FILE SYSTEM IMPLEMENTATION

Mrs.S.Lakshmi Priya, AP/CSE

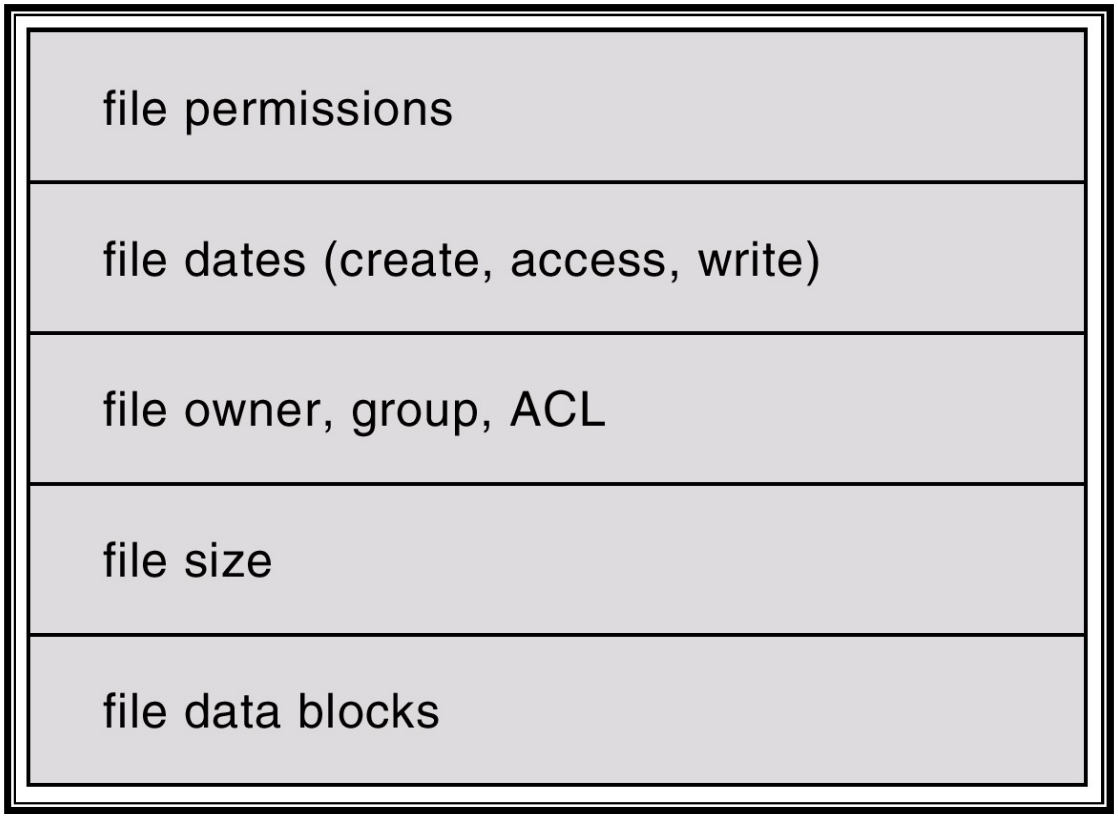
Topics to be discussed

- Directory Implementation
- Allocation Methods
- Free Space Management
- Efficiency and Performance
- Recovery
- Log-Structured File System

File-System Structure

- File structure
  - Logical storage unit
  - Collection of related information
- File system resides on secondary storage (disks).
- File system organized into layers.
- *File control block* – storage structure consisting of information about a file.

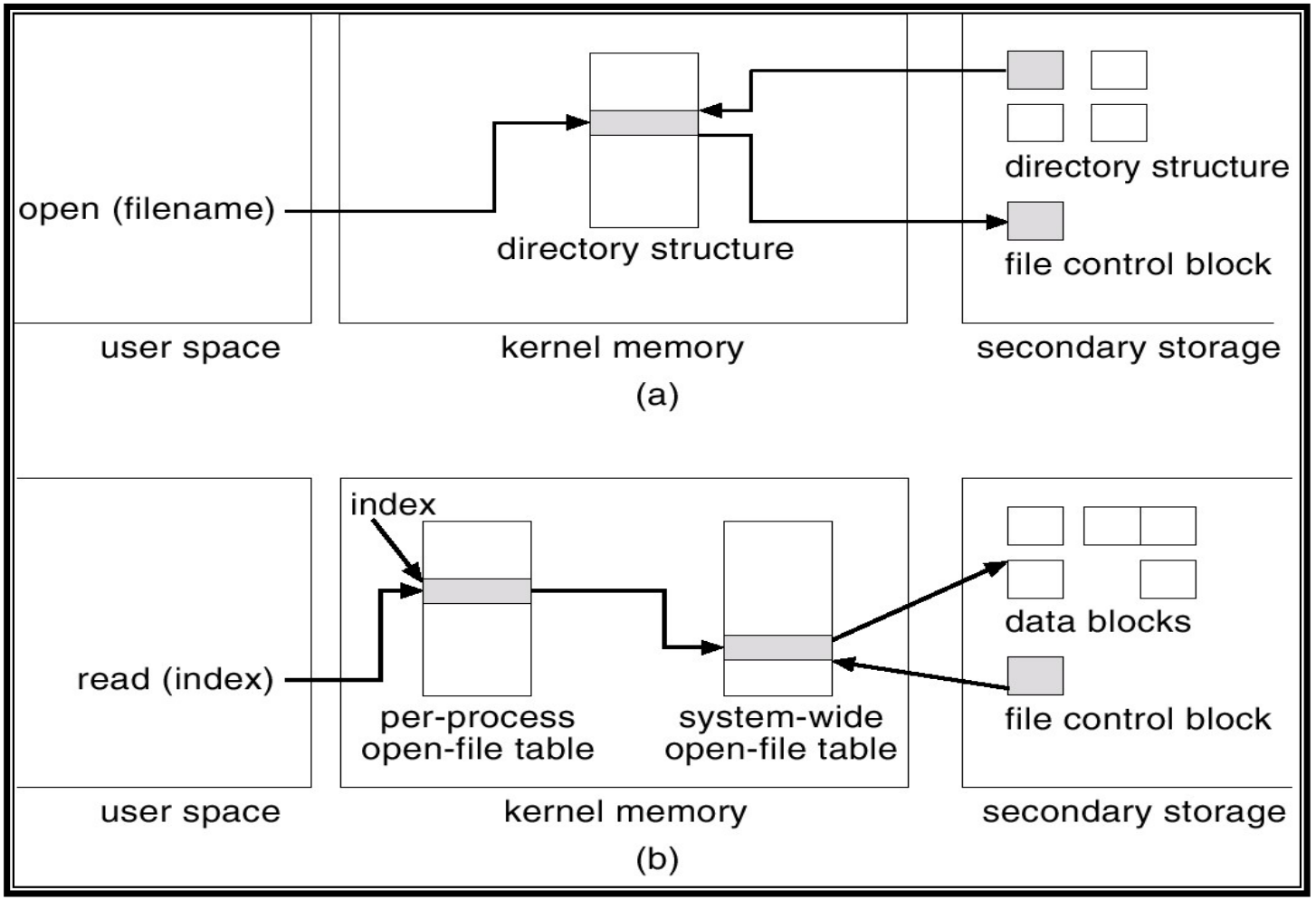
A Typical File Control Block



In-Memory File System Structures

- The following figure illustrates the necessary file system structures provided by the operating systems.
- Figure 12-3(a) refers to opening a file.
- Figure 12-3(b) refers to reading a file.

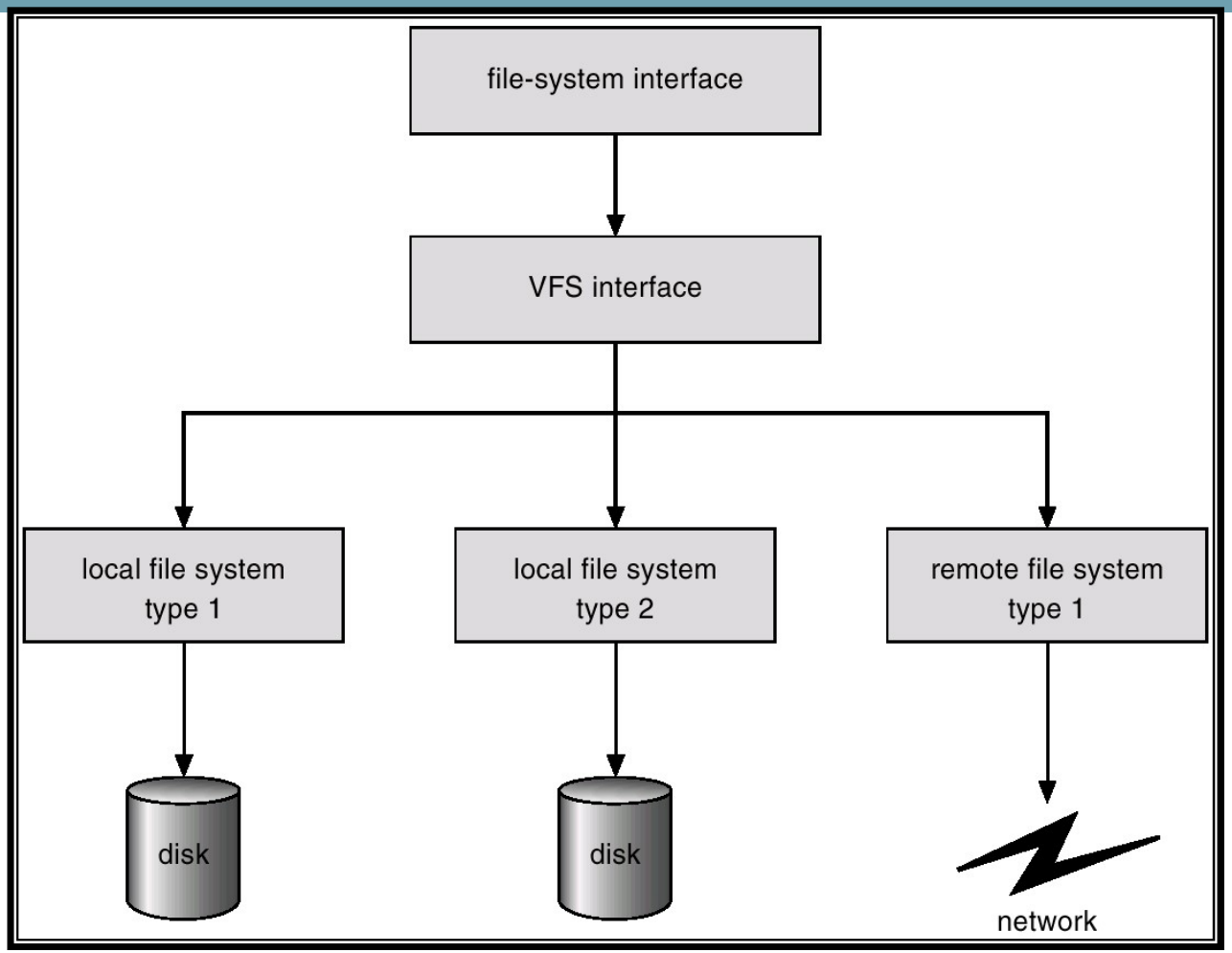
In-Memory File System Structures



Virtual File Systems

- Virtual File Systems (VFS) provide an object-oriented way of implementing file systems.
- VFS allows the same system call interface (the API) to be used for different types of file systems.
- The API is to the VFS interface, rather than any specific type of file system.

Schematic View of Virtual File System



Directory Implementation

- Linear list of file names with pointer to the data blocks.
  - simple to program
  - time-consuming to execute – linear search involved for file operations
- Hash Table – linear list with hash data structure.
  - decreases directory search time
  - collisions – situations where two file names hash to the same location
  - Difficulties:
    - fixed size
    - Dependency of hash function on that size.

Allocation Methods

An allocation method refers to how disk blocks are allocated for files:

- Contiguous allocation
- Linked allocation
- Indexed allocation

1.Contiguous Allocation

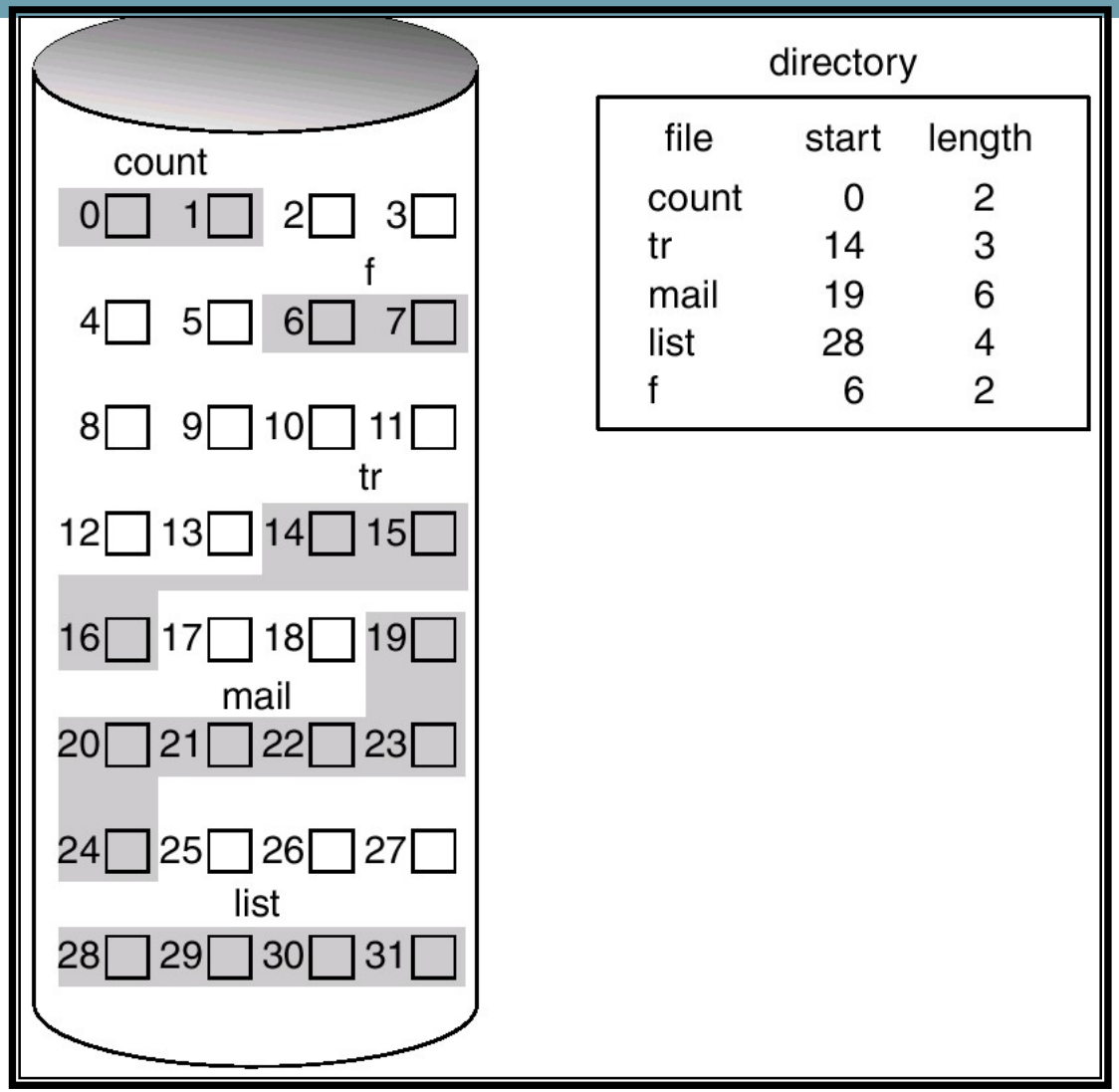
- Each file occupies a set of contiguous blocks on the disk.
- Disk addresses define a linear ordering on the disk.
- Simple – only starting location (block #) and length (number of blocks) are required.
- If file is n blocks, and starts at block b then, it occupies, blocks b,b+1,b+2….b+n-1.

- Both sequential (b+1) and direct access(b+i) of files are supported.

Contiguous Allocation Contd.,

- Wastage of space (dynamic storage-allocation problem).
- Difficulties:
  - External fragmentation may occur.
  - Problem in determining how much space is needed for a file.
    - If too little space is allocated, and then the owner wants to expand the file, two possibilities exist.
      - User program may be terminated with an error message.
      - Find a larger hole, copy the file contents and release the previous hole. – time consuming
    - If too much space is allocated, causes internal fragmentation.

Contiguous Allocation of Disk Space



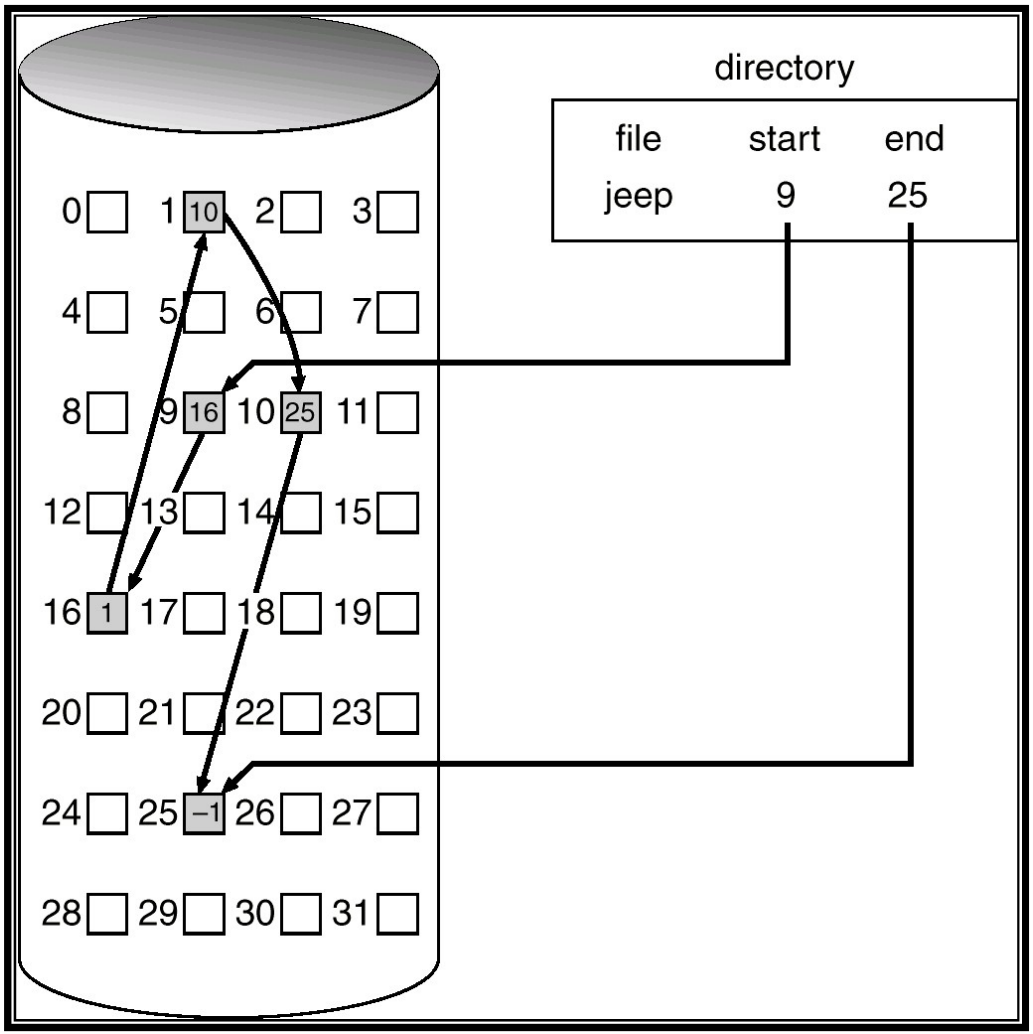
Extent-Based Systems

- Modified version of contiguous allocation scheme.
- A file is allocated contiguous chunks of space initially, and then if needed a extent is added which is another chunk of contiguous space.
- The location of file's blocks is recorded and block count plus a link to the first block of next extent.
- Many newer file systems (I.e. Veritas File System) uses this scheme.

2.Linked Allocation

- Each file is a linked list of disk blocks: blocks may be scattered anywhere on the disk.
  - Starting address is stored in the dir entry.
  - Each block of the file will contain a pointer to the next one.
  - EOF is marked by a special value.
  - If the file is appended, just fine a free block, write the content and link it to the current last block.
- No external fragmentation
- Size of the file does not need to be declared when that file is created.

Linked Allocation



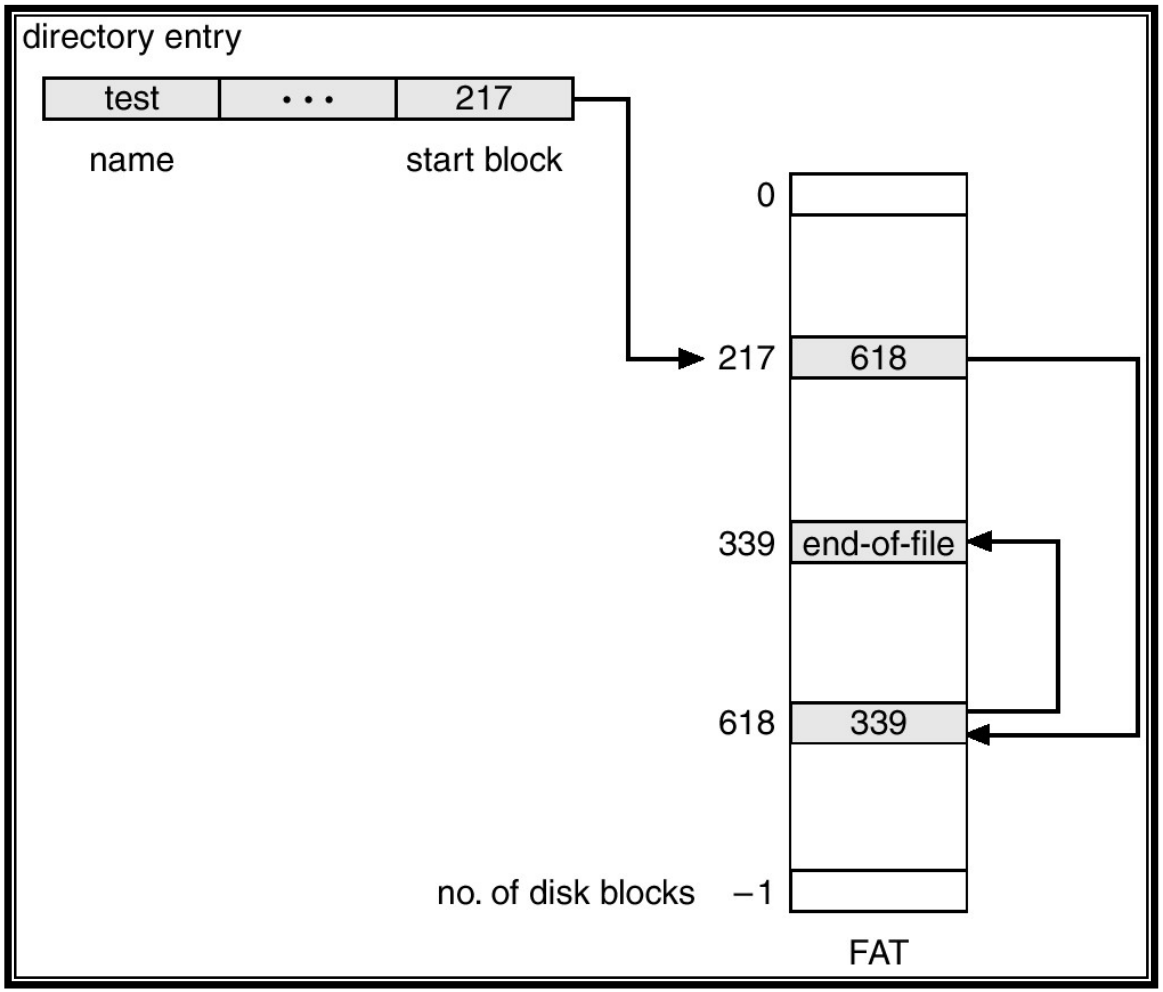
Linked Allocation (Cont.)

- Disadvantages:
  - Can be used only for sequential access files.
  - Space required for the pointers
    - Collect blocks into multiples called clusters and allocate clusters instead of blocks – problem: internal fragmentation.
  - Reliability – if a pointer is lost or damaged?
- Variation on linked allocation – FAT
  - A section of disk at the beginning of each partition is set aside to contain the table.

File Allocation Table

- The table has one entry for each disk block and is indexed by the block number.
- The dir entry has the block number of the first block of the file.
- The table entry indexed by that block number then contains the block number of the next block.
- The chain continues until the last block, which has a special end-of-file value as the table entry.
- Unused blocks are indicated by 0.
- Allocating a new block:
  - Find the first 0 valued entry and replace the previous EOF value with the address of the new block.

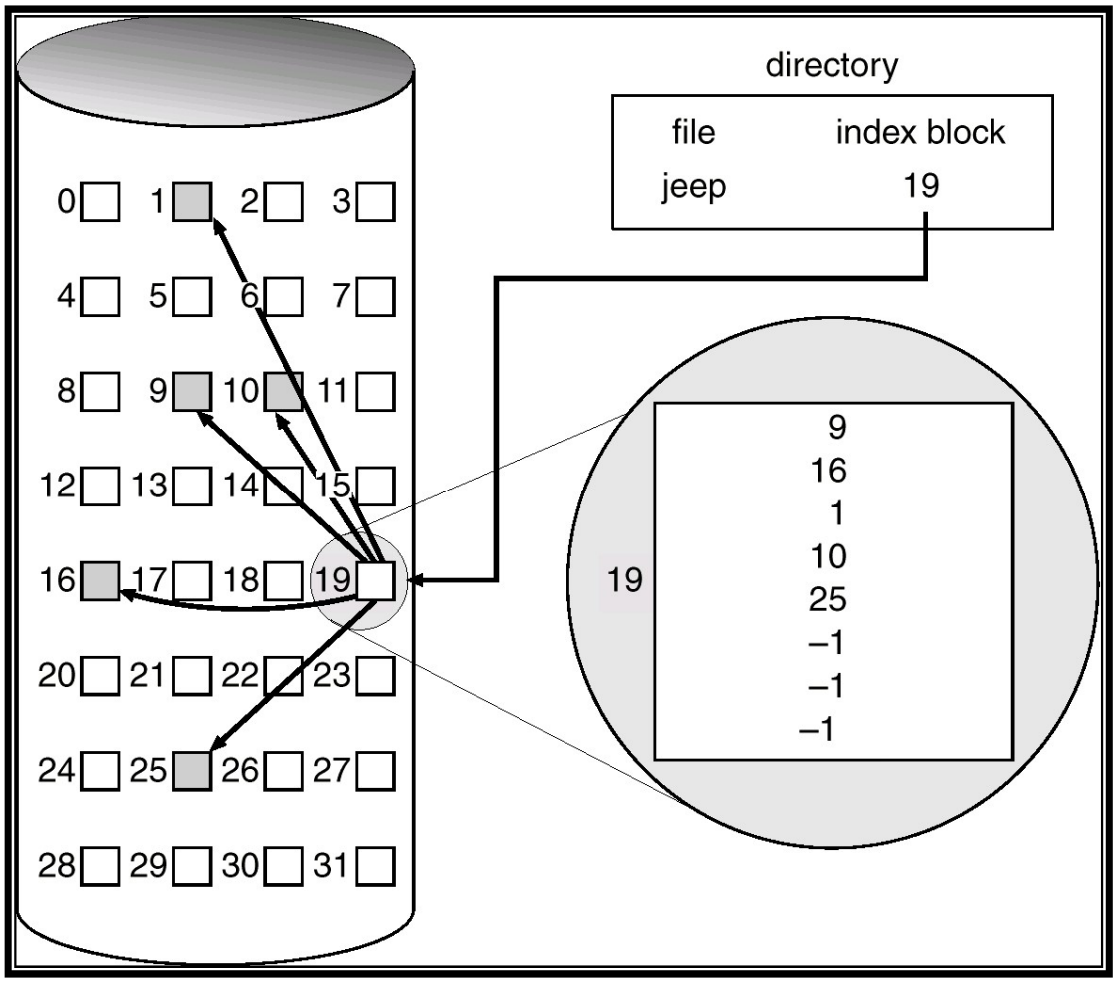
File-Allocation Table



3.Indexed Allocation

- Linked allocation doesn't support random access since the pointers are scattered all over the disk.
- Brings all pointers together into the index block.
- Index block:
  - Each file has its own index block which is an array of disk block addresses.
  - The ith entry in the index block points to the ith block of the file.
  - The directory entry contains the address of the index block.

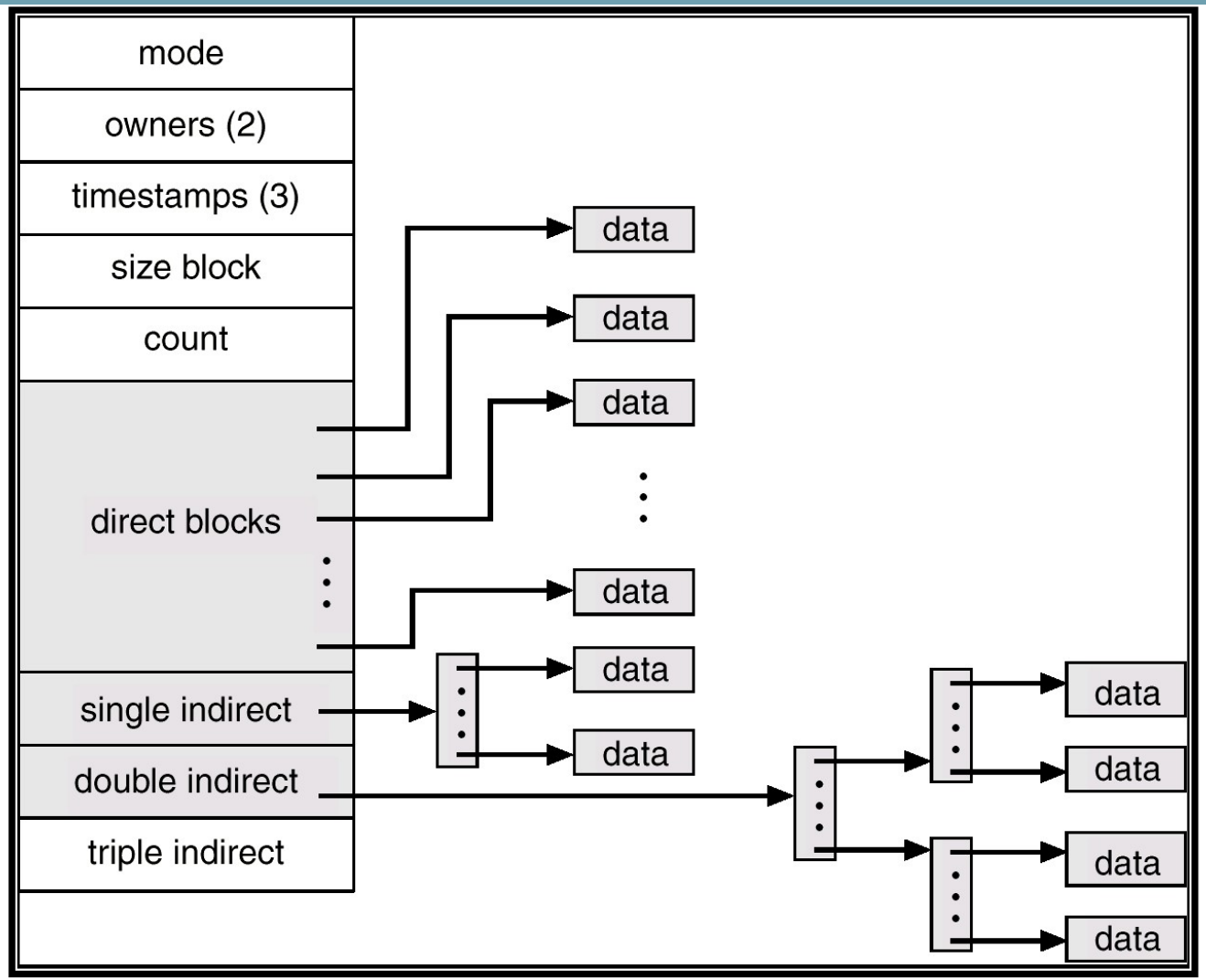
Example of Indexed Allocation



Indexed Allocation (Cont.)

- Supports Random access
- No external fragmentation, but have overhead of index block.
- How large should the index block be?
  - Linked scheme: a index block is one disk block. If not enough, link several index blocks together.
  - Multilevel Index: index of index blocks
  - Combined scheme: Keep the first 15 pointers of the index block in the file's inode.
    - First 12 –direct blocks
    - Single indirect, double indirect and triple indirect.

Combined Scheme: UNIX (4K bytes per block)



Free Space Management

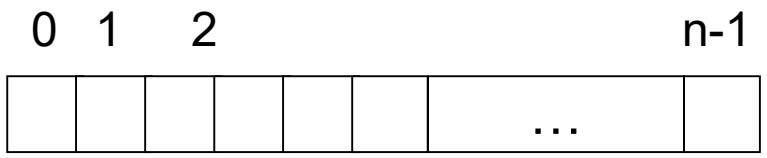
To keep track of free disk space the system maintains a free-space list.

Implemented as

Bit vector, Linked list, Grouping, Counting

Free-Space Management

- Bit vector - Each block is represented by a bit
  - For n blocks,



$$bit[i] = \begin{cases} 0 \Rightarrow block[i] \text{ allocated} \\ 1 \Rightarrow block[i] \text{ free} \end{cases}$$

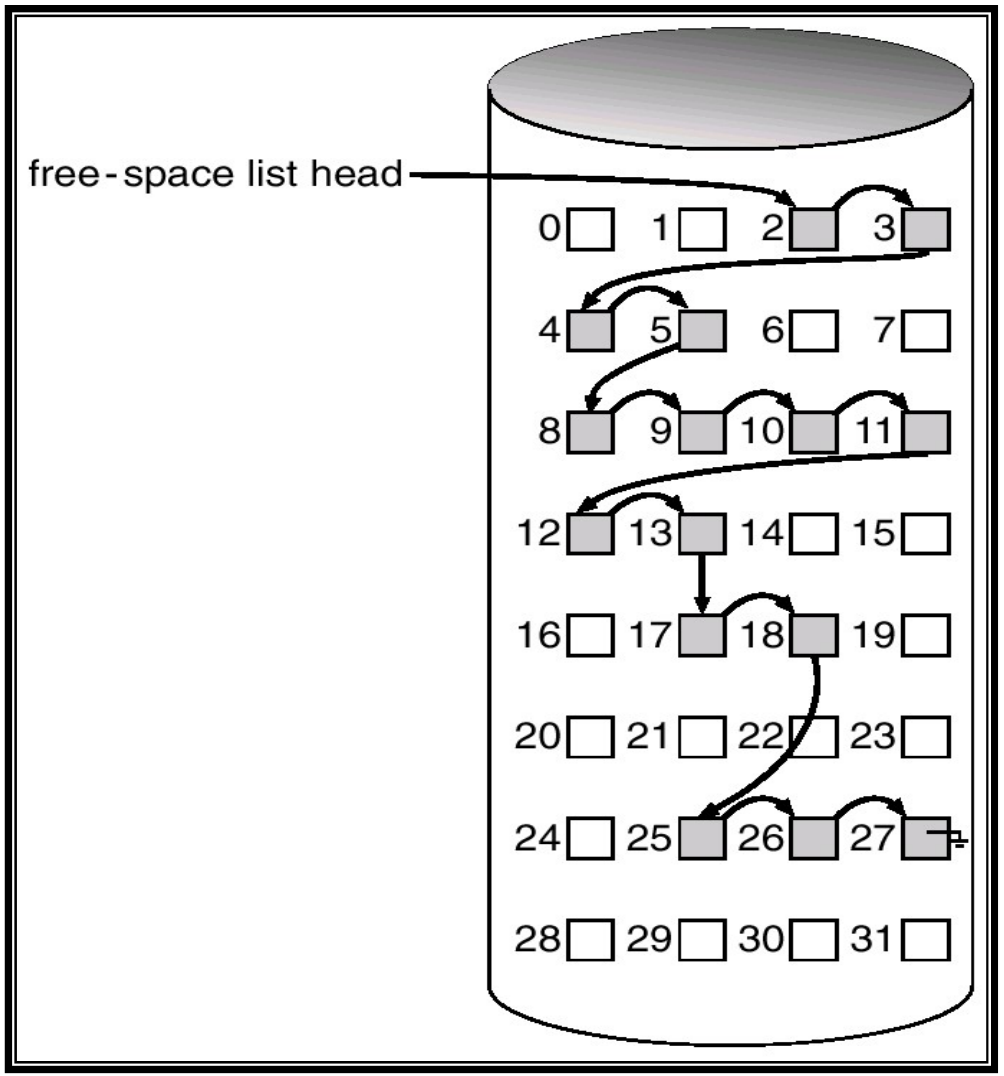
  - The Apple MAC OS checks sequentially each word in the bit map to see whether that value is not 0,since a 0-valued word will have all 0 bits .
  - The first non-zero word is scanned for first 1 bit and is allotted to the requesting process.
  - The calculation of block number is

(number of bits per word) * (number of 0-value words) + offset of first 1 bit

Free-Space Management (Cont.)

- Linked list (free list)
  - Link together all the free disk blocks, keeping a pointer to the first free block in a special location on disk and caching it in memory.
- Grouping
  - Store the addresses of n free blocks in the first free block
  - The first n-1 of these blocks are actually free.
  - The last block contains the addresses of another n blocks.
- Counting
  - Rather than keeping a list of n free disk addresses, keep the address of first free block and number n of free contiguous blocks that follow the first block.
  - Each entry in the free space list then consists of a disk address and a count.

Linked Free Space List on Disk



Summary

- How file system implemented
- Data structures needed
  - Kernel
  - On disk
- File Allocation methods
  - Contiguous
  - Linked
  - Indexed
- Free space management