Practica 4 (Memoria Virtual - E/S - Cache) - Resolucion

Trábajo Práctico - 4.pdf

1. **Memoria Virtual**

a. Describa qué beneficios introduce este esquema de administración de la memoria.

-No es necesario cargar toda la imagen del proceso en memoria, el SO va a cargar partes del proceso a medida que este las necesita. Ese espacio cargado se llama conjunto residente

-La MMU no siempre va a calcular la direccion fisica con la tabla de paginas, por lo que en la tabla de paginas habra un bit de referencia al marco en memoria fisica (bit V: indica si pagina esta en memoria, bit M indica si pagina fue modificada)

-En conclusion mantiene en RAM solo loq ue el proceos este utilizando y deja lo demas en disco, proporcionando una abstraccion entre la memoria fisica y la memoria virtual. Esto permite que los SO sean ams eficientes al asignar memoria a los procesos y puedan aprovechar al maximo los recursos con los que se cuenta

b. ¿En que se debe apoyar el Kernel para su implementación?

-Se apoya en el HW ya que es el que detecta si una instruccion quiere acceder a una direccion que no esta cargada en memoria y es a partir de eso que el SO soluciona el fallo. El SO le brinda al HW la informacion necesaria para que pueda ocuparse de esa deteccion. Ademas en cuanto a los bits de control de la tabla de paginas es el HW el que consulta si el Kernel activo o desactivo dichos bits

c. Al implementar está técnica utilizando paginación por demanda, las tablas de páginas de un proceso deben contar con información adicional además del marco donde se encuentra la página. ¿Cuál es está información? ¿Por qué es necesaria?

-Bits de validez mencionados anteriormente

-Bit V: indica si la pagina esta en memoria (activo/desactivo el Kernel, lo consulta el HW)

-Bit M: indica si la pagina fue modificada, y si lo fue, se deben reflejar cambios en memoria secundaria (lo activa el HW, lo consulta y desactiva el SO)

2. **Fallos de Página (Page Faults):**

a. ¿Cuándo se producen?

-Se producen cuando una instruccion en ejecucion hace referencia a una direccion logica que, su pagina fisica, aun no fue cargada en memoria

b. ¿Quién es responsable de detectar un fallo de página?

-Lo detecta el HW, genera una interrupcion y el SO se encarga de resolverlo

c. Describa las acciones que emprende el Kernel cuando se produce un fallo de página.

-1) Se genera la interrupcion

-2) SO bloquea al proceso, la CPU agarra otro

-3) El SO busca un marco libre en memoria y genera una operacion de E/S en el disco para copiar en dicho marco la pagina a la cual se esta haciendo referencia y todavia no esta en memoria

-4) La E/S avisa por interrupcion cuando termina

-5) El SO actualiza la tabla de paginas del proceso (bit V en 1 y coloca la direccion base del marco donde se coloca la pagina)

-6) El proceso pasa de bloquedo a ready

-7) Cuando la CPU vuelve a tomar el proceso, es decir, se le asigna nuevamente, el proceso comienza desde la interrupcion que genero el fallo de pagina

3.

Suponga que la tabla de páginas para un proceso que se está ejecutando es la que se muestra a continuación:

Página	BitV	BitR	BitM	Marco
0	1	1	0	4
1	1	1	1	7
2	0	0	0	-
3	1	0	0	2
4	0	0	0	-
5	1	0	1	0

Asumiendo que:

-El tamaño de la página es de 512 bytes

-Cada dirección de memoria referencia 1 byte

-Los marcos se encuentran contiguos y en orden en memoria (0, 1, 2..) a partir de la dirección física 0.

-¿Qué dirección física, si existe, correspondería a cada una de las siguientes direcciones virtuales? (No gestionar ningún fallo de página en caso de producirse)

a. 1052

b. 2221

c. 5499

d. 3101

Num pagina: Dir Logica DIV tamaño pagina

Desplazamiento: Dir Logica MOD tamaño pagina

Direccion fisica: Base marco + desplazamiento

Segun la tabla las paginas cargadas en memoria son: 0, 1 , 3 y 5

Dir	Pagina	Desplazamiento	Fallo de Pagina
1052	2	28	Si
2221	4	173	Si
5499	10	379	Ni idea
3101	6	20	Ni idea

4. **Analice cómo impacta el tamaño de una página (pequeña o grande) en paginación por demanda.**

-Tamaño de pagina pequeño:

-Ventajas:

-Menos espacio no utilizado dentro de una pagina, reduce fragmentacion interna

-Mas paginas alocadas en memoria

-Si hay paginas con errores, se pierde menos memoria en comparacion con pag grande

-Desventajas:

-Se necesitan mas entradas en al tabla de paginas, volviendola mas grande e ineficiente en tiempos de busqueda y carga

-Mas fragmentacion externa

-Tamaño de pagina grande:

-Ventajas:

-Se necesitan menos entradas en la tabla de paginas para mapear el mismo espacio de direcciones

-Menor fragmentacion interna

-Menor sobrecarga del SO al gestionar con un numero menor de paginas, es mas rapido mover paginas grandes a la RAM ya que esta diseñada para mover bloques de datos

-Desventajas:

-Si una pagina grande tiene errores se pierde mas memoria en comparacion a una pag pequeña

-Mas fragmentacoin interna

5. **Asignación de marcos a un proceso (Conjunto de trabajo o Working Set).**

Con la memoria virtual paginada, no se requiere que todas las páginas de un proceso se encuentren en memoria. El Kernel debe controlar cuantas páginas de un proceso puede tener en la memoria principal. Existen 2 políticas que se pueden utilizar:

Asignación Fija

Asignación Dinámica.

a. Describa cómo trabajan estas 2 políticas.

-Asignacion dinamica: el numero de marcos asignados para el proceso varia

-Asignacion fija: cada proceso tiene un numero fijo de marcos

-Asignacion equitativa: si tenes 100 marcos y 5 procesos, le das 20 marcos a cada uno

-Asignacion proporcional: se asigna proporcionalmente al tamaño del proceso

P1=100 paginas , P2=250 paginas y tenes 100 marcos. Demanda total es 350 paginas. *La proporcion de P1 es 100/350 * 100, y la de P2 es 250/350 * 100*

b.

Dada la siguiente tabla de procesos y las páginas que ellos ocupan, y teniéndose 40 marcos en la memoria principal, cuántos marcos le corresponderá a cada proceso si se usa la técnica de Asignación Fija:

Proceso	Total de Páginas Requeridas
1	15
2	20
3	20
4	8

- **Reparto Equitativo** → m / p → si tenes 40 marcos y 4 procesos = 10
- **Reparto Proporcional**

-P1 → 15/63 * 40 = 9,52 → 10

-P2 → 20/63 * 40 = 12.7 → 13

-P3 → 20/63 * 40 = 12.7 → 13

-P4 → 8/63 * 40 = 5.08 → 5

c. ¿Cuál de los 2 repartos usados en b) resultó más eficiente? Justifique

-Es mas eficiente el proporcional porque se ajusta mas a cada proceso, si tomamos P4 en el equitativo usa 10 cuando literalmente solo necesita la mitad, por lo que estarias desperdiciando el doble de marcos de asignacion

6. **Reemplazo de páginas (selección de una víctima).**

¿Qué sucede cuando todos los marcos en la memoria principal están usados por las páginas de los procesos y se produce un fallo de página? El Kernel debe seleccionar una de las páginas que se encuentra en memoria principal (en un marco) como víctima, la que será reemplazada por la nueva página que produjo el fallo.

-Considere los siguientes algoritmos de selección de víctimas básicos:

LRU

FIFO

OPT (Óptimo)

Segunda Chance

a. Clasifique estos algoritmos de malo a bueno de acuerdo a la tasa de fallos de página que se obtienen al utilizarlos.

-FIFO → FIFO 2da chance → LRU → Optimo

b. Analice su funcionamiento. ¿Cómo se podrían implementar?

-Algoritmo FIFO: trata a los frames en uso como una cola circular, la pagina mas antigua en la memoria es la reemplazada. La pagina puede ser necesitada pronto

-Algoritmo LRU (Least recently used): requiere soporte de HW para mantener una marca de tiempo de acceso a las paginas. Favorece a las paginas mas recientemente accedidas. Con lo que vimos se puede implementar con el bit de accedido (setea en 1 cuando se usa y cuando se checkea que se usa cada tanto lo borras y lo volves a poner en 0)

-Algoritmo 2da chance: un avance del FIFO tradicional que beneficia a las paginas referenciadas. Si detecto que tu bit de uso esta en 1 te doy una segunda oportunidad y te mando al final de la cola con R = 0, basicamente asume que probablemente pueda ser usada nuevamente. Si esta en 0 es victima

-Algoritmo NRU (Non recently Used): usa los bits R y M para determinar cual sacar primer, favorece a las paginas que fueron usadas recientemente:

c. Sabemos que la página a ser reemplazada puede estar modificada. ¿Cómo detecta el Kernel esta situación? ¿Qué acciones adicionales deben realizarse cuando se encuentra ante esta situación?

-La pagina que causo el fallo se coloca en un framed esginado a la descarga asincronica

-El SO envia orden de descarga asincronica mientras se ejecuta otro proceso

-El frame de descarga asincronica pasa a ser el contenedor de la pagina victima que fue descargada

7. **Alcance del reemplazo.**

Al momento de tener que seleccionar una página victima, el Kernel puede optar por 2 políticas a utilizar:

Reemplazo local

Reemplazo global

a. Describa cómo trabajan estas 2 políticas.

-Reemplazo local: El fallo de pagina de un proceso solo puede reenplazar sus propias paginas. No cambia la cantidad de frames asignado. El SO es capaz de determinar la tasa de fallo de paginas de cada proceso

-Reemplazo global: El fallo de paginas de un proceso puede reemplazar pagina de cualquier proceso. Cambia la cantidad de frames del proceso ya que un proceso se puede appropiar de frames de otro. El SO no es capaz de

determinar la tasa de fallo de pagina de cada proceso

b. ¿Es posible utilizar la política de “Asignación Fija” de marcos junto con la política de “Reemplazo Global"? Justifique.

-Y no, le das marcos fijos a cada proceso y en reemplazo global los procesos se pueden adueñar de marcos de otros procesos, rompiendo esa regla de que cada proceso tiene exactamente la misma cantidad de procesos que los demas

8. Considere la siguiente secuencia de referencias de páginas:

1, 2, 15, 4, 6, 2, 1, 5, 6, 10, 4, 6, 7, 9, 1, 6, 12, 11, 12, 2, 3, 1, 8, 1, 13, 14, 15, 3, 8

a. Si se disponen 5 marcos. ¿Cuántos fallos de página se producirán si se utilizan las siguientes técnicas de selección de víctima? (Considere una política de Asignación Dinámica y Reemplazo Global)

i. Segunda Chance

Segunda Chance																												
MARCOS	**1**	**2**	**15**	**4**	**6**	**2**	**1**	**5**	**6**	**10**	**4**	**6**	**7**	**9**	**1**	**6**	**12**	**11**	**12**	**2**	**3**	**1**	**8**	**1**	**13**	**14**	**15**	**3**	**8**
1	1	1	1	1	1	1	1*	1	1	1	4	4	4	4	4	4	12	12	12*	12*	12*	12*	12	12	12	12	15	15	15
2		2	2	2	2	2*	2*	2	2	2	2	2	7	7	7	7	7	11	11	11	11	11	8	8	8	8	8	3	3
3			15	15	15	15	15	5	5	5	5	5	5	9	9	9	9	9	9	2	2	2	2	2	13	13	13	13	8
4				4	4	4	4	4	4	10	10	10	10	10	1	1	1	1	1	1	3	3	3	3	3	14	14	14	14
5					6	6	6	6	6*	6*	6	6*	6*	6*	6*	6*	6	6	6	6	6	1	1	1*	1*	1*	1	1	1
PF?	X	X	X	X	X			X		X	X		X	X	X		X	X		X	X	X	X		X	X	X	X	X
QUEUE	~~1*~~	~~2*~~	~~15~~	4	~~6*~~	~~1~~	~~2~~	~~5~~	~~10~~	~~6~~	4	~~7~~	~~9~~	~~1~~	~~6~~	~~12*~~	~~11~~	~~2~~	~~3~~	~~1*~~	~~12~~	~~8~~	~~13~~	14	1	15	3	8	
																							0,1 seg x 22 fallos = 2,2 segundos						

ii. FIFO

FIFO																												
MARCOS	**1**	**2**	**15**	**4**	**6**	**2**	**1**	**5**	**6**	**10**	**4**	**6**	**7**	**9**	**1**	**6**	**12**	**11**	**12**	**2**	**3**	**1**	**8**	**1**	**13**	**14**	**15**	**3**	**8**
1	1	1	1	1	1	1	1	5	5	5	5	5	5	5	5	6	6	6	6	6	6	1	1	1	1	1	1	3	3
2		2	2	2	2	2	2	2	2	10	10	10	10	10	10	10	12	12	12	12	12	12	8	8	8	8	8	8	8
3			15	15	15	15	15	15	15	15	15	15	7	7	7	7	7	11	11	11	11	11	11	11	13	13	13	13	13
4				4	4	4	4	4	4	4	4	4	4	9	9	9	9	9	9	2	2	2	2	2	2	14	14	14	14
5					6	6	6	6	6	6	6	6	6	6	1	1	1	1	1	3	3	3	3	3	3	3	15	15	15
PF?	X	X	X	X	X			X		X			X	X	X	X	X	X		X	X	X	X		X	X	X	X	
QUEUE	~~1~~	~~2~~	~~15~~	4	~~6~~	~~5~~	~~10~~	~~7~~	~~9~~	~~1~~	~~6~~	~~12~~	~~11~~	~~2~~	~~3~~	~~1~~	8	13	14	15	3	8							
																							0,1 seg x 21 fallos = 2,1 segundos						

iii. LRU

LRU																												
MARCOS	**1**	**2**	**15**	**4**	**6**	**2**	**1**	**5**	**6**	**10**	**4**	**6**	**7**	**9**	**1**	**6**	**12**	**11**	**12**	**2**	**3**	**1**	**8**	**1**	**13**	**14**	**15**	**3**	**8**
1	1	1	1	1	1	1	1	1	1	1	1	1	7	7	7	7	7	11	11	11	11	11	8	8	8	8	8	3	3
2		2	2	2	2	2	2	2	2	2	4	4	4	4	4	4	12	12	12	12	12	12	12	12	13	13	13	13	13
3			15	15	15	15	15	5	5	5	5	5	5	9	9	9	9	9	9	2	2	2	2	2	2	14	14	14	14
4				4	4	4	4	4	4	10	10	10	10	10	1	1	1	1	1	1	3	3	3	3	3	3	15	15	15
5					6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	1	1	1	1	1	1	1	8
PF?	X	X	X	X	X			X		X			X	X	X		X	X		X	X	X	X		X	X	X	X	X
QUEUE																													
																							0,1 seg x 21 fallos = 2,2 segundos						

iv. OPT

OPTIMO																												
MARCOS	**1**	**2**	**15**	**4**	**6**	**2**	**1**	**5**	**6**	**10**	**4**	**6**	**7**	**9**	**1**	**6**	**12**	**11**	**12**	**2**	**3**	**1**	**8**	**1**	**13**	**14**	**15**	**3**	**8**
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	13	13	13	13	13
2		2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	8	8	8	8	8	8	8
3			15	15	15	15	15	5	5	10	10	10	7	9	9	9	12	12	12	12	12	12	12	12	12	15	15	15	
4				4	4	4	4	4	4	4	4	4	4	4	4	4	4	11	11	11	11	11	11	11	11	14	14	14	14
5					6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	3	3	3	3	3	3	3	3	3
PF?	X	X	X	X	X			X		X			X	X			X	X			X		X		X	X	X		
QUEUE																													
																							0,1 seg x 14 fallos = 1,4 segundos						

b. Suponiendo que cada atención de un fallo se página requiere de 0,1 seg. Calcular el tiempo consumido por atención a los fallos de páginas para los algoritmos de a).

ejer8 (pf).xlsx

9. Sean los procesos A, B y C tales que necesitan para su ejecución las siguientes páginas:

A: 1, 3, 1, 2, 4, 1, 5, 1, 4, 7, 9, 4

B: 2, 4, 6, 2, 4, 1, 8, 3, 1, 8

C: 1, 2, 4, 8, 6, 1, 4, 1

Si la secuencia de ejecución es tal que los procesos se ejecutan en la siguientes secuencia:

1. B demanda 2 páginas

2. A demanda 3 páginas

3. C demanda 2 páginas

4. B demanda 3 páginas

5. A demanda 3 páginas

6. C demanda 2 páginas

7. B demanda 2 páginas

8. C demanda 4 páginas

9. A demanda 3 páginas

10. B demanda 3 páginas

11. C termina

12. A demanda 3 páginas

13. B termina

14. A termina

a. Considerando una política de Asignación Dinámica y Reemplazo Global y disponiéndose de 7 marcos. ¿Cuántos fallos de página se producirán si se utiliza la técnica de selección de víctimas:

i. LRU

LRU (DIN - RG)																												
MARCOS	B2	B4	A1	A3	A1	C1	C2	B6	B2	B4	A2	A4	A1	C4	C8	B1	B8	C6	C1	C4	C1	A5	A1	A4	B3	B1	B8	A7	A9	A4
1	B2	B2	B2	B2	B2	B2	B2	B2	B2	B2	B2	B2	B2	B2	B2	B1	B1	B1	B1	B1	B1	B1	B1	A4	A4	A4	A4	A4	A4	A4
2		B4	B4	B4	B4	B4	B4	B4	B4	B4	B4	B4	B4	B4	B4	B4	B8	B8	B8	B8	B8	B8	B8	B8	B3	B3	B3	B3	B3	B3
3			A1	A1	A1	A1	A1	A1	A1	A1	A1	A4	A4	A4	A4	A4	A4	A4	C1	C1	C1	C1	C1	C1	C1	C1	C1	C1	A7	A7	A7
4				A3	A3	A3	A3	A3	A3	A3	A2	A2	A2	A2	A2	A2	A2	C6	C6	C6	C6	C6	C6	C6	C6	C6	B1	B1	B1	B1	B1
5						C1	C1	C1	C1	C1	C1	C1	A1	A1	A1	A1	A1	A1	A1	A1	A1	A5	A5	A5	A5	A5	A5	A5	A9	A9
6							C2	C2	C2	C2	C2	C2	C2	C4	C4	C4	C4	C4	C4	C4	C4	C4	C4	C4	C4	C4	B8	B8	B8	B8
7								B6	B6	B6	B6	B6	B6	B6	C8	C8	C8	C8	C8	C8	C8	C8	A1	A1	A1	A1	A1	A1	A1	A1
PF ?	X	X	X	X		X	X	X			X	X	X	X	X	X	X	X			X	X	X	X	X	X	X	X		
QUEUE																														
CANTIDAD DE FP = 24																														

ii. Segunda Chance

2DA CHANCE (DIN - RG)																														
MARCOS	B2	B4	A1	A3	A1	C1	C2	B6	B2	B4	A2	A4	A1	C4	C8	B1	B8	C6	C1	C4	C1	A5	A1	A4	B3	B1	B8	A7	A9	A4
1	B2	B2	B2	B2	B2	B2	B2	B2	B2*	B2*	B2	B2	B2	B2	B2	B1	B1	B1	B1	B1	B1	B1	B1	A4	A4	A4	A4	A4	A4	A4*
2		B4	B4	B4	B4	B4	B4	B4	B4	B4*	B4	B4	B4	B4	B4	B4	B8	B8	B8	B8	B8	B8	B8	B8	B3	B3	B3	B3	B3	B3
3			A1	A1	A1*	A1*	A1*	A1*	A1*	A1*	A1*	A1*	A1*	A1*	A1*	A1*	A1*	A1	A1	A1	A1	A1	A1*	A1*	A1*	A1	A1	A1	A1	A1
4				A3	A3	A3	A3	A3	A3	A3	A2	A2	A2	A2	A2	A2	A2	C6	C6	C6	C6	C6	C6	C6	C6	B1	B1	B1	B1	B1
5						C1	C1	C1	C1	C1	C1	A4	A4	A4	A4	A4	A4	A4	C1	C1	C1*	C1*	C1*	C1*	C1*	C1*	C1	A7	A7	A7
6							C2	C2	C2	C2	C2	C2	C2	C4	C4	C4	C4	C4	C4	C4*	C4*	C4	C4	C4	C4	C4	B8	B8	B8	B8
7								B6	B6	B6	B6	B6	B6	B6	C8	C8	C8	C8	C8	C8	C8	A5	A5	A5	A5	A5	A5	A9	A9	
PF ?	X	X	X	X		X	X	X			X	X		X	X	X	X	X	X			X		X	X	X	X	X	X	
QUEUE	~~B2*~~	~~B4*~~	~~A1*~~	~~A3~~	~~A1~~	~~C1~~	~~C2~~	~~B6~~	~~B2~~	~~B4~~	~~A1*~~	~~A2~~	~~A4~~	~~C4*~~	~~C8~~	~~B1~~	~~B8~~	~~A1*~~	~~C6~~	~~C1*~~	~~C4~~	A5	A4	B3	A1	B1	C1	B8	A7	A9
CANTIDAD DE FP = 22																		en los ultimos 3-4 excel me bugeo y relleno la cola pero esta bien											

b. Considerando una política de Asignación Fija con reparto equitativo y Reemplazo Local y disponiéndose de 9 marcos. ¿Cuántos fallos de página se producirán si se utiliza la técnica deselección de víctimas:

i. LRU

LRU (FIJA - RL)																														
MARCOS	B2	B4	A1	A3	A1	C1	C2	B6	B2	B4	A2	A4	A1	C4	C8	B1	B8	C6	C1	C4	C1	A5	A1	A4	B3	B1	B8	A7	A9	A4
1	B2	B2	B2	B2	B2	B2	B2	B2	B2*	B2	B2	B2	B2	B2	B2	B2	B8	B8	B8	B8	B8	B8	B8	B8	B8	B8	B8*	B8	B8	B8
2		B4	B4	B4	B4	B4	B4	B4	B4	B4*	B4	B4	B4	B4	B4	B4	B4	B4	B4	B4	B4	B4	B4	B4	B3	B3	B3	B3	B3	B3
3			A1	A1	A1*	A1	A1	A1	A1	A1	A1	A1	A1*	A1	A1	A1	A1	A1	A1	A1	A1	A1	A1*	A1	A1	A1	A1	A1	A9	A9
4				A3	A3	A3	A3	A3	A3	A3	A3	A4	A4	A4	A4	A4	A4	A4	A4	A4	A4	A4	A4	A4*	A4	A4	A4	A4	A4	A4*
5						C1	C1	C1	C1	C1	C1	C1	C1	C1	C8	C8	C8	C8	C4	C4	C4	C4	C4	C4	C4	C4				
6							C2	C2	C2	C2	C2	C2	C2	C2	C2	C2	C6	C6	C6	C6	C7	C7	C7	C7	C7	C7				
7								B6	B6	B6	B6	B6	B6	B6	B6	B1	B1	B1	B1	B1	B1	B1	B1	B1	B1*	B1	B1	B1	B1	
8											A2	A2	A2	A2	A2	A2	A2	A2	A2	A2	A5	A5	A5	A5	A5	A5	A7	A7	A7	
9														C4	C4	C4	C4	C4	C1	C1	C1*	C1	C1	C1	C1	C1	C1	C1	C1	
PF ?	X	X	X	X		X	X	X			X	X		X	X	X	X	X	X		X		X	X			X		
QUEUE A	~~A1*~~	~~A3~~	~~A1~~	~~A2~~	~~A4~~	~~A1*~~	~~A5~~	~~A1~~	A4*	A7	A9																			
QUEUE B	~~B2*~~	~~B4*~~	~~B6~~	~~B2~~	~~B4~~	B1	B8*	B3																						
QUEUE C	~~C1~~	~~C2~~	~~C4~~	~~C8~~	C6	C1																								
CANTIDAD DE FP = 20																														

ii. Segunda Chance

2DA Chance (FIJA - RL)																														
MARCOS	B2	B4	A1	A3	A1	C1	C2	B6	B2	B4	A2	A4	A1	C4	C8	B1	B8	C6	C1	C4	C1	A5	A1	A4	B3	B1	B8	A7	A9	A4
1	B2	B2	B2	B2	B2	B2	B2	B2	B2*	B2*	B2*	B2*	B2*	B2*	B2*	B2	B8	B8	B8	B8	B8	B8	B8	B8	B8	B8	B8*	B8*	B8*	B8*
2		B4	B4	B4	B4	B4	B4	B4	B4	B4*	B4*	B4*	B4*	B4*	B4*	B4	B4	B4	B4	B4	B4	B4	B4	B4	B3	B3	B3	B3	B3	B3
3			A1	A1	A1*	A1*	A1*	A1*	A1*	A1*	A1*	A1	A1*	A1*	A1*	A1*	A1*	A1*	A1*	A1*	A1*	A1	A1	A1	A1	A1	A1	A1	A9	A9
4				A3	A3	A3	A3	A3	A3	A3	A3	A4	A4	A4	A4	A4	A4	A4	A4	A4	A4	A4	A4	A4	A4*	A4*	A4*	A4*	A4*	A4
5						C1	C1	C1	C1	C1	C1	C1	C1	C1	C8	C8	C8	C8	C8	C4	C4	C4	C4	C4	C4	C4	C4			
6							C2	C2	C2	C2	C2	C2	C2	C2	C2	C2	C2	C6	C6	C6	C6	C6	C6	C6	C6	C6	C6			
7								B6	B6	B6	B6	B6	B6	B6	B6	B1	B1	B1	B1	B1	B1	B1	B1	B1	B1	B1*	B1*	B1*	B1*	B1*
8											A2	A2	A2	A2	A2	A2	A2	A2	A2	A2	A2	A5	A5	A5	A5	A5	A5	A7	A7	A7
9														C4	C4	C4	C4	C4	C1	C1	C1*	C1*	C1*	C1*	C1*	C1*	C1*			
PF ?	X	X	X	X		X	X	X			X	X			X	X	X	X	X	X		X			X			X	X	
QUEUE A	~~A1*~~	~~A3~~	~~A2~~	~~A1*~~	~~A4*~~	~~A5~~	A1	A4	A7	A9																				
QUEUE B	~~B2*~~	~~B4*~~	~~B6~~	~~B2~~	~~B4~~	B1*	B8	B3																						
QUEUE C	~~C1~~	~~C2~~	~~C4~~	~~C8~~	C6	C1*	C4																							
CANTIDAD DE FP = 20																														

ejer9 (pf).xlsx

10.Sean los procesos A, B y C tales que necesitan para su ejecución las siguientes páginas:

A: 1, 2, 1, 7, 2, 7, 3, 2

B: 1, 2, 5, 2, 1, 4, 5

C: 1, 3, 5, 1, 4, 2, 3

Si la secuencia de ejecución es tal que los procesos se ejecutan en la siguiente manera:

1. C demanda 1 página

2. A demanda 2 páginas

3. C demanda 1 página

4. B demanda 1 página

5. A demanda 1 página

6. C modifica la página 1

7. B demanda 2 páginas

8. A demanda 1 página

9. C demanda 1 página

10.B modifica la página 2

11.A modifica la página 2

12.B demanda 2 páginas

13.A demanda 1 página

14. **B demanda 2 páginas**
15. **C demanda 2 páginas**
16. **C demanda 1 página**
17. **A demanda 1 página**
18. **B termina**
19. **A demanda 2 páginas**
20. **C demanda 1 página**
21. **A termina**
22. **C termina**

Considerando una política de Asignación Dinámica y Reemplazo Global y disponiéndose de 7 marcos, debiéndose guardar 1 marco para la gestión de descarga asincrónica de páginas modificadas ¿Cuántos fallos de página se producirán si se utiliza la técnica de selección de víctima:

a. Segunda Chance

Segunda Chance

MARCOS	C1	A1	A2	C3	B1	A1	CM1	B2	B5	A7	C5	BM2	AM2	B2	B1	A2	B4	B5	C1	C4	C2	A7	BT	A3	A2	C3	AT	CT
1	C1	C1	C1	C1	C1	C1	C1 M*	C1 M*	C1 M	C1M	C1M	C1M																
2		A1	A1	A1	A1	A1*	A1*	A1*	A1	A1	A1	A1	A1	A1	B1	B1	B1	B1	B1	C4	C4	C4	C4	C4	C4	C4	C4	
3			A2	A2	A2	A2	A2	A2	B5	B5	B5	B5	B5	B5	B5	B5	B4	B4	B4	B4	C2	C2	C2	C2	C2	C2	C2	
4				C3	C3	C3	C3	C3	C3	A7	A7	A7	A7	A7	A7	A7	A7	B5	B5	B5	B5	A7	A7	A7	A7	A7		
5					B1	B1	B1	B1	B1	B1	C5	C5	C5	C5	C5	C5	C5	C5	C1	C1	C1	C1	C1	C1	C1	C3	C3	
6								B2	B2	B2	B2	BM2*	BM2	BM2*	BM2*	BM2*	BM2*	BM2*	BM2*	BM2	BM2	BM2		A3	A3	A3		
7													AM2	AM2	AM2	AM2*	AM2*	AM2*	AM2*	AM2	AM2	AM2	AM2	AM2	AM2*	AM2		
PF?	X	X	X	X	X			X	X	X	X		X		X		X	X	X	X	X	X		X		X		
QUEUE	~~C1*~~	~~A1*~~	~~A2~~	~~C3~~	~~B1~~	~~B2*~~	~~C1M~~	~~A1~~	~~B5~~	~~A7~~	~~C5~~	~~BM2*~~	~~AM2*~~	~~B1~~	~~B4~~	~~B5~~	~~C1~~	~~BM2~~	AM2*	C4	C2	A7	C3					
CANT FALLOS = 19																		T		T	T	T	T					

b. FIFO

FIFO

MARCOS	C1	A1	A2	C3	B1	A1	CM1	B2	B5	A7	C5	BM2	AM2	B2	B1	A2	B4	B5	C1	C4	C2	A7	BT	A3	A2	C3	AT	CT
1	C1	C1	C1	C1	C1	C1	CM1	CM1											C1	C1	C1	C1	C1	C1	C1	C1	C1	
2		A1	A1	A1	A1	A1	A1	A1	A1	A7	A7	A7	A7	A7	A7	A7	A7	A7	A7	A7	C2	C2	C2	C2	C2	C2	C2	
3				A2	A2	A2	A2	A2	A2	A2	A2	C5	C5	C5	C5	C5	C5	C5	C5	C5	C5	A7	A7	A7	A7	A7		
4					C3	C3	C3	C3	C3	C3	C3	C3	AM2	AM2	AM2	AM2	AM2	AM2	AM2	AM2	AM2	AM2	AM2	AM2	AM2			
5						B1	B1	B1	B1	B1	B1	B1	B1	B1	B1	B1	B4	B4	B4	B4	B4	B4		A3	A3	A3		
6									B2	B2	B2	BM2	BM2	BM2	BM2	BM2	BM2	BM2								C3	C3	
7									B5	B5	B5	B5	B5	B5	B5	B5	B5	B5	B5	C4	C4	C4	C4	C4	C4	C4	C4	
PF?	X	X	X	X	X			X	X	X	X		X				X		X	X	X	X		X		X		
QUEUE	~~C1M~~	~~A1~~	~~A2~~	~~C3~~	~~B1~~	~~BM2~~	~~B5~~	~~A7~~	~~C5~~	~~AM2~~	~~B4~~	C1	C4	C2	A7	A3	C3											
CANT FALLOS = 17											T	T	T	T	T	T	T											

c. LRU

LRU

MARCOS	C1	A1	A2	C3	B1	A1	CM1	B2	B5	A7	C5	BM2	AM2	B2	B1	A2	B4	B5	C1	C4	C2	A7	BT	A3	A2	C3	AT	CT
1	C1	C1	C1	C1	C1	C1	CM1	CM1	CM1	CM1	CM1	CM1	CM1	CM1						C4	C4	C4	C4	C4	C4	C4	C4	
2		A1	A1	A1	A1	A1	A1	A1	A1	A1	A1	A1	AM2	AM2	AM2	AM2	AM2	AM2	AM2	AM2								
3			A2	A2	A2	A2	A2	A2	B5	B5	B5	B5	B5	B5	B5	B5	B4	B4	B4	B4	B4	B4	B4	A3	A3	A3		
4				C3	C3	C3	C3	C3	C3	A7	A7	A7	A7	A7	A7	A7	A7	B5	B5	B5	B5	B5			A2	A2		
5					B1	B1	B1	B1	B1	B1	C5	C5	C5	C5	C5	C5	C5	C5	C1	C1	C1	C1	C1	C1	C1	C3	C3	
6								B2	B2	B2	B2	BM2	BM2	BM2	BM2	BM2	BM2	BM2	BM2		C2	C2	C2	C2	C2	C2	C2	
7															B1	B1	B1	B1	B1	B1	B1	A7	A7	A7	A7	A7		
PF?	X	X	X	X	X			X	X	X	X		X		X		X	X	X	X	X	X		X	X	X		
QUEUE	CM1	A1	A2	C3	B1	B2																						
CANT FALLOS = 20																												

ejer10 (pf con M).xlsx

11. **Hiperpaginación (Trashing)**

a. ¿Qué es?

-Ocurre cuando un sistema pasa mas tiempo paginando que ejecutando procesos

b. ¿Cuáles pueden ser los motivos que la causan?

-Ocurre cuando los procesos en ejecucion no tienen asignados suficientes marcos para mantenr su working set. Provoca una serie de fallos de pagina consetucivos ya que las paginas reemplazadas pertenecen al conjunto de

trabajo de algun proceso y esto geenra un ciclo. Surge por la falta de memoria RAM suficiente para satisfacer toda la llegada de ejecuciones de procesos

c. ¿Cómo la detecta el SO?

-Cuando un proceso pasa mas tiempo paginando que en ejecucion, tu p esta llegando a 1 y baja el performance del sistema y entra en thrashing. Siendo p una tasa que mide la proporcion de fallos respecto a los accesos a memoria de un proceso

-Para identificarlo vemos el posible ciclo de trashing:

-Monitoreo de CPU

-Si hay baja de utilizacion aumenta el grado de multiprogramacion y admitis mas procesos

-Si el algoritmo de reemplazo es global se sacan frames a otros procesos

-Un proceso necesita mas frames y aumentan los page faults y el robo de frames a otros procesos

-Swapping de paginas y encolamiento de dispotivos, baja el uso de Cpu y se repite el ciclo

d. Una vez que lo detecta, ¿qué acciones puede tomar el Kernel para eliminar este problema?

-Se limita usando algoritmos de reemplazo local, evitando que otros procesos te roben frames y reduciendo los page faults

-Se usa la estrategia de working set (paginas del proceso en memoria) o el algoritmo de PFF (frecuencia de fallos de pagina)

-Los procesos tienen referencias a memoria que fluctuan en las mismas localidades, es decir, las referencias del proceso se ejecutan recurrentemente en las mismas zonas. Se intenta aprovechar este principio de localidad para intentar hacer una buena implementacion de paginacion por demanda

-Mantener la localidad de cada proceso en memoria principal. Requiere de una correcta gfestion del conjunto residente (working set)

-Utilizar reemplazo local: control de la tasa p

-Algoritmo de reemplazo acorde: seleccion de victima que no forme parte de la localidad actual

-Tamaño acorde: que contenga la localidad sin frames de mas

12.Analice y compare las técnicas de PFF (Page Fault Frequency) y del Working Set. ¿Como las mismas permiten determinar la existencia de trashing?

-PFF: Controla la frecuencia de fdallos de pagina de cada proceso. Mide cada cuanto tiempo ocurre y usa ese valor para ajustar dinamicamente la cantidad de marcos asignados a un proceso

-Se definen dos umbrales:

-Umbral inferior (L): si el proceso tiene muy pocos fallos, significa que tiene *demasiados* marcos → se le pueden quitar

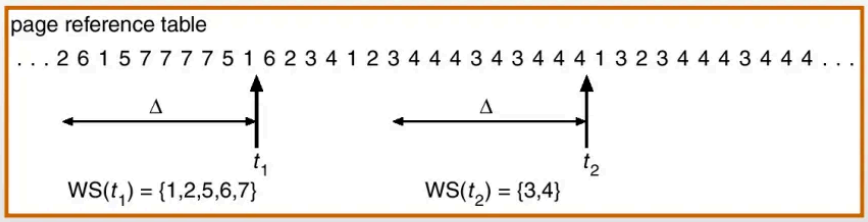
-Umbral superior (U): si el proceso tiene muchos fallos, significa que tiene *pocos* marcos → se le deben dar mas

-Working set: Conjunto de paginas de un proceso que ha utilizado dado un instante de tiempo

-Dado un instante de tiempo en el que freno el sistema se analiza el pasado con una ventana(delta) y se arma un conjunto que contiene las paginas que fueron utilizadas en esa ventana de tiempo (conjuntos de WS al frenarse en t1 y t2)

-El delta no puede ser muy grande porque tenes que tratar de detectar la localidad, si es muy grande podes detectar un proceso que trabajo en una localidad pero actualmente ya no trabaja mas en ella y estarias tratando de tener una localidad mas grande de la que se necesita. Tampoco puede ser muy chico por falta de cobertura de localidad

☑ Δ=10



- ☑ m = cantidad frames disponibles
- ☑ WSS_i= tamaño del working set del proceso p_i.
- ☑ WSS_i=D;
- ☑ D= demanda total de frames.
- ☑ Si D>m, habrá ***thrashing***.

13.**Considere un sistema cuya memoria principal se administra mediante la técnica de paginación por demanda que utiliza un dispositivo de paginación, algoritmo de reemplazo global LRU y una política de asignación que reparte marcos equitativamente entre los procesos. El nivel de multiprogramación es actualmente, de 4.**

-<u>Ante las siguientes mediciones:</u>

a) Uso de CPU del 13%, uso del dispositivo de paginación del 97%.

b) Uso de CPU del 87%, uso del dispositivo de paginación del 3%.

c) Uso de CPU del 13%, uso del dispositivo de paginación del 3%.

-<u>Analizar para cada caso:</u>

-¿Qué sucede?

-¿Puede incrementarse el nivel de multiprogramación para aumentar el uso de la CPU?

-¿La paginación está siendo útil para mejorar el rendimiento del sistema?

a) Hiperpaginacion, el sistema pasa mas tiempo paginando que utilizando la CPU para procesos. Si incrementas el nivel de multiprogramacion en teoria creo que esto deberia empeorar asique no. La paginacion no esta siendo util para mejorar el rendimiento del sistema

b) No hay hiperpaginacion, los procesos estan usando la CPU y no se quedan esperando la resolucion de los page faults. Se puede aumentar el grado de multiprogramacion pero muy poco ya que la CPU esta casi al maximo(?. La paginacion esta siendo util para el rendimiento del sistema

c) No hay hiperpaginacion pero la CPU no se esta aprovechando del todo. Quizas este al pedo porque hay muchos lotes de procesos en espera de ejecucion y todavia no se ejecutaron o ni idea. Se puede aumentar el nivel de

multiprogramacion ya que disminuiria el tiempo de ocio de la CPU. La paginacion no esta siendo util ya que no se esta aprovechando al maximo la CPU

14.**Considere un sistema cuya memoria principal se administra mediante la técnica de paginación por demanda. Considere las siguientes medidas de utilización:**

-Utilización del procesador: 20%

-Utilización del dispositivo de paginación: 97,7%

-Utilización de otros dispositivos de E/S: 5%

-Cuales de las siguientes acciones pueden mejorar la utilización del procesador:

a) Instalar un procesador más rápido

-No mejoraria nada porque tenes una paginacion del 98% e hiperpaginacion. Por mas que mejores el procesador los procesos van a seguir esperando la resolucion de PF

b) Instalar un dispositivo de paginación mayor

-Mejoraria notablemente ya que la hiperpaginacion se reduciria o hasta podria desaparecer

c) Incrementar el grado de multiprogramación

-Empeoraria ya que habra mas procesos causando PF

d) Instalar mas memoria principal

-Mejoraria ya que das mas espacio para almacenar mas paginas y reducirias los tiempos de espera para que se resuelvan justamente los fallos de pagina, ya que va a haber mas

e) Decrementar el quantum para cada proceso

-Seguirian esperando de igual forma asique creo que no cambia nada

15.**La siguiente fórmula describe el tiempo de acceso efectivo a la memoria al utilizar paginación para la implementación de la memoria virtual:**

TAE = At + [(1 − p) * Am] + [p * (Tf + Am)]

-Donde:

➢ **TAE = tiempo de acceso efectivo**

➢ **p = tasa de fallo de página (0 <= p <=1)**

➢ **Am = tiempo de acceso a la memoria real**

➢ **Tf = tiempo se atención de una fallo de pagina**

➢ **At = tiempo de acceso a la tabla de páginas. Es igual al tiempo de acceso a la memoria (Am) si la entrada de la tabla de páginas no se encuentra en la TLB.**

Suponga que tenemos una memoria virtual paginada, con tabla de páginas de 1 nivel, y donde la tabla de páginas se encuentra completamente en la memoria.

Servir una falla de página tarda 300 nanosegundos si hay disponible un marco vacío o si la página reemplazada no se ha modificado, y 500 nanosegundos si se ha modificado. El tiempo de acceso a memoria es de 20 nanosegundos y el de acceso a la TLB es de 1 nanosegundo

a. Si suponemos una tasa de fallos de página de 0,3 y que siempre contamos con un marco libre para atender el fallo ¿Cuál será el TAE si el 50% de las veces la entrada de la tabla de páginas se encuentra en la TLB (hit)?

-p = 0,3

-Am = 20 ns

-At= 0.5(1ns) + 0.5(20 ns) = 0.5 + 10 = 10.5 ns

TAE = 10.5 + (1 - 0.3) (20) + 0.3 (300 + 20)

TAE = 10.5 + 0.7 (20) + 0.3 (320)

TAE = 120.5 ns

b. Si suponemos una tasa de fallos de página de 0,3; que el 70% de las ocasiones la página a reemplazar se encuentra modificada. ¿Cuál será el TAE si el 60% de las veces la entrada de la tabla de páginas se encuentra en la TLB (hit)?

-p = 0.3

-Am = 20 ns

-70% modificada → Tf = 0.3(300) + 0.7(500)

-TLB hit = 60% → At = 0.6(1 ns) + 0.4(20 ns)

TAE = 8.6 + (1 − 0.3)(20) + 0.3 (440 + 20)

TAE = 8.6 + 0.7 (20) + 0.3 (460)

TAE = 8.6 + 14 + 138

TAE = 160.6 ns

c. Si suponemos que el 60% de las veces la página a reemplazar está modificada, el 100% de las veces la entrada de la tabla de páginas requerida se encuentra en la TLB (hit) y se espera un TAE menor a 200 nanosegundos.

-Am = 20 ns

-Tf = 0.4(300) + 0.6(500)

-TLB hit = 100% → At = 1 ns

-TAE < 200 ns

TAE = 1 + (1 − p)(20) + p(420 + 20)

TAE = 1 + 20 − 20p + 440p

TAE = 21 + 420p

-Se quiere TAE < 200:

21 + 420p < 200

420p < 179

p < 179/420

p < **0.426**

16.Considere el siguiente programa:

```
#define Size 64
int A[Size; Size], B[Size; Size], C[Size; Size];
 int register i, j;
    for (j = 0; j < Size; j ++)
        for (i = 0; i < Size; i++)
            C[i; j] = A[i; j] + B[i; j];
```

Si asumimos que el programa se ejecuta en un sistema que utiliza paginación por demanda para administrar la memoria, donde cada página es de 1Kb. Cada número entero (int) ocupa 4 bytes. Es claro que cada matriz requiere de 16 páginas para almacenarse. Por ejemplo: A[0,0]..A[0,63], A[1,0]..A[1,63], A[2,0]..A[2,63] y A[3,0]..A[3,63] se almacenará en la primera página.

Asumamos que el sistema utiliza un working set de 4 marcos para este proceso. Uno de los 4 marcos es utilizado por el programa y los otros 3 se utilizan para datos (las matrices). También consideramos que para los índices "i" y "j" se

utilizan 2 registros, por lo que no es necesario el acceso a la memoria para estas 2 variables.

a. Analizar cuántos fallos de páginas ocurren al ejecutar el programa (considere las veces que se ejecuta C[i,j] = A[i,j] + B[i,j])

-64×64 = 4096

-4096 × 4 bytes = 16384 bytes → 16 kb

-Cada matriz ocupa 16 paginas

- 16 + 16 + 16 = 48 fallos → **48 × 64 (columnas) = 3072 fallos** ?

b. Puede ser modificado el programa para minimizar el número de fallos de páginas. En caso de ser posible indicar la cantidad de fallos de páginas que ocurren.

17.Considere las siguientes secuencias de referencias a páginas de los procesos A y B, donde se muestra en instante de tiempo en el que ocurrió cada una (1 a 78):

Proceso A							
1	2	3	4	5	6	7	8
1	2	1	3	4	3	3	2
40	41	42	43	44	45	46	47
2	2	1	2	1	1	2	3

Proceso B							
1	2	3	4	5	6	7	8
1	2	3	3	3	4	4	5
40	41	42	43	44	45	46	47
2	2	3	3	4	4	4	3

a. Considerando una ventana Δ=5, indique cuál sería el conjunto de trabajo de los procesos A y B en el instante 24 (WSA(24) y WSB(24))

-WSA(24) = 4, 5 , 6 (5 hacia la izquierda desde el instante 24)
-WSB(24) = 2, 3, 5 ,6 (5 hacia la izquierda desde el instante 24)

b. Considerando una ventana Δ=5, indique cuál sería el conjunto de trabajo de los procesos A y B en el instante 60 (WSA(60) y WSB(60))

-WSA(60) = 1, 2, 3, 4, 5

-WSB(60) = 3, 4, 5

c. Para el los WS obtenidos en el inciso a), si contamos con 8 frames en el sistema ¿Se puede indicar que estamos ante una situación de trashing? ¿Y si contáramos con 6 frames?

-No ya que WSA requiere de 3 frames y el WSB requiere de 4, es decir, 7 en total por lo que con 8 alcanza. En caso de tener 6 frames estariamos en trashing

d. Considerando únicamente el proceso A, y suponiendo que al mismo se le asignaron inicialmente 4 marcos, donde el de reemplazo de páginas es realizado considerando el algoritmo FIFO. ¿Cuál será la tasa de fallos en el instante 38 de páginas suponiendo que la misma se calcula contando los fallos de páginas que ocurrieron en las últimas 10 unidades de tiempo?

-6

e. Para el valor obtenido en el inciso d), si suponemos que el S.O. utiliza como límites superior e inferior de tasa de fallos de páginas los valores 2 y 5 respectivamente ¿Qué acción podría tomar el Kernel respecto a la cantidad de marcos asignados al proceso?

-Al tener como limite inferior 2 y superior 5 y tener 6 de tasa de fallo de pagina en el instante 38 estariamos en situacion de trashing. Creo que una buena solucion seria agregar 1 o 2 marcos mas al Proceso A ya que eso sin dudas reduciria la tasa de fallo y quedaria dentro de los limites establecidos en el problema

Administración de E/S

18.Dispositivos

a. Los dispositivos, según la forma de transferir los datos, se pueden clasificar en 2 tipos:

i. Orientados a bloques

ii. Orientados a flujos

b. Describa las diferencias entre ambos tipos.

-Orientados a bloques:

-Transferencia de datos se realiza en bloques de tamanio fijo. Cada bloque se lee o escribe como unidad completa

-Los datos se agrupan en bloques antes de ser transferidos entre la memoria y el dispositivo. Cada bloque itene una estructura fija y se identifica por numero de bloqu

-Transferencias en bloques mas eficientes en rendimiento

-Orientados a flujos:

-Transferencia de datos de manera continua sin estructura fija

-No se dividen los datos en bloques, se transmiten como flujo constante de bytes

-Eficientes para operaciones secuenciales donde se requiere este flujo

-No hay unidad de transferencia predeterminada

c. Cite ejemplos de dispositivos de ambos tipos.

-Orientados a bloques → SSD, disco magnetico

-Orientados a flujos → Impresora, teclado, mouse, etc

d. Enuncie las diferencias que existen entre los dispositivos de E/S y que el Kernel debe considerar.

-Tipos de dispositivos

-Diferentes formas de realizar E/S dependiendo del dispositivo

-Velocidad

-Caracteristicas de dispositivos

19.**Técnicas de E/S. Describa cómo trabajan las siguientes técnicas de E/S**

a. E/S programada

-Cpu controla directamente todas las operaciones de transferencia de datos entre la memoria y los perifericos. Emite comandos y supervisa el estado del dispositivo periferico y espera activamente hasta que la E/S se complete

b. E/S dirigida por interrupciones

-CPU emite un comando de E/S al periferico y continua con otras tareas. Cuando el periferico termina, envia una interrupcion a la CPU para avisarle que los datos estan listos

c. DMA (Acceso Directo a Memoria)

-Se usa el controlador de DMA para transmitir los datos directamente entre el periferico y la memoria sin uso de CPU. La CPU solo se encarga de iniciar el controlador con la direccion de inicio y la cantidad de datos a transferir y luego se va a hacer otras tareas, dejandole el resto del trabajo al DMA

20.**La técnica de E/S programa puede trabajar de dos formas:**

Indique cómo trabajan estas 2 técnicas.

a. E/S mapeada

-Dispositivos y memoria comparten el espacio de direcciones. Las operaciones de E/S pasan a ser como de R/W en la memoria por lo que no hay instrucciones especiales para E/S

b. E/S aislada

-Hay un espacio de direcciones para E/S. Se necesitan puertos y lineas de E/S

21.**Enuncie las metas que debe perseguir un SO para la administración de la entrada/salida.**

-Eficencia: gestionar recursos de E/S para minimziar tiempos de espera y maximizar el uso de los dispositivos

-Desempenio: mejorar rendimiento con tecnicas de cache, buffering y planificacion eficiente

-Abstraccion de HW: proveer una interfaz que oculte los detalles del HW para facilitar la portabilidad y el desarrollo de aplicaciones

-Concurrencia: operaciones simultaneas de E/S y gestionar correctamente el acceso compartido entre procesos

-Gestion de dispositivos: manejar diferentes tipos de dispositivos mediante controladores adecuadamente asignados

22.**Drivers**

a. ¿Qué son?

-Interfaces entre el SO y el HW,m forman parte del espacio de memoria del Kernel y se cargan como modulos de este. Contienen codigo dependiente del dispositivo para poder manejarlo y se encarga de traducir los requerimientos abstractos en los comandos para el dispositivo escribiendo sobre los registros del controlador, accediento a la memoria mapeada y encolando requerimientos

b. ¿Qué funciones mínimas deben proveer?

-Inicializacion y configuracion:

-Preparar dispositivo paar su uso, realizar diagnosticos iniciales para verificar que funcione bien

-Interfaz de comunicacion:

-Proveer funciones para enviar comandos al dispositivo

-Facilitar transferencia de datos entre el sistema y el dispositivo

-Gestion de interrupciones:

-Manejar interrupciones generadas por el dispositvo para avisar al SO sobre eventos importantes

-Control de errores:

-Detectar y reportar errores e implementar mecanismos basicos para manejar los fallos del dispositivo

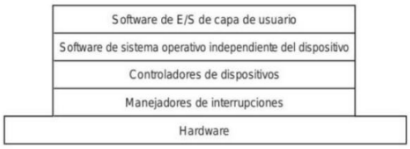
-Operaciones de E/S basicas:

-Leer/Escribir datos del dispositivos desde y hacia el

-Asegurar que als operaciones se realicen en el orden correcto

c. ¿Quién determina cuáles deben ser estas funciones?

23.**Realice un gráfico que marque la relación entre el Subsistema de E/S, los drivers, los controladores de dispositivos y los dispositivos.**



(Diapositiva de teoria)

24 - 25 (repetidas). **Describa mediante un ejemplo los pasos mínimos que se suceden desde que un proceso genera un requerimiento de E/S hasta que el mismo llega al dispositivo.**

-Ejemplo: un dispositivo ejecuta el comando open ('home/juan/p')

-El gestor de E/S tiene que identificar en que particion se encuentra el directorio a abrir

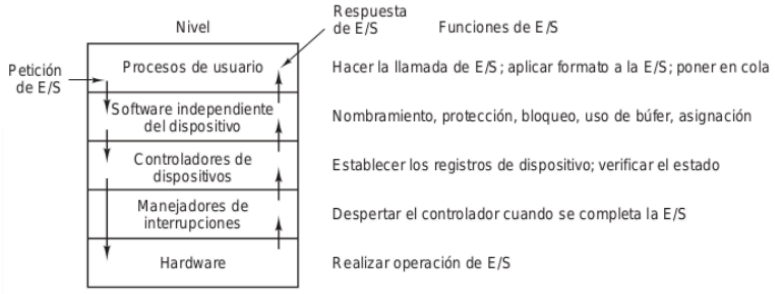
-Una vez identifica en que direccion se encuentra el archivo al sistema de archivos le llega operacion con el resto del path

imaginando que home esta en /dev/hda1 y filesystem→ EXT3

seria → open ('dev/hda', 'juan/p')

-El filesystem necesita el driver ya que no se tiene a mano el/los bloques correspondientes a la solicitud de apertura de archivo. Por ejemplo se le pide el bloque 3 de X dispositivo y el driver lo proporcionara

-Se llega a nivel de HW y se hace un IN en el disco



(Diagrama resumen, (diapositivas 17-21) - Ejemplo de clase)

26.**Enuncie qué servicios provee el Kernel para la administración de E/S.**

-Buffering → Almacenamiento de datos en memoria mientras se transfieren

-Solucionar problemas de velocidad entre los dispositivos subiendo datos a memoria (si te entra informacion mas rapido de lo que lo podes sacar se te empiezan a sobreescribir, por lo que los sube a memoria en caso de ser mas rapido)

-Solucionar problemas de tamaño de los datos entre dispositivos (el tamaño de paquete que entra es diferente al tamaño de paquete que puedo enviar, los sacas, los pones en memoria y los procesas ahi para reacomodar el tamaño al cual es aceptable)

-Caching → Mantener copias de los datos de acceso reciente en memoria (mejora velocidad de E/S ya que si esta disponible no necesitas acceder a disco)

-Spooling → Administra cola de requerimientos de un dispositivo

-Dispositivos de acceso exclusivo no pueden atender distintos requerimientos al mismo tiempo (por ejemplo impresora) lo mete en una capa al software que simula una cola para el dispositivo

-Coordinar acceso concurrente al dispositivo

-Si en toda la cadena ocurre un error el subsistema/gestor de E/S es el que se tiene que encargar de informarla de manera 'entendible' para el usuario

-Formas de realizar E/S:

-Bloqueante: el proceso se bloquee hasta que la oepracion de E/S se completa

-No bloqueante: el proceso de E/S retorna cuanto antes es posible

Administración de Discos

27.**Describa en forma sintética, cómo es la organización física de un disco, puede utilizar gráficos para mayor claridad.**

-Platos:

-Superficies circulares recubiertas de material magnético, donde se almacena la información. Cada disco puede tener múltiples platos apilados.

-Pistas:

-Concentros circulares en la superficie de cada plato, organizados desde el borde hacia el centro.

-Sectores:

-Divisiones de las pistas en segmentos más pequeños, que son las unidades mínimas de almacenamiento.

-Cilindros:

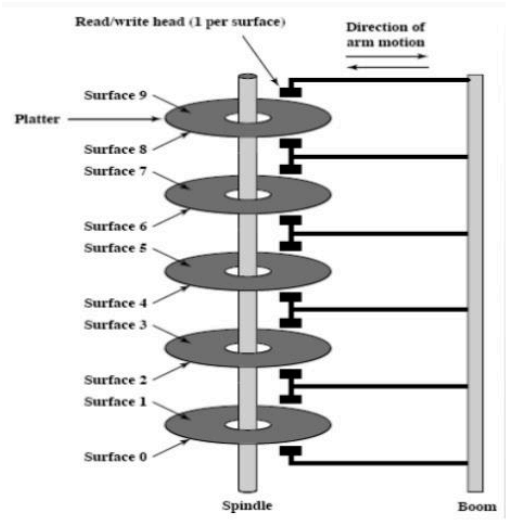
-Conjunto de pistas alineadas verticalmente en los platos, formando un cilindro lógico que puede ser accedido simultáneamente.

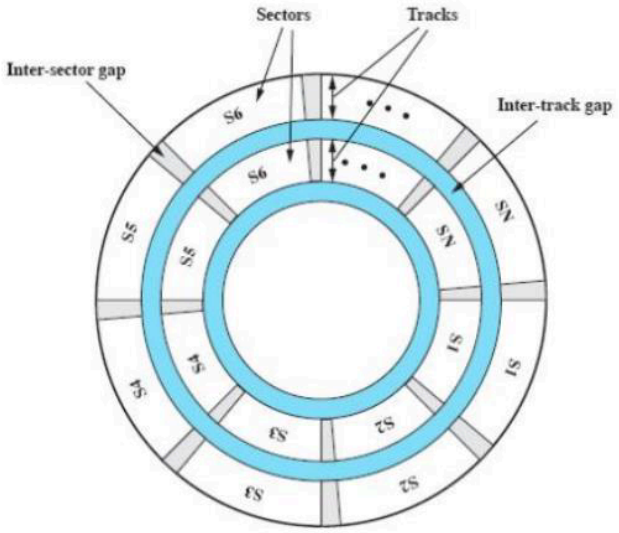
-Cabezas de lectura/escritura:

-Brazo mecánico con cabezas que se mueven sobre las pistas de los platos para leer o escribir datos

-Motor:

-Controla la rotación de los platos y el movimiento de las cabezas, permitiendo el acceso a las posiciones deseadas.





(Diapositivas e info de las diapositivos de la clase)

28.**La velocidad promedio para la obtención de datos de un disco está dada por la suma de los siguientes tiempos:**

De una definición para cada uno.

➢ ***Seek Time***

-Tiempo que tarda en posicionarse la cabeza en el cilindro

➢ ***Latency Time***

-Tiempo que transcurre desde que la cabeza se posiciona en el cilindro hasta que el sector buscado pase por abajo de la cabeza

➢ ***Transfer Time***

-Tiempo de transferencia del bloque del disco a la memoria

29.**Suponga un disco con las siguientes características:**

➢ **7 platos con 2 caras utilizables cada uno.**

➢ **1100 cilindros, con 300 sectores por pista, donde cada sector es 512 bytes.**

➢ **Seek Time de 10 ms**

➢ **9000 RPM .**

➢ **Velocidad de Transferencia de 10 MiB/s (Mebibytes por segundos).**

a. Calcule la capacidad total del disco.

- (7 × 2) x 1100 × 300 × 512 bytes

-2.365.440.00 bytes / 2 ^ 30

-**2.2 GB**

b. ¿Cuántos sectores de disco se requerirían si se quiere almacenar 3 MiB(MebiBytes) de datos?

-3 mib x 2^20 = 3.145.728 bytes

-Sectores = 3.145.728 / 512 → **6144**

c. Calcule el tiempo de latencia.

-9000 RPM / 60 s → 150 RPS (Revoluciones por Segundo)

-1/150 → 0.00666 (lo que tarda una vuelta en segundos)

→ 0.0066 × 1000 → 6.67 ms

Tiempo de latencia → 6.67/2 → 3.33 MS

30. **Originalmente, los sectores de un disco se identificaban por la terna de valores (C,H,S) esto es Cylinder-head-sector (Cilindro, Cabeza , Sector), con C desde 0 hasta la cantidad de cilindros menos 1, H desde 0 hasta la cantidad de cabezas lecto/grabadoras menos 1 y S desde 1 hasta la cantidad de sectores por pista.**

En los '90 los controladores de disco reemplazaron la identificación de los sectores por un modelo numérico denominado LBA (Logical Block Addressing). La numeración comenzaba en 0 y alcanzaba la cantidad total de sectores menos 1. Cada sector en formato (C,H,S) se puede traducir al LBA a partir de la siguiente fórmula:

A = (C * Nheads + H) * Nsectors+ (S -1)

Donde:

➔ A es el valor LBA a obtener

➔ Nheads es la cantidad de cabezas lecto/grabadoras

➔ Nsectors es la cantidad de sectores por pista

También se puede obtener el valor de la terna (C,H,S) a partir de un valor LBA con las

siguientes fórmulas:

*C = LBA div (Nheads * Nsectors)*	*H =Resto de (LBA mod (Nheads * Nsectors)) / Nsectors*	*S = (LBA mod (Nheads * Nsectors)) mod Nsectors + 1*

a. Si se tiene un disco con 1020 cilindros, 16 cabezas y 63 sectores por pista, calcule la cantidad total de sectores que tendrá el disco y determine el valor LBA para las siguientes ternas (C,H,S):

i. (3, 2, 1) **iii. (1019, 15, 63)**

ii. (0, 0, 1) **iv. (500, 8, 44)**

b. Determine el valor de la terna (C,H,S) para los siguientes valores LBA:

i. 1 **iii. 14890**

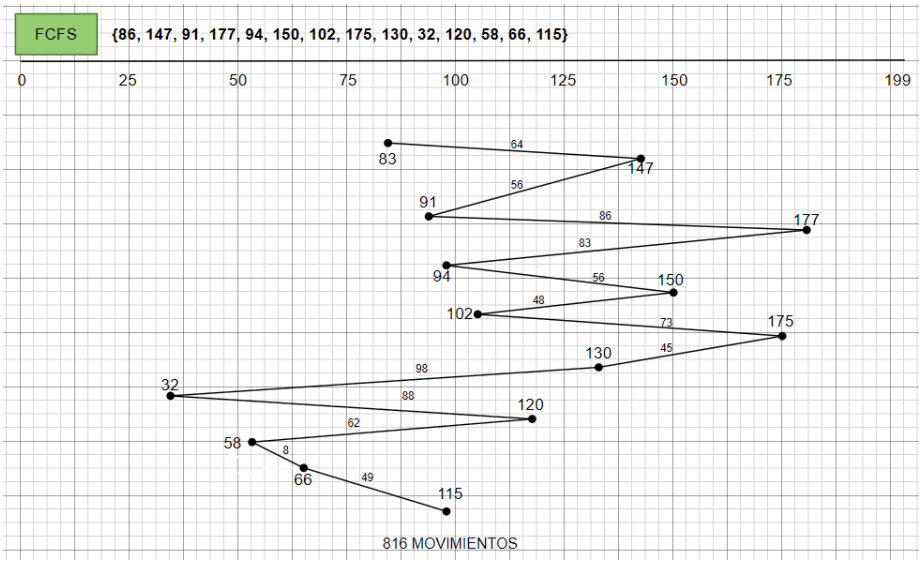
ii. 100 **iv. 45687**

31.**El *Seek Time* es el parámetro que posee mayor influencia en el tiempo real necesario para transferir datos desde o hacia un disco. Es importante que el Kernel planifique los diferentes requerimientos al disco para minimizar el movimiento de la cabeza lecto-grabadora.**

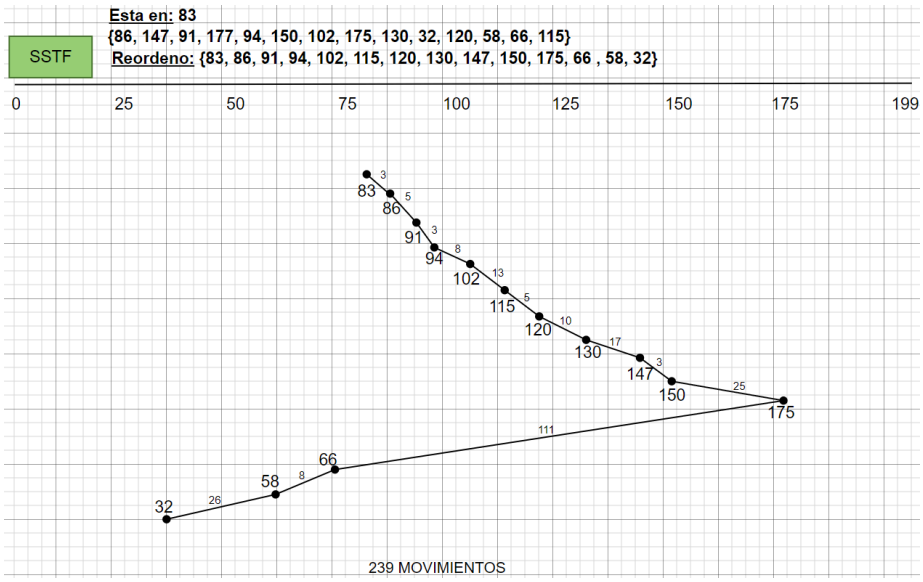
Analicemos las diferentes políticas de planificación de requerimientos a disco con un ejemplo: Supongamos un *Head* con movimiento en 200 *tracks* (numerados de 0 a 199), que está en el track 83 atendiendo un requerimiento y anteriormente atendió un requerimiento en el track 75.

Si la cola de requerimientos es: {86, 147, 91, 177, 94, 150, 102, 175, 130, 32, 120, 58, 66, 115}. Realice los diagramas para calcular el total de movimientos de head para satisfacer estos requerimientos de acuerdo a los siguientes algoritmos de scheduling de discos:

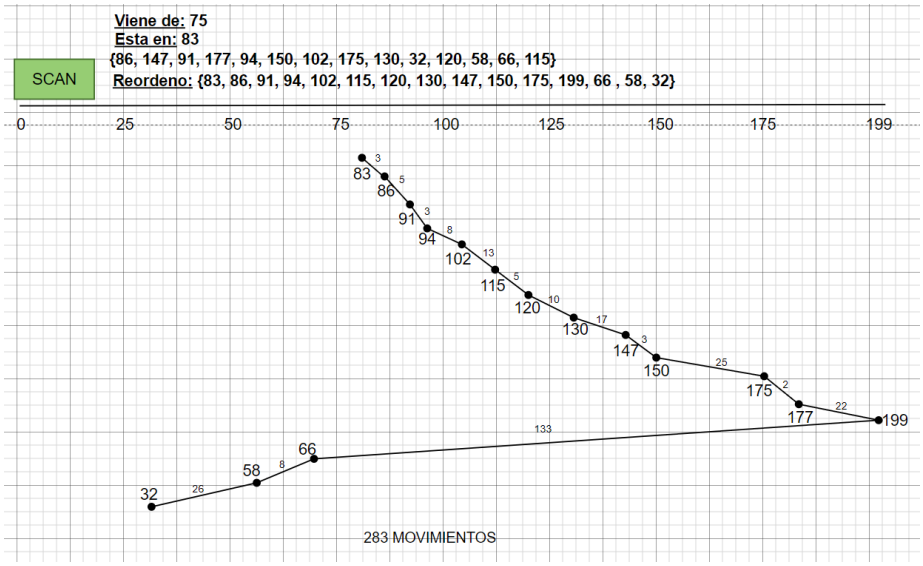
a. FCFS (*First Come, First Served*)



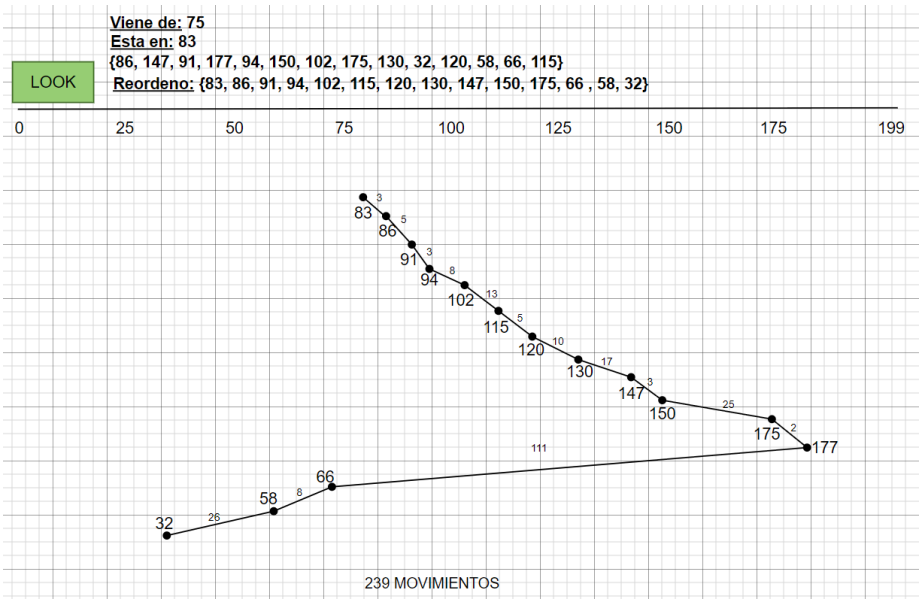
b. SSTF (*Shortest Seek Time First*)


Esta en: 83
{86, 147, 91, 177, 94, 150, 102, 175, 130, 32, 120, 58, 66, 115}
Reordeno: {83, 86, 91, 94, 102, 115, 120, 130, 147, 150, 175, 66 , 58, 32}
SSTF
0
25
50
75
100
125
150
175
199
83
86
91
94
102
115
120
130
147
150
175
66
58
32
3
5
3
8
13
5
10
17
3
25
111
26
8
239 MOVIMIENTOS

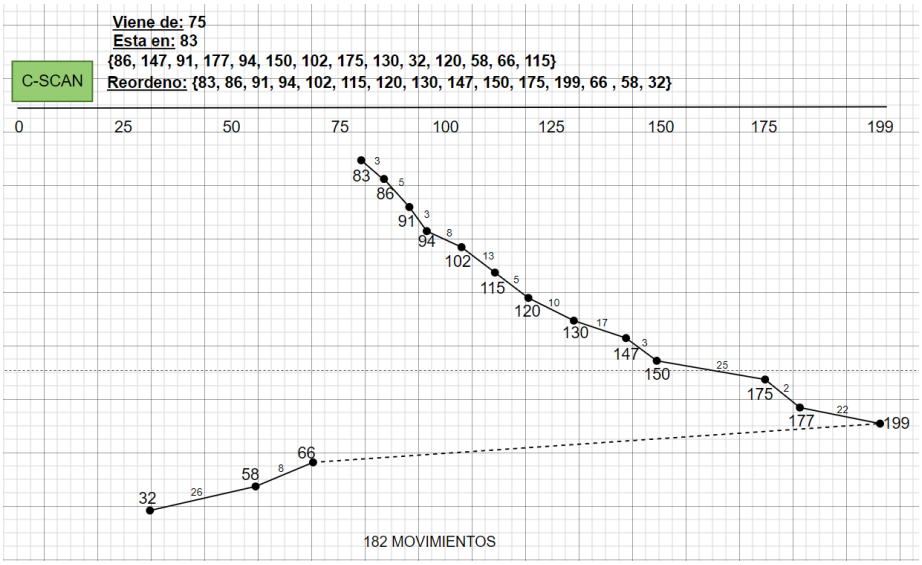
c. Scan


Viene de: 75
Esta en: 83
{86, 147, 91, 177, 94, 150, 102, 175, 130, 32, 120, 58, 66, 115}
Reordeno: {83, 86, 91, 94, 102, 115, 120, 130, 147, 150, 175, 199, 66 , 58, 32}
SCAN
0
25
50
75
100
125
150
175
199
83
86
91
94
102
115
120
130
147
150
175
177
199
66
58
32
3
5
3
8
13
5
10
17
3
25
2
22
133
26
8
283 MOVIMIENTOS

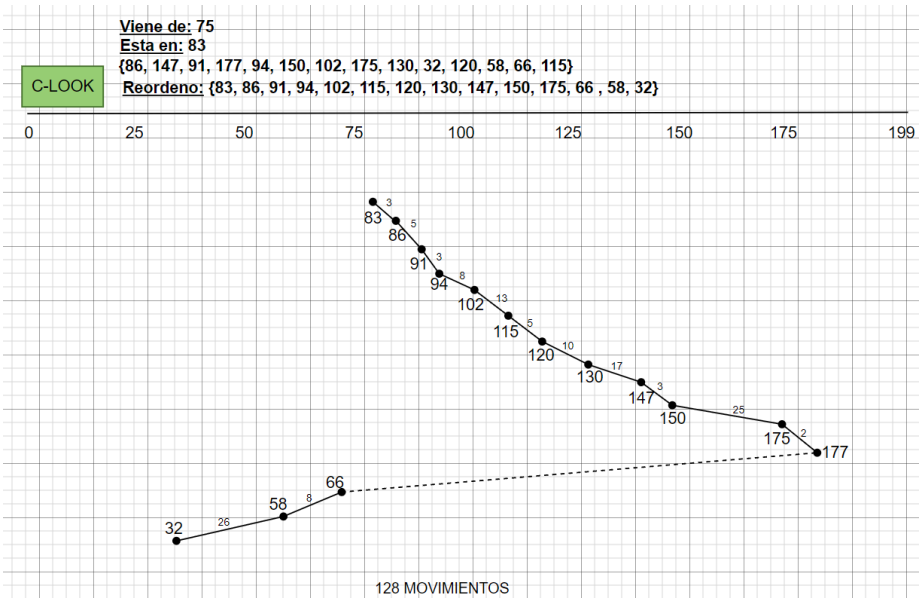
d. Look



e. C-Scan (*Circular Scan*)



f. C-Look (*Circular Look*)



32.**¿Alguno de los algoritmos analizados en el ejercicio anterior pueden causar inanición de requerimientos?**

-SSTF puede

33.**Supongamos un disco con 300 pistas (numerados de 0 a 299), donde la cabeza se encuentra en la pista 143 atendiendo un requerimiento y anteriormente atendió un requerimiento en la pista 125.**

Si la cola inicial de requerimientos es: {126, 147, 81, 277, 94, 150, 212, 175, 140, 225, 280, 50, 99, 118, 22, 55} y luego de 30 movimientos se incorporan los requerimientos de las pistas {75, 115, 220 y 266}. Realice los diagramas para calcular el total de movimientos de head para satisfacer estos requerimientos de acuerdo a los siguientes algoritmos de scheduling de discos:

a. FCFS

b. SSTF

c. Scan

d. Look

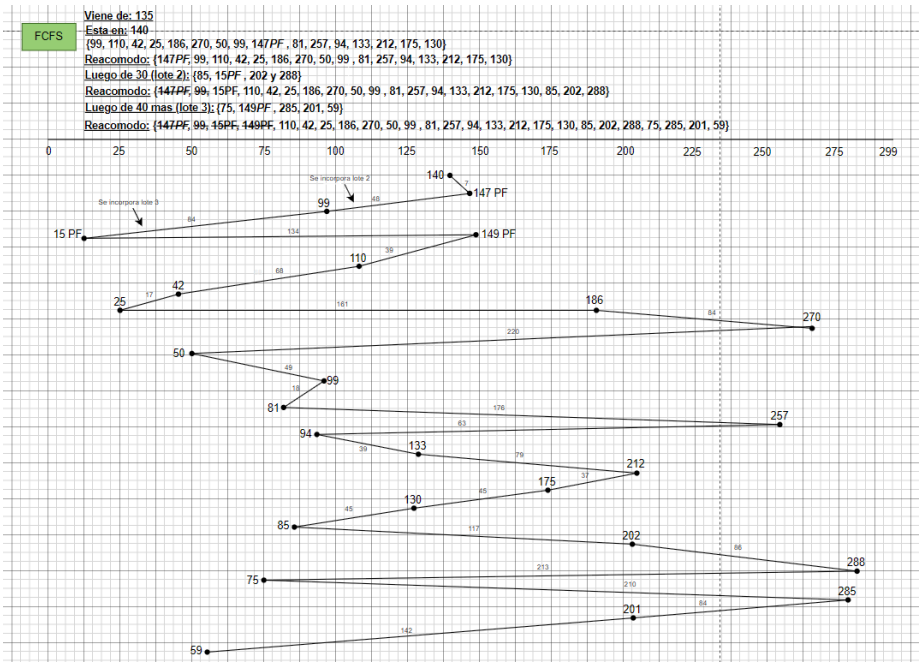
e. C-Scan

f. C-Look

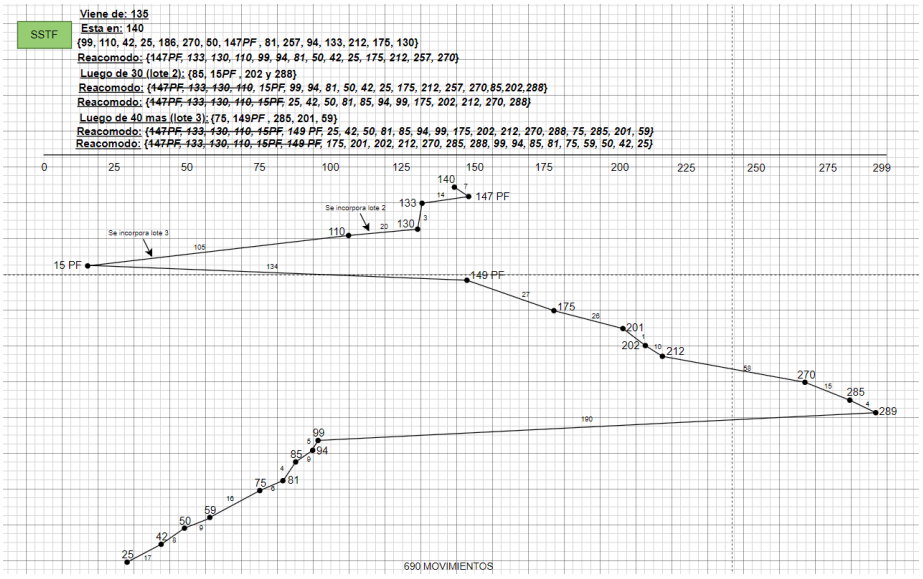
34.**Supongamos un disco con 300 pistas (numerados de 0 a 299), donde la cabeza se encuentra en la pista 140 atendiendo un requerimiento y anteriormente atendió unrequerimiento en la pista 135.**

Si la cola de requerimientos es: {99, 110, 42, 25, 186, 270, 50, 99, 147*PF* , 81, 257, 94, 133, 212, 175, 130} y después de 30 movimientos se incorporan los requerimientos de las pistas {85, 15*PF* , 202 y 288} y después de otros 40 movimientos más se incorporan los requerimientos de las pistas {75, 149*PF* , 285, 201 y 59}. Realice los diagramas para calcular el total de movimientos de head para satisfacer estos requerimientos de acuerdo a los siguientes algoritmos de scheduling de discos:

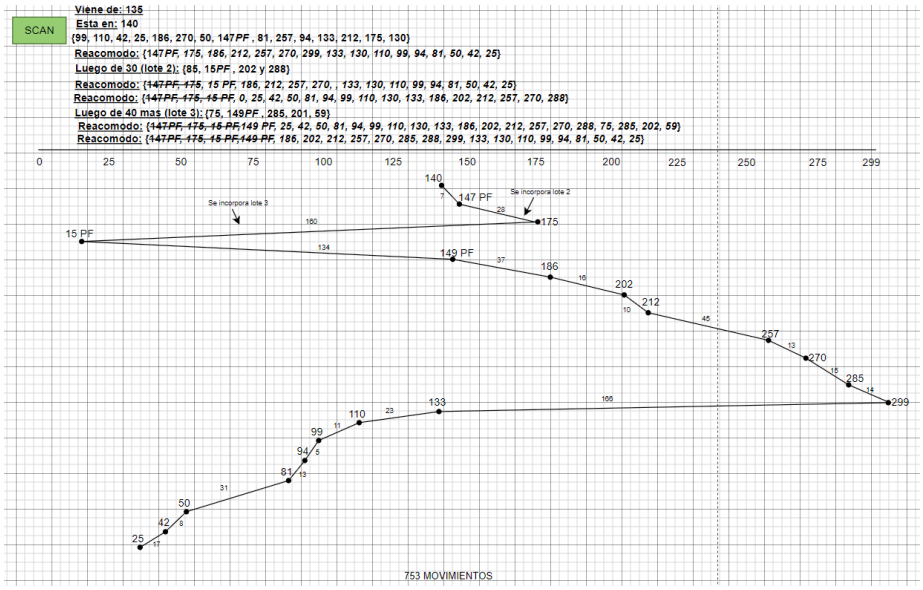
a. FCFS


FCFS
Viene de: 135
Esta en: 140
{99, 110, 42, 25, 186, 270, 50, 99, 147PF , 81, 257, 94, 133, 212, 175, 130}
Reacomodo: {147PF, 99, 110, 42, 25, 186, 270, 50, 99 , 81, 257, 94, 133, 212, 175, 130}
Luego de 30 (lote 2): {85, 15PF , 202 y 288}
Reacomodo: {147PF, 99, 15PF, 110, 42, 25, 186, 270, 50, 99 , 81, 257, 94, 133, 212, 175, 130, 85, 202, 288}
Luego de 40 mas (lote 3): {75, 149PF , 285, 201, 59}
Reacomodo: {147PF, 99, 15PF, 149PF, 110, 42, 25, 186, 270, 50, 99 , 81, 257, 94, 133, 212, 175, 130, 85, 202, 288, 75, 285, 201, 59}
0 25 50 75 100 125 150 175 200 225 250 275 299
Se incorpora lote 2
Se incorpora lote 3
140
147 PF
99
15 PF
149 PF
110
42
25
186
270
50
99
81
257
94
133
212
175
130
85
202
288
75
285
201
59

b. SSTF


SSTF
Viene de: 135
Esta en: 140
{99, 110, 42, 25, 186, 270, 50, 147PF , 81, 257, 94, 133, 212, 175, 130}
Reacomodo: {147PF, 133, 130, 110, 99, 94, 81, 50, 42, 25, 175, 212, 257, 270}
Luego de 30 (lote 2): {85, 15PF , 202 y 288}
Reacomodo: {147PF, 133, 130, 110, 15PF, 99, 94, 81, 50, 42, 25, 175, 212, 257, 270,85,202,288}
Reacomodo: {147PF, 133, 130, 110, 15PF, 25, 42, 50, 81, 85, 94, 99, 175, 202, 212, 270, 288}
Luego de 40 mas (lote 3): {75, 149PF , 285, 201, 59}
Reacomodo: {147PF, 133, 130, 110, 15PF, 149 PF, 25, 42, 50, 81, 85, 94, 99, 175, 202, 212, 270, 288, 75, 285, 201, 59}
Reacomodo: {147PF, 133, 130, 110, 15PF, 149 PF, 175, 201, 202, 212, 270, 285, 288, 99, 94, 85, 81, 75, 59, 50, 42, 25}
0 25 50 75 100 125 150 175 200 225 250 275 299
140
147 PF
133
Se incorpora lote 2
110
130
Se incorpora lote 3
15 PF
149 PF
175
201
202
212
270
285
289
99
94
85
81
75
59
50
42
25
690 MOVIMIENTOS

c. Scan



d. Look

-Igual que scan pero sin llegar al borde

e. C-Scan

-Igual que scan pero sin cambiar el sentido dps de los PF

f. C-Look

-Igual que el look pero sin cambiar el sentido dps de los PF

Administración de Archivos

35. **Dados las siguientes técnicas de administración de espació de un archivo:**

➢ **Asignación contigua**

➢ **Asignación enlazada**

➢ **Asignación indexada**

a. Describa cómo trabaja cada una.

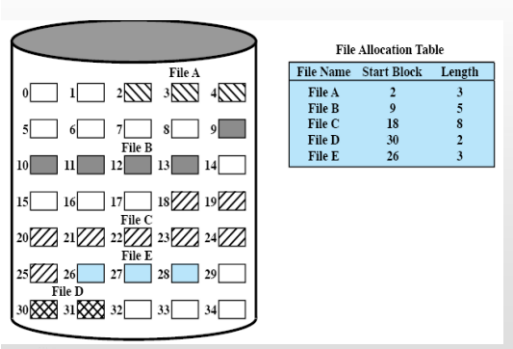
b. Cite ventajas y desventajas de cada una.

c. Indique para cada técnica si la misma puede producir fragmentación interna y/o externa.

-Asignacion contigua:

-Utiliza pre asignacion y el filesystem aloja a los clusters de manera consecutiva. Utiliza una FAT simple. Si tenes un cluster grande genera fragmentacion interna

-Al querer agregar un bloque de tamanio X que no tiene esa cantidad de bloques consecutivos se generaria fragmentacion externa (podes tener dos fragmentaciones por lo que tendrias que desfragmentar usando compactacion)



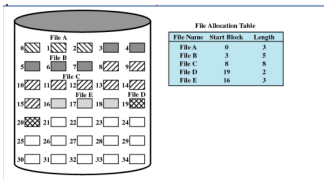


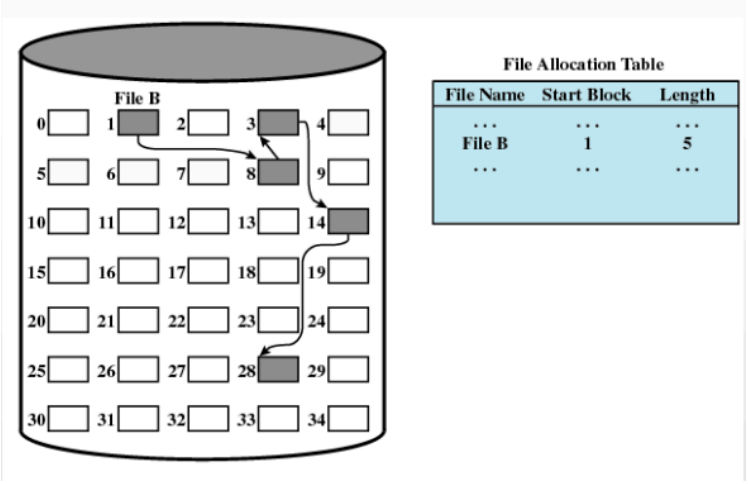
Figure 12.8 Contiguous File Allocation (After Compaction)

-Por ejemplo al querer agregar un archivo de 6 bloques no entraria y generaria f.e. Al compactarlo para solucionar la fragmentacion quedaria como en la imagen derecha

-Asignacion enlazada:

-Se hace en base a bloques individuales y cada bloque tiene un puntero al proximo bloque del archivo y el tamanio. FAT simple ya que se almacena bloque de inicio y tamanio del archivo

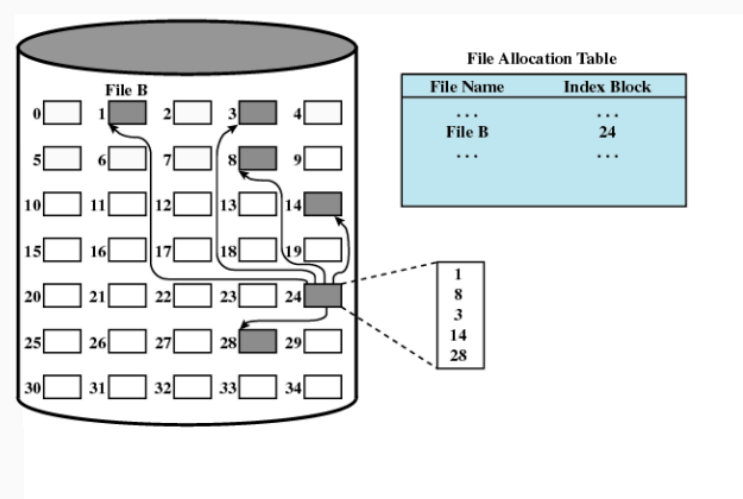
-No tiene fragmentacion externa, los archivos pueden crecer bajo demanda. Pueden consolidarse bloques del mismo archivo y ponerlos cerca para garantizar cercania en los bloques del mismo archivo



-Asignacion indexada:

-FAT tiene puntero al bloque indice. Tenes un puntero al bloque indice el cual apunta a todos los archivos

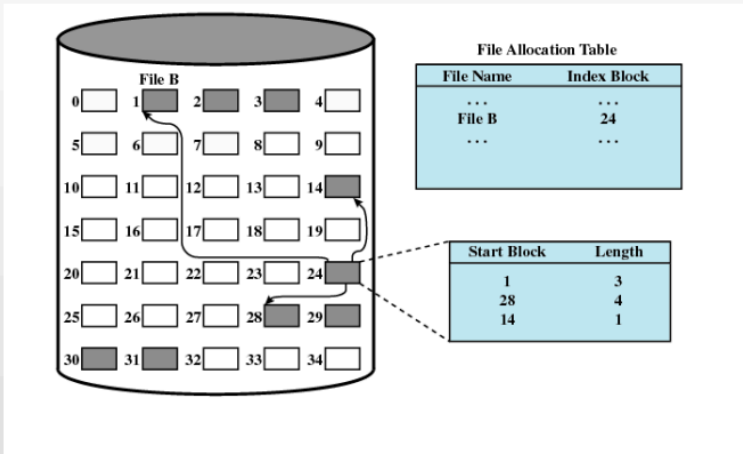
-El bloque indice no contiene datos del archivo, solo contiene un indice a los bloques que componen ese archivo. No se produce fragmentacion externa



-Variante por secciones:

-Asignar por secciones, a esa estructura de indices se trata de achicarla y a cada entrada de bloque de indice se agrega el campo de longitud del bloque

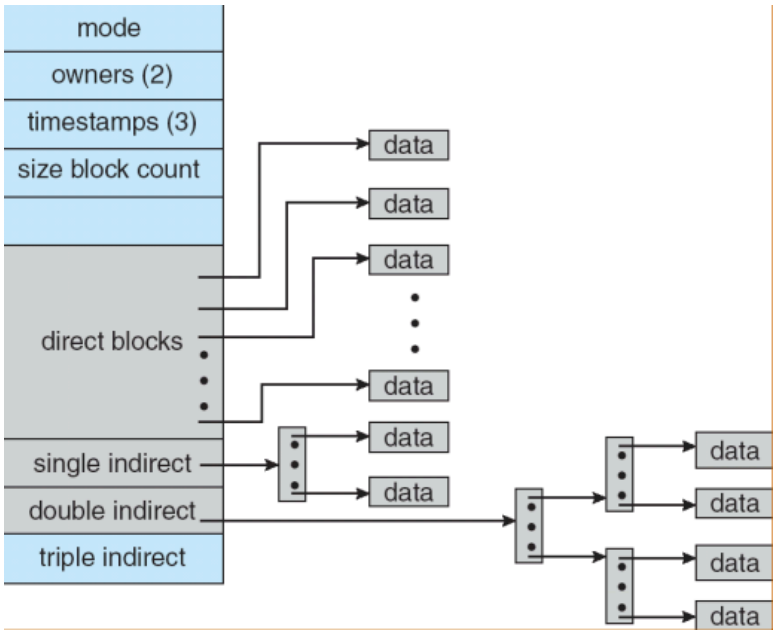
-El indice apunta al primer bloque de un conjunto almacenado de mantera contigua



-Variante por niveles de indireccion:

-Existen bloques directos de datos (apuntan directamente a datos). Los otros bloques se consideran bloquen indices y apuntan a varios bloques de datos.

-A medida que el archivo crece se utilizan bloques de indireccion los cuales se estructuran con punteros a datos o en su defecto a otro bloque de indireccion (puede haber varios niveles de indireccion)



Cada I-NODO contiene 9 direcciones a los bloques de datos, organizadas de la siguiente manera:

- 7 de direccionamiento directo.
- 1 de direccionamiento indirecto simple
- 1 de direccionamiento indirecto doble

Si cada bloque es de 1KB y cada dirección usada para referenciar un bloque es de 32 bits:

- ✓ ¿Cuántas referencias (direcciones) a bloque pueden contener un bloque de disco?

  1 KB / 32 bits = 256 direcciones

- ✓ ¿Cuál sería el tamaño máximo de un archivo?

  $(7 + 256 + 256^2) * 1\ KB = 65799\ KB = 64,25\ MB$

36. **Gestión de espacio libre. Dados los siguientes métodos de gestión de espacio libre en un disco:**

➢ **Tabla de bits**

➢ **Lista Enlazada**

➢ **Agrupamiento**

➢ **Recuento**

a. Describa cómo trabajan estos métodos.

b. Cite ventajas y desventajas de cada uno.

-Tabla de bits:

-Vector con 1 bit por cada bloque de disco → 0 libre, 1 ocupado

-Ventaja → facil de encontrar bloques libres

-Desventaja → tamanio de vector en memoria: tamanio disco en bytes/tamanio bloque en sistema de archivo

☑ Ejemplo

00111

00001

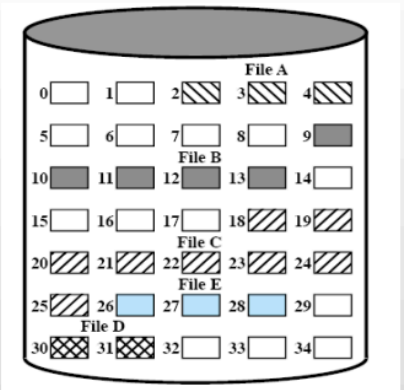
11110

00011

11111

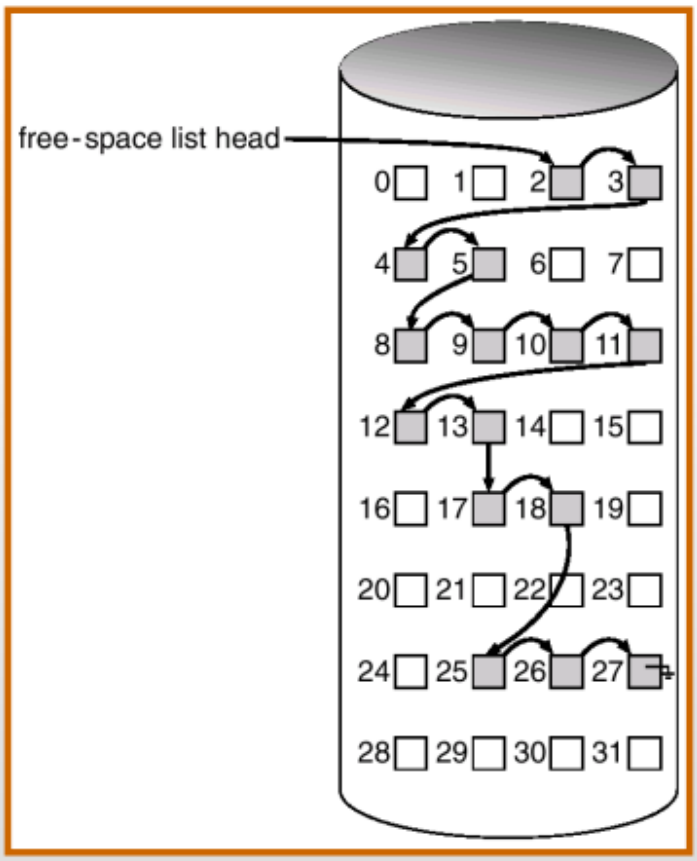
11110

11000



-Bloques encadenados: puntero a primer bloque libre. Cada bloque libre tiene un puntero a otro bloque libre

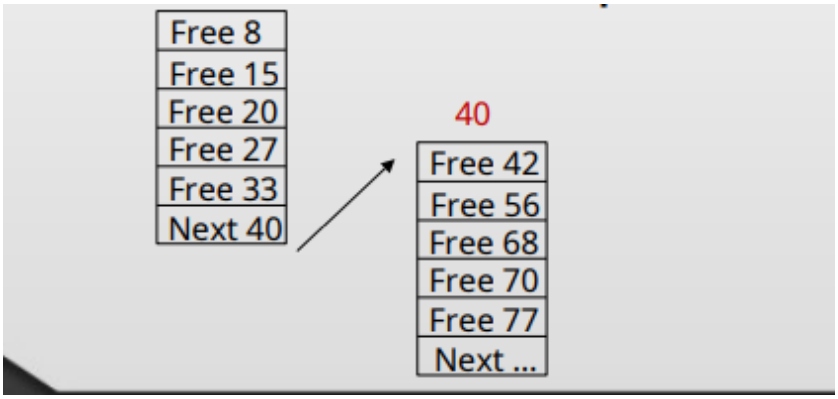
Desventaja → inteficiente para buscar bloques libres, varias E/S para obtener un grupo libre



-Indexacion:

-El primer bloque libre contiene direcciones de N bloques libres. Es una variante de la de bloques libres encadenados.

-Hasta n-1 tenes bloques libres y en N tenes la referencia a otro bloque con N-1 direcciones de bloques libres

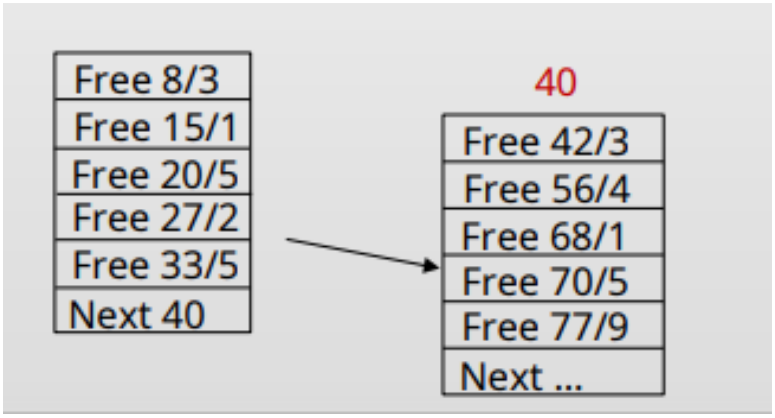


-Recuento:

-Varios bloques contiguos pueden ser solicitados y tiene en cuenta eso. Es una varian de de indexacion y busca achicar estructuras ya que, en lugar de tener N direcciones libres, se tiene:

-Direccion del primer bloque libre

-Los N bloques libres que le siguen (#bloque, N)



37.**Datos un disco con las características vistas en el ejercicio 29:**

- ➢ 7 platos con 2 caras utilizables cada uno.
- ➢ 1100 cilindros, con 300 sectores por pista, donde cada sector es 512 bytes.
- ➢ Seek Time de 10 ms
- ➢ 9000 RPM .
- ➢ Velocidad de Transferencia de 10 MiB/s (Mebibytes por segundos).

a. Calcule el tiempo de transferencia real de un archivo de 15 MiB(Mebibytes) almacenado en el disco de manera secuencial (todos sus bloques almacenados de manera consecutiva)

-9000 RPM → 1 min = 60 s x 1000 = 6000 ms

-3,33 ms → Latencia promedio

-15 MiB x 1s/10 MiB = 1,5s → 1500 ms

-Tiempo total = Seek time + latencia promedio + tiempo de transferencia del archivo (al estar secuencializado el almacenamiento con 1 seek alcanza)

-**Tiempo total = 10ms + 3,33 ms + 1500 = 1513,33 ms**

b. Calcule el tiempo de transferencia real de un archivo de 16 MiB(Mebibytes) almacenado en el disco de manera aleatoria.

-9000 RPM → 1 min = 60 s x 1000 = 6000 ms

-3,33 ms → Latencia promedio

-16 MiB x 2^20 = 16.777.216 bytes

-16.777.216/512 = 32768 → Numero de bloques

-512 bytes / 10 MiB → 0.0488 → Tiempo de transferencia por bloque

-Tiempo total = (Seek time + latencia promedio + tiempo de transferencia por bloque) x cantidad de bloques → Al ser aleatorio cada bloque necesita un seek + latencia

-**Tiempo total = (10 ms + 3,33 ms + 0,0488 ms) × 32.768 = 438.000 ms = 7,3 min**

38.**Unidad de Asignación. Las técnicas vistas pueden referenciar a sectores físicos de medio de almacenamiento o clusters (conjunto consecutivo de sectores físicos). Cite ventajas y desventajas de la utilización de cada forma de unidad de asignación.**

-Asignacion por sectores:

-Ventajas:

-Menor desperdicio: cada archivo ocupa exactamente los sectores que necesita sin desjar espacio sin usar

-Mayor precision y aprovechamiento de datos

-Mejor control por parte del SO: acceso granular a los datos

-Desventajas:

-Sobrecarga administrativa: el sistema debe manejar tablass muy grandes (FAT grande) ya que hay un registro por cada sector

-Mayor fragmentacion externa: unidad pequenia, los archivos tienden a repartirte no contiguamente

-Menor rendimiento en archivos grandes: leer archivos implica muchas operaciones de direccionamiento y saltos entre sectores

-Asignacion por clusters:

-Ventajas:

-Menor tamanio de tabla de asignacion: en FAT trabajar con clusters reduce la FAT y acelera busquedas

-Mejor rendimiento en archivos medianos o grandes: un solo cluster equivale a varios sectores (menos saltos y menos operaciones de lectura de tabla)

-Menor fragmentacion externa: unidad mayor, es mas probable que queden en bloques contiguos

-Desventajas:

-Mayor fragmentacion interna: un archivo puede no llenar el cluster al completo y el espacio restante se desperdicia

-Peor aprovechamiento del disco para muchos archivos pequenios: desperdicio acumulado

-Menor precision en el uso del espacio: una unidad de asignacion mas grande siempre implica menor precision

39.**Si se tiene un disco con 10 platos con 2 caras utilizables cada uno (20 cabezas), 350 cilindros y con 500 sectores por pista ¿Cuántos cluster pueden haber si se toma un tamaño de 4 sectores? ¿Y si el cluster es de 8 sectores?**

-Platos: 10

-20 cabezas

-Cilindros: 350

-Sectores por pista: 500

20 × 350 xs 500 = 3.500.000 sectores

-Si te toma un tamanio de 8 sectores por cluster → 3.500.000/4 = 875.000 clusters

-Si te toma un tamanio de 8 sectores por cluster → 3.500.000/8 = 437.500 clusters

40.**Asumiendo un sistema de archivos que utiliza como unidad de asignación un cluster de 8 sectores de disco y que el primer cluster de datos (cluster 0) se comienza a almacenar en el sector LBA 120. Determine el los valores LBA de los sectores que corresponden a los siguientes números de cluster:**

-Dado que cada cluster tiene 8 sectores tenemos que saber a cuantos sectores de distancia esta X cluster, entonces es N bloques de 8 (n x 8)

-Una vez averiguas donde empieza le sumas 7 para saber donde termina

a. 0

-LBA inicio = 120 + (0 x 8) = 120

-LBA fin = 120 + 7 = 127

LBA de 120 a 127

b. 123

-LBA inicio = 120 + (123 × 8) = 120 + 984 = 1104

-LBA fin = 1104 a 1111

LBA de 1104 a 1111

c. 299

-LBA inicio = 120 + (299 × 8) = 120 + 2392 = 2512

-LBA fin = 2519

LBA de 2512 a 2519

d. 1001

-LBA inicio = 120 + (1001 × 8) = 120 + 8008 = 8128

-LBA fin = 8135

LBA de 8128 a 8135

41.**Describa la organización del FileSystem de Unix System V. Tenga en cuenta la organización del mismo, estructuras de datos, etc.**

42.**Gestión de archivos en *UNIX* System V**

El sistema de archivos de UNIX utiliza una variante del esquema de asignación indexada para la administración de espacio de los archivos. Cada archivo o directorio está representado por una estructura que mantiene, entre otra información, las direcciones de los bloques que contienen los datos del archivo: el I-NODO.

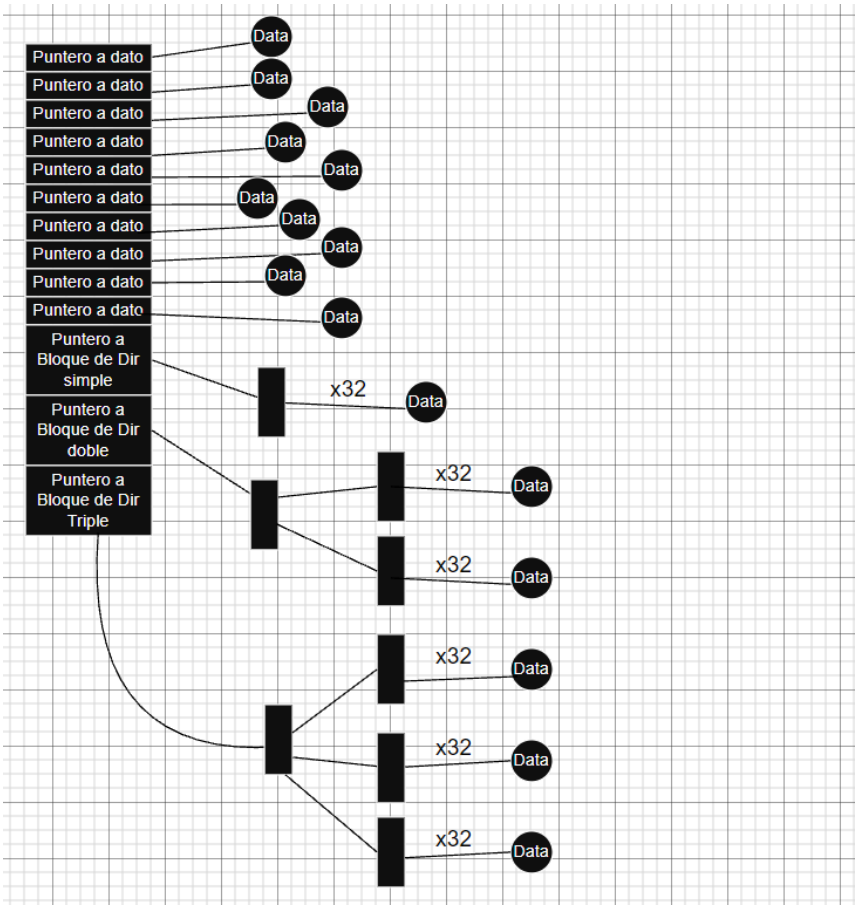
Cada I-NODO contiene 13 campos para direcciones a los bloques de datos, organizadas de la siguiente manera:

- **10 de direccionamiento directo.**
- **1 de direccionamiento indirecto simple.**
- **1 de direccionamiento indirecto doble.**

➢ **1 de direccionamiento indirecto triple.**

a. Realice un gráfico que represente la estructura del *I-NODO* y de los bloques de datos. Cada cluster es de 1 Kib(Kibibits).

Data = 1 KiB



b. Si cada dirección para referenciar un bloque es de 32 bits:

i. ¿Cuántas referencias (direcciones) a bloque pueden contener un cluster?

-Direccion = 32 bits

-Cluster = 1 Kibibits= 1024 bits

-**1024/32 = 32 direcciones caben en un cluster**

ii. ¿Cuál sería el tamaño máximo de un archivo?

(10 + 32 + 32^2 + 32^3) x 1024 = 34.643.616 bits

43.**Dado el FileSystem de UNIX System V (visto en clase), indique qué cambios se producen el en mismo dada las siguientes situaciones:**

a. Se agrega un nuevo archivo "Archivo1" en el directorio "Directorio1"

-Se agrega inodo libre y se inicializa

-Se agrega una entrada en el directorio

-"Archivo1" → i-nodo X"

-Se actualiza el i-nodo del directorio

-Se reduce lista de inodos libres

b. Se agrega nuevo contenido el archivo “Archivo1”

-Se asignan bloques de datos libres

-Se actualizan punteros directos o indirectos al i nodo

-Actualizar tamanio de i-nodo

-Se reduce lista de bloques libres

c. Se modifica el nombre del archivo “Archivo1” a “Archivo2”

-Soloe modifica la entrada del directorio

-”Archivo2” → i-nodo X” (apunta al mismo i-nodo)

d. Se hace un link duro a “Archivo2” en el mismo directorio con nombre “LinkArchivo2”.

-Se agrega una entrada de directorio. “LinkArchivo2” → i-nodo X”

e. Se borra un archivo “Archivo3”.

-Se elimina su entrada del direcotrio

-Se decrementa el contador de links de i nodo

-Si llega a 0 se liberan bloques de datos en la free list y se libera el i nodo y vuelve a la free list

44.**Se tiene un FileSystem con un modelo indexado con niveles de indirección, donde se mantiene un i-nodo con 10 índices de direccionamiento directo, 1 índice para indirecto simple y 2 para indirecto doble. Considerando además que utilizan 32 bits para referenciar direcciones a bloques/clusters.**

Suponga que existen dos siguientes archivos con sus tamaños logicos:

- ***ArchivoA*: de 55 KibiBytes**
- ***ArchivoB*: de 12 KibiBytes**
- ***ArchivoC*: de 1075 KibiBytes**

a. Si se utiliza como unidad de asignación el sector de disco de 512bytes

i. ¿Cuántas direcciones a sectores puede contener un sector de disco?

-32 bits / 8 → 4 bytes

- 512/4 → 128 direcciones

ii. ¿Cuál sería el tamaño máximo que un archivo podría alcanzar?

(10 + 128 + (128^2 × 2)) x 512 bytes = 16.777.216 bytes

iii.¿Cuántos sectores se necesitan para almacenar cada uno de los archivos antes enumerados? Indique el tamaño físico que se requirió para almacenar cada uno.

-Archivo A = 55 kibibytes = 56320 bytes

-56320 / 512 = 110 sectores

-Archivo B = 12 kibibytes = 12288 bytes

-12288 / 512 = 24 sectores

-Archivo C = 1075 kibibytes = 1.100.800 bytes

-1.100.800 / 512 = 2150 sectores

iv.Indique la fragmentación interna que se produce para cada uno de los archivos antes enumerados.

-0 en todos porque ninguna division me dio con decimal?

b. Si se utiliza como unidad de asignación el cluster de disco de 2 KibiBytes

i. ¿Cuántas direcciones a clusters puede contener un cluster de disco?

-2 kibibytes = 2048 bytes

-32 bits = 4 bytes

-2048 / 4 = 512 direcciones

ii. ¿Cuál sería el tamaño máximo que un archivo podría alcanzar?

(10 + 512 + (512^2) x 2) x 2048 bytes = 1,074,995,200 bytes

iii.¿Cuántos clusters se necesitan para almacenar cada uno de los archivos antes enumerados? Indique el tamaño físico que se requirió para almacenar cada uno.

-Archivo A = 55 kibibytes = 56320 bytes

-56320 / 2048 = 27.5 → 28

-Archivo B = 12 kibibytes = 12288 bytes

-12288 / 2048 = 6

-Archivo C = 1075 kibibytes = 1.100.800 bytes

-1.100.800 / 2048 = 537,37 → 538

iv.Indique la fragmentación interna que se produce para cada uno de los archivos antes enumerados.

-A: 57,344 − 56,320 = 1,024 bytes

-B: 12,288 − 12,288 = 0 bytes

-C: 1,100,864 − 1,100,800 = 64 bytes

45.**Para los incisos a. y b. el ejercicio anterior, y considerando que el fileSystem se utiliza en un disco con las siguientes características:**

➢ **6 platos, con 2 caras útiles cada uno**

➢ **2500 pistas por cara y 100 sectores por pista de 512 bytes cada uno.**

➢ **7200 RPM**

➢ **seek type de 10,5 ms**

➢ **velocidad de transferencia de 146 MiB/seg (Mebibytes por segundo)**

Calcule el tiempo de transferencia de los 3 archivos (ArchivoA, ArchivoB y ArchivoC) del ejercicio previo.

-7200 RPM / 60 = 120 RPS = 1/120 = 8.33 ms → Latencia promedio rotacion

-Seek → 10,5 ms

-Velocidad → 146 MiB/s

-Archivo A:

Tiempo = 55 KiB / 145 MiB/s

TIempo = 55Kib x 1024 / 145 MiB/s (paso KiB a MiB)

Tiempo = 0.00036S → 0.36 MS

Tiempo de transferencia = 10,5 ms + 8.33 ms + 0.36 = 19.19 ms

-Archivo B:

Tiempo = 12 KiB / 145 MiB/s

TIempo = 12Kib x 1024 / 145 MiB/s (paso KiB a MiB)

Tiempo = 0,078 ms

Tiempo de transferencia = 10,5 ms +8.33 ms + 0,078 ms = 18.9 ms

-Archivo C:

Tiempo = 1075 KiB / 145 MiB/s

TIempo = 1075Kib x 1024 / 145 MiB/s (paso KiB a MiB)

Tiempo = 7,5 ms

Tiempo de transferencia = 10,5 ms +8.33 ms + 7,5 ms = 26.33 ms

46.**Analice las siguientes fórmulas necesarias para localizar un I-NODO en la tabla de i-nodos de un filesystem similar a System V (visto en la teoría)**

nro bloque = ((nro de inodo -1)/nro. de inodos por bloque) + bloque de comienzo de la lista de inodos.

Desplazamiento del inodo en el bloque = ((nro de inodo - 1) módulo (número de inodos por bloque)) * medida de inodo del disco.

a. Según la primera fórmula, asumiendo que en el bloque 2 está en el comienzo de la lista de inodos y que hay 8 inodos por bloque: calcule donde se encuentra el inodo 8 y el 9. ¿Dónde estarían para un bloque de disco con 16 inodos?

-Busco bloque de inodo 8 con 8 inodos por bloque:

-Numero de bloque = ((8 - 1) / 8) + 2

-Numero de bloque = 7/8 + 2

-Numero de bloque = 2.87

-Busco bloque de inodo 8 con 16 inodos por bloque:

-Numero de bloque = ((8 - 1) / 16) + 2

-Numero de bloque = 7/16 + 2

-Numero de bloque = 2.43

-Busco bloque de inodo 9 con 8 inodos por bloque:

-Numero de bloque = ((9 - 1) / 8) + 2

-Numero de bloque = 1 + 2

-Numero de bloque = 3

-Busco bloque de inodo 9 con 16 inodos por bloque:

-Numero de bloque = ((9 - 1) / 16) + 2

-Numero de bloque = 0,5 + 2

-Numero de bloque = 2,5

b. De acuerdo a la segunda fórmula, si cada inodo del disco ocupa 64 bytes y hay 8 inodos por bloque de disco, el inodo 8 comienza en el desplazamiento 448 del bloque de disco. ¿Dónde empieza el 6? Si fueran inodos de 128 bytes y 24 inodos por bloque: ¿dónde empezaría el inodo 8?

-Inodo 6 desplazamiento en bloque:

Desplazamiento en bloque = (((6 - 1) MOD 8) x 64)

Desplazamiento en bloque = (5 MOD 8) x 64

Desplazamiento en bloque = 5 × 64 = 320

-Inodo 8 desplazamiento en bloque:

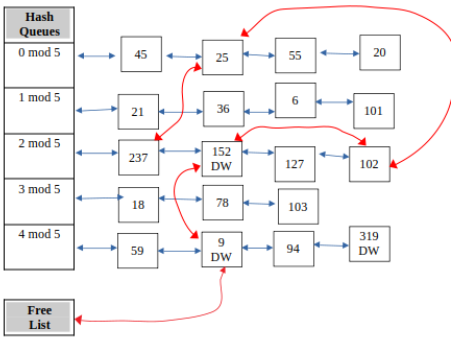
Desplazamiento en bloque = (((8 - 1) MOD 24) x 128)

Desplazamiento en bloque = (7 MOD 24) x 128

Desplazamiento en bloque = 7 × 128 = 896

Caché de Disco

47.**Dado el siguiente gráfico que representa el estado de Buffer Cache en Unix System V (Visto en clase):**



Responda a los siguientes incisos (considere cambios en los headers, las hash queue, free list, operaciones de E/S, etc):

- **a. ¿Qué sucedería si un proceso P1 requiere el bloque 45?**
- **b. ¿Qué sucedería si un proceso P1 solicita el bloque 113?**
- **c. ¿Qué sucedería si un proceso P1 solicita el bloque 102?**
- **d. ¿Qué sucedería si un proceso P1 solicita el bloque 9?**