Sistemas Operativos
2° Parcial 2C2022 - TT - 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/la docente del curso.

Teoría

1.

	Particionamiento fijo	**Particionamiento dinámico**	**Segmentación paginada**
Overhead	Bajo	Medio/Alto (creacion/destruccion de particiones; compactación)	Alto (creación/destrucción de tablas de segmentos/páginas, búsqueda de frames)
Fragmentacion	Tiene fragmentación Interna solamente	Tiene fragmentación externa solamente	Tiene fragmentación interna solamente

2. El FCB es una estructura de datos usada para administrar archivos. Existe un FCB por cada archivo del sistema. En el mismo se guardan atributos y punteros al archivo. La estructura existe en disco porque es metainformación referida a los archivos en reposo. También se carga a memoria cuando el archivo es abierto por al menos un proceso, en particular en la tabla global de archivos abiertos
3. La interrupción de PF será atendida por el Sistema Operativo el cual validará si la referencia se encuentra dentro del espacio de direcciones del proceso. Al ser válida, deberá encontrar un frame libre para cargar la página en cuestión. Al disponer de un frame libre, disparará la operación de lectura de la página en disco. El frame destino se marcará como ocupado y se lockeará para evitar que al finalizar la operación de lectura el mismo pertenezca a otro proceso. Una vez cargada la página en memoria se marcará la página como presente y se actualizará la TLB. Finalmente se reiniciará la instrucción que dio origen al PF.
4. Las estrategias más utilizadas son el bit vector o la lista enlazada de bloques libres. El bit vector utiliza un bit por cada bloque del sistema y marca los bloques ocupados con 1 y los libres con 0. En la implementación más simple de la lista enlazada se encuentran encadenados las referencias a los bloques libres del sistema.
   Por su estructura, el bit vector es la estructura de gestión de espacio libre que menos espacio utiliza en disco.

5.

a. Falso. Los softlink solo contienen al path al archivo, no la referencia al inodo del mismo.
b. Falso. Aún con segmentación paginada, utilizar paginación jerárquica permite que las tablas de páginas de cada nivel sean más pequeñas lo cual resulta en un uso más eficiente de la memoria principal donde solo se cargan las porciones de la misma que están siendo accedidas.

Práctica

1.

a) Se verifican los máximos direccionables con los diferentes bloques:

- 256 bytes: 2^32 x 2^8 bytes = 2^40 bytes = 1 TB => No alcanza para direccionar la partición
- 512 bytes: 2^32 x 2^9 bytes = 2^41 bytes = 2 TB => Alcanza para direccionar la partición, pero contradice el dato de que el tamaño máximo real quedó distinto del teórico
- 1024bytes: 2^32 x 2^10 bytes = 2^42 bytes = 4 TB => Alcanza y sobra para direccionar la partición. Por lo tanto, este tamaño de bloque fue el elegido.

b) Se espera direccionar una nueva partición de 1 TB (2^40 bytes).

2^28 x 2^13 bytes = 2^41 byes = 2TB => Teóricamente sobra para direccionar la partición, por lo tanto no es necesaria una tabla fat de 2^ 28 entradas.

Para calcular cuántas entradas necesitamos: 2^40 bytes / 2^13 bytes = 2^27 entradas.

Cada entrada ocupa 4 bytes en disco, por lo que el tamaño de la misma es 2^27 x 2^2 bytes = 2^29 bytes = 512 MB

2. Tenemos 32 bits de dirección lógica y paginación jerárquica de 2 niveles. Sabemos que las páginas son de 64KiB por lo cual utilizamos 16 bits de la dirección para el offset. Cómo todas las tablas de páginas tienen la misma cantidad de entradas, sabemos que los niveles poseen la misma cantidad de bits cada uno, en este caso 8.

a)

1) 01AAFFFEh: En el primer nivel buscaremos 01, en el segundo AA y FFFE corresponde al offset.
Se accede a la tabla de primer nivel para buscar la dirección de la entrada 01 y ocurre un PF. Tendremos que ir a disco para buscar la tabla de 2do nivel y cargarla en memoria. Se actualiza la presencia de la entrada 01 en la tabla de 1er nivel. Se reinicia la instrucción, se accede a la tabla de primer nivel para buscar la de 2do nivel y en la misma se busca la entrada AA la cual arroja PF. Se accede a disco para buscar la página correspondiente, se carga en memoria, se actualiza la tabla de páginas y se reinicia la instrucción. Finalmente se puede acceder a los dos niveles de TP para realizar la traducción y finalmente acceder a la dirección física en memoria principal.

2) 01AAFFFFh: En el primer nivel buscaremos 01, en el segundo AA y FFFF corresponde al offset.

Se accede a los dos niveles de TP para realizar la traducción y finalmente a la dirección física en memoria principal.

3) 01AB0000h: En el primer nivel buscaremos 01, en el segundo AB y 0000 corresponde al offset.
Se puede acceder a la tabla de segundo nivel que ya está en memoria principal pero al acceder a la entrada AB hay un PF. Se accede a disco para buscar la página correspondiente, se carga en memoria, se actualiza la tabla de páginas y se reinicia la instrucción. Finalmente se puede acceder a los dos niveles de TP para realizar la traducción y finalmente acceder a la dirección física en memoria principal.

4) En el primer nivel buscaremos 01, en el segundo AB y 0000 corresponde al offset.
Se accede a los dos niveles de TP para realizar la traducción y finalmente a la dirección física en memoria principal.

b) Utilizar TLB hubiese ahorrado los accesos a las TP una vez que ambas estuviesen cargadas en memoria en cada referencia.

c) Se cumple el concepto de localidad espacial, ya que las referencias se realizan en direcciones contiguas.

3. Como la asignación de bloques es secuencial se recuperan 20 bloques del archivo. (del 0 al 9 y del 30 al 39).
   a) Hay 8 marcos asignados al proceso.
   1 ocupado con la imagen inicial del proceso con el contador de bloques leídos. Otros 7 para los primeros bloques del archivo recuperado (donde no hay reemplazo). Es necesario realizar 13 reemplazos para leer los siguientes bloques del archivo. Al ser LRU la página 0 del proceso nunca se reemplaza porque es utilizada para actualizar el contador de bloques leídos.

   b) La información recuperada tiene un tamaño de 20 MiB, lo que es equivalente a 2560 bloques de 8 KiB del ext2.
   Hay 1024 punteros por bloque y los bloques son de 8 KiB.
   - Con los PD guardo 5 bloques. Restan 2555 bloques de datos para escribir en el ext2.
   - Con IS guardo 1024 bloques. Restan 1531 bloques de datos para escribir en el ext2. Utilizo el primer bloque de punteros.
   - Con ID guardo los restantes 1531 bloques de datos.. Necesito un bloque de puntero del cual utilizo los dos primeros punteros. Una IS para escribir 1024 bloques de datos y otra para los 507 bloques de datos restantes.
   En Resumen:
   2560 Bloques de datos
   1 (IS) + 3 (ID) Bloques de punteros.
   Total: 2564 Bloques.