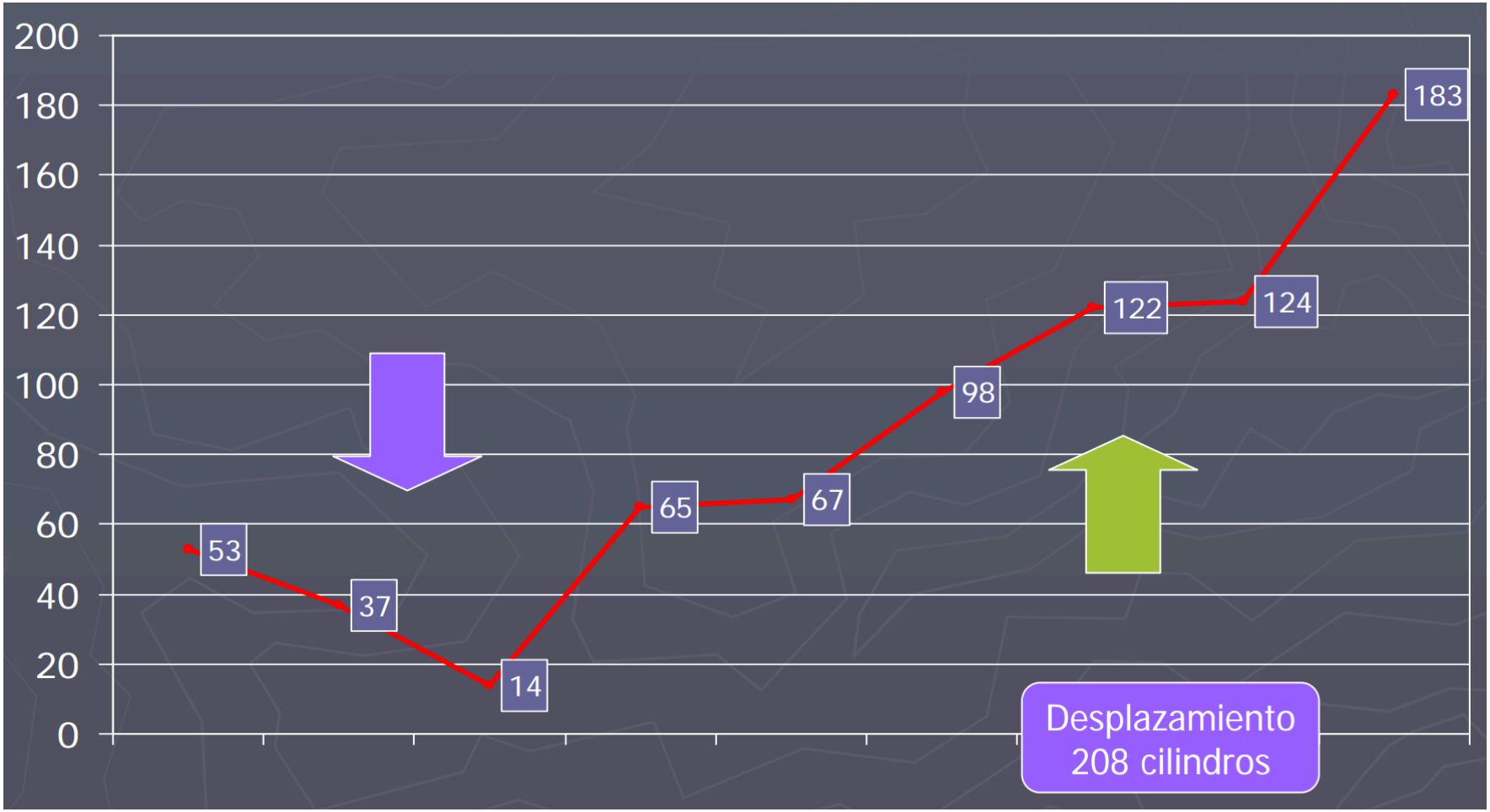
Proxima pista a la que se accede	59	36	22	5	105	115	124	142	181	
Numero de pistas que se atraviesan	36	23	14	17	100	10	9	18	39	LMB = 29.5

Algoritmo C-Look

Proxima pista a la que se accede	59	36	22	5	181	142	124	115	105	
Numero de pistas que se atraviesan	36	23	14	17	176	39	18	9	10	LMB = 38.0

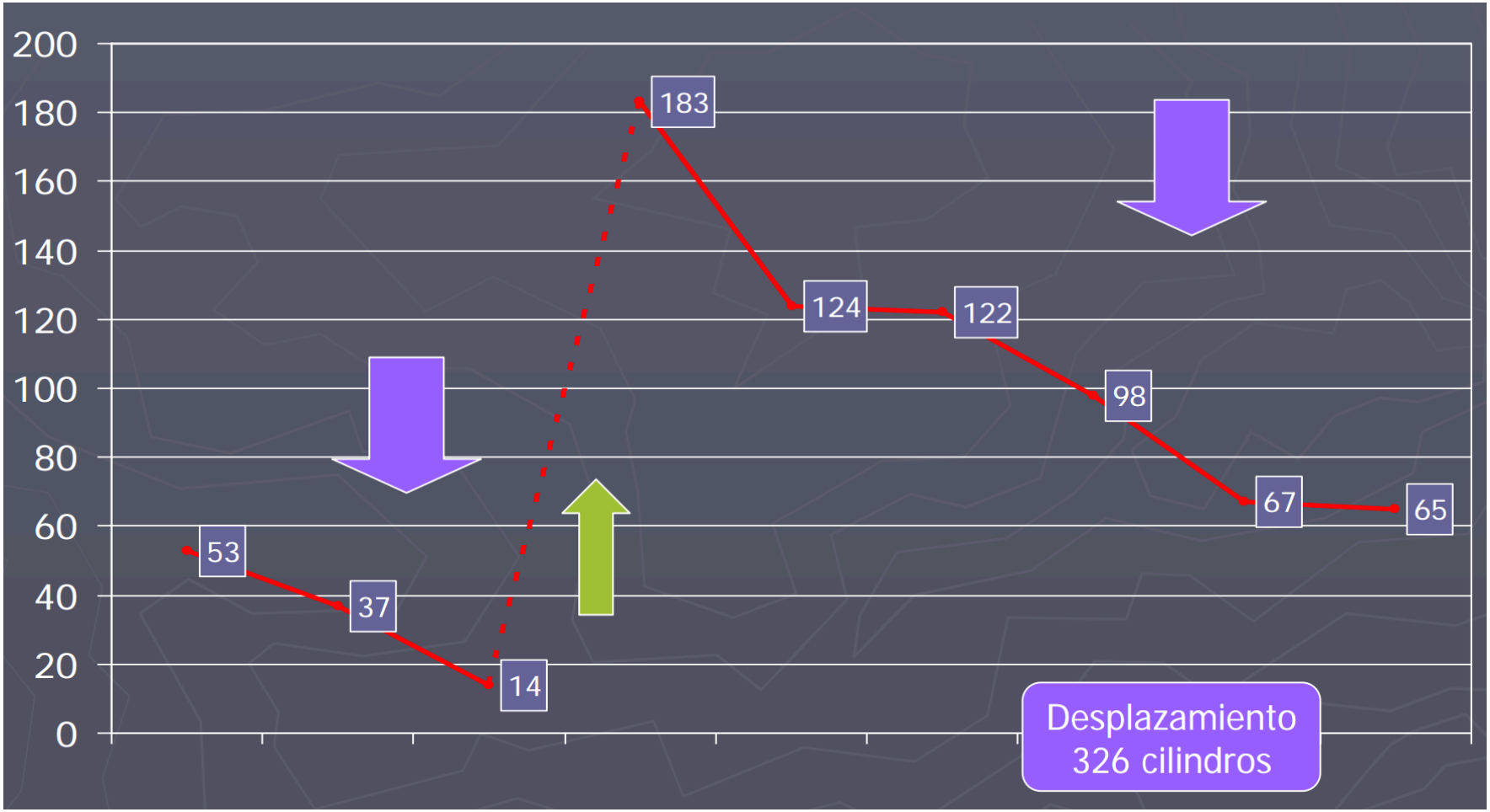
Planificación de Disco

Planificación LOOK



Planificación de Disco

Planificación C-LOOK



Cache de Disco

- El término memoria cache se utiliza normalmente aplicado a una memoria que es más pequeña y más rápida que la memoria principal y que se interpone entre la memoria principal y el procesador.
- Dicha memoria cache reduce el tiempo medio de acceso a memoria explotando el principio de la proximidad.
- El mismo principio se puede aplicar a la memoria del disco. Específicamente, una cache de disco es un buffer en memoria principal para almacenar sectores del disco.
- La cache contiene una copia de algunos de los sectores del disco.
- Cuando se hace una petición de E/S solicitando un determinado sector, se comprueba si el sector está en la cache del disco.
- En caso afirmativo, se sirve la petición desde la cache.

Cache de Disco

Consideraciones de Diseño.

- Cuando se satisface una petición de E/S de la cache de disco, se deben entregar los datos de la cache al proceso solicitante.
    - Esta operación se puede hacer o bien copiando el bloque de datos almacenado en la memoria principal asignada a la cache de disco hasta la memoria asignada al proceso de usuario, o bien utilizando simplemente la técnica de la memoria compartida pasando un puntero al bloque correspondiente de la cache de disco.
    - Esta última estrategia ahorra el tiempo de la transferencia de memoria a memoria y también permite el acceso compartido por parte de otros procesos.

Cache de Disco

Consideraciones de Diseño.

Un algoritmo ***Least frequently used*** **-** LFU sencillo conlleva el siguiente problema.

- Puede ocurrir que algunos bloques se accedan en términos globales de forma relativamente infrecuente, pero cuando se hace referencia a ellos, se producen referencias repetidas durante cortos intervalos de tiempo debido a la proximidad.
- Cuando se acaba el intervalo, el valor del contador de referencias puede llevar a conclusiones erróneas, no reflejando el grado de probabilidad de que el bloque se acceda de nuevo en un breve plazo de tiempo.
- Por tanto, el efecto de la proximidad puede realmente causar que el algoritmo LFU tome decisiones inadecuadas.

Cache de Disco

Consideraciones de Diseño.

Para resolver esta deficiencia del algoritmo LFU, se propone una técnica denominada remplazo basado en la frecuencia.

- Los bloques se organizan lógicamente en una pila, como en el algoritmo LRU. Una cierta porción de la parte superior de la pila se considera separada como una sección nueva.
- Cuando hay un acierto en la cache, el bloque accedido se mueve a la parte superior de la pila. Si el bloque ya estaba en la sección nueva, su contador de referencias no se incrementa; en caso contrario, se incrementa en 1.

Cache de Disco

Consideraciones de Rendimiento.

- El aspecto del rendimiento de la cache se reduce en sí mismo a una cuestión de si se puede alcanzar una determinada tasa de fallos.
- Esto dependerá del comportamiento de las referencias al disco con respecto a la proximidad y del algoritmo de remplazo, así como de otros factores de diseño. Sin embargo, la tasa de fallos principalmente está en función del tamaño de la cache del disco.