File System

Fan Wu

Department of Computer Science and Engineering

Shanghai Jiao Tong University

Spring 2020

File Concept

- A file is a named collection of related information that is recorded on secondary storage.

- Types:
  - Text
  - Source/object programs
  - Executable programs
  - Database records
  - Graphic images
  - Multimedia

File Types – Name, Extension

file type	usual extension	function
executable	exe, com, bin or none	ready-to-run machine-language program
object	obj, o	compiled, machine language, not linked
source code	c, cc, java, pas, asm, a	source code in various languages
batch	bat, sh	commands to the command interpreter
text	txt, doc	textual data, documents
word processor	wp, tex, rtf, doc	various word-processor formats
library	lib, a, so, dll	libraries of routines for programmers
print or view	ps, pdf, jpg	ASCII or binary file in a format for printing or viewing
archive	arc, zip, tar	related files grouped into one file, sometimes compressed, for archiving or storage
multimedia	mpeg, mov, rm, mp3, avi	binary file containing audio or A/V information

File Attributes

- **Name** – information kept in human-readable form
- **Identifier** – unique tag (number) identifies file within file system
- **Type** – needed for systems that support different types
- **Location** – pointer to file location on device
- **Size** – current file size
- **Protection** – controls who can do reading, writing, executing
- **Time, date, and user identification** – data for protection, security, and usage monitoring

- Information about files are kept in the directory structure, which is maintained on the disk

File Operations

- **Create**
- **Write**
- **Read**
- **Reposition within file**
- **Delete**
- **Truncate**

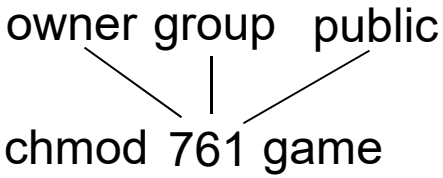
- The other operations can be implemented by the primitive ones.

Access Lists and Groups

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

			RWX
a) **owner access**	7	⇒	1 1 1
b) **group access**	6	⇒	RWX 1 1 0
c) **public access**	1	⇒	RWX 0 0 1

- Ask administrator 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.



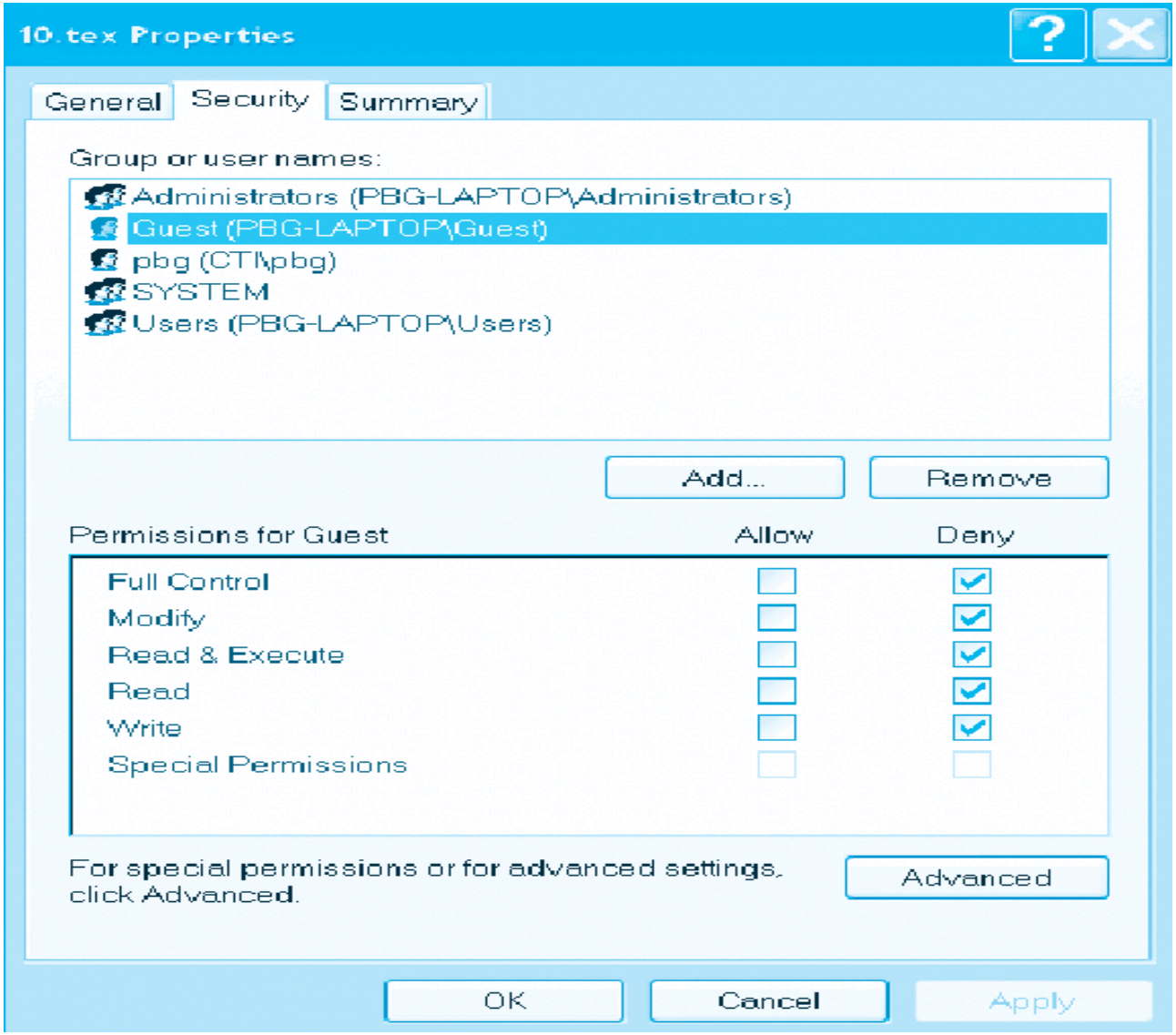
Attach a group to a file

chgrp G game

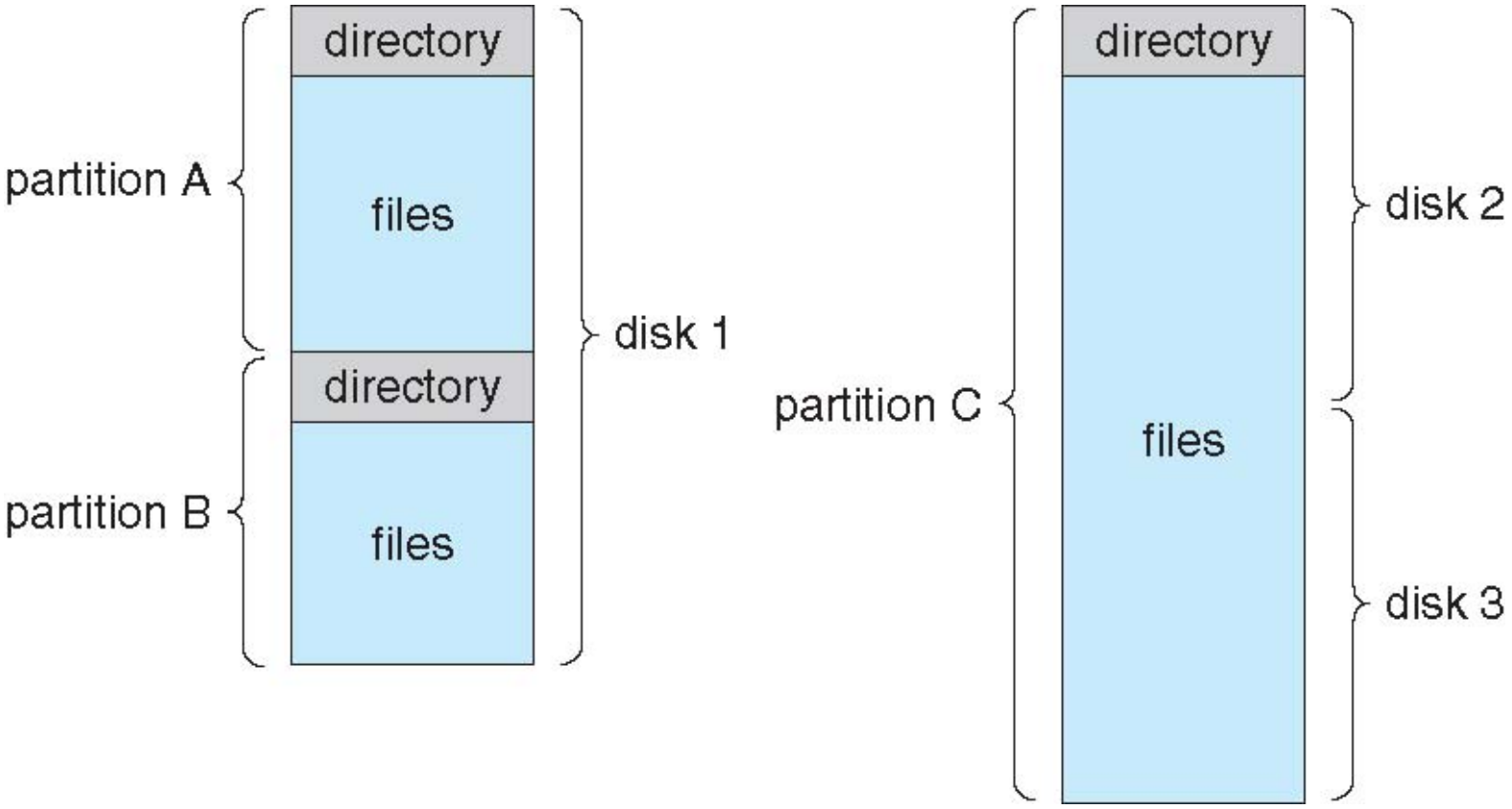
A Sample UNIX Directory Listing

-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-r--r--	1 pbg	staff	9423	Feb 24 2003	program.c
-rwxr-xr-x	1 pbg	staff	20471	Feb 24 2003	program
drwx--x--x	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/

Windows XP Access-Control List Management

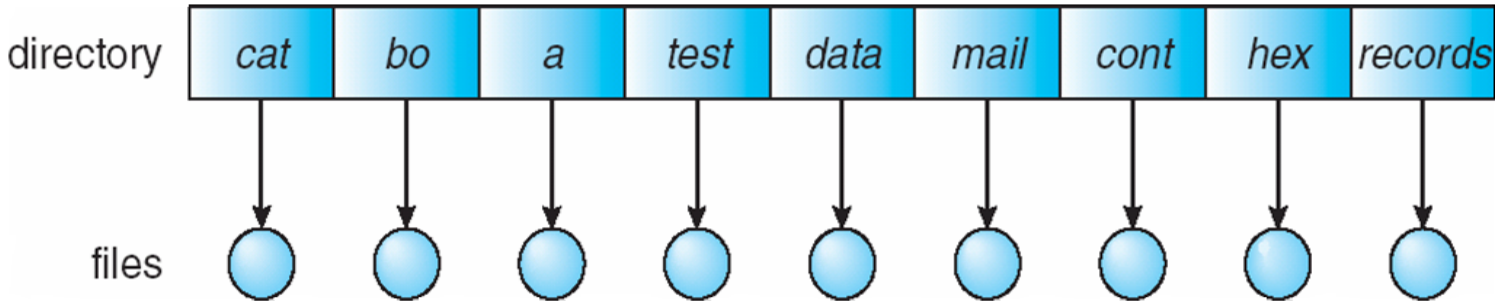


A Typical File-system Organization



Single-Level Directory

- A single directory for all users



Efficiency problem

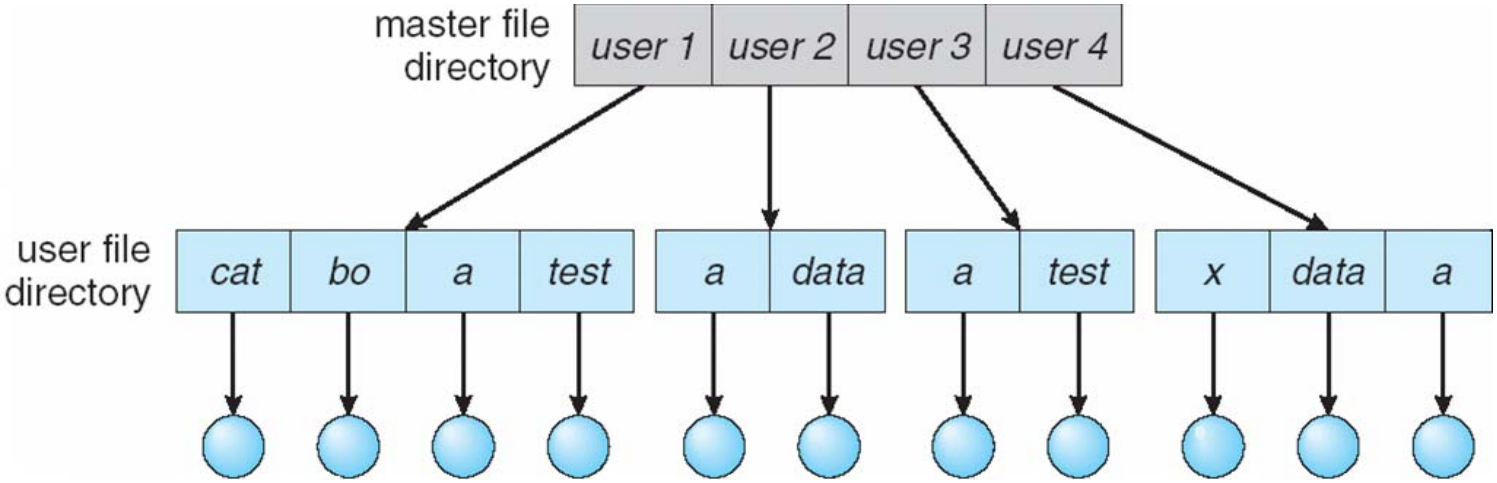
Naming problem

Protection of users' private files

Grouping problem

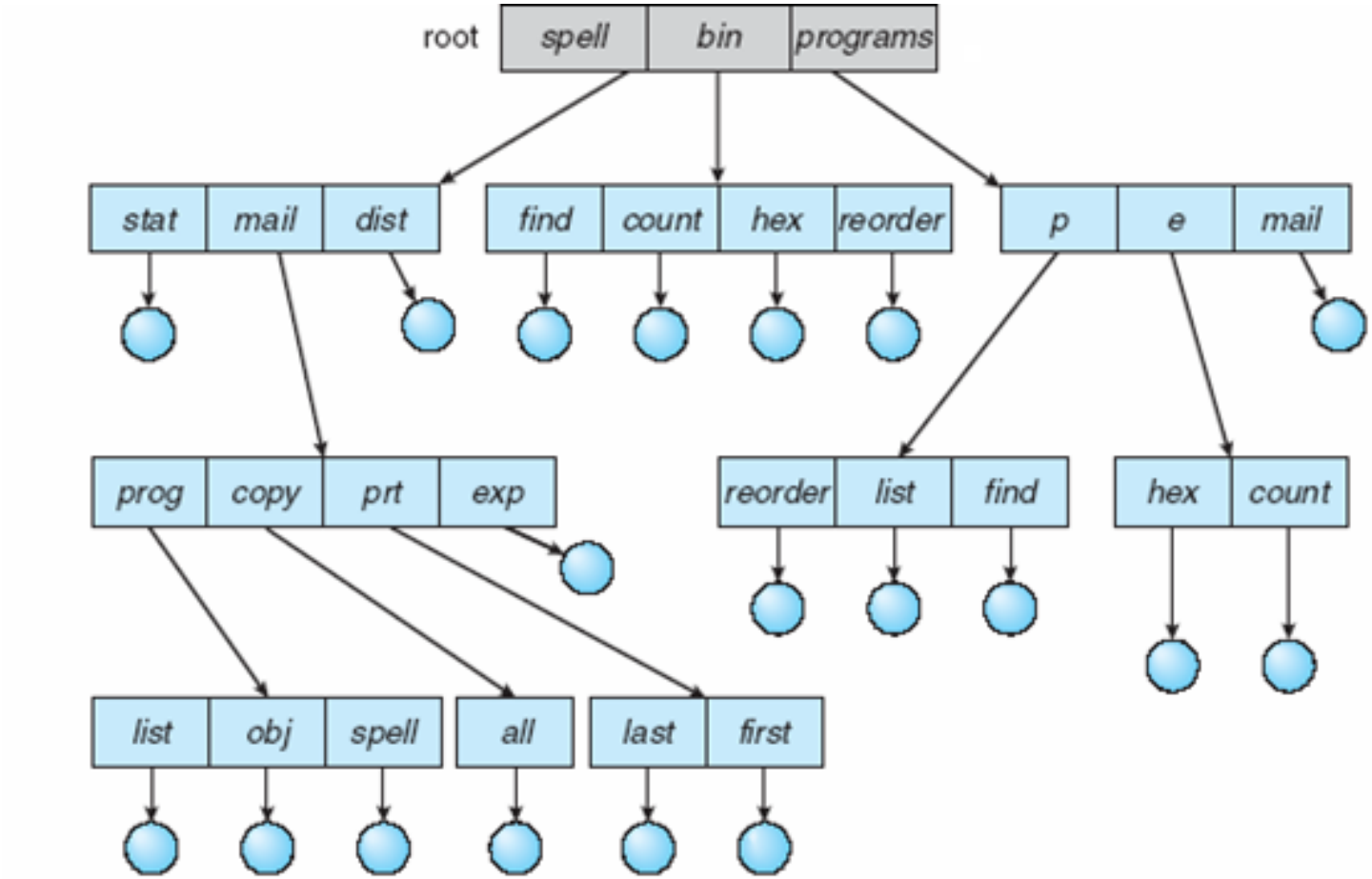
Two-Level Directory

- Separate directory for each user



- Can have the same file name for different user
- A little bit more efficient searching
- Path name
- No grouping capability

Tree-Structured Directories



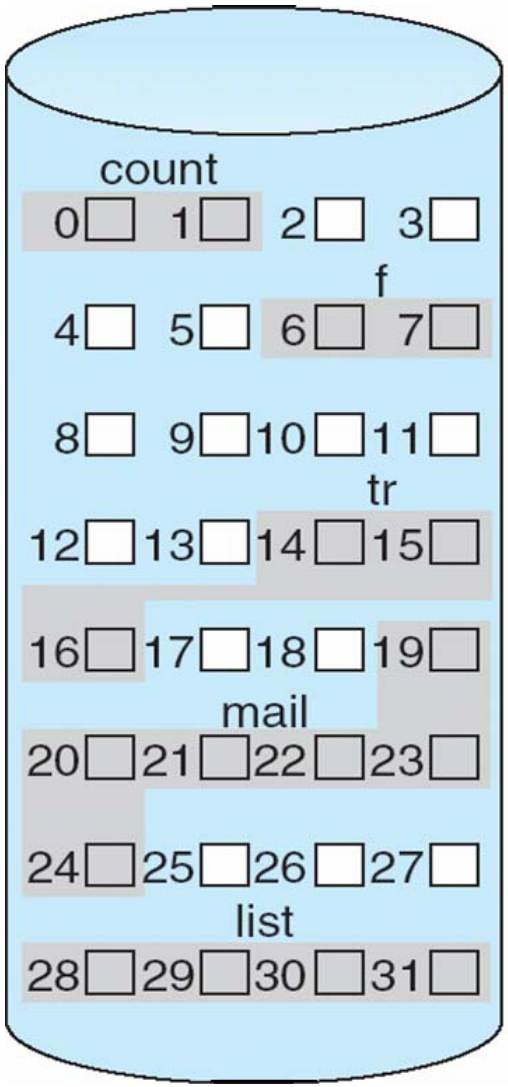
File Allocation Methods

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

Contiguous Allocation

- **Contiguous allocation** – each file occupies set of contiguous blocks
  - Simple – only starting location (block #) and length (number of blocks) are required
  - Best performance in most cases
  - Problems include finding space for file, knowing file size, external fragmentation, need for **compaction off-line** (**downtime**) or **on-line**

Contiguous Allocation of Disk Space

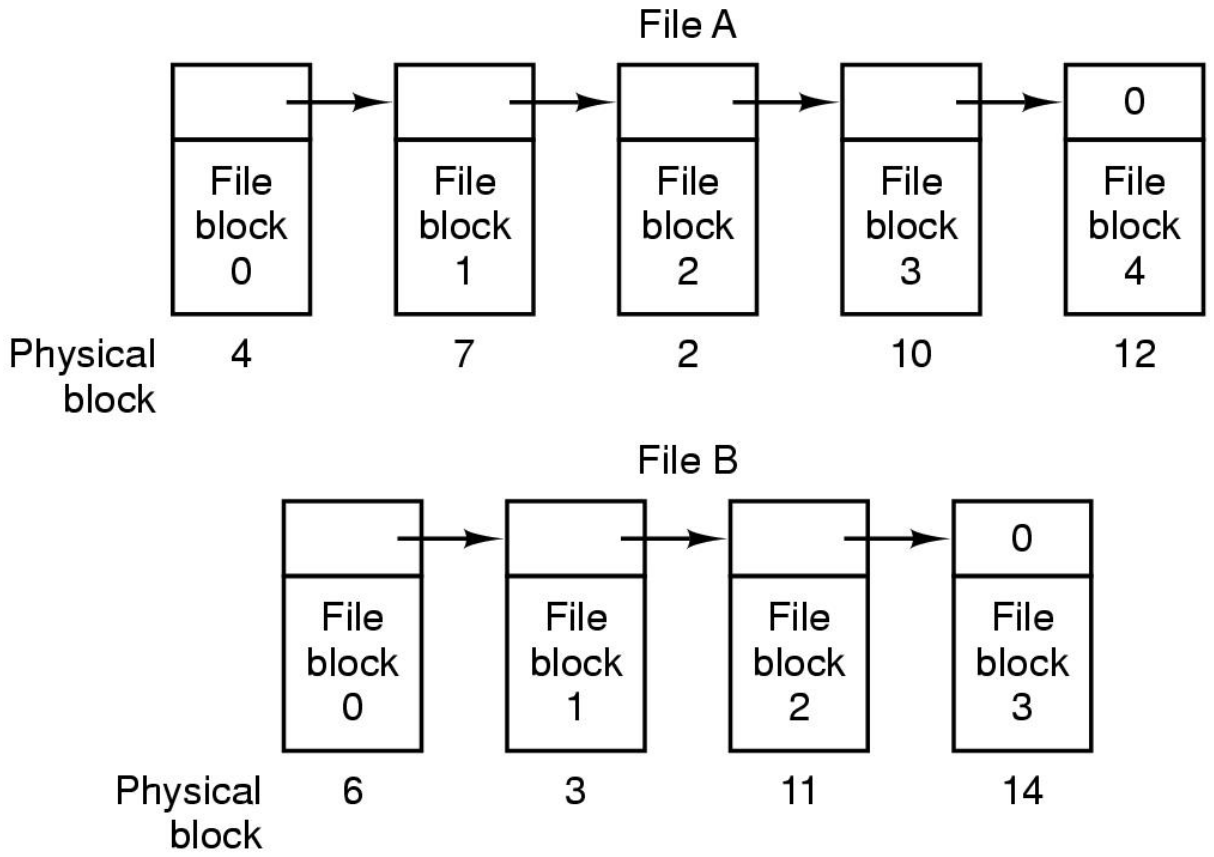


directory

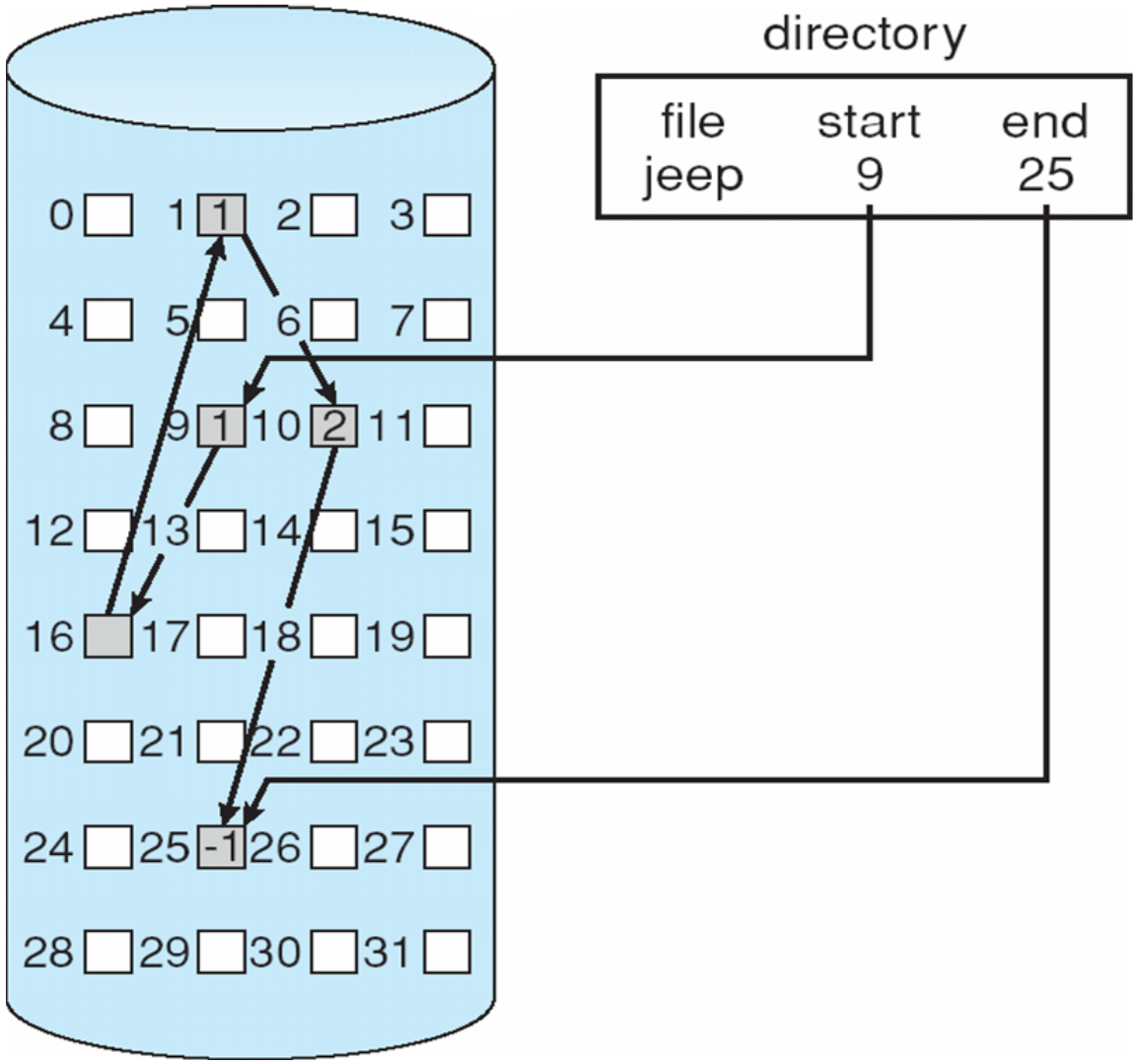
file	start	length
count	0	2
tr	14	3
mail	19	6
list	28	4
f	6	2

Linked Allocation

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



Linked Allocation (Cont.)

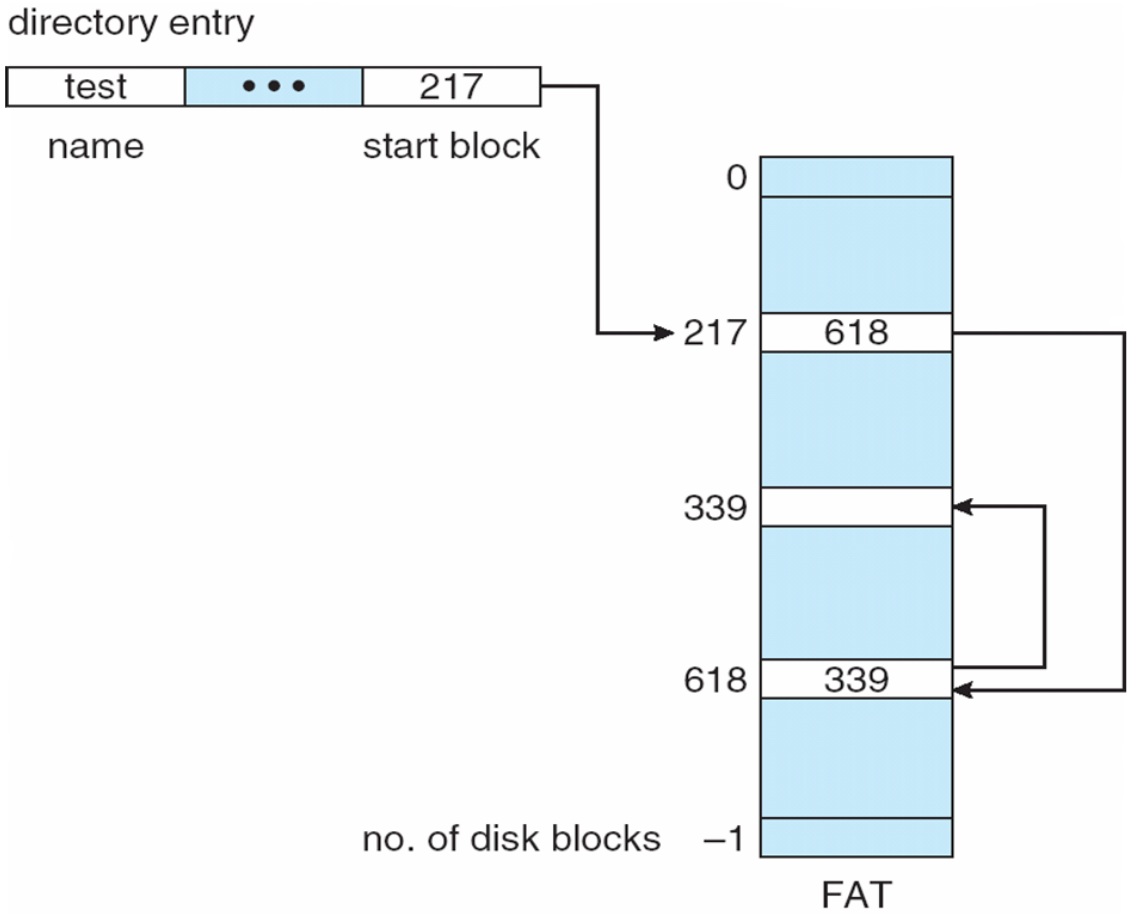


Linked Allocation (Cont.)

- **Linked allocation** – each file is a linked list of blocks
  - Each block contains pointer to next block
  - File ends at nil pointer
  - No external fragmentation
  - Free space management system called when new block needed
  - Improve efficiency by clustering blocks into groups but increases internal fragmentation
  - Reliability can be a problem
  - Locating a block can take many I/Os and disk seeks

File-Allocation Table (FAT)

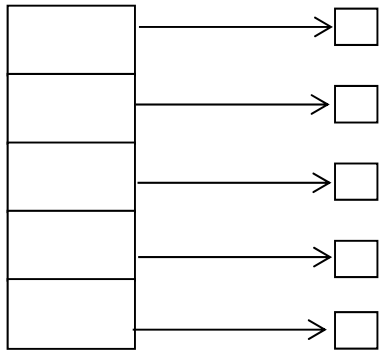
- FAT (File Allocation Table) variation
  - Beginning of volume has a table, indexed by block number
  - Much like a linked list, but faster on disk and cacheable
  - New block allocation simple
- DOS / Windows



Indexed Allocation

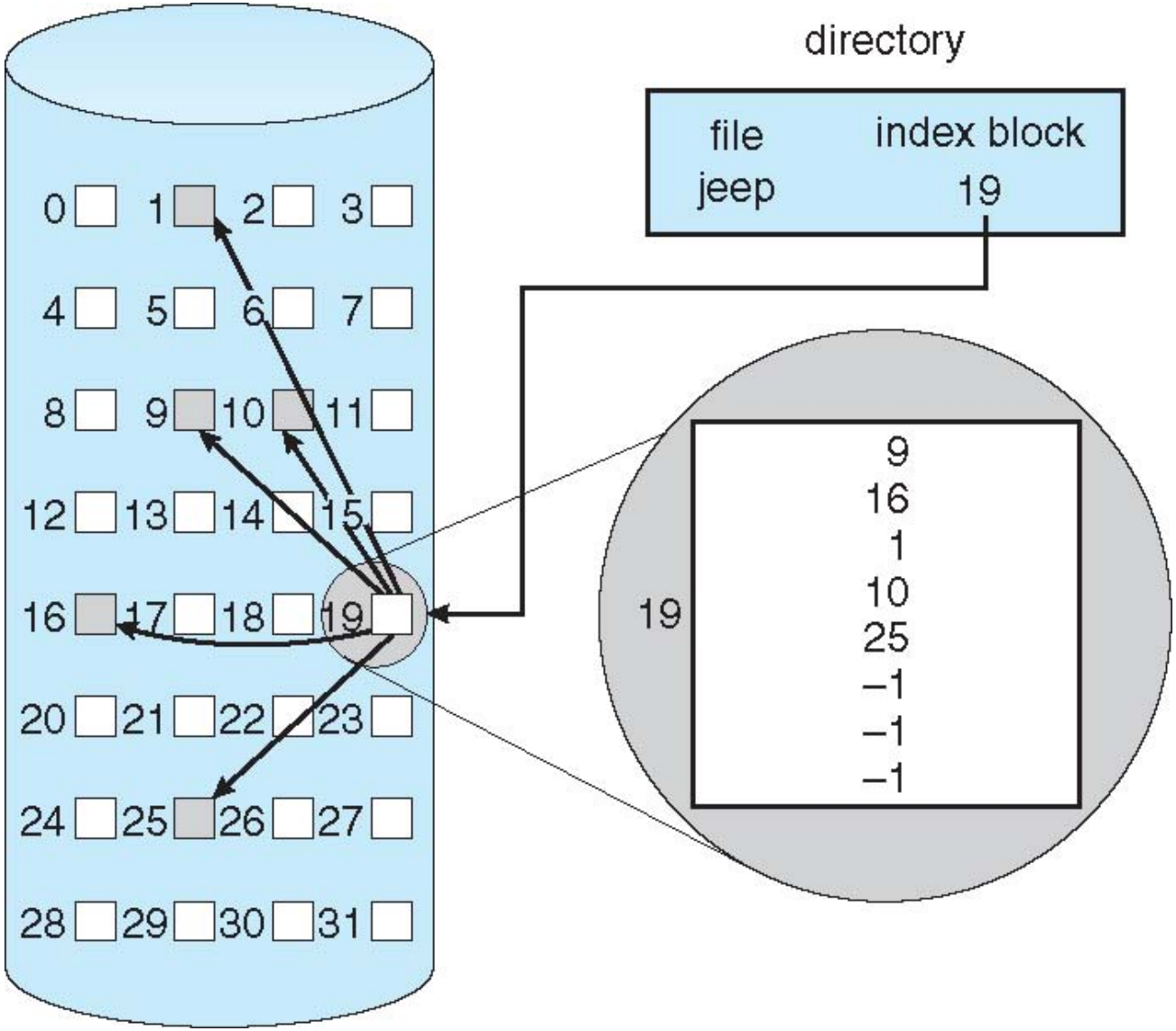
- **Indexed allocation**
  - Each file has its own **index block**(s) of pointers to its data blocks

- Logical view



index table

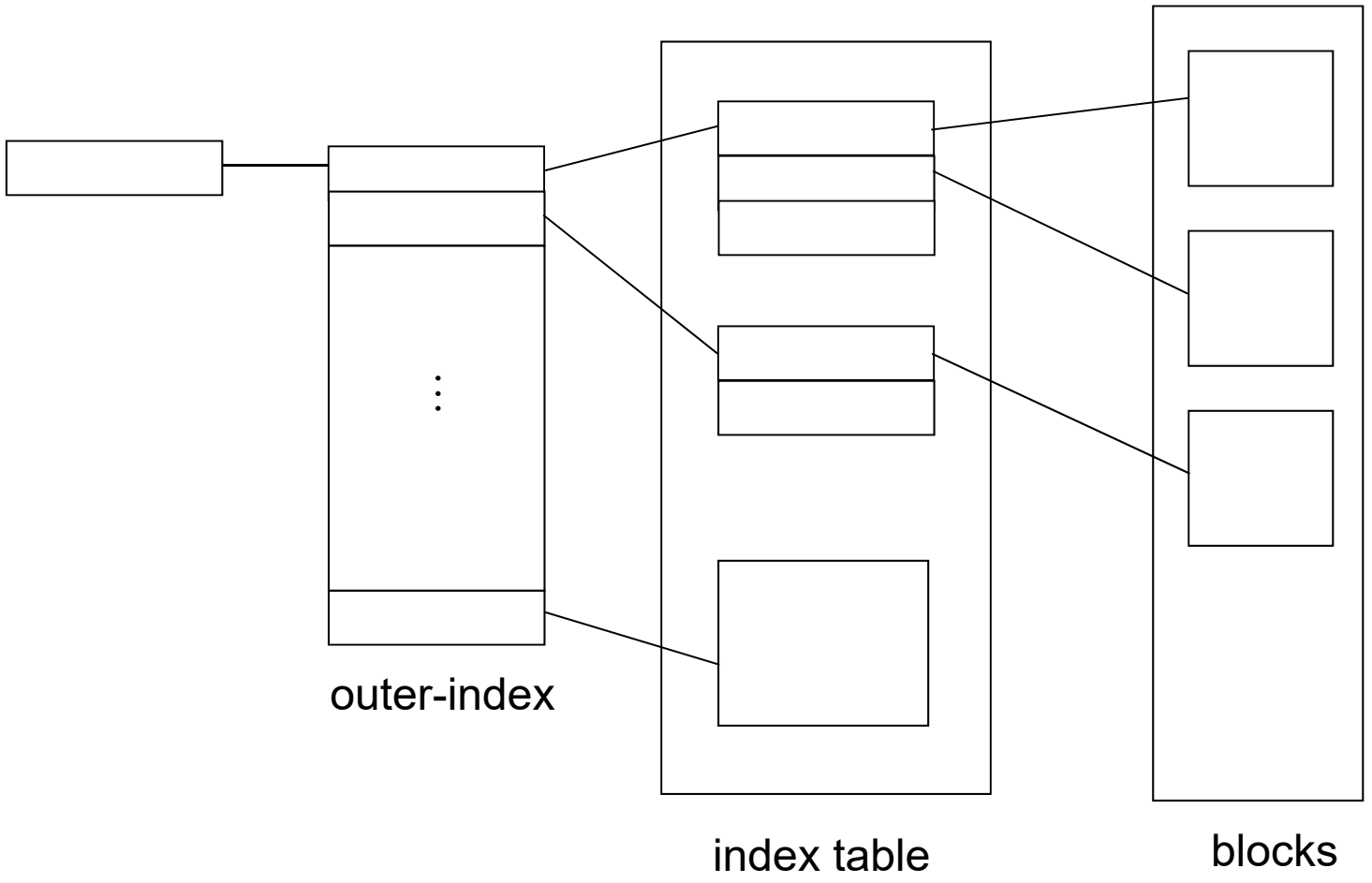
Example of Indexed Allocation



Indexed Allocation (Cont.)

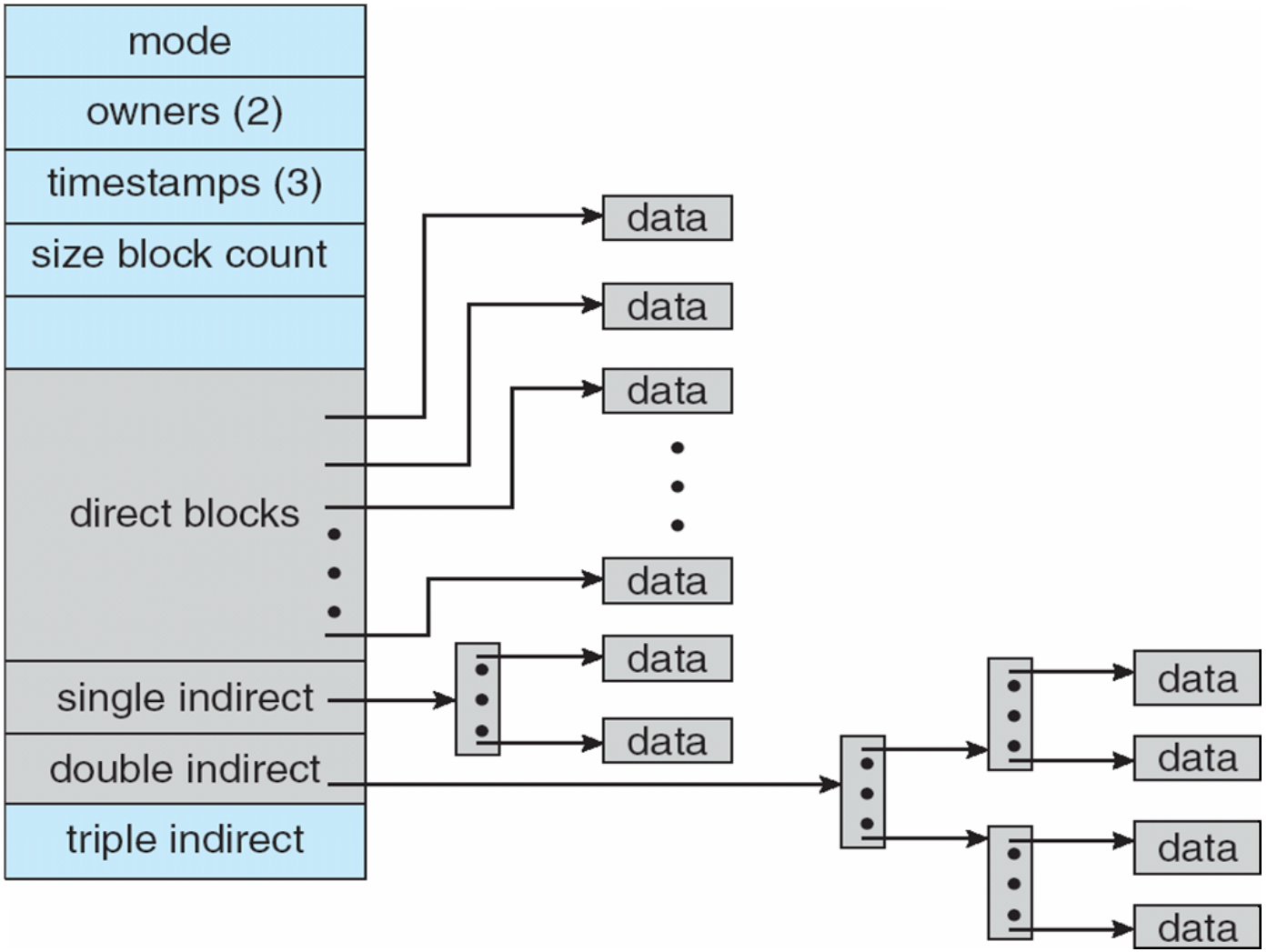
- Random access
- Without external fragmentation
- Need index table

Indexed Allocation – Mapping (Cont.)



Multilevel Index

Indexed Allocation – Mapping (Cont.)



Combined Scheme with **UNIX I-node**

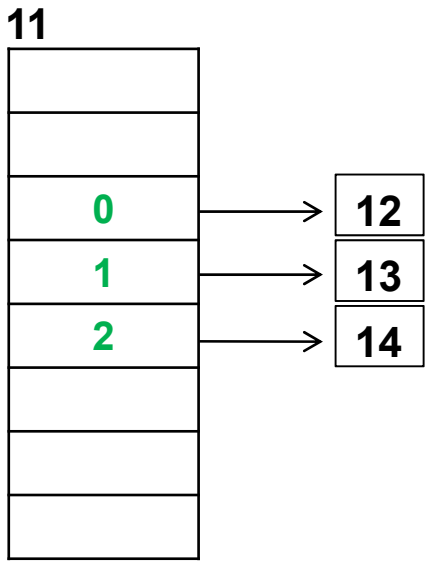
I-node Example

- Suppose a file system is constructed using blocks of 32 bytes. A pointer needs 4 bytes. The I-node structure is as follows (word, value):

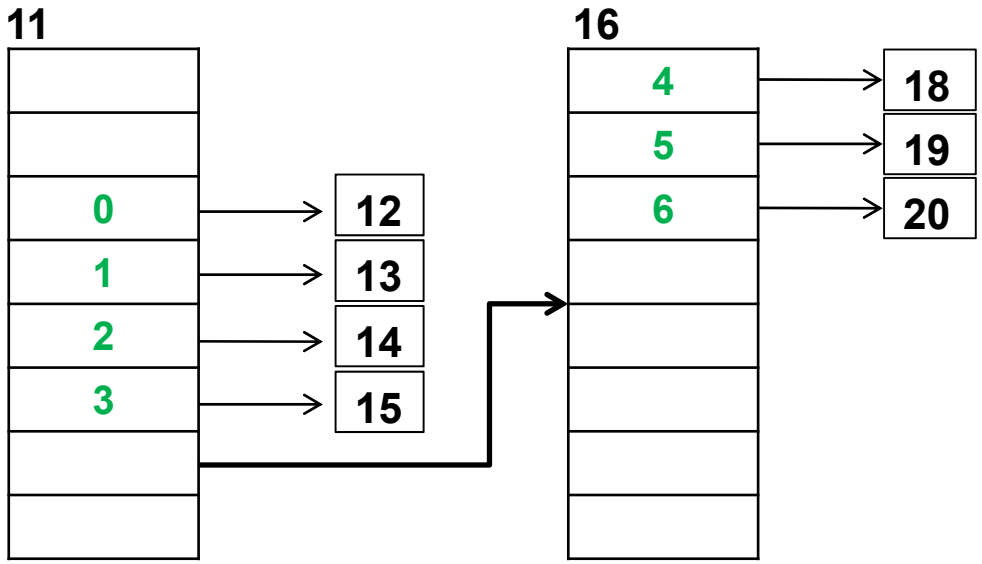
0	Permission word
1	File Size
2	Direct block
3	Direct block
4	Direct block
5	Direct block
6	Single-indirect
7	Double-indirect

- Assume that free blocks are allocated in logical order starting with block 11. Also it has been determined that blocks 17 and 32 are bad and cannot be allocated.
- Draw a block diagram showing the structure of the I-node and the blocks that are allocated for
  - Original file size of 3 blocks
  - Adding 4 blocks
  - Adding 17 blocks

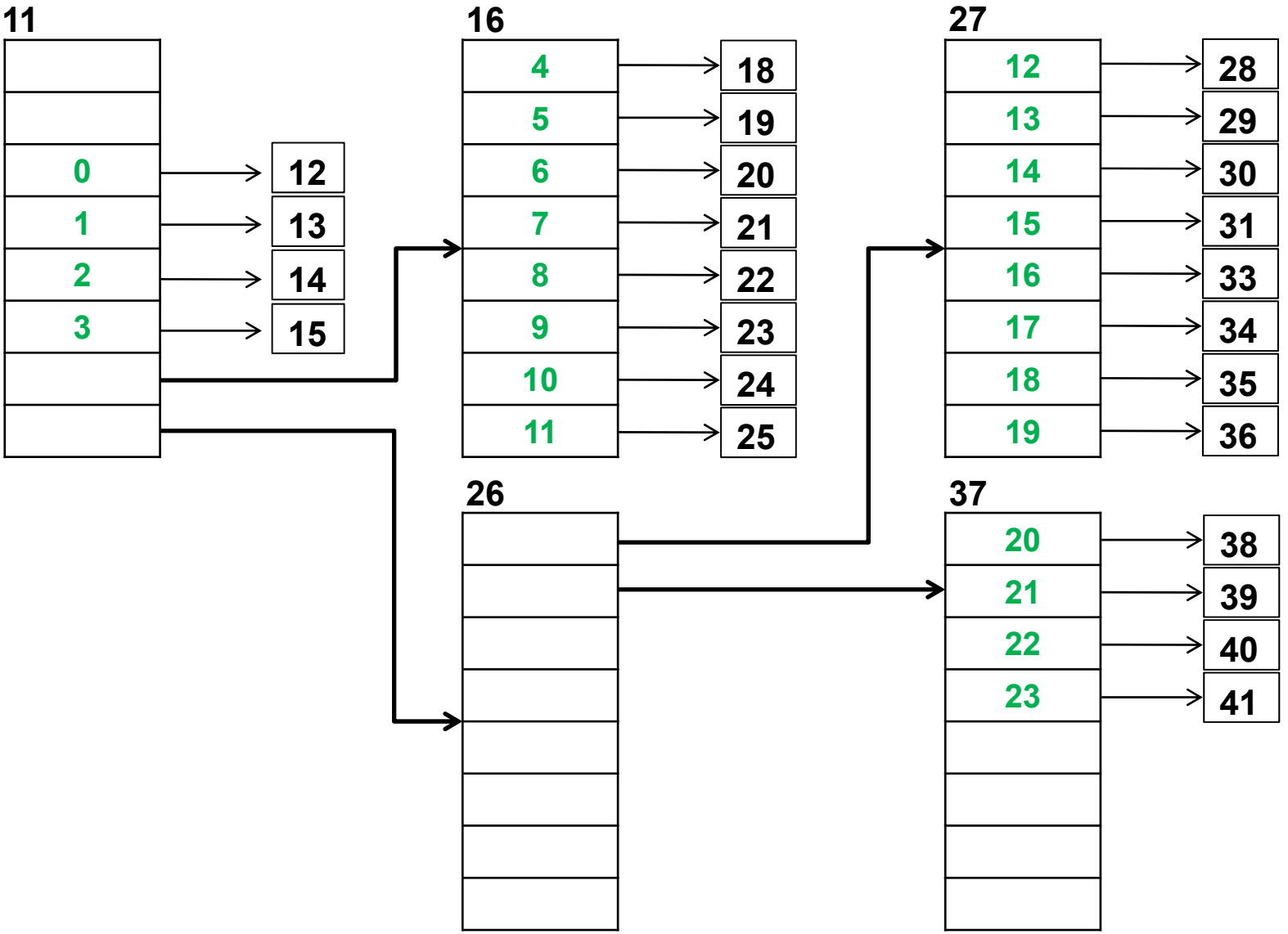
I-node: Original file size of 3 block



I-node: Adding 4 blocks



I-node: Adding 17 blocks



Pop Quiz

- Suppose a file system is constructed using blocks of 32 bytes. A pointer needs 4 bytes. The I-node structure is as follows (word, value):

0	Permission word
1	File Size
2	Direct block
3	Direct block
4	Direct block
5	Single-indirect
6	Double-indirect
7	Triple-indirect

- Assume that free blocks are allocated in logical order starting with block 100. Also it has been determined that blocks 107, 108, 109, and 112 are bad and cannot be allocated.
- Draw a block diagram showing the structure of the I-node and the blocks that are allocated for
  - Original file size of 3 blocks
  - Adding 7 blocks
  - Adding 24 blocks
  - Adding 64 blocks

Homework

- Reading
  - Chapter 10 and 11