Curso: Sistemas Operativos
5. Sistema de Archivos
(Parte II)

M.C. Laura Sandoval Montaño

Sistema de Archivos.
Administración del almacenamiento secundario.

Como se mencionó en la clase anterior, uno de los componentes del sistema de archivos es el Administrador del almacenamiento secundario, el cual se encarga de asignar espacio a los archivos en dispositivos de almacenamiento secundario.

Generalidades.

La asignación de espacio en memoria secundaria presenta problemas similares que en la asignación de memoria real bajo un régimen de multiprogramación con particiones variables.

Si deseamos almacenar archivos en áreas adyacentes del disco, será necesario juntar dichas áreas; pero con las operaciones frecuentes de asignación y liberación del espacio en disco, propicia que éste se fragmente cada vez más. Así, cada nueva asignación hace que los archivos se desperdiguen en bloques cada vez más separados entre sí, lo que ocasiona problemas de desempeño.

Las técnicas que pueden aliviar este problema es realizar la compactación o recolección de basura. De esta forma, los archivos se reorganizan para que ocupen áreas adyacentes del disco (compactación) o bien los archivos que no estén referenciados en sus directorios, desalojarlos del disco para contar con espacio disponible (recolección de basura).

Sistema de Archivos.
Administración del almacenamiento secundario.

Por otro lado, si el sistema operativo trabaja en un sistema de paginación en asignación de memoria real, la cantidad mínima de información transferida entre el almacenamiento secundario y el primario es una página, por lo que es razonable asignar memoria secundaria en bloques del tamaño de una página o múltiplos de ese tamaño.

Cuando un proceso ha hecho referencia a un elemento en una página, es probable que haga referencia a otros elementos de esa misma página; también es probable que haga referencia a elementos en páginas contiguas.

Por tanto, es conveniente almacenar las páginas contiguas desde el punto de vista lógico de memoria virtual de un usuario como páginas físicamente contiguas en memoria secundaria, sobre todo si se guardan varias páginas en cada bloque físico.

Clasificación.

La asignación de espacio a los archivos en dispositivos de almacenamiento secundario se clasifican, primeramente, en dos tipos:

- Asignación contigua.
- Asignación no contigua

Administración del almacenamiento secundario.

Asignación contigua.

En la asignación contigua, los archivos se asignan a zonas contiguas de memoria secundaria. Los usuarios especifican por adelantado el tamaño del área requerida para guardar el archivo que se creará. Si no se dispone de suficiente espacio contiguo no se podrá crear el archivo. Como ejemplos de medios secundarios de asignación contigua están las cintas y cartuchos magnéticos, utilizados generalmente para hacer respaldos.

Ventajas: Se realizan menos accesos de a memoria secundaria debido a que registros lógicos contiguos les corresponde registros físicos contiguos. La realización de directorios es más sencilla ya que para cada archivo, basta registrar la dirección de inicio y la longitud del archivo.

Desventajas: Al eliminar un archivo, se libera espacio que ocupaba, formándose un hueco que puede asignársele a otro. Este proceso hace que se queden huecos cada vez más pequeños y que al final no puedan grabarse más archivos por no caber. Lo que hay que hacer es un proceso de Condensación o Compactación de huecos.

La asignación contigua puede no ser conveniente cuando los archivos son dinámicos, es decir, puedan crecer o encogerse con el tiempo. Los usuarios deben hacer una buena estimación del tamaño final de su archivo, cuando consideran que van a crecer, para no ocasionar grandes desperdicios.

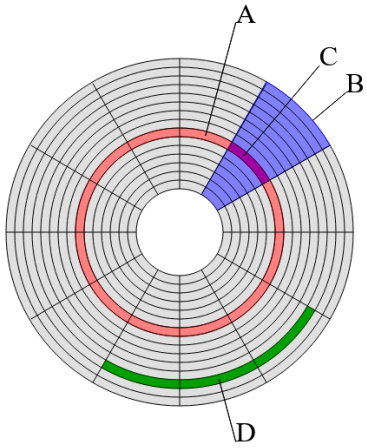
Administración del almacenamiento secundario.
Asignación no contigua.

Por las problemáticas presentadas en la asignación contigua, específicamente referente al tamaño tan dinámico que se presenta en los archivos, es que se crean nuevas técnicas donde da más flexibilidad de almacenar este tipo de archivos en memoria secundaria. Existen varios esquemas de asignación no contigua de memoria. Veremos algunos.

Encadenada orientada a los sectores.

Se emplea comúnmente en los discos y hace referencia a los sectores en los que se divide el disco:

Estructura de disco que muestra: (A) una pista (roja), (B) un sector geométrico (azul), (C) un **sector** de una pista (magenta), y (D) un grupo de sectores de pista contiguos (verde).



El archivo se divide en partes cuyo tamaño es de un sector de una pista (C). Se van colocando las partes en sectores de pista contiguos (de la misma pista, como se señala en D); si no es posible, el último registro de un sector de pista apunta al inicio del sector de pista donde se encuentra la siguiente parte del archivo, y así sucesivamente hasta tener todas las partes del archivo en el disco.

Asignación no contigua.

Encadenada orientada a los sectores (continuación).

Como los sectores que pertenecen al mismo archivo contienen apuntadores de uno a otro, se forma una lista encadenada. Una lista de espacio libre contiene entradas para todos los sectores libres del disco.

Cuando un archivo necesita crecer, el proceso solicita más sectores de la lista de espacio libre. Los archivos cuyo tamaño decrece, devuelven sectores a la lista de espacio libre. No hay necesidad de compactar.

Desventajas. Los registros de un archivo pueden estar desperdigados en todo el disco y la lectura de registros lógicos contiguos implica búsquedas largas. Requiere más tiempo de ejecución para mantener la estructura de lista encadenada.

Asignación de bloques

Los bloques se forman por sectores de pista contiguos (D), también denominados clústers; así se maneja de manera más eficiente la memoria secundaria y se reduce el tiempo extra de ejecución. Se trata de una mezcla de métodos de asignación contigua y no contigua.

El sistema trata de asignar bloques nuevos a un archivo eligiendo los bloques libres más cercanos a los bloques ya existentes del archivo. Cada acceso al archivo implica la determinación del bloque apropiado, así como del sector apropiado dentro del bloque.

Asignación no contigua.

Asignación de bloques (continuación)

Existen varias formas de poner en práctica los sistemas de asignación de bloques:

- Encadenamiento de bloques,
- Encadenamiento de bloques de índice y
- La correspondencia de archivos orientada a los bloques.

Revisemos cada una de ellas

Encadenamiento de bloques.

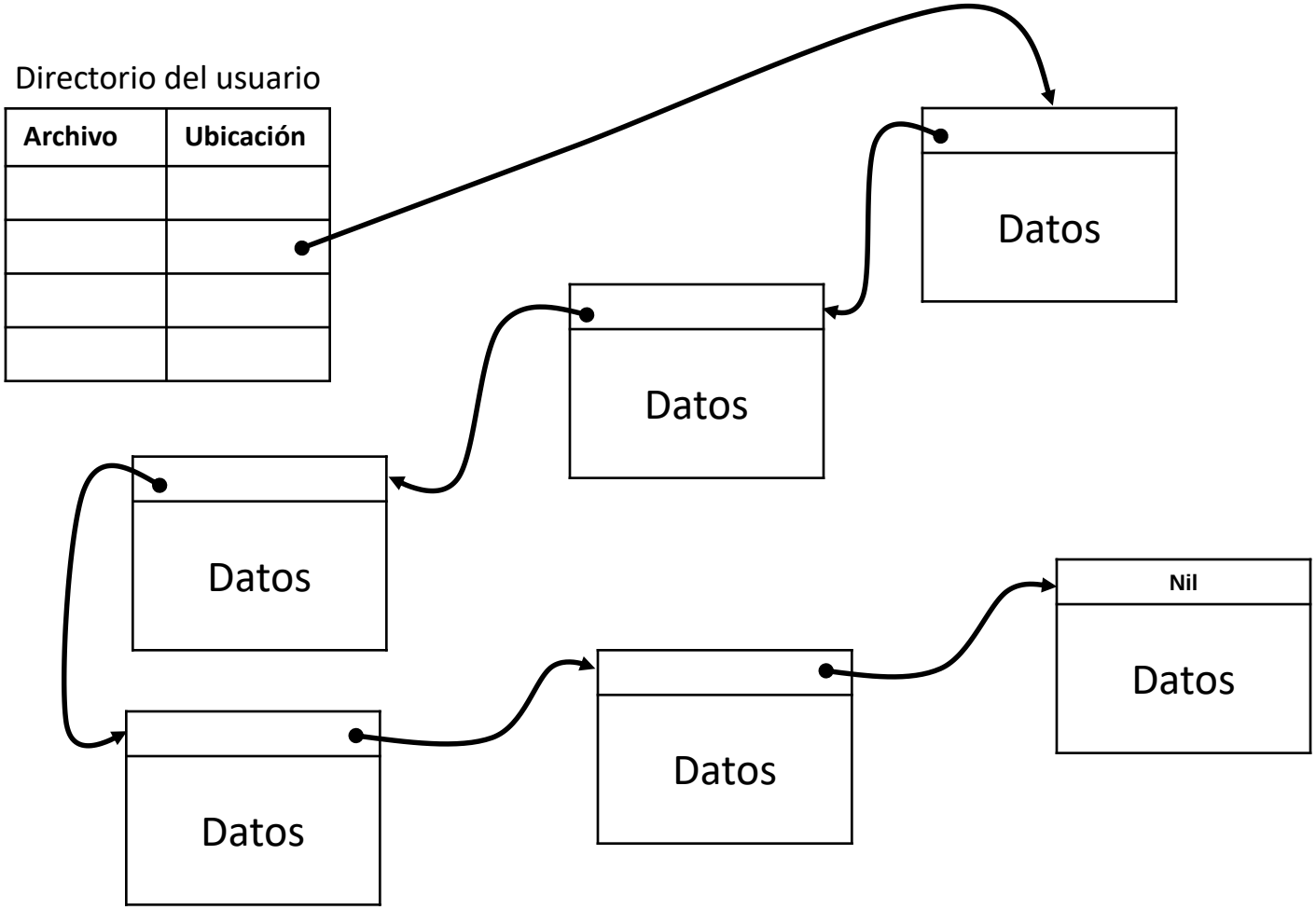
En esta técnica, las entradas del directorio del usuario apuntan al primer bloque de cada archivo. Los bloques de longitud fija que forman un archivo contienen cada uno dos partes: un bloque de datos y un apuntador al siguiente bloque. (Véase la figura en la siguiente diapositiva).

Para localizar un registro específico, es necesario seguir la cadena de bloques hasta encontrar el bloque apropiado y después buscar en ese bloque hasta encontrar el registro apropiado. Esta búsqueda puede ser lenta, pues se requieren accesos de bloque en bloque.

La inserción y la eliminación son sencillas ya que se logran modificando el apuntador del bloque anterior.

Asignación no contigua.

Encadenamiento de bloques.



Asignación no contigua.

Encadenamiento de bloques.

Ejercicio 1

Un cierto sistema de archivos utiliza el método de asignación por encadenamiento de bloques. El tamaño de bloque es de 4 sectores de una pista. La capacidad de cada sector es de 512 bytes. De los siguientes archivos, calcula cuántos bloques estarán encadenados. El apuntador de cada bloque (que apunta al siguiente bloque) ocupa 4 bytes.

Considera que son archivos con tamaño de registro lógico fijo y de acceso secuencial. Un bloque debe contener registros lógicos completos, por lo que puede haber fragmentación interna, que no es la misma para el último bloque, en la que si crece el archivo, puede ocupar ese espacio libre.

Archivo	Tamaño archivo(bytes)	Tamaño registro lógico
Arch1	10,800	120
Arch2	25,300	230

Encadenamiento de bloques.

Ejercicio 1

Respuesta
El tamaño de bloque es 512 bytes * 4 = 2,048 bytes; le restamos 4 bytes que son los que ocupa el apuntador para referenciar a otro bloque y nos quedan 2,044 bytes para datos del archivo.

Cálculos para Arch1:
de registros por bloque: trunc(2,044/120)= 17 ; quedando un fragmento de 4 bytes
de registros del archivo: 10,800/120 = 90
de bloques: round(90/17) = 6
Los primeros 5 bloques tienen un fragmento de 4 bytes, mientras que el 6º bloque está ocupado por 5 registros (600 bytes), dejando un espacio libre de 1,444 bytes para futuro crecimiento del archivo.

Cálculos para Arch2:
de registros por bloque: trunc(2,044/230)= 9 ; quedando un fragmento de 204 bytes
de registros del archivo: 25,300/230 = 110
de bloques: round(110/9) = 13
Los primeros 12 bloques tienen un fragmento de 204 bytes, mientras que el 13º bloque está ocupado por 2 registros (460 bytes), dejando un espacio libre de 1,584 bytes para futuro crecimiento del archivo.

Asignación no contigua.

Asignación de bloques (continuación)

Encadenamiento de bloques de índice .

En esta técnica, los apuntadores se colocan en bloques de índice separados. Cada bloque de índice contiene un número fijo de entradas. Cada entrada, a su vez, contiene un identificador de registro y un apuntador a ese registro.

Si se necesita más de un bloque de índice para describir a un archivo, se encadenará una serie de bloques de índice. (Véase la figura en la siguiente diapositiva).

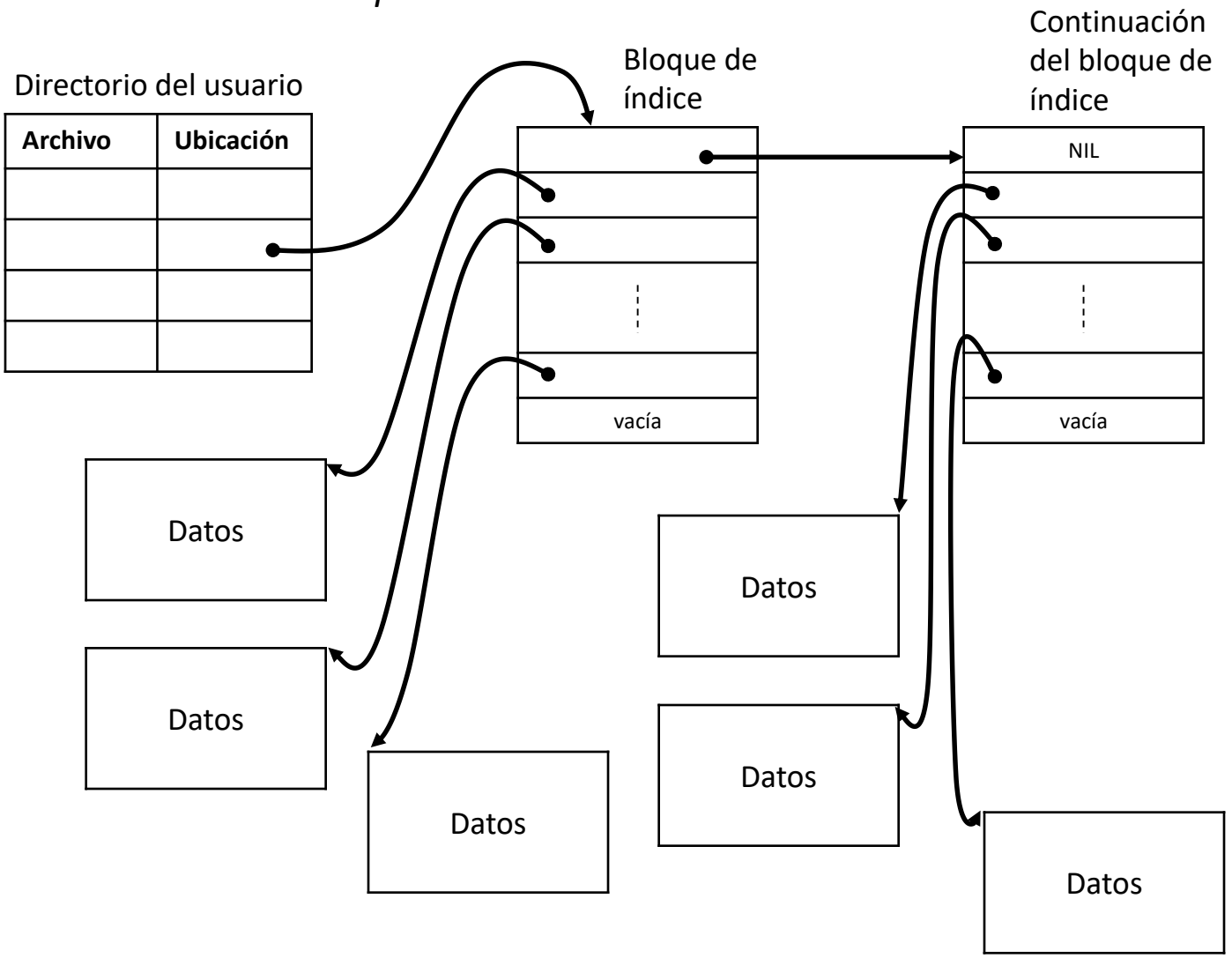
Para localizar un registro específico, puede efectuarse dentro de los bloques de índice. Estos bloques pueden encontrarse juntos en almacenamiento secundario para reducir al mínimo las búsquedas.

En algunos sistemas en los que se requiere rapidez de búsqueda, los bloques de índice se pueden mantener en memoria principal. Una vez que es localizado el registro apropiado por medio de los bloques de índice, se transfiere el registro de datos que contiene a ese registro a la memoria principal.

Desventaja: Las inserciones de registros pueden requerir la reconstrucción completa de los bloques de índice, razón por la cual algunos sistemas dejan vacía cierta porción de los bloques de índice para prevenir inserciones futuras.

Asignación no contigua.

Encadenamiento de bloques de índice.



Asignación de bloques.

Encadenamiento de bloques de índice .

Ejercicio 2.

Un cierto sistema de archivos utiliza el método de asignación por encadenamiento de bloques de índice. El tamaño de bloque es de 1024 bytes. Los bloques pueden ser de índices o de datos. Para almacenar un índice (dirección a otro bloque) utiliza 8 bytes. De los siguientes archivos, calcula cuántos bloques de índice y de datos emplea en cada uno para almacenarlo. Calcula hasta cuánto puede crecer para que no ocupe otro bloque de índice.

Archivo	Tamaño archivo(bytes)	Tamaño registro lógico
Arch1	208,896	64
Arch2	128,000	256

Respuesta
Calculemos cuántos índices caben en un bloque: 1024/8 = 128 ; entonces cada bloque de índice puede apuntar a un bloque de índice y a 127 bloques de datos. Ahora debemos calcular cuántos bloques de datos ocupa cada archivo y por ende, cuántos bloques de índice requiere.

Asignación de bloques.

Encadenamiento de bloques de índice .

Ejercicio 2 (continuación).

Cálculos para Arch1:

Bloques de datos que ocupa: round(208,896/1024)=204 ; en cada bloque de datos entran 16 registros lógicos. El total de registro lógicos es 208,896/64 = 3,264.

El número de bloques de índice que ocupa: round(204/127) = 2 ; en el último bloque de índices hay 77 índices que apuntan a bloques de datos y quedan 50 entradas libres.

Entonces el archivo puede crecer en 50*1024= 51,200 bytes, donde representan 800 registros lógicos, sin ocupar otro bloque de índices

Cálculos para Arch2:

Bloques de datos que ocupa: round(128,000/1024)=125 ; en cada bloque de datos entran 4 registros lógicos. El total de registro lógicos 128,000/256=500.

El número de bloques de índice que ocupa: round(125/128) = 1 ; en este bloque de índices hay 125 índices que apuntan a bloques de datos y quedan 2 entradas libres.

Entonces el archivo puede crecer en 2*1024= 2048 bytes, donde representan 8 registros lógicos

Asignación no contigua.

Asignación de bloques (continuación)

Correspondencia de archivos orientada a los bloques.

En este caso, el sistema utiliza número de bloques en vez de apuntadores. Éstos se convierten con relativa facilidad en direcciones reales en bloques debido a la geometría del disco. Un mapa de archivos contiene una entrada por cada bloque del disco. (Véase la figura en la siguiente diapositiva).

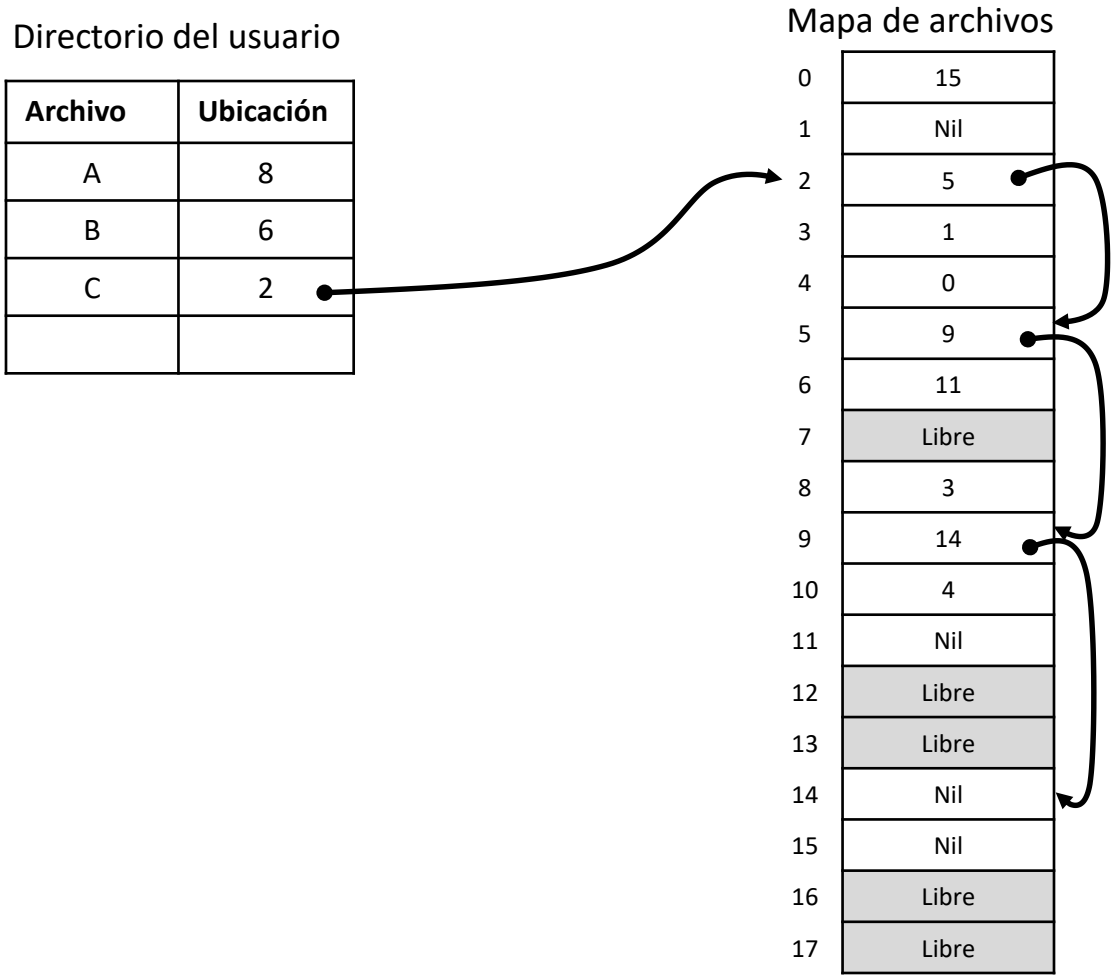
Las entradas en el directorio del usuario apunta a la primera entrada de cada archivo en el mapa de archivos. Cada entrada de este último contiene el número de bloque del siguiente bloque de ese archivo. Así, es posible localizar todos los bloques de un archivo siguiendo las entradas en el mapa de archivos.

Algunas entradas en el mapa de archivos se marcan Libres a fin de indicar que el bloque está disponible para ser asignado. El sistema puede revisar linealmente el mapa de archivos para localizar un bloque libre, o bien puede mantener un lista de bloques libres.

Una ventaja de este esquema es que las adyacencias físicas en el disco se reflejan en el mapa de archivos. Así, cuando se debe asignar un nuevo bloque, es relativamente fácil localizar un bloque bastante cercano a los demás bloques del archivo. En este esquema, las inserciones y eliminaciones son directas.

Asignación no contigua.

Correspondencia de archivos orientada a los bloques.



Administración de Archivos.

El Administrador de Archivos es otro componente del Sistema de Archivos que ofrece mecanismo para almacenar, compartir y asegurar archivos, y hacer referencia a ellos.

Bloque de control de archivos o descriptor de archivos

En la administración de archivos, es muy importante contar con información oportuna y actualizada de todos y cada uno de los archivos del sistema. Para esto, los sistemas de archivos manejan una estructura llamada *bloque de control de archivos* o *descriptor de archivos*. Esta estructura es controlado por el sistema de operativo, para cada archivo, y podría incluir:

- Nombre simbólico del archivo
- Ubicación del archivo en almacenamiento secundario
- Organización interna del archivo (secuencial, aleatorio, relativo a índice, etc.)
- Tipo de dispositivo
- Datos para el control de acceso
- Tipo (archivo de datos, programa, imagen, etc)
- Fecha y hora de creación
- Tamaño

Por lo regular, los descriptores de archivos se mantienen en memoria secundaria. Se transfieren a memoria principal cuando se abre un archivo.

Administración de Archivos.

Control de acceso.

El control de acceso a los archivos es otra tarea del administrador de archivos. Se refiere a que el archivo pueda ser utilizado sólo por el usuario o proceso que tenga permisos. Los permisos son los modos válidos de acceso para un archivo; pueden ser de lectura, escritura, adición, ejecución y una combinación de ellos. Por ejemplo, si un proceso abre un archivo en modo lectura, al proceso sólo se le permitirá leerlo.

En un sistema operativo multitareas, más de un proceso podría abrir el mismo archivo a la vez; lo que cada uno de ellos pueda hacer y cómo esto repercute en lo que vean los procesos, dependerá del control del acceso.

Existen varias formas de controlar el acceso a archivos.

Matriz para control de acceso. En una matriz bidimensional se incluye a todos los usuarios y todos los archivos del sistema. La entrada Aij es 1 si al usuario i se le permite el acceso al archivo j; de lo contrario, Aij=0. Para que esta forma de control sea más útil, es preciso utilizar códigos que indiquen los diferentes modos de acceso. Esta matriz puede crecer mucho a medida que crezca el número de usuarios y archivos, lo que propiciaría que su mantenimiento fuera impráctico.

Administración de Archivos.

Control de acceso (continuación).

Control de acceso por clases de usuarios. Al definir clases de usuarios, esta técnica requiere mucho menos espacio para manipularla. Una clasificación de usuarios muy usada es la siguiente:

- Propietario: casi siempre es el usuario que creó el archivo.
- Usuario especificado: El dueño especifica qué otro usuario tiene acceso al archivo.
- Grupo o proyecto: Grupo de usuarios que participan en el desarrollo de un proyecto y requieren compartir archivos con diferentes modos de acceso.
- Público: Cualquier usuario del sistema puede tener acceso a los archivos

Respaldo y recuperación.

Es bien sabido que la destrucción de información, ya sea accidental o intencional es un grave problema en el que se enfrenta el dueño o usuario de la información.

Se pueden tomar medidas físicas para evitar el acceso no autorizado al cuarto de cómputo, para evitar robos o daños de equipo; sin embargo, se pueden presentar hechos tan simples, como el aterrizaje de la cabeza lectora del disco propiciando que lo dañe y en consecuencia la destrucción de datos.

Administración de Archivos.

Respaldo y recuperación (continuación)

La técnica más empleada para garantizar la alta disponibilidad de los datos es realizar respaldos periódicos, es decir crear con regularidad una o más copias de los archivos del sistema y colocarlos en un lugar seguro.

Quizás, estos respaldos no sean suficientes, pues podrían perderse todas las modificaciones de un archivo cuando ocurre un incidente después del último respaldo.

Otra técnica es crear un *archivo de bitácora* de todas las transacciones copiándolas en otro disco. Esta redundancia puede ser costosa, pero en caso de fallos es posible reconstruir todo el trabajo perdido.

El respaldo periódico tiene algunos puntos débiles:

- Quizás sea necesario suspender el sistema durante la operación de respaldo
- Los sistemas de archivos pueden ser enormes; un respaldo completo podría requerir un tiempo considerable.
- Cuando ocurre una falla, la recuperación puede requerir también mucho tiempo. Y no se recuperarán los cambios que se hicieron después del último respaldo.

Integridad de Archivos.

Los mecanismos de integridad de los archivos que un sistema de archivos maneje, garantiza que no se corrompa la información en un archivo. La información que contenga el archivo debe ser consistente y completa.

Dado que los archivos pueden emplearse como mecanismo de comunicación entre procesos, los sistemas operativos multitarea implementan mecanismos de bloqueo para evitar que varios procesos, intentando emplear de forma concurrente a un archivo, lo corrompan.

Es así que se podrán usar los bloqueos para actualizar un archivo de un modo seguro, o un programa secundario podrá dejar de intentar leer un archivo que se encuentra en un estado de transición mientras otro programa está escribiendo en él.

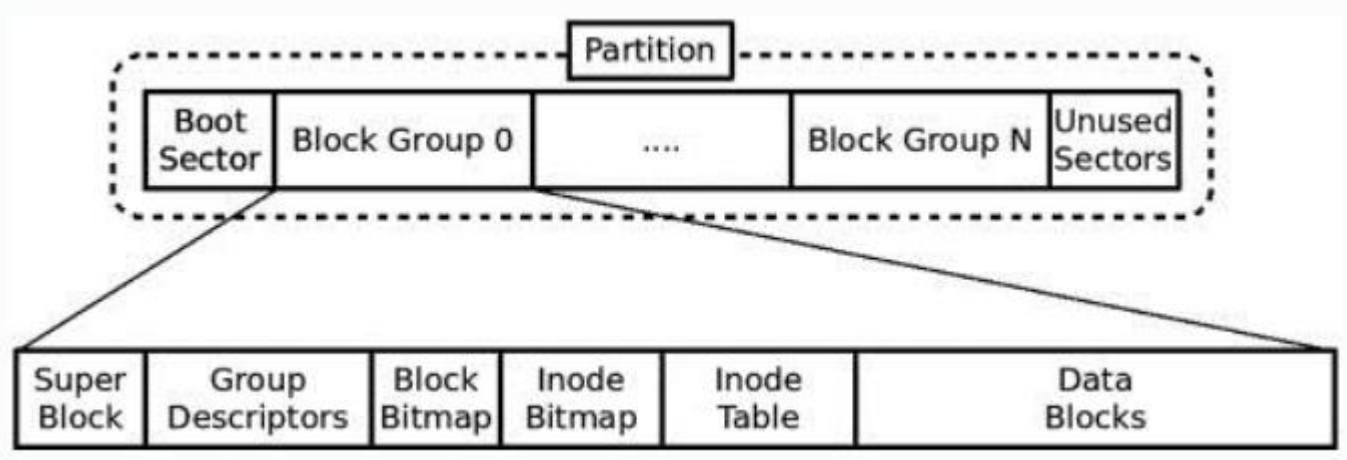
Hay dos tipos de bloqueos (candados)

- *Compartido*. Es un bloqueo para realizar lectura. Varios procesos pueden adquirir al mismo tiempo un bloqueo de lectura, e indica que todos los que tengan este candado tienen la certeza de que el archivo no sufrirá modificaciones.
- *Exclusivo.* Lo adquiere un solo proceso, e indica que realizará operaciones que modifiquen al archivo.

El Sistema de Archivos en Linux (Parte 2).

Estructura del sistema de archivos ext4 de Linux.

El sistema de archivos actual de Linux se llama ext4 (Extended File System, versión 4). Su estructura, por cada partición del disco, está conformada por un bloque reservado para el arranque del sistema y por *grupos de bloques (block group*) del mismo tamaño. A su vez, cada grupo de bloques está compuesto por el superbloque, los descriptores de grupo, el mapa de bits de bloque de datos, el mapa de bits de los inodes*, la tabla de inodes del grupo y los bloques de datos del grupo.



* Inode es la estructura de datos para localizar la información de cada archivo.

El superbloque y los descriptores de grupo de cada grupo poseen los mismos valores en todos los grupos. Esta redundancia de información permite que el sistema sea más robusto.

Estructura del sistema de archivos ext4 de Linux.

Para reducir las dificultades de rendimiento debido a la fragmentación, la asignación de bloques se realiza tratando de mantener los bloques de cada archivo dentro del mismo grupo.

Ahora describiré brevemente cada uno de los componentes del grupo de bloques.

- *Superbloque*. Describe el estado global del sistema de archivos; contiene la siguiente información: número total de inodes de todo el sistema de archivos, numero total de bloques del sistema de archivos, número de bloques reservados para el administrador, número total de bloques libres, número total de inodes libres, parámetro para calcular el tamaño del bloque lógico, parámetro para calcular el tamaño del fragmento de bloque lógico, número de bloques incluidos en un grupo, número de fragmentos de bloques lógicos incluidos en el grupo, número de inodes contenidos en un grupo, entre otros. La pérdida del superbloque supone la pérdida de todos los datos almacenados en el sistema de archivos, es por esto, que se repite en cada grupo de bloques.
- *Descriptores de grupo*. Cada descriptor de grupo está definido por una estructura de datos con la siguiente información: apuntador al bloque de datos que contiene el mapa de bits de los bloques de datos del grupo, apuntador al bloque de datos que contiene el mapa de bits de los inodes del grupo, apuntador al primer bloque de la tabla de inodes, número de bloques de datos libres en el grupo, número de inodes libres en el grupo, número de inodes asociados a directorios en el grupo.

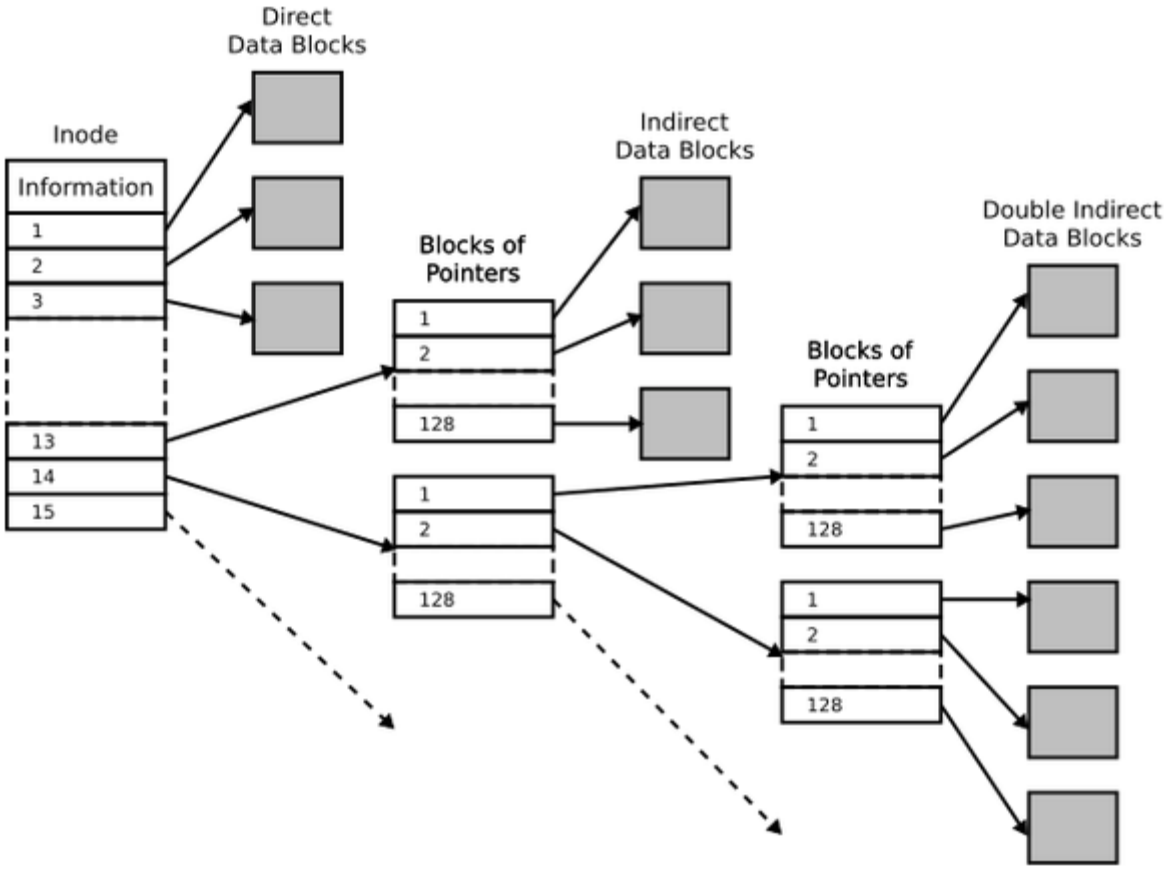
Estructura del sistema de archivos ext4 de Linux.

Los descriptores de grupo están colocados uno a continuación de otro y todos forman la tabla de descriptores de grupo.

- *Mapas de bits de bloques y de inodes*. Sirven para determinar en cada grupo el estado de cada bloque de datos y de cada inode. Existe una relación biunívoca entre cada bit del mapa de bits y cada bloque o inode, de modo que si el bit 938 del mapa de bits de datos está activo quiere decir que ese bloque de datos el 938 está asignado y viceversa. Esta es una forma eficiente de localizar bloques libres cuando sean necesarios.
- *Tabla de inodes*. El inode o nodo índice es la estructura de datos básica empleada por el sistema de archivos para localizar la información relativa a cada archivo o directorio. Cada grupo mantiene todos los inodes en una tabla.
  Los inodes contienen toda la información acerca del archivo o directorio que representan. Los principales campos de que consta la estructura inode son: tipo de archivo, derechos de acceso (lectura, escritura o ejecución para el propietario, grupo y resto de usuarios), identificador del usuario propietario del archivo (UID), tamaño del archivo en bytes, fecha de último acceso, fecha de última modificación del inode, fecha de última modificación del contenido del archivo, identificador de grupo al que pertenece el propietario del archivo (GID), número de enlaces duros a este archivo, varias banderas para manejo de datos y un arreglo de apuntadores a los bloques de datos del archivo.

Estructura del sistema de archivos ext4 de Linux.

El arreglo de apuntadores a los bloques de datos de cada archivo está conformado por 12 apuntadores directos, uno indirecto simple, uno indirecto doble y uno triple. Véase la siguiente figura.



Estructura del sistema de archivos ext4 de Linux.

Funcionamiento del arreglo de apuntadores a bloques de datos:

Apuntador directo o simple: apunta directamente al bloque de datos.

Apuntador indirecto simple: apunta a un bloque con índices o apuntadores a bloques de datos.

Apuntador indirecto doble: apunta a un bloque de índices; cada índice apunta a otro bloque de índices, los cuales apuntan a bloques de datos.

Apuntador indirecto triple: apunta a un bloque de índices; cada índice apunta a otro bloque de índices, a su vez, cada uno apunta a un bloque de índices, que finalmente apuntan a bloques de datos.

Ejercicio 3.

Se tienen los siguientes archivos almacenados en disco bajo sistema operativo Linux.

Calcula cuántos apuntadores ocupan y de qué tipo del arreglo de apuntadores a bloques de datos de su inode. Considera que el tamaño de bloque es de 1024 bytes y ocupa 8 bytes para un índice o dirección.

Archivo	Tamaño(bytes)
File1	8,704
File2	3,145,728

Ejercicio 3 (continuación).

Respuesta

Cada bloque de índices tiene 1024/8 = 128 índices.

Cálculo para File1

de bloques que ocupa: round(8,704/1,024)= 9

Como en el arreglo de apuntadores del inode tiene 12 apuntadores directos, sólo ocupa 9 apuntadores directos.

Cálculo para File2

de bloques que ocupa: round(3,145,728/1,024)= 3,072

Vemos claramente que no son suficientes los 12 apuntadores directos, por lo que se tiene que ocupar el indirecto simple para apuntar a 128 bloques más.

Aún así no son suficientes. Por lo que se ocupa el indirecto doble que puede apuntar a un total de 128*128 = 16,384 bloques de datos. Esto sobrepasa en mucho lo requerido para el archivo. Veamos entonces qué tantos índices se ocupan del apuntador indirecto doble:

Se usa este apuntador, para apuntar a 3,072 - 12 - 128 = 2,932 bloques, por lo que calculemos cuántos índices se ocupan del bloque de índices del primer nivel: round(2,932/128) = 23 índices. Donde, 22 bloques de índices de segundo nivel ocupan sus 128 índices para apuntar a bloques de datos y el bloque de índices 23, ocupa sólo 116 de sus 128 índices.

Preguntas base para siguiente examen. Contestarlas en sus notas.

- Diferencia entre compactación del disco y su recolector de basura.
- Distinguir las ventajas y desventajas de la asignación contigua y no contigua de memoria secundaria.
- Identificar los diferentes esquemas de asignación no contigua de memoria.
- Funciones y contenido del bloque de control de archivos.
- Diferencias entre los tipos de control de acceso a los archivos.
- ¿Por qué el archivo de bitácora es un complemento en el respaldo de archivos?
- ¿Cuál es la función primordial de los bloqueos y cuáles son sus tipos?

Ejercicio A

Un cierto sistema de archivos utiliza el método de asignación por encadenamiento de bloques. El tamaño de bloque es de 8 sectores de una pista. La capacidad de cada sector es de 128 bytes. Para un archivo cuyo tamaño es de 6,600 bytes y cuyo registro lógico es de tamaño fijo de 55 bytes, calcula cuántos bloques estarán encadenados. El apuntador de cada bloque (que apunta al siguiente bloque) ocupa 4 bytes.

Calcula además el valor del fragmento interno y el tamaño de espacio disponible del último bloque para el crecimiento del archivo.

Preguntas base para siguiente examen (continuación). Contestarlas en sus notas.

Ejercicio B.

Un cierto sistema de archivos utiliza el método de asignación por encadenamiento de bloques de índice. El tamaño de bloque es de 1024 bytes. Los bloques pueden ser de índices o de datos. Para almacenar un índice (dirección a otro bloque) utiliza 4 bytes. Para un archivo cuyo tamaño es de 404,480 bytes y cuyo registro lógico es de tamaño fijo de 32 bytes, calcula cuántos bloques de índice y de datos emplea en cada uno para almacenarlo. Calcula hasta cuánto puede crecer para que no ocupe otro bloque de índice.

Del sistema de archivos de Linux:

- ¿Qué partes del los grupos de bloques se repiten en cada grupo con la misma información y a qué se debe esto?
- ¿A cuántos bloques de datos se pueden referenciar a través del apuntador indirecto triple si el tamaño de bloque es de 1024 bytes y el índice ocupa 8 bytes?

Ejercicio C.

En el sistema de archivos ext4, considerando que el tamaño de bloque es de 1024 bytes y ocupa 8 bytes para un índice o dirección, calcula el tamaño máximo (en bytes) de un archivo si ocupara hasta la 2ª entrada del primer nivel del apuntador indirecto triple.