Ejercicios relación 4

Los autores no se hacen responsables de ningún fallo provocado o influido por estos ejercicios

`Parte del dibujo del 14 sin hacer`

(la diferencia entre los dos grupos es que los que tienen dudas están resueltos a media)

1. Sea un Sistema Operativo que sólo soporta un directorio (es decir, todos los archivos existentes estarán al mismo nivel), pero permite que los nombres de archivo sean de longitud variable. Apoyándonos únicamente en los servicios proporcionados por este Sistema Operativo, deseamos construir una "utilidad" que "simule" un sistema jerárquico de archivos. ¿Es esto posible? ¿Cómo?

Podríamos realizar la estructura jerárquica imponiedo el nombre de "directorios" y almacenándolos junto con archivos que referencien al resto de archivos almacenados por este, por ejemplo tener: Datos/

Datos/altura.txt

Datos/peso.txt

Alternativa: Podemos añadir a cada nombre de archivo un índice (entero, carácter) que indique en que nivel se encuentra.

2. En un entorno multiusuario, cada usuario tiene un directorio inicial al entrar en el sistema a partir del cual puede crear archivos y subdirectorios. Surge, entonces, la necesidad de limitar el tamaño de este directorio para impedir que el usuario consuma un espacio de disco excesivo. ¿De qué forma el Sistema Operativo podría implementar la limitación de tamaño de un directorio?

Implementando el sistema de cutas visto en prácticas donde cada ususario tiene asociado un número máximo de i-nodos y un número máximo de bloques.

3. En la siguiente figura se representa una tabla FAT. Al borde de sus entradas se ha escrito, como ayuda de referencia, el número correspondiente al bloque en cuestión. También se ha representado la entrada de cierto directorio. Como simplificación del ejemplo, suponemos que en cada entrada del directorio se almacena: Nombre de archivo/directorio, el tipo (F=archivo, D=directorio), la fecha de creación y el número del bloque inicial.

Los 512B que ocupa un bloque los aprovechamos en su totalidad para direccionar datos. Por eso, Datos1, que ocupa 10B, lo almacenaremos en un único bloque. Datos2 cuyo tamaño es 1200B-> 3 bloques. Caras -> 4 bloques.

Nombre	Tipo	Fecha	Nº de bloque
Datos	F	8-2-90	3
Datos1	F	1-3-90	4
Datos2	F	2-3-90	5
D	D	3-3-90	10
Cartas	F	13-3-90	12

Rellene la figura para representar lo siguiente:

- **Creación del archivo DATOS1 con fecha 1-3-90, y tamaño de 10 bytes.**
- **Creación del archivo DATOS2 con fecha 2-3-90, y tamaño 1200 bytes.**
- **El archivo DATOS aumenta de tamaño, necesitando 2 bloques más.**
- **Creación del directorio D, con fecha 3-3-90, y tamaño 1 bloque.**
- **Creación del archivo CARTAS con fecha 13-3-90 y tamaño 2 kBytes.**

1		10	*
2		11	13
3	15	12	11
4	*	13	14
5	6	14	*
6	7	15	8
7	*	16	
8	9	17	
9	*	18	

Nota: La anterior tabla es después de modificar.

4. Si usamos un Mapa de Bits para la gestión del espacio libre, especifique la sucesión de bits que contendría respecto a los 18 bloques del ejercicio anterior.

Los ceros indican que el bloque está vacio y el 1 que el bloque está ocupado.

0011 1111 1111 1110 00

5. Si se pierde el primer puntero de la lista de espacio libre, ¿podría el Sistema Operativo reconstruirla? ¿Cómo?

Si se borra el primer puntero de la lista lo que se tendría que hacer es recorrer todos los archivos del árbol del sistema de archivos para conocer los bloques ocupados y, por descarte, los no ocupados.

6. El espacio libre en disco puede ser implementado usando una lista encadenada con agrupación o un mapa de bits. La dirección en disco requiere D bits. Sea un disco con B bloques, en que F están libres. ¿En qué condición la lista usa menos espacio que el mapa de bits?

Si requiere B bloques, el mapa de bits ocupará de manera estable $B/8 Bytes$. Mientras tanto, la lista ocupará siempre $F \cdot 4$(tamaño puntero)$Bytes$. Entonces, serán iguales en $F \cdot 4 = B/8 \Rightarrow F = B/32$. Es decir, cuando una trigesimo segunda parte o menos de la memoria esté libre, utilizar punteros ocupará menos espacio.

7. Entre los posibles atributos de un archivo, existe un bit que marca un archivo como temporal y por lo tanto esta sujeto a destrucción automática cuando el proceso acaba ¿Cuál es la razón de esto? Después de todo un proceso siempre puede destruir sus archivos, si así lo decide.

Jose Antonio Ocete, [05.02.17 00:17] porque a los programadores se les puede olvidar borrar los archivos temporales

Jose Antonio Ocete, [05.02.17 00:17] asi es mas comodillo pa tos

8. Algunos SO proporcionan una llamada al sistema (RENAME) para dar un nombre nuevo a un archivo existente ¿Existe alguna diferencia entre utilizar esta llamada para renombrar un archivo y copiar el archivo a uno nuevo, con el nuevo nombre y destruyendo el antiguo?

La diferencia sería que rename sería muchísimo menos costosa dado que lo único que haría sería meterse en el i-nodo y cambiar en nodo mientras que en la otra habría que crear una estructura nueva integra.

9. Un i-nodo de UNIX tiene 10 direcciones de disco para los diez primeros bloques de datos, y tres direcciones más para realizar una indexación a uno, dos y tres niveles. Si cada bloque índice tiene 256 direcciones de bloques de disco, cuál es el tamaño del mayor archivo que puede ser manejado, suponiendo que 1 bloque de disco es de 1KByte?

$10 * 1KB + 256 * 1KB + 256^2 * 1KB + 256^3 * 1KB = 16GB$

10. Sobre conversión de direcciones lógicas dentro de un archivo a direcciones físicas de disco. Estamos utilizando la estrategia de indexación a tres niveles para asignar espacio en disco. Tenemos que el tamaño de bloque es igual a 512 bytes, y el tamaño de puntero es de 4 bytes. Se recibe la solicitud por parte de un proceso de usuario de leer el carácter número N de determinado archivo. Suponemos que ya hemos leído la entrada del directorio asociada a este archivo, es decir, tenemos en memoria los datos PRIMER-BLOQUE y TAMAÑO. Calcule la sucesión de direcciones de bloque que se leen hasta llegar al bloque de datos que posee el citado carácter.

Como el tamaño de cada bloque (tanto bloques índices como bloques de datos) ocupan 512B y como cada puntero ocupa 4B, cada bloque contiene $512/4 = 128$ punteros. Como se trata de un esquema de indexación a tres niveles y tenemos el primer bloque cargado en memoria, necesitamos acceder a disco 3 veces. Buscamos los tres índices para acceder al n-ésimo Byte buscado.

$$i = \frac{NB}{128 \cdot 128 \cdot 512B}; \quad N' = N\%(128 \cdot 128 \cdot 512B)$$

$$j = \frac{NB}{128 \cdot 512B}; \quad N'' = N'\%(128 \cdot 512B)$$

$$k = \frac{NB}{512B}; \quad N''' = N''\%(512B)$$

Hay que traer a memoria los bloques i, j, k y leer con desplazamiento N"'.

11. ¿Qué método de asignación de espacio en un sistema de archivos elegiría para maximizar la eficiencia en términos de velocidad de acceso, uso del espacio de almacenamiento y facilidad de modificación (añadir/borrar /modificar), cuando los datos son:

- **modificados infrecuentemente, y accedidos frecuentemente de forma aleatoria:**

  Asignación contigua.

Como se accede infrecuentemente a los archivos no es necesario mantener un método de asignación que permita modificarlos eficientemente. Además, es necesario optimizar la velocidad de acceso directo y el espacio de almacenamiento, por ello, la mejor opción será el método de asignación contigua.

- **modificados con frecuencia, y accedidos en su totalidad con cierta frecuencia:**

  No contiguo - Enlazado - Tabla FAT.

  Dado que queremos modificar los datos de manera eficiente, además de acceder a todos los datos en su totalidad, vamos a necesitar un sistema que nos ofezca la posibilidad de ir añadiendo de manera indefinida y con una velocidad óptima.

- **modificados frecuentemente y accedidos aleatoriamente y frecuentemente:**

  Asignación No contigua - Indexada.

  Este método nos ofrece la mejor velocidad de acceso aleatorio. Además, pese a que ocupe un poco más, la frecuencia de uso de este sistema justifica el gasto.

12. ¿Cuál es el tamaño que ocupan todas las entradas de una tabla FAT32 que son necesarias para referenciar los cluster de datos, cuyo tamaño es de 16 KB, de una partición de 20 GB ocupada exclusivamente por la propia FAT32 y dichos cluster de datos?

Como la partición ocupa $5{\cdot}2^{32}B$ y cada cluster 2^{14} B, nos quedan $5{\cdot}2^{18}$ conjuntos, referenciados cada uno por un puntero de tamaño 32. Por lo tanto la tabla FAT ocupará $5 \cdot 2^{23}$B.

13. Cuando en un sistema Unix/Linux se abre el archivo /usr/ast/work/f, se necesitan varios accesos a disco. Calcule el número de accesos a disco requeridos (como máximo) bajo la suposición de que el i-nodo raíz ya se encuentra en memoria y que todos los directorios necesitan como máximo 1 bloque para almacenar los datos de sus archivos.

respuesta realizada en base a la diapositiva 39

- Accede al i-nodo n1 de la raiz.
- Accede al bloque b1 del directorio /usr.
- Accede al i-nodo n2 de /usr/ast.
- Accede al bloque b2 del directorio /usr/ast.
- Accede al i-nodo n3 de la /usr/ast/work.
- Accede al bloque b3 del directorio /usr/ast/work.

total: 6 accesos

14. represente gráficamente cómo y dónde quedaría reflejada y almacenada toda la información referente a la creación anterior de un enlace simbólico y absoluto ("hard") a un mismo archivo, pract1.

En el enlace simbólico necesita un inodo nuevo, entrada de directorio y (a lo mejor) un bloque de datos. Sin embargo en el enlace duro esto último no sería necesario.

si alguien se siente con fuerza, que haga la representación gráfica

15. En un sistema de archivos ext2 (Linux), ¿qué espacio total (en bytes) se requiere para almacenar la información sobre la localización física de un archivo que ocupa 3 Mbytes?. Suponga que el tamaño de un bloque lógico es de 1 Kbytes y se utilizan direcciones de 4 bytes. Justifique la solución detalladamente.

Ahora lo cambiamos, estaba mal

16. En la mayoría de los sistemas operativos, el modelo para manejar un archivo es el siguiente:

- **Abrir el archivo, que nos devuelve un descriptor de archivo asociado a él.**
- **Acceder al archivo a través de ese descriptor devuelto por el sistema.**

¿Cuáles son las razones de hacerlo así? ¿Por qué no, por ejemplo, se especifica el archivo a manipular en cada operación que se realice sobre él?

Porque un archivo puede tener varíos nombres dados por el mismo o incluso por distintos usuarios, sin embargo, por su descriptor te puedes referir a el de forma inequívoca.

17. Sea un directorio cualquiera en un sistema de archivos ext2 de Linux, por ejemplo, DirB. De él cuelgan algunos archivos que están en uso por uno o más procesos. ¿Es posible usar este directorio como punto de montaje? Justifíquelo.

Si se puede. No se podría en el caso de que el directorio en si fuese utilizado por un proceso, sin embargo en este caso al estar solo algún archivo se podría.