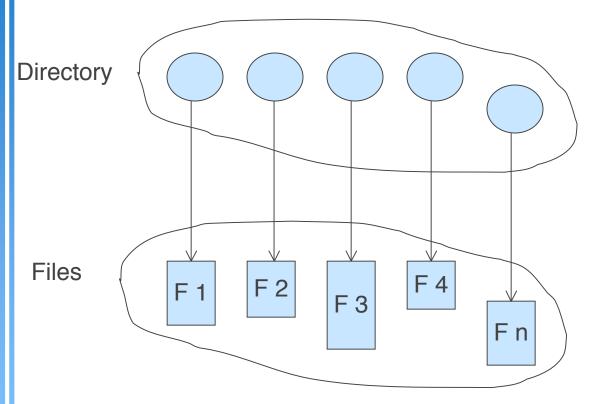
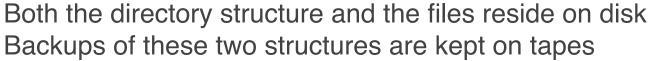
Directory Structure

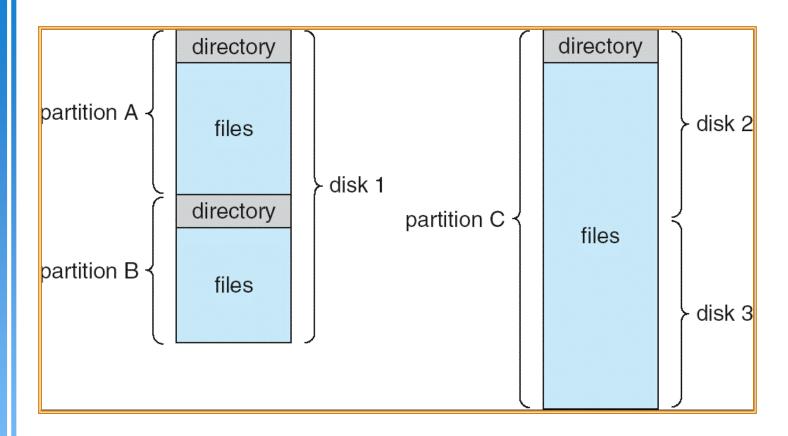
A collection of nodes containing information about all files







A Typical File-system Organization





Operations Performed on Directory

- Search for a file
- Create a file
- Delete a file
- List a directory
- Rename a file
- ▶ Traverse the file system



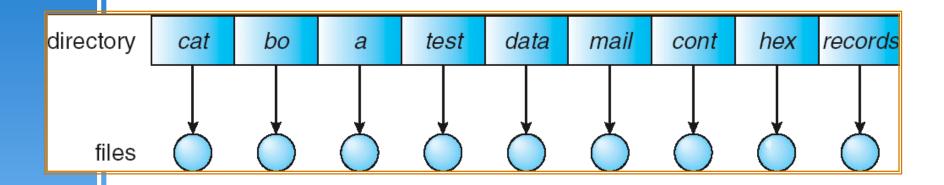
Organize the Directory (Logically) to Obtain

- Efficiency locating a file quickly
- Naming convenient to users
 - Two users can have same name for different files
 - The same file can have several different names
- Grouping logical grouping of files by properties, (e.g., all Java programs, all games, ...)



Single-Level Directory

A single directory for all users



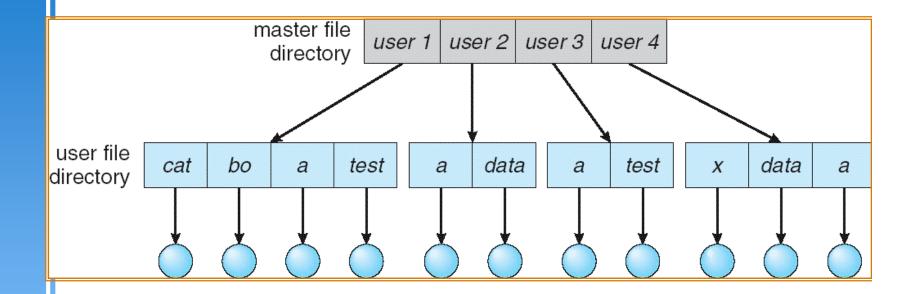
Naming problem

Grouping problem



Two-Level Directory

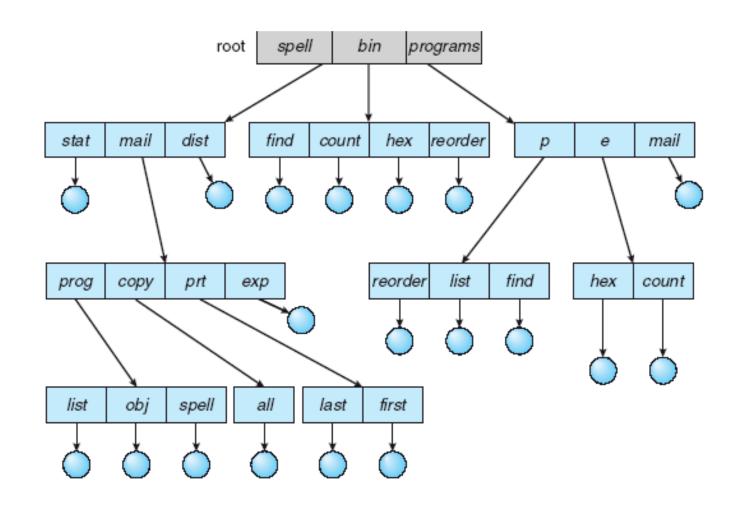
Separate directory for each user





- Path name
- v Can have the same file name for different user
- v Efficient searching
- No grouping capability

Tree-Structured Directories





Tree-Structured Directories (Cont)

- Efficient searching
- Grouping Capability
- Current directory (working directory)
 - cd /spell/mail/prog
 - type list



Tree-Structured Directories (Cont)

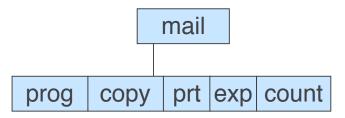
- Absolute or relative path name
- Creating a new file is done in current directory
- Delete a file

```
rm <file-name>
```

Creating a new subdirectory is done in current directory

```
mkdir <dir-name>
```

Example: if in current directory /mail mkdir count

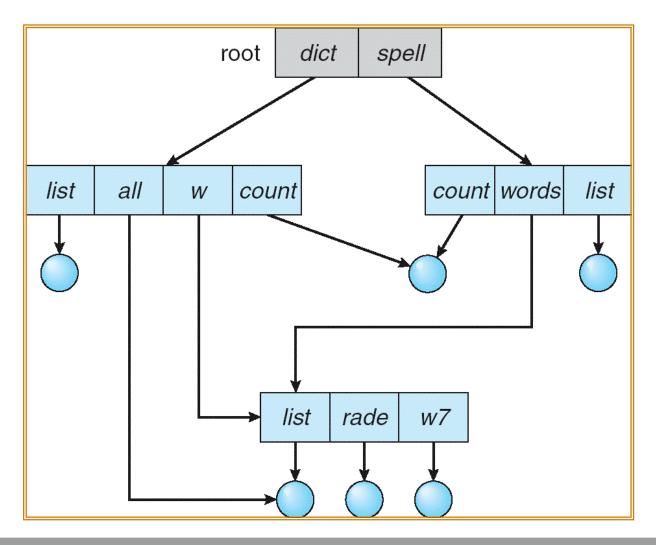


Deleting "mail" ⇒ deleting the entire subtree rooted by "mail"



Acyclic-Graph Directories

Have shared subdirectories and files



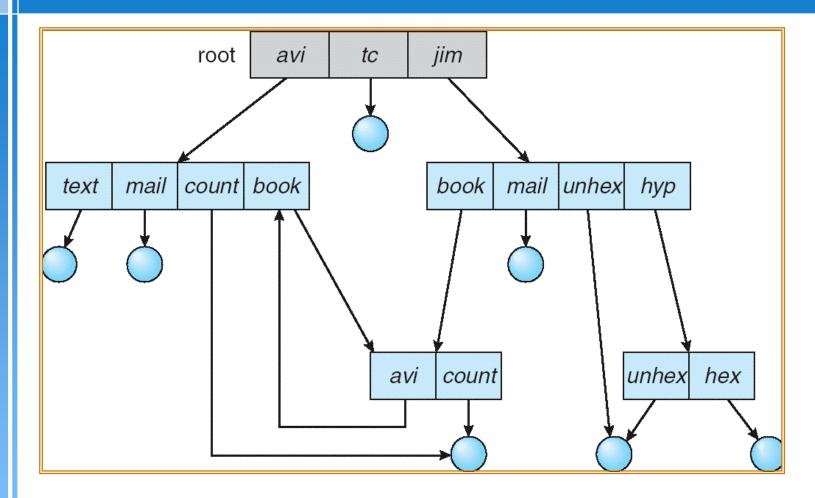


Acyclic-Graph Directories (Cont.)

- Two different names (aliasing)
- If dict deletes list ⇒ dangling pointer Solutions:
 - Backpointers, so we can delete all pointers
 Variable size records a problem
 - Backpointers using a daisy chain organization
 - Entry-hold-count solution
- New directory entry type
 - Link another name (pointer) to an existing file
 - Resolve the link follow pointer to locate the file



General Graph Directory





General Graph Directory (Cont.)

- ▶ How do we guarantee no cycles?
 - Allow only links to file not subdirectories
 - Garbage collection
 - Every time a new link is added use a cycle detection algorithm to determine whether it is OK

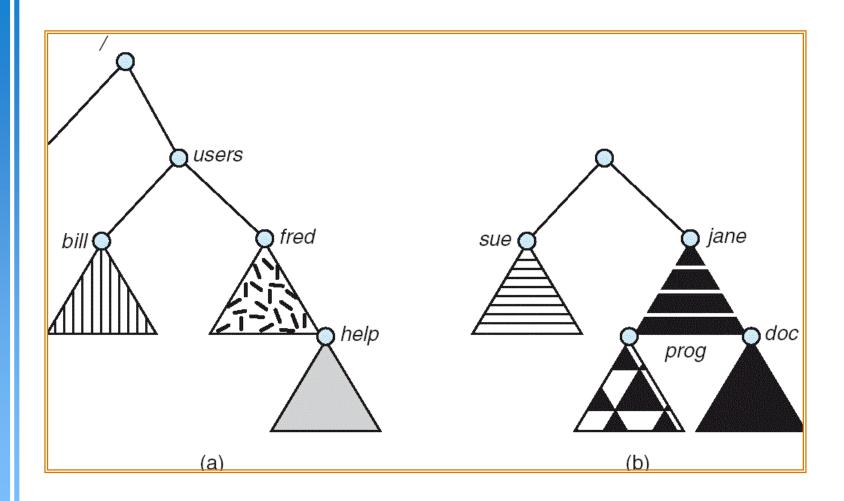


File System Mounting

- A file system must be mounted before it can be accessed
- ▶ A unmounted file system (i.e. Fig. 11-11(b)) is mounted at a mount point

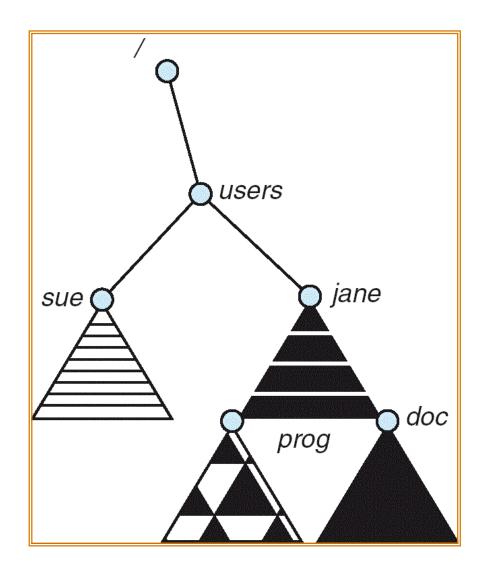


(a) Existing. (b) Unmounted Partition





Mount Point





File Sharing

- Sharing of files on multi-user systems is desirable
- Sharing may be done through a protection scheme
- On distributed systems, files may be shared across a network
- Network File System (NFS) is a common distributed file-sharing method



File Sharing – Multiple Users

- User IDs identify users, allowing permissions and protections to be per-user
- Group IDs allow users to be in groups, permitting group access rights



File Sharing – Consistency Semantics

- Consistency semantics specify how multiple users are to access a shared file simultaneously
 - Similar to Ch 7 process synchronization algorithms
 - Tend to be less complex due to disk I/O and network latency (for remote file systems
 - Andrew File System (AFS) implemented complex remote file sharing semantics
 - Unix file system (UFS) implements:
 - Writes to an open file visible immediately to other users of the same open file
 - Sharing file pointer to allow multiple users to read and write concurrently
 - AFS has session semantics
 - Writes only visible to sessions starting after the file is closed



Protection

- ▶ File owner/creator should be able to control:
 - what can be done
 - by whom
- Types of access
 - Read
 - Write
 - Execute
 - Append
 - Delete
 - List

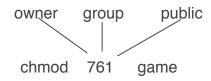


Access Lists and Groups

- Mode of access: read, write, execute
- Three classes of users

		RWX
7	\Rightarrow	111
		RWX
6	\Rightarrow	110
		RWX
1	\Rightarrow	0 0 1
	7 6 1	6 ⇒

- Ask manager to create a group (unique name), say G, and add some users to the group.
- For a particular file (say *game*) or subdirectory, define an appropriate access.

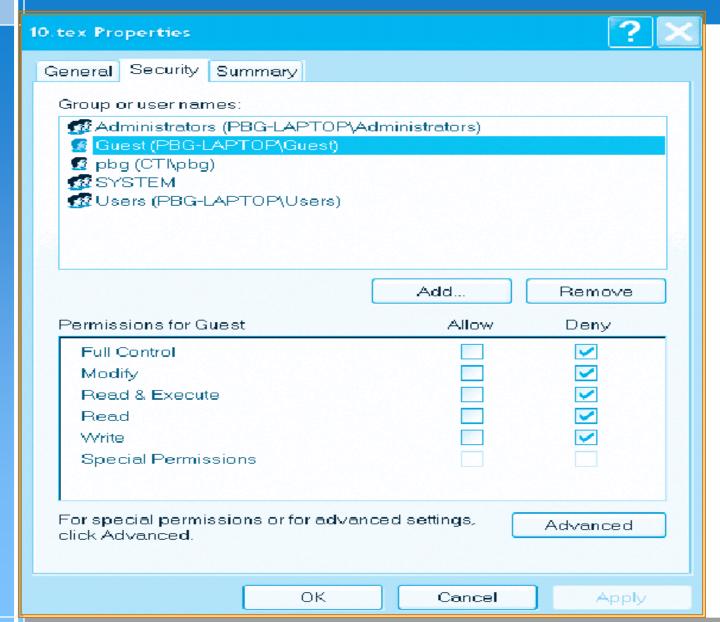


Attach a group to a file

chgrp G game



Vindows XP Access-control List Management





A Sample UNIX Directory Listing

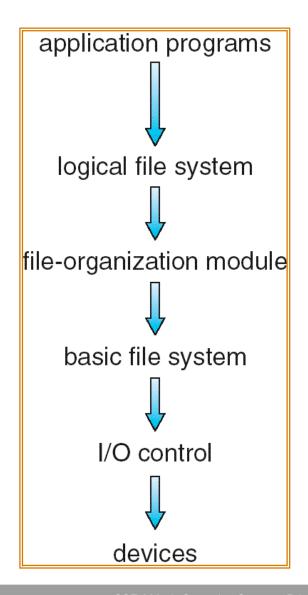
-rw-rw-r	1 pbg	staff	31200	Sep 3 08:30	intro.ps
drwx	5 pbg	staff	512	Jul 8 09.33	private/
drwxrwxr-x	2 pbg	staff	512	Jul 8 09:35	doc/
drwxrwx	2 pbg	student	512	Aug 3 14:13	student-proj/
-rw-rr	1 pbg	staff	9423	Feb 24 2003	program.c
-rwxr-xr-x	1 pbg	staff	20471	Feb 24 2003	program
drwxxx	4 pbg	faculty	512	Jul 31 10:31	lib/
drwx	3 pbg	staff	1024	Aug 29 06:52	mail/
drwxrwxrwx	3 pbg	staff	512	Jul 8 09:35	test/

Chapter 11: File System Implementation

- File structure
 - Logical storage unit
 - Collection of related information
- File system resides on secondary storage (such as disks)
- 1. Boot control block information needed to boot
- 2. Volume control block information about volume/partitions (# blocks, size of blocks, free block count, free block pointers)
- 3. Directory structure (inode)
- 4. Per file control blocks
- File system organized into layers



Layered File System





A Typical File Control Block

▶ File control block – storage structure consisting of information about a file

file permissions

file dates (create, access, write)

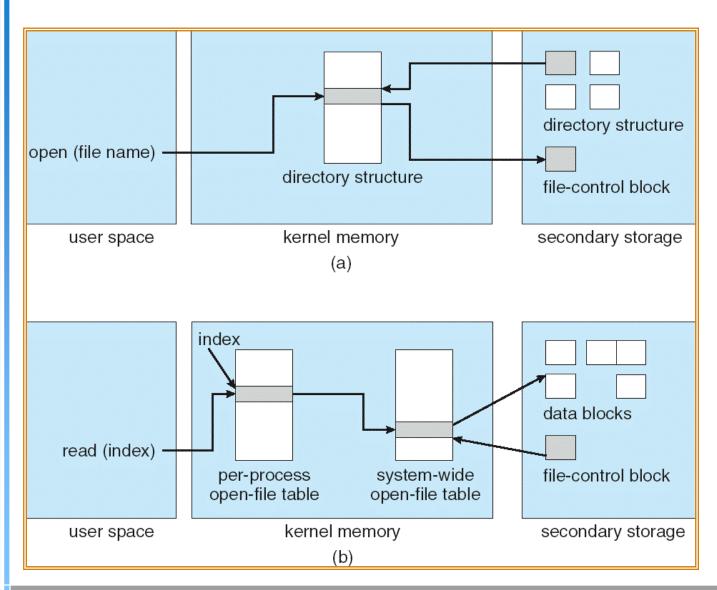
file owner, group, ACL

file size

file data blocks or pointers to file data blocks



In-Memory File System Structures



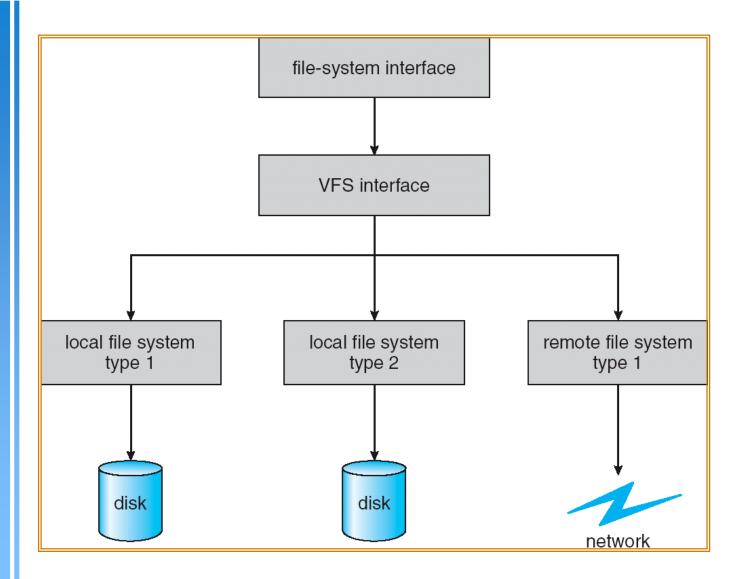


Virtual File Systems

- There are many different file systems available on any operating systems
 - Windows: NTFS, FAT, FAT32
 - Linux: ext2/ext3, ufs, vfat, ramfs, tmpfs, reiserfs, xfs ...
- Virtual File Systems (VFS) provide an objectoriented 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

- Directories hold information about files
- Linear list of file names with pointer to the data blocks.
 - simple to program
 - time-consuming to execute
- ▶ Hash Table linear list with hash data structure.
 - decreases directory search time
 - collisions situations where two file names hash to the same location
 - fixed size



Allocation Methods

- An allocation method refers to how disk blocks are allocated for files:
- Contiguous allocation
- Linked allocation
- Indexed allocation

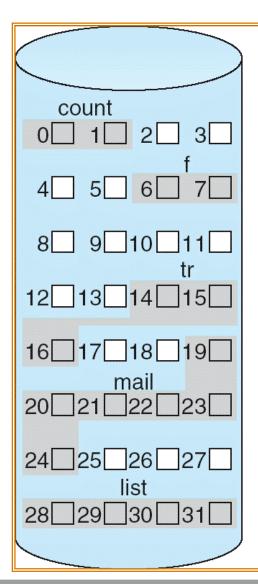


Contiguous Allocation

- Each file occupies a set of contiguous blocks on the disk
- Simple only starting location (block #) and length (number of blocks) are required
- Random access
- Wasteful of space (dynamic storageallocation problem)
- ▶ Files cannot grow



Contiguous Allocation of Disk Space



directory

start |

file start length count 0 2 tr 14 3 mail 19 6 list 28 4 f 6 2



Extent-Based Systems

- Many newer file systems (I.e. Veritas File System) use a modified contiguous allocation scheme
- Extent-based file systems allocate disk blocks in extents
- An extent is a contiguous block of disks
 - Extents are allocated for file allocation
 - A file consists of one or more extents.



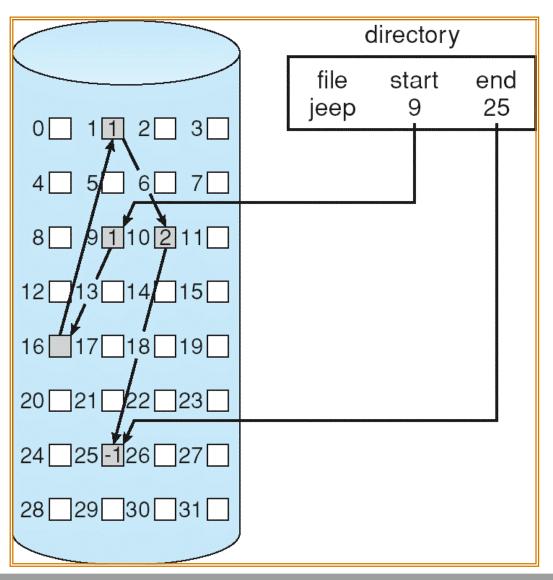
Linked Allocation

▶ Each file is a linked list of disk blocks: blocks may be scattered anywhere on the disk.

- ▶ Simple need only starting address
- ▶ Free-space management system no waste of space
- No random access

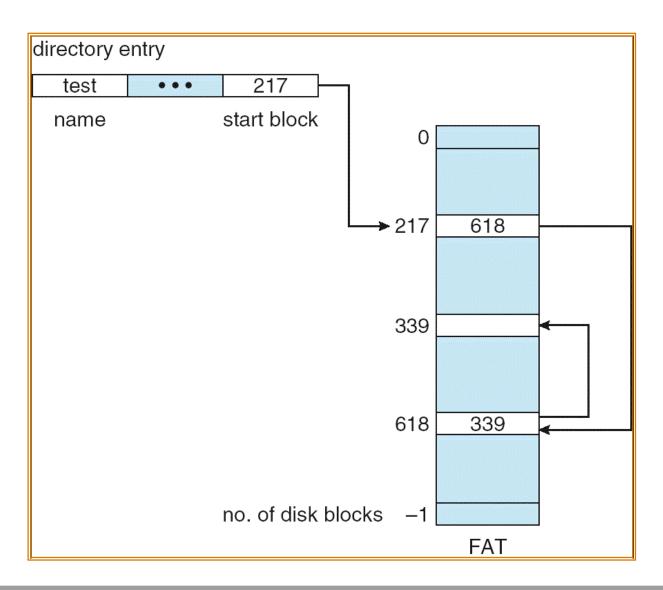


Linked Allocation





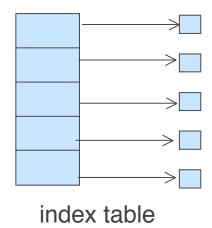
File-Allocation Table (DOS FAT)





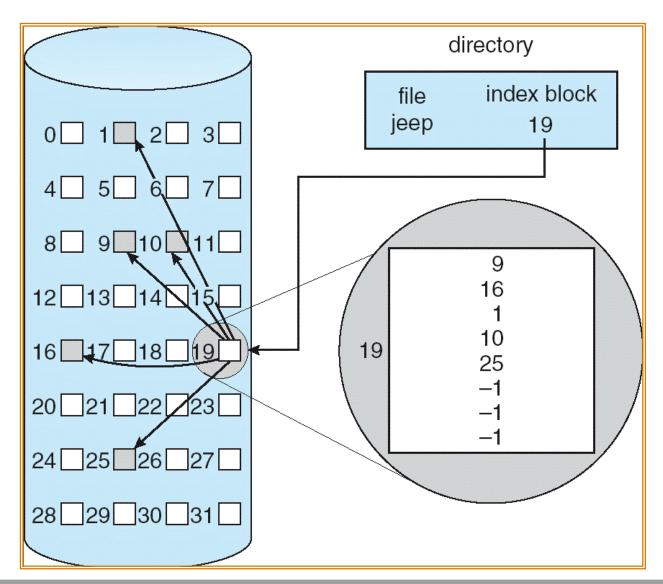
Indexed Allocation

- ▶ Brings all pointers together into the *index block*.
- ▶ Logical view.





Example of Indexed Allocation



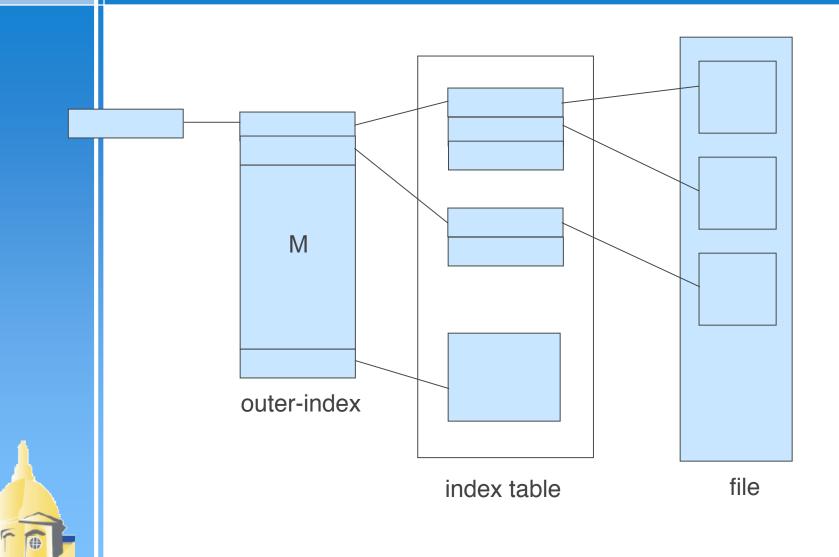


Indexed Allocation (Cont.)

- Need index table
- Random access
- Dynamic access without external fragmentation, but have overhead of index block.
- Mapping from logical to physical in a file of maximum size of 256K words and block size of 512 words. We need only 1 block for index table.



Indexed Allocation – Mapping (Cont.)



Combined Scheme: UNIX (4K bytes per block)

