**Respaldos**  
Incluso teniendo una estrategia ingeniosa para manejar los bloques defectuosos, es importante respaldar los archivos con frecuencia. Los sistemas de archivos en disco flexible se pueden respaldar con sólo copiar todo el disco en uno en blanco.

Los sistemas de archivos en discos winchester pequeños se pueden respaldar vaciando todo el disco en cinta magnética. Una estrategia que es fácil de implementar, pero desperdicia la mitad del espacio del almacenamiento es instalar en cada computadora dos unidades de disco en lugar de una.

Una alternativa al vaciado diario de todo el sistema de archivos es realizar vaciados incrementales. La forma más sencilla de vaciado incremental consiste en efectuar un vaciado completo periódicamente, digamos cada semana o cada mes, y realizar un vaciado diario de sólo aquellos archivos que han sido modificados después del último vaciado completo. Un esquema mejor sería vaciar sólo aquellos archivos que han cambiado desde que se vaciaron por última vez.

Para implementar este método, se debe mantener en disco una lista de los tiempos de vaciado para cada archivo. El programa de vaciado revisa cada archivo del disco. La escritura en el disco espejo se retrasa un poco, efectuándose cuando el sistema está ocioso. Un sistema así puede seguir funcionando en modo degradado si un disco falla, y esto permite reemplazar el disco que falló y recuperar los datos sin tener que parar el sistema.  
 **Consistencia del sistema de archivos**  
Otra área en la que la confiabilidad es importante es la consistencia del sistema de archivos. Muchos sistemas de archivos leen bloques, los modifican y los vuelven a escribir después. Si el sistema se cae antes de que todos los bloques modificados se hayan escrito en disco, el sistema de archivos puede quedar en un estado inconsistente. Este problema se vuelve crítico si algunos de los bloques que no se han escrito contienen nodos-i, entradas de directorio o la lista libre.

Para enfrentar el problema de un sistema de archivos inconsistente, la mayor parte de las computadoras cuentan con un programa de utilería que verifica la consistencia del sistema de archivos. Estos verificadores del sistema de archivos examinan cada sistema de archivos con independencia de los demás. Para comprobar la consistencia de los bloques, el programa construye dos tablas, cada una de las cual contiene un contador para cada bloque, que inicialmente vale 0. Después, el programa examina la lista o mapa de bits de bloques libres, para encontrar todos los bloques que no están en uso.

Si el sistema de archivos es consistente, cada bloque tendrá un 1 ya sea en la primera tabla o en la segunda, como se ilustra en la Figura 5.17a. En un sistema de archivos consistente, ambos conteos coinciden. Si el conteo de enlaces es mayor que el número de entradas de directorio, entonces, aunque se borraran todos los archivos de todos los directorios el conteo seguiría siendo mayor que cero y el nodo-i no se eliminaría. Este error no es grave, pero desperdicia espacio en el disco con archivos que no están en ningún directorio, así que debe corregirse asignando el valor correcto al conteo de enlaces del nodo-i.

Imagen que contiene Texto

Descripción generada automáticamente

Cuando esto suceda, el sistema de archivos lo marcará como desocupado y liberará todos sus bloques. El resultado de esta acción es que uno de los directorios ahora apunta a un nodo-i desocupado, cuyos bloques pronto pueden ser asignados a otros archivos. Estas dos operaciones, verificar bloques y verificar directorios, a menudo se integran por razones de eficiencia. Si un número de nodo-i es mayor que el número de nodos-i que hay en el disco, es señal de que el directorio ha sido dañado.

Podría ser útil al menos informar de la existencia de archivos que dan más derechos a terceros que al propietario. Al escribir los bloques críticos rápidamente, reducimos en buena medida la probabilidad de que una caída arruine el sistema de archivos.

Incluso con esta medida para mantener intacta la integridad del sistema de archivos, no es deseable mantener los bloques de datos demasiado tiempo en el caché antes de escribirlos en el disco. Consideremos la situación de una persona que está usando una computadora personal para escribir un disco.

Incluso si nuestro escritor le dice periódicamente al editor que grabe en el disco el archivo que está editando, hay una buena probabilidad de que todo estará aún en el caché y nada en el disco. Si el sistema se cae, la estructura del sistema de archivos no estará corrompida, pero se habrá perdido todo un día de trabajo. Lo que hace UNlX es tener una llamada al sistema, SYNC, que obliga la escritura inmediata en disco de todos los bloques modificados. Lo que hace MS-DOS es grabar todos los bloques modificados en el disco tan pronto como se modifican.

Los caches en los que todos los bloques modificados se escriben en el disco de inmediato se denominan caches de escritura inmediata, y requieren mucha más actividad de E/S de disco que los de otro tipo. UNIX acumula todos los caracteres en el caché y escribe el bloque en disco una vez cada 30 segundos, o cuando el bloque se quita del caché.  
  
**Sistemas de archivos estructurados por diario**  
Los cambios tecnológicos están ejerciendo presión sobre los sistemas de archivos actuales. El único parámetro que no está mejorando a pasos agigantados es el tiempo de búsqueda en disco. La combinación de estos factores implica que en muchos sistemas de archivos está apareciendo un cuello de botella del rendimiento.

Investigaciones efectuadas en Berkeley intentaron aliviar este problema diseñando un tipo totalmente nuevo de sistema de archivos, LFS . La idea en que se basa el diseño LFS es que conforme las CPU se hacen más rápidas y las memorias RAM se hacen más grandes, los caches de disco están aumentando aceleradamente.

Las escrituras pequeñas son muy ineficientes, ya que una escritura en disco de 50 microsegundos típicamente va precedida por una búsqueda de 10 ms y un retardo rotacional de 6 ms. Con estos parámetros, la eficiencia del disco decae a una fracción de 1%.

Si bien estas escrituras podrían diferirse, hacerlo expone el sistema de archivos a problemas de consistencia graves en caso de ocurrir una caída antes de llevarse a cabo las escrituras. Siguiendo este razonamiento, los diseñadores de LFS decidieron reimplementar el sistema de archivos de UNIX de forma tal que se aprovechara todo el ancho de banda del disco, incluso en casos en los que la carga de trabajo consiste en su mayor parte en escrituras aleatorias pequeñas.

En este diseño, siguen existiendo nodos-i y tienen la misma estructura que en UNIX, pero ahora están dispersos por todo el diario, en lugar de estar en una posición fija en el disco. La entrada i de este mapa apunta al nodo-i i en el disco.

El mapa se mantiene en disco, pero también en caché, así que las partes de uso más intenso están en la memoria casi todo el tiempo. Resumiendo, todas las escrituras se colocan inicialmente en buffers en la memoria, y periódicamente se escriben en el disco en un solo segmento, al final del diario.