Sistemas Operativos
2° Parcial 1C2017 - TM - Resolución

Aclaración: La mayoría de las preguntas o ejercicios no tienen una única solución. Por lo tanto, si una solución particular no es similar a la expuesta aquí, no significa necesariamente que la misma sea incorrecta. Ante cualquier duda, consultar con el profesor del curso.

Teoría:

1)

a) Falso. Si bien buddy system sí lo sufre, porque define las particiones en forma dinámica (externa) pero con valores controlados posibles (fragmentación interna), no se cumple con segmentación paginada. En este último caso, toda la memoria va a estar dividida en frames, por lo que nunca habrá fragmentación externa.

b) Verdadero. Ej: se usa una tabla de páginas jerárquica, entonces: se quiere acceder a una dirección, no está en la TLB, se accede a la TP (1er acceso) se ve que P == 0, se tira PF se va a buscar la pág a disco, no hay frames libres por lo que hay que reemplazar una que está con M == 1 por lo que hay que descargarla (2do acceso para leer la pág reemplazada y volcarla a disco, 3er acceso para actualizar la TP y poner P = 0; 4to acceso para escribir la página con lo que se trae de disco, 5to acceso para actualizar la TP con P=1). Ahora dentro de esa página se accede para recuperar el frame final real del dato, ya que ésta era una parte de la TP paginada. Se encuentra que P = 0; y puede ocurrir lo mismo que ocurrió en el caso anterior, por lo que podrían sumar 4 accesos más. Finalmente estaría el acceso deseado a la página.

2) Dado que se asigna una entrada por frame, no es posible utilizar memoria virtual sin realizar ninguna modificación.

3) La ventaja que trae N-STEP-SCAN es que al atender los pedidos de a ráfagas es más justo porque respeta el orden de llegada evitando también la posibilidad de starvation. Por otro lado, podría llegar a generar tiempos de atención más lentos comparados a SCAN, ya que debe ir de extremo a extremo más veces en lugar de atender todos los pedidos nuevos que llegan en el camino.

4) EXT2 utiliza asignación de bloques indexada, es decir, cada archivo tiene un índice (o varios en este caso) que contienen los punteros a los bloques de datos del mismo, permitiendo el acceso directo. En el caso particular de EXT en lugar de tener un único índice, se guardan algunos de los punteros en el mismo FCB (inodo) y luego se tienen bloques de punteros de diferente nivel según la configuración.

Por otro lado, FAT32 utiliza encadenada, es decir, cada bloque dice quién le sigue, pero con la mejora de la tabla FAT, que es una para todo el FS a comparación de los índices que son por cada archivo. Esta tabla permite lograr un acceso directo como en EXT2.
Podría ocurrir ya que en EXT probablemente tenga que asignar bloques de punteros además de bloques de datos. Ej: bloque de 1KiB, con 10 ptrs directos, al menos va a necesitar acceder a un bloque extra para el bloque de punteros de 1er nivel.

5) a) Desde un hardlink sería lo mismo que desde la entrada de directorio original. Se carga en memoria el inodo correspondiente y luego se determinará con qué puntero se accede al BD

b) Desde un softlink se cargará el inodo, se obtendrá su contenido, que es un path al archivo original. Luego se tendrá que ir recorriendo directorio por directorio (obteniendo el inodo de cada uno ya que son archivos, buscando las entradas correctas dentro de sus bloques de datos) hasta llegar a la entrada de dir del archivo, recuperando así su número de inodo. A partir de acá sería igual que a)

Práctica:

1)

4GiB / 1024 KiB / 8 / 1024 → 512 cilindros

La pista inicial es la 70. Los pedidos se ordenan, tomando colas de a 3

70 -> 511 (tope) -> 0 (base) -> 10, 30, 50 (primera cola, para entonces ya llegó la segunda tanda)
50 -> 60 -> 210 -> 220 (segunda cola)
220 -> 511 (tope) -> 0 (base) -> 50 (tercera cola)

2)

Bloques 1KiB, ptrs de 4b => 2^10/2^2 = 2 ^8 = 256 ptrs por bloque
EL FS puede direccionar 2^32 * 2^10 = 4TiB
Tamaño máx teórico archivo = (10 BD + 2 * 256 BD + 256*256 BD) 1KiB = 64,5 MiB
Dicho formato de inodo no permite direccionar ese tamaño de archivo

Se podría agregar un puntero de triple indirección, eso le agregaría (2^8)^3 * 2^10 = 2^34 = 16GiB de direccionamiento, ya superando el tamaño de la partición.
(hay múltiples opciones)

b)
i) Dentro de EXT hay que leerlo => 10MiB /1KiB =>10 2^20 /2^10 = 10240 bloques de datos a leer
Con los primeros 10 ptrs directos se leen 10 BD -> 10 accesos a BD
Con los dos ptrs ind simple se leen 512 BD -> 512 accesos a BD y 2 accesos a BP
Con el ptr ind doble se leen los BDs restantes (10240 - 522 = 9718). Como cada puntero del bloque de punteros tiene una capacidad de direccionamiento de 256 bloques, significa que se requieren (9718/256 = 37,9..) 38 bloques de punteros de segundo nivel
=> 9718 accesos a BD y 39 accesos a BP

En EXT se acceden a 10240 BDs y 41 BPs

Luego hay que escribir el archivo en FAT => 10MiB/2KiB => 5120 clusters
En este caso ese es el mismo número de accesos a bloques que serán necesarios.

ii)
Dentro de EXT:
Se marcan los bloques asignados como libres en el bitmap de bloques
Se marca el inodo como libre en el bitmap de inodos
Se borra la entrada de directorio
Se modifica en el Superbloque la cantidad de bloques e inodos disponibles (lo mismo en el descriptor del grupo de bloques)

En FAT:
Se crea la entrada de directorio, se completa la FAT con el valor de los bloques siguientes.

3)
a)

Las DL 300A = 0011| 0000| 0000| 1010
Son 4 bit para Nro página ósea las paginas validas son 0 a 15
Tamaño de página es 2^12 = 4096
Los pedidos son:

Referencia	Pagina
PB – A100h – L	10
PA – 2013h – E	2
PA – 5E12h – L	5
PC – 1AEFh – E	1
PB – B010h – L	11

Marco	Página	10 (L)	2 (E)	5 (L)	1 (E)	11 (L)
3	→ 11 U M	11 U M	11 - M	11 - M	11 - M	11 U M
2	3 U -	3 U -	3 - -	**5 U -**	5 U -	5 U -
9	4 U -	4 U -	4 - -	→ 4 - -	→ 4 - -	→ 4 - -
11	6 - -	**10 U -**	10 - -	10 - -	10 - -	10 - -
7	5 - M	→ 5 - M	**2 U M**	2 U -	2 U -	2 U -
4	1 U M	1 U M	→ 1 - M	1 - M	1 U M	1 - M
5	0 U M	0 U M	0 - M	0 - M	0 - M	0 - M
		PF	PF	PF		
			Escritura			

3 Fallos de página

b) La página 5 fue escrita en disco cuando se la reemplazó por página 2
c) DF = marco * tamaño de página + desplazamiento
DF Frame 7 = 7 *4096 + 19 = 28691 en decimal