Sistemas operativos

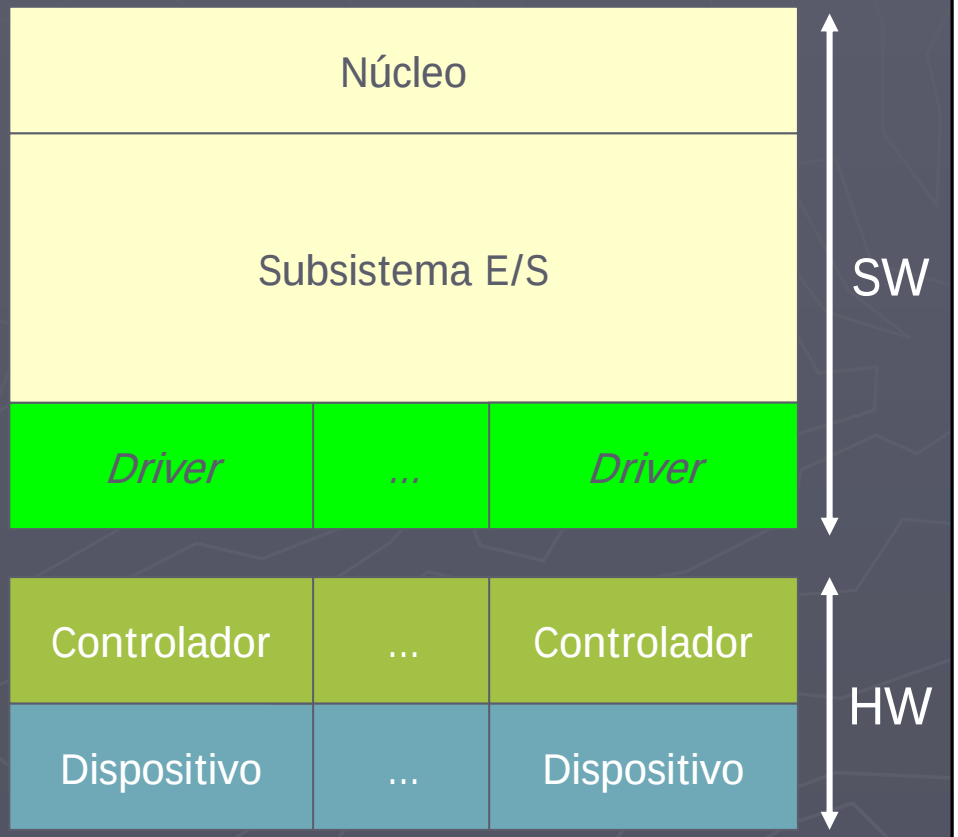
Tema 9: Gestión de E/S

Gestión de E/S

- Un sistema de computación puede incorporar múltiples dispositivos de E/S:
  - De interfaz de usuario: ratones, teclados, pantallas, etc.
  - De almacenamiento: discos, unidades de cinta, etc.
  - De comunicaciones: módems, tarjetas de red, puertos Firewire, etc.
- Los dispositivos de E/S son generalmente mucho más lentos que la CPU y la memoria → cuello de botella.
- Principales objetivos de la gestión de E/S: uniformidad y eficiencia.

Uniformidad

- Proporcionar una interfaz sencilla para acceder a los dispositivos de E/S.
  - Operaciones genéricas: abrir, leer, conectar, etc.
  - Clases de dispositivos: E/S por bloques, flujo de *bytes*, *sockets* de red, etc.
- *Software* y *hardware* especializado: *drivers* y controladores.



Eficiencia

- Gestionar el acceso a los dispositivos de E/S para optimizar el rendimiento global del sistema.
  - Almacenamiento intermedio: *buffering* y *spooling*.
  - Planificación: establecer un orden para servir las solicitudes a dispositivos E/S.
    - Repartir equitativamente el acceso a dispositivos entre múltiples procesos.
    - Reducir el tiempo de espera medio de E/S.

Buffering

- Uso de zonas de memoria propias del SO para almacenar datos mientras se transfieren entre un dispositivo y un proceso, o entre dos dispositivos.
- Se usan *buffers* para:
  - Simultanear la E/S de un proceso con su ejecución.
  - Maximizar la utilización de la CPU y los dispositivos de E/S.

Algunos usos del *buffering*

- Amortiguar diferencias de velocidad entre productores y consumidores de información.
  - Los datos de un productor lento se acumulan en un *buffer* para consumirlos posteriormente de una tacada.
- Garantizar la semántica de copiado en operaciones de salida.
  - Si un proceso ordena escribir los datos de una región de memoria, se copian en un *buffer* del SO.
  - El proceso puede desentenderse de la operación de salida, y modificar los datos en cualquier momento;
    se escribirán siempre los datos originales.
- Permitir intercambiar procesos con E/S pendiente.

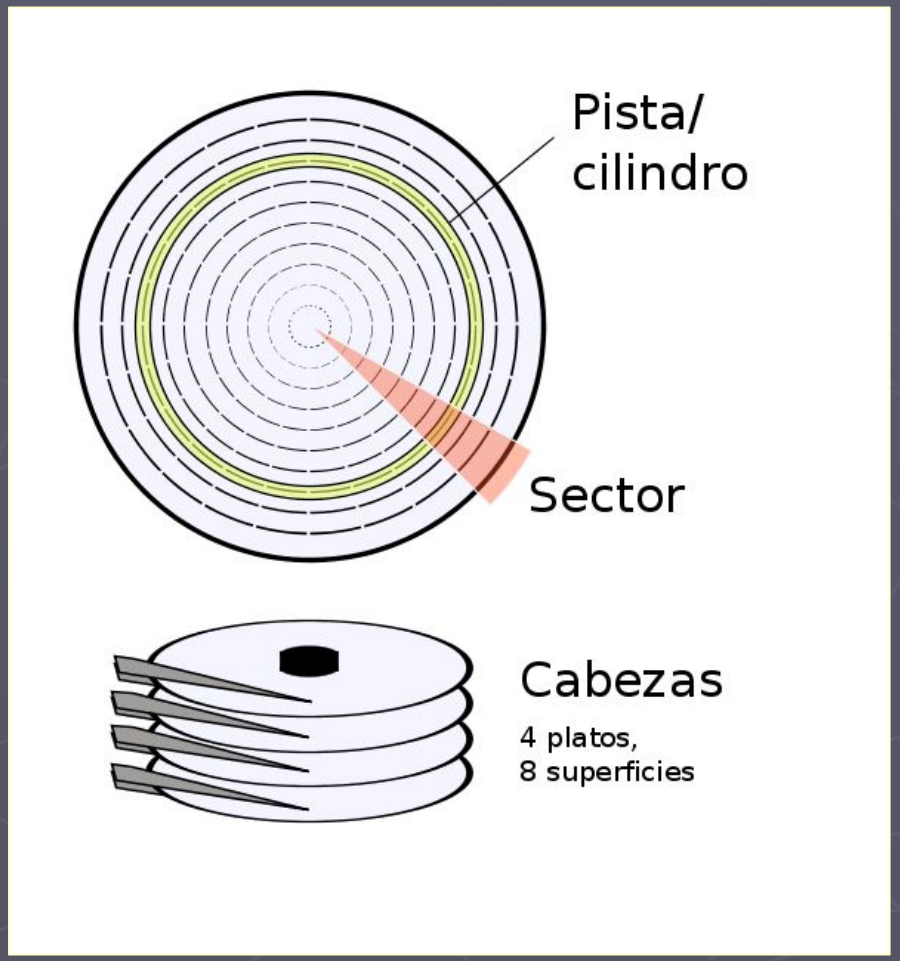
Spooling

- Un *spool* es un *buffer* de gran tamaño que se aloja en disco, no en memoria.
- Se suele usar para almacenar temporalmente la salida dirigida a dispositivos que no aceptan flujos de datos intercalados (e.g. impresoras).
  - El SO intercepta la salida de los procesos, y la guarda en sucesivos ficheros.
  - Un proceso residente envía ficheros al dispositivo cuando está disponible.
- Se utiliza un *spool* por dispositivo, no por cada proceso que ordena una operación de salida.

Planificación de discos

Discos de cabezas móviles

- Superficies magnéticas + cabezas de L/E.
  - Las superficies se dividen en pistas y sectores.
  - Las cabezas se mueven al unísono, delimitando cilindros.
- Las operaciones de L/E indican número de pista o cilindro, superficie y sector.



Discos de cabezas móviles

- El tiempo que tarda en atenderse una solicitud de L/E se desglosa en:
  - Tiempo de búsqueda, para situar las cabezas en el cilindro al que se desea acceder.
    - Arranque, desplazamiento y detención.
  - Tiempo de latencia, esperando a que el sector deseado pase por debajo de la cabeza.
    - Valor promedio: medio giro.
  - Tiempo de transferencia, determinado por la tasa de datos del disco.
  - Tiempo de espera en la cola de E/S.

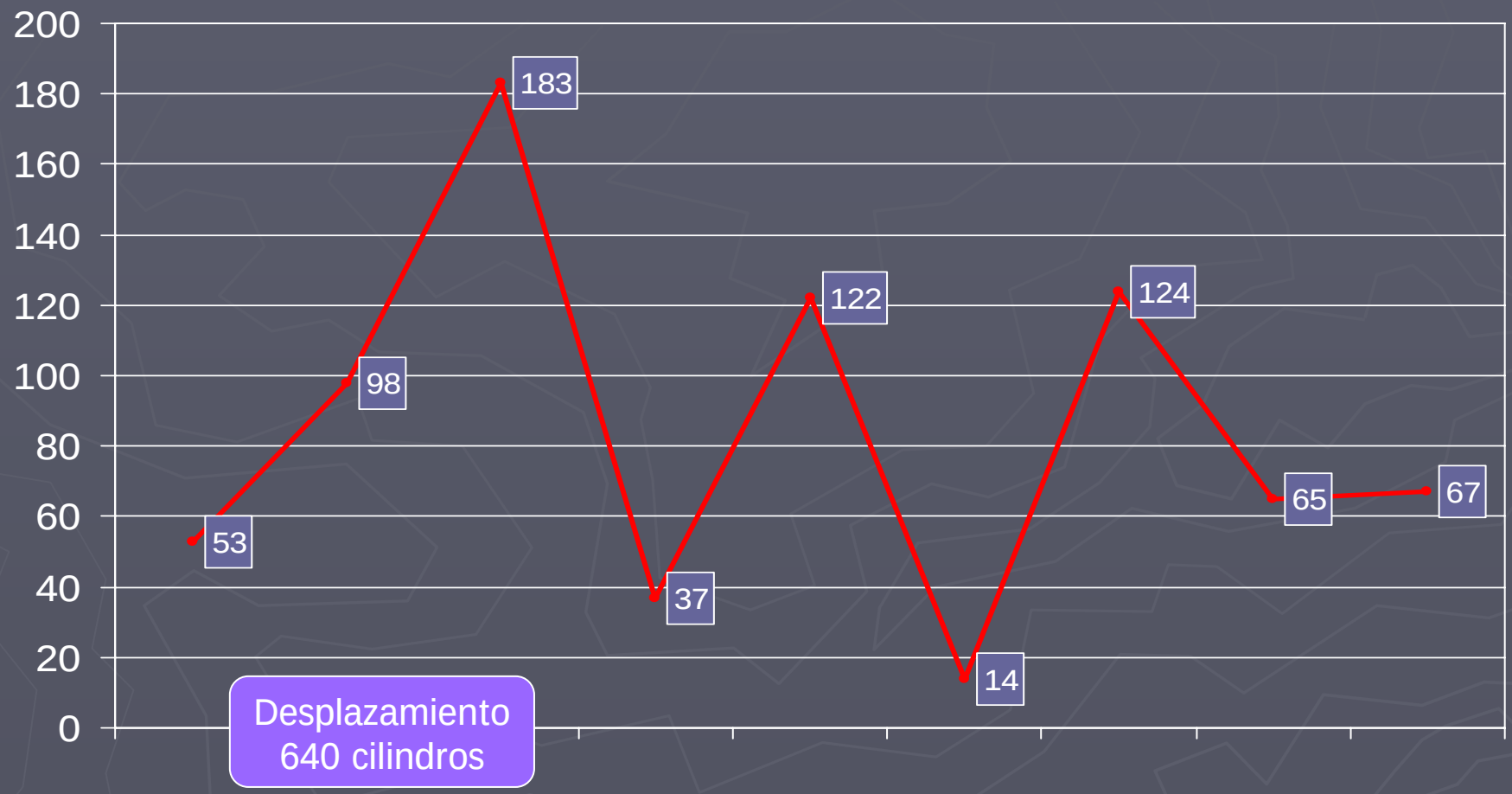
Planificación

- Los tiempos de búsqueda y latencia dependen de la última solicitud servida.
  - La planificación busca un orden de servicio para reducir esos tiempos, sin perder de vista el de espera en cola.
  - Los algoritmos más habituales (para discos de cabezas móviles) se centran en los tiempos de búsqueda.
- Ejemplo:
  - Disco de 200 cilindros (200 pistas/superficie).
  - Cola de solicitudes a los cilindros 98, 183, 37, 122, 14, 124, 65 y 67.
  - Cabezas inicialmente posicionadas en el cilindro 53.

Algoritmo FCFS

- FCFS *(First Come, First Served):* se atienden las solicitudes en orden de llegada.
  - Fácil de programar, y equitativo en los tiempos de espera en cola.
  - Al no tener en cuenta la geometría del disco, se pueden registrar grandes desplazamientos de las cabezas.
    - Tiempos de espera elevados.

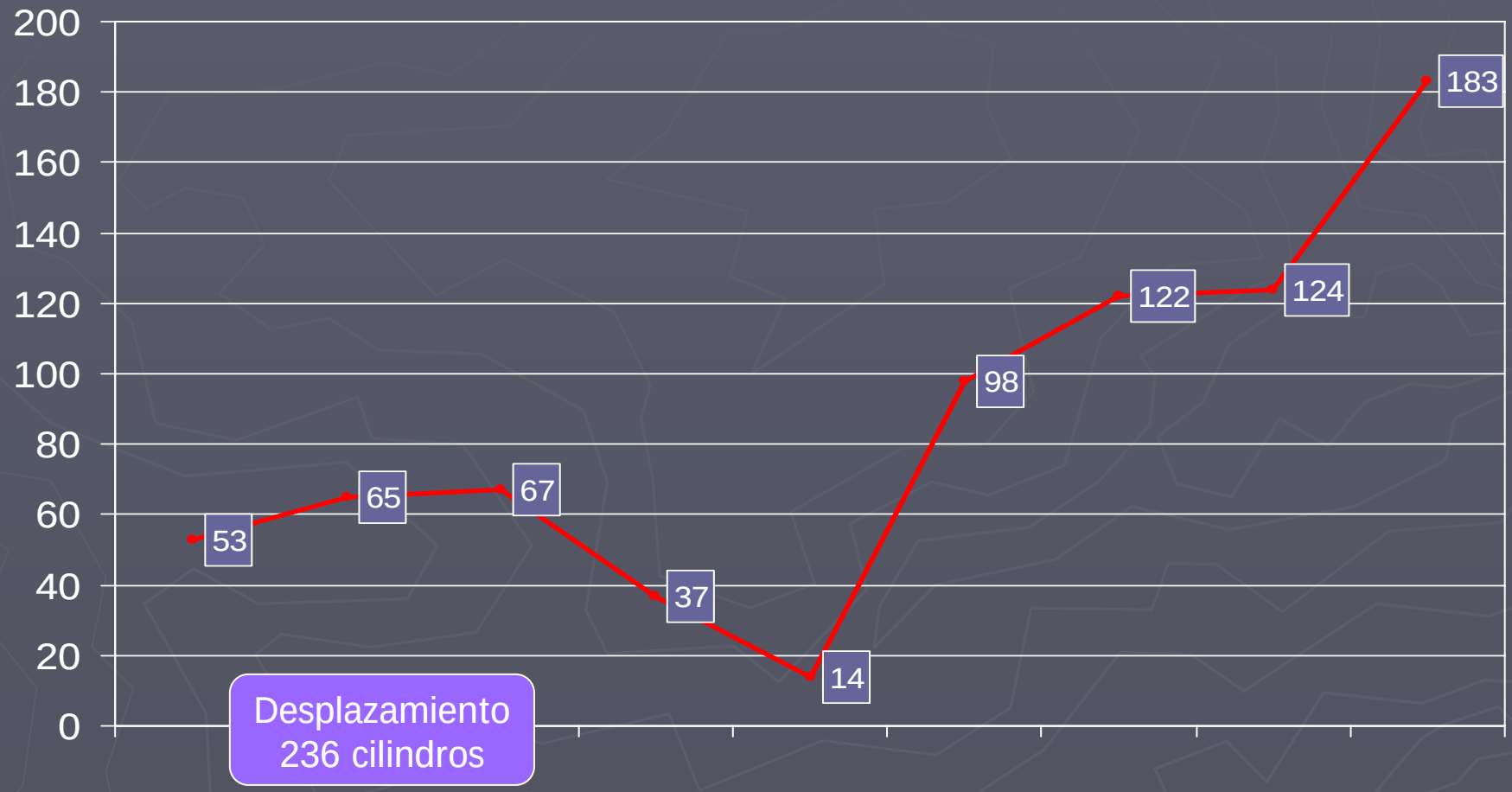
Ejemplo con FCFS



Algoritmo SSTF

- SSTF *(Shortest Seek Time First):* se atiende la solicitud con el menor tiempo de búsqueda desde la posición actual de las cabezas.
  - Las peticiones de L/E en zonas alejadas pueden sufrir inanición.

Ejemplo con SSTF



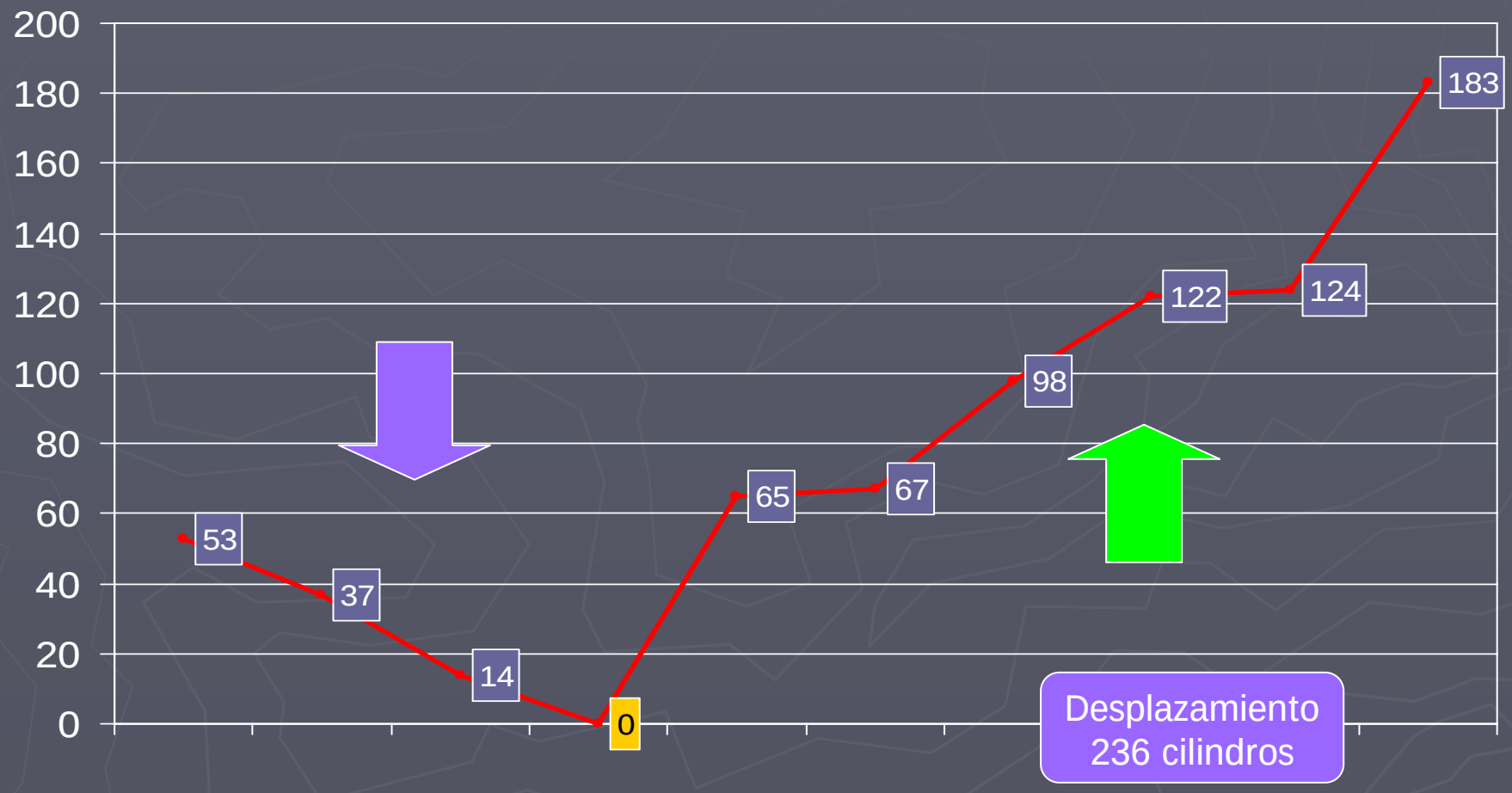
Algoritmo SSTF

- SSTF no es óptimo, en el sentido de minimizar el desplazamiento de las cabezas para un conjunto de solicitudes dado.
  - El algoritmo óptimo supone un coste computacional excesivo, y mantiene el riesgo de inanición.

Algoritmo SCAN

- SCAN: las cabezas se mueven de un extremo a otro del disco, atendiendo las solicitudes que se van encontrando.
  - Tiempos de servicio acotados, y más variables en los extremos que en el centro.

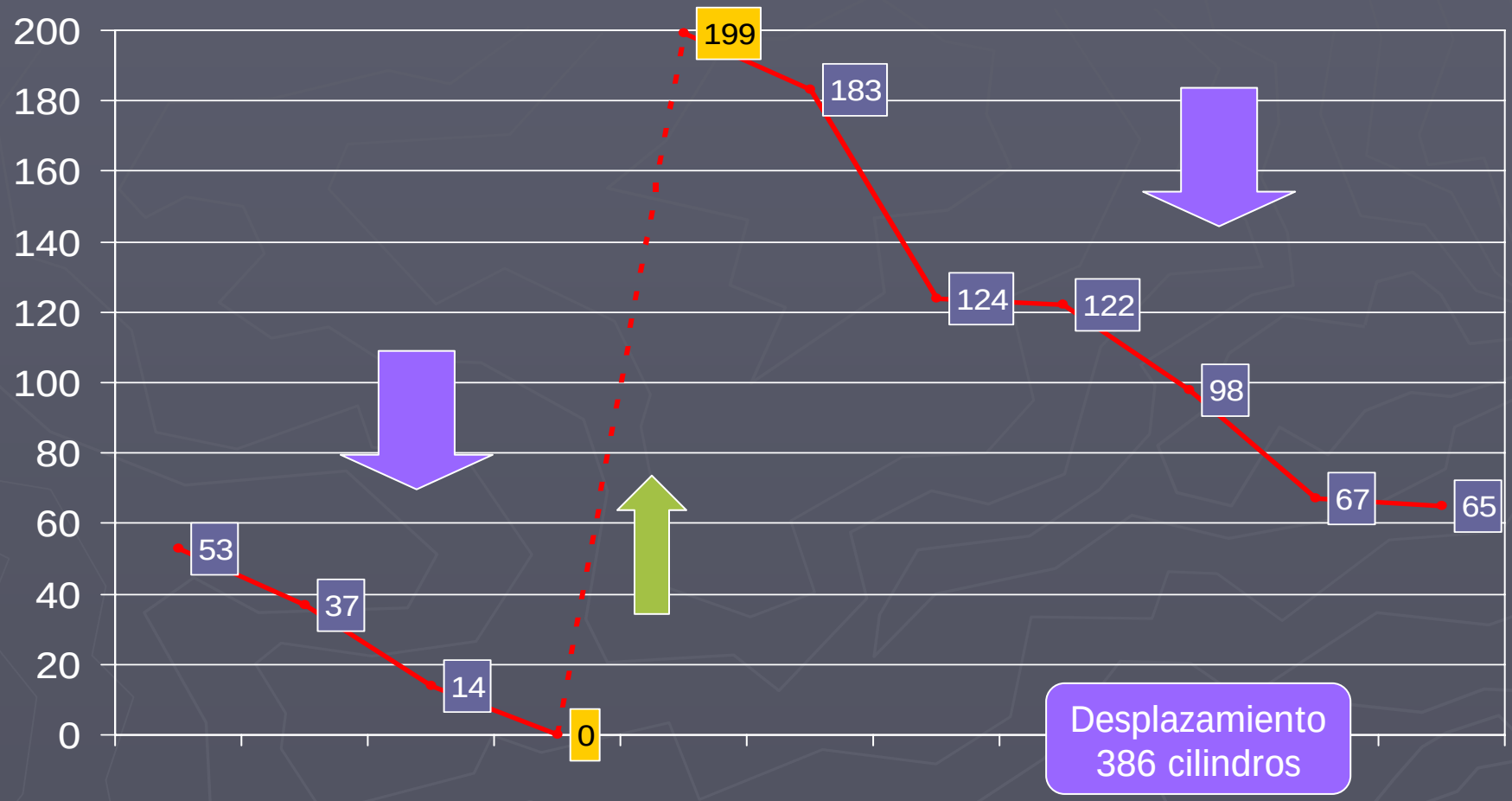
Ejemplo con SCAN



Algoritmo C-SCAN

- Con SCAN, al llegar a un extremo y cambiar de sentido, se encuentran por lo general pocas solicitudes.
  - La mayor densidad estará en el extremo opuesto, con las solicitudes que llevan más tiempo esperando.
- C-SCAN *(Circular SCAN):* las cabezas se mueven del primer cilindro al último atendiendo solicitudes, y retornan al principio.
  - Tiempos de espera más uniformes.
  - El retorno consume relativamente poco tiempo, porque se hace sin paradas.

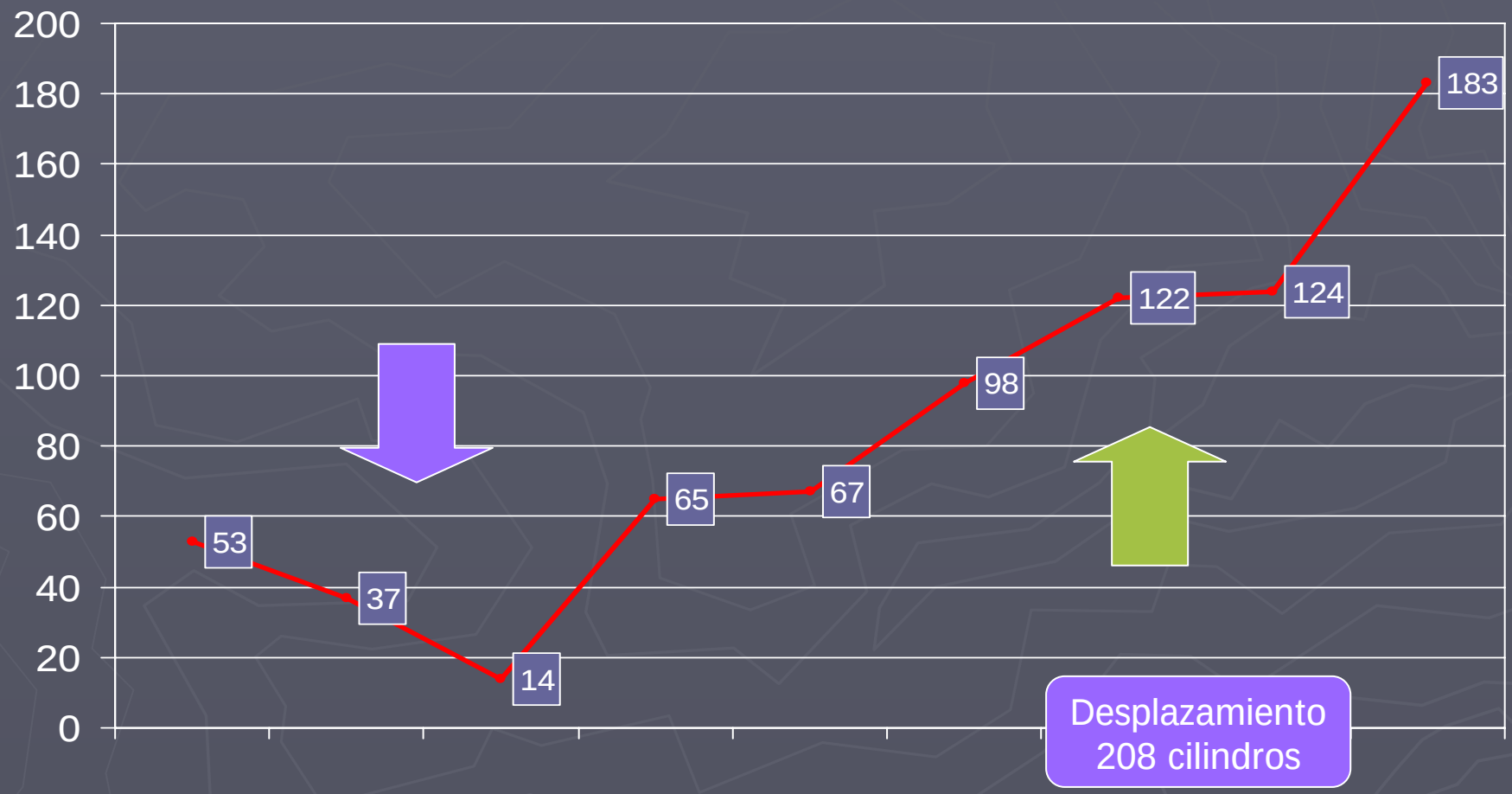
Ejemplo con C-SCAN



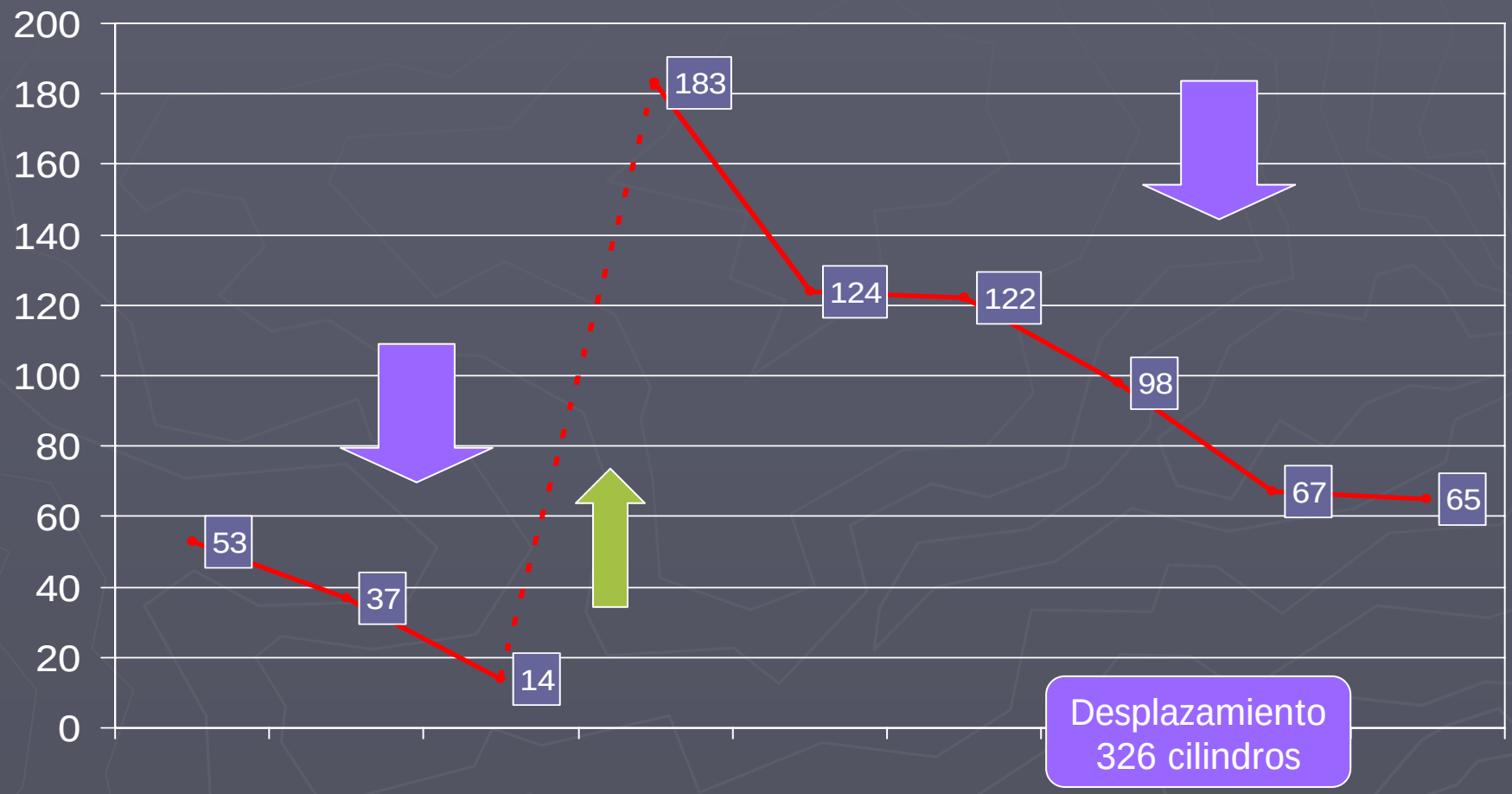
Algoritmos LOOK y C-LOOK

- Variantes de SCAN y C-SCAN: las cabezas no se mueven hasta el extremo, sino hasta la última solicitud pendiente en el sentido del movimiento.

Ejemplo con LOOK



Ejemplo con C-LOOK



Elección del algoritmo

- Con poca carga de E/S, todos los algoritmos tienen un rendimiento similar.
- En condiciones de carga elevada,
  - FCFS es equitativo pero ineficiente.
  - SSTF puede provocar inanición.
  - (C-)SCAN tendrá un rendimiento similar a (C-)LOOK, porque siempre habrá solicitudes en los extremos.
- El algoritmo más empleado en sistemas de propósito general es C-SCAN.
  - Sistemas de tiempo real o multimedia requieren soluciones específicas.

Fin