

Operating Systems

Prepared by Dr. Alaa Eldeen Sayed Ahmed

IT263 / CYB510
File Systems

Lecture 9



- Objectives:
 - File organisation and Access
 - File Directories
 - File Sharing
 - Record Blocking
 - Secondary Storage Management

- **Files** are the central element to most application
- A File is a named collection of related information that is recorded on secondary storage such as magnetic disks, magnetic tapes and optical disks.
- **File systems** are a crucial part of any operating system, providing a structured way to store, organize, and manage data on storage devices such as hard drives, SSDs, and USB drives.

- A File system acts as a bridge between the operating system and the physical storage hardware, allowing users and applications to create, read, update, and delete files in an organized and efficient manner
- Desirable properties of files:
 - Long-term existence
 - Sharable between processes
 - Structure

- File systems also provide functions which can be performed on files, typically:
 - **Create:** A new file is defined and positioned within the structure of files.
 - **Delete:** A file is removed from the file structure and destroyed.
 - **Open:** An existing file is declared to be “opened” by a process, allowing the process to perform functions on the file.

- **Close:** The file is closed with respect to a process, so that the process no longer may perform functions on the file, until the process opens the file again.
- **Read:** A process reads all or a portion of the data in a file.
- **Write:** A process updates a file, either by adding new data that expands the size of the file or by changing the values of existing data items in the file.

Criteria for File Organization

- Important criteria include:
 - Short access time
 - Ease of update
 - Economy of storage
 - Simple maintenance
 - Reliability

File Organizations(Directories)

- A directory structure is the way an operating system arranges files that are accessible to the user.
- Contains information about files
 - Attributes
 - Location
 - Ownership
- Directory itself is a file owned by the operating system
- Provides mapping between file names and the files themselves

- File Name
 - Name as chosen by creator (user or program).
 - Must be unique within a specific directory.
- File type
 - Doc
 - Exe
 - Jpeg
 - Pdf

Directory Elements: Address Information

- File Organisation
 - For systems that support different organizations
 - **FAT (File Allocation Table):** An older file system used by older versions of Windows and other operating systems.
 - **NTFS (New Technology File System):** A modern file system used by Windows. It supports features such as file and folder permissions, compression, and encryption.
 - **ext (Extended File System):** A file system commonly used on [Linux](#) and [Unix](#)-based operating systems.
 - **HFS (Hierarchical File System):** A file system used by macOS.
 - **APFS (Apple File System):** A new file system introduced by Apple for their Macs and iOS devices.

Directory Elements: Address Information

- Volume
 - Indicates device on which file is stored
- Starting Address
- Size Used
 - Current size of the file in bytes, words, or blocks
- Size Allocated
 - The maximum size of the file

Directory Elements: Usage Information

- Date Created
- Identity of Creator
- Date Last Read Access
- Identity of Last Reader
- Date Last Modified
- Identity of Last Modifier
- Date of Last Backup
- Current Usage
 - Current activity, locks, etc

Operations performed on a Directory

- A directory system should support a number of operations including:
- **Search:** When a user or application references a file, the directory must be searched to find the entry corresponding to that file.
- **Create file:** When a new file is created, an entry must be added to the directory.

Operations performed on a Directory

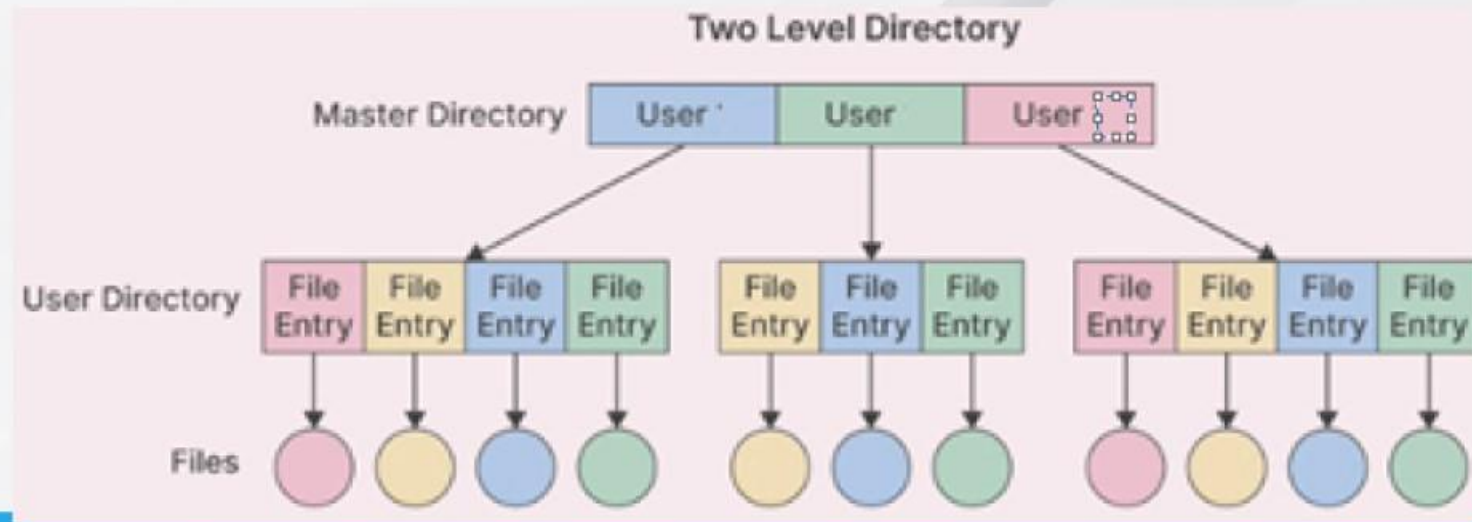
- **Delete file:** When a file is deleted, an entry must be removed from the directory.
- **List directory:** All or a portion of the directory may be requested. Generally, this request is made by a user and results in a listing of all files owned by that user, plus some of the attributes of each file
- **Update directory:** Because some file attributes are stored in the directory, a change in one of these attributes requires a change in the corresponding directory entry.

Types of directory structures

- There are four types of directories:
 - **Two-Level Directory Structure**
 - **Tree-Structured Directory**
 - Acyclic-Graph Directory Structure
 - General-Graph Directory Structure

Two-Level Scheme for a Directory

- One directory for each user and a master directory
 - Master directory contains entry for each user
 - Provides address and access control information
- Each user directory is a simple list of files for that user (separate directory for each user contains user's files)

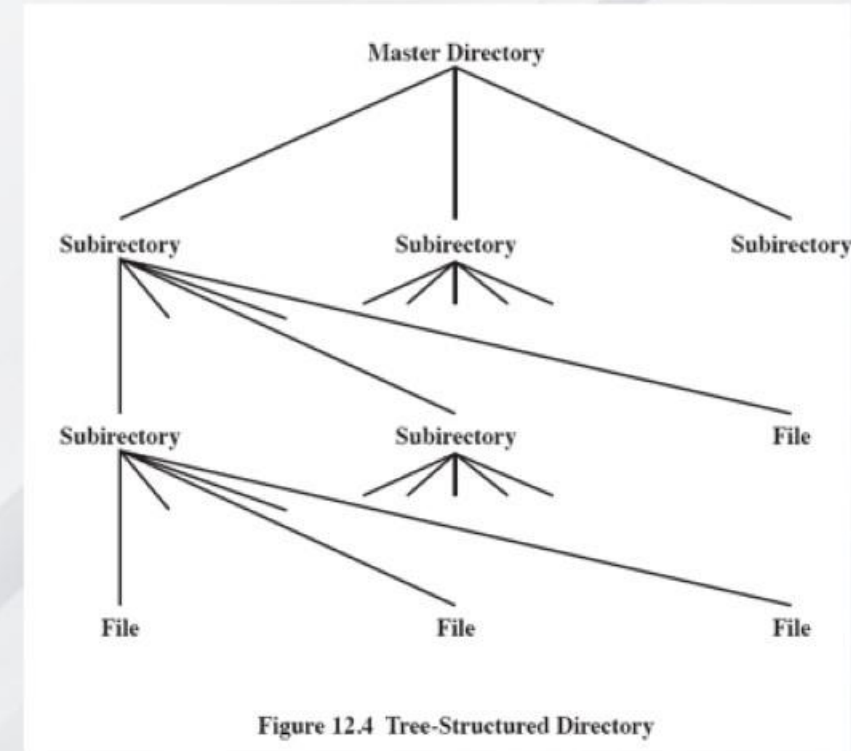


Two-Level Scheme for a Directory

- Enables the usage of the same file name across several user directories.
- Prevents a user from accessing another user's directory without their authorization.
- Different users can have the same file name (/user2/me and /user3/me).
- Searching is more efficient.
- One user cannot share a file with another user in a two-level directory without permission.
- The two-level directory also has the problem of not being scalable.

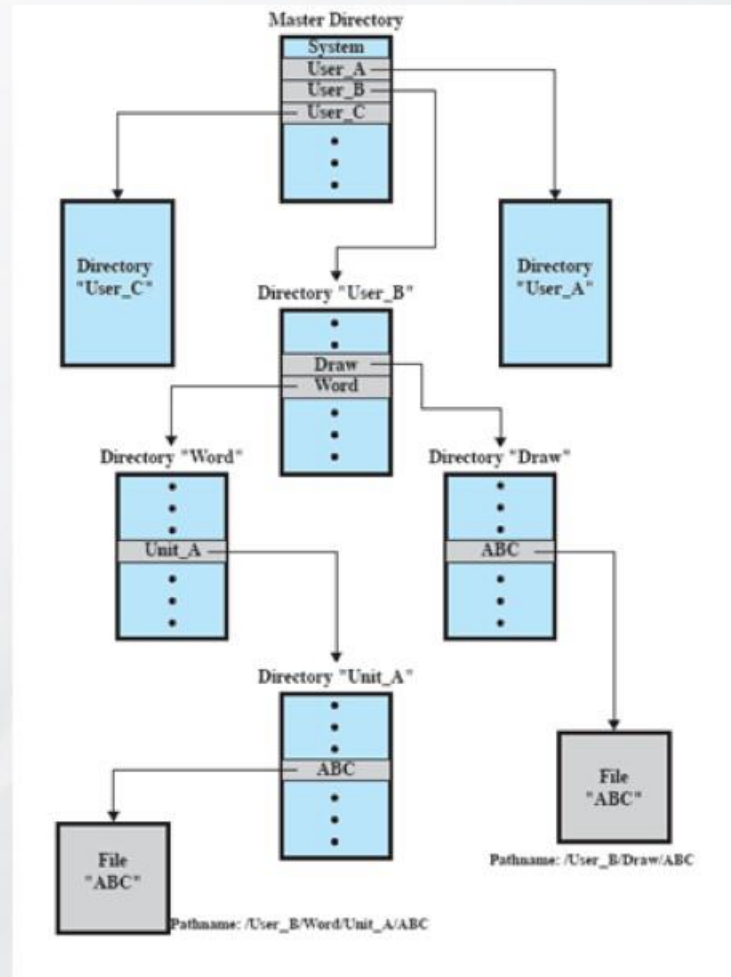
Hierarchical, or Tree-Structured Directory

- Master directory with user directories underneath it
- Each user directory may have subdirectories and files as entries
- The directory, which is organized in a tree form, is extremely scalable.
- Searching in the tree-structure directory is efficient



- Users need to be able to refer to a file by name
 - Files need to be named uniquely, but users may not be aware of all filenames on a system
- The tree structure allows users to find a file by following the directory path
 - Duplicate filenames are possible if they have different pathnames

Example of Tree-Structured Directory



- File sharing refers to the process of sharing or distributing electronic files between two or more users or computers.
- Two issues
 - Access rights
 - Management of simultaneous access

- The file system should provide a number of options so that the way in which a particular file is accessed can be controlled.
- Typically, users or groups of users are granted certain access rights to a file.
- A wide variety of access rights have been used by various systems
 - often as a hierarchy where one right implies previous
- **None**
 - User may not even know of the files existence
- **Knowledge**
 - User can only determine that the file exists and who its owner is

- **Execution**
 - The user can load and execute a program but cannot copy it
- **Reading**
 - The user can read the file for any purpose, including copying and execution
- **Appending**
 - The user can add data to the file but cannot modify or delete any of the file's contents

- **Updating**
 - The user can modify, delete, and add to the file's data.
- **Changing protection**
 - User can change access rights granted to other users
- **Deletion**
 - User can delete the file

- **Owner**
 - Usually the files creator, usually has full rights
- **Specific Users**
 - Rights may be explicitly granted to specific users
- **User Groups**
 - A set of users identified as a group
- **All**
 - everyone

Simultaneous Access

- When access is granted to append or update a file to more than one user, the operating system or file management system must enforce discipline.
- User may lock entire file when it is to be updated
- User may lock the individual records during the update
- Mutual exclusion and deadlock are issues for shared access

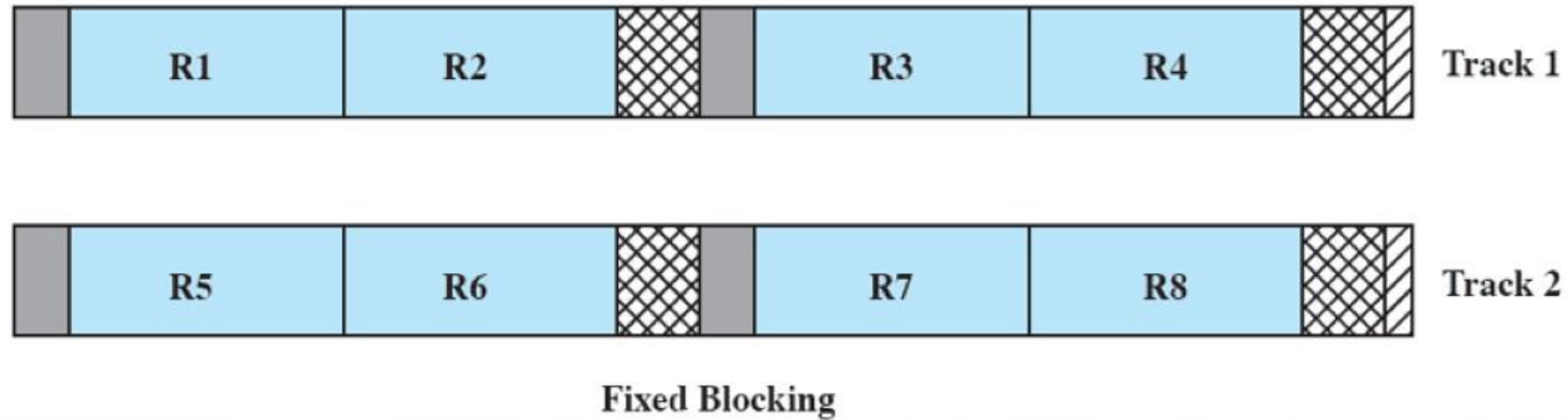
Blocks and Records

- Records are the logical unit of access of a structured file
 - Whereas blocks are the unit of I/O with secondary storage. For I/O to be performed, records must be organized as blocks.
- Three approaches are common
 - Fixed length blocking
 - Variable length spanned blocking
 - Variable-length unspanned blocking

Fixed Blockings

- Fixed-length records are used, and an integral number of records are stored in a block.
- Unused space at the end of a block is ***internal fragmentation***

Fixed Blockings



Data



Gaps due to hardware design



Waste due to block fit to track size



Waste due to record fit to block size

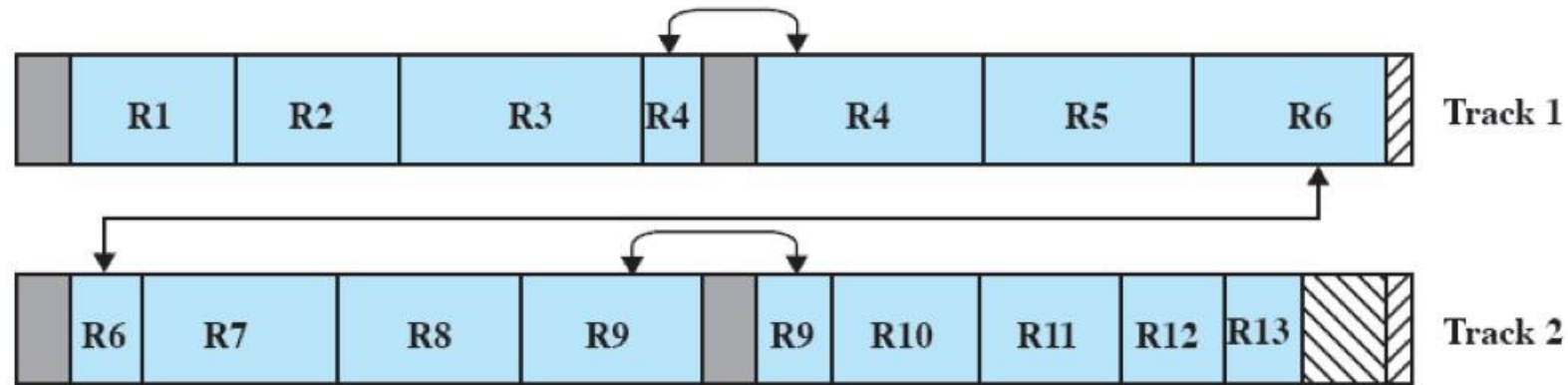


Waste due to block size constraint
from fixed record size

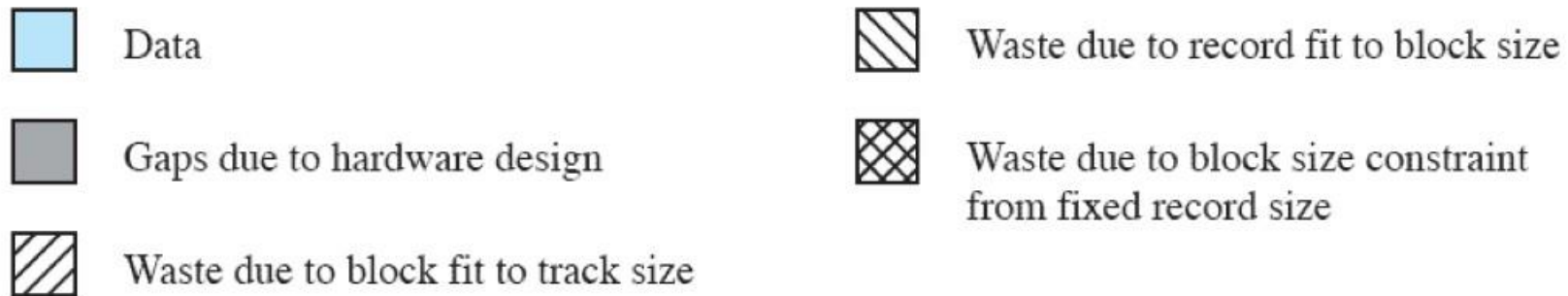
Variable length Spanned Blocking

- Variable-length records are used and are packed into blocks with no unused space.
- Some records may span multiple blocks
 - Continuation is indicated by a pointer to the successor block

Variable Blocking: Spanned



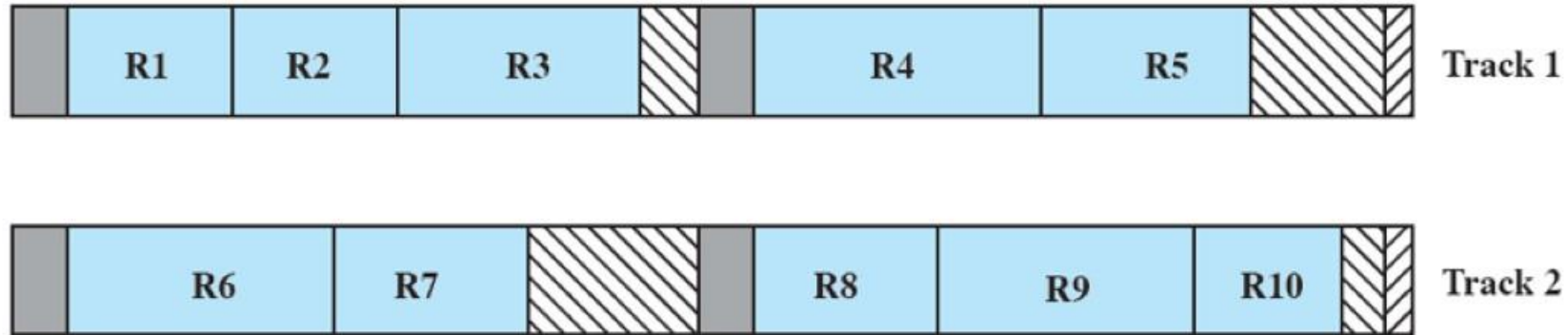
Variable Blocking: Spanned



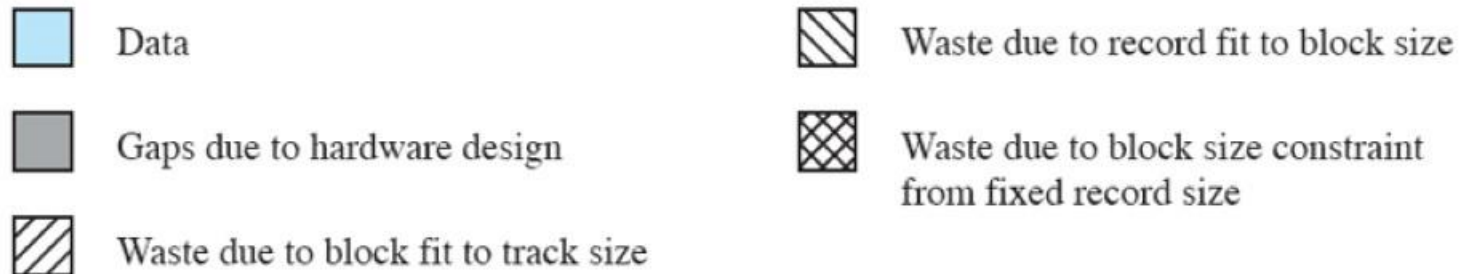
Variable-length unspanned blocking

- Uses variable length records without spanning
- Wasted space in most blocks because of the inability to use the remainder of a block if the next record is larger than the remaining unused space.

Variable Blocking: Unspanned



Variable Blocking: Unspanned



Secondary Storage Management

- On secondary storage, a file consists of a collection of blocks.
 - The operating system or file management system is responsible for allocating blocks to files.
- This raises two management issues.
 - First, space on secondary storage must be allocated to files,
 - second, it is necessary to keep track of the space available for allocation.

Preallocation vs Dynamic Allocation

- A **preallocation** policy requires that the maximum size of a file be declared at the time of the file creation request.
- Difficult to reliably estimate the maximum potential size of the file
- Tend to overestimated file size so as not to run out of space
- In **dynamic allocation**, which the space is allocated to a file in portions as needed

- Two extremes:
 - Portion large enough to hold entire file is allocated
 - Allocate space one block at a time
- In choosing a portion size, there is a **tradeoff** between efficiency from the point of view of a single file versus overall system efficiency.

File Allocation Method

- Three methods are in common use:
 - contiguous,
 - chained, and
 - indexed.

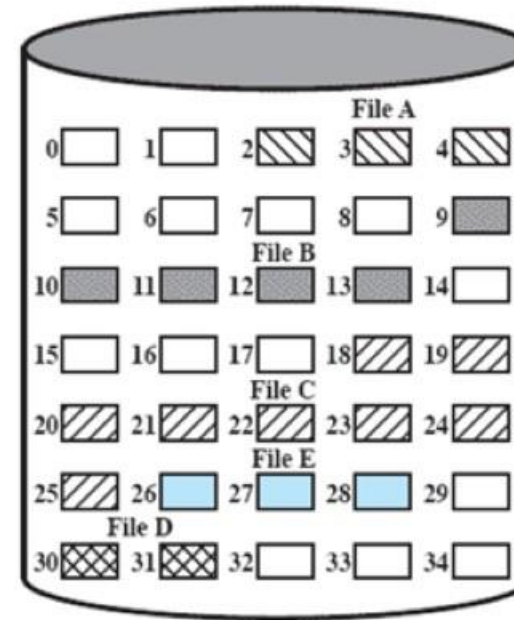
Contiguous Allocation

- A single contiguous set of blocks is allocated to a file at the time of file creation
- The file allocation table needs just a single entry for each file, showing the starting block and the length of the file.
- Contiguous allocation is the best from the point of view of the individual sequential file.
 - Multiple blocks can be read in at a time to improve I/O performance for sequential processing.
 - It is also easy to retrieve a single block.

External fragmentation

External fragmentation will occur, making it difficult to find contiguous blocks of space of sufficient length.

From time to time, it will be necessary to perform a compaction algorithm to free up additional space on the disk.



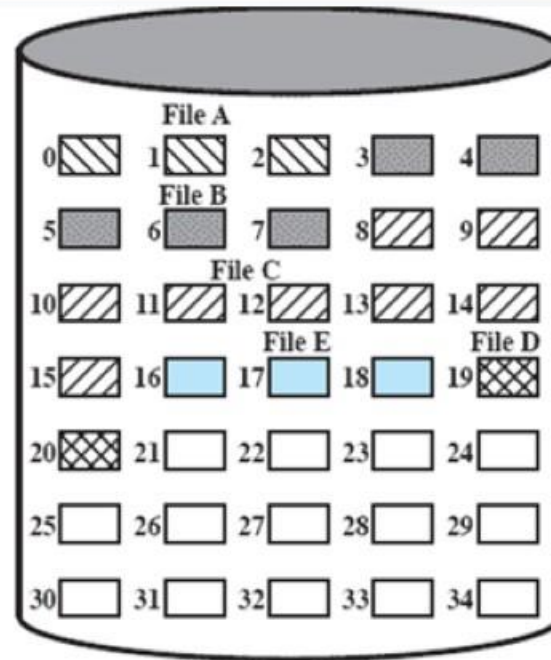
File Allocation Table

File Name	Start Block	Length
File A	2	3
File B	9	5
File C	18	8
File D	30	2
File E	26	3

Contiguous File Allocation

lecture 09

External fragmentation



File Allocation Table

File Name	Start Block	Length
File A	0	3
File B	3	5
File C	8	8
File D	19	2
File E	16	3

Contiguous File Allocation (After Compaction)

Chained Allocation

- Allocation on basis of individual block
- Each block contains a pointer to the next block in the chain
- Only single entry in the file allocation table
 - Starting block and length of file
- There is no external fragmentation to worry about because only one block at a time is needed.
- Best suited for sequential files

Chained Allocation

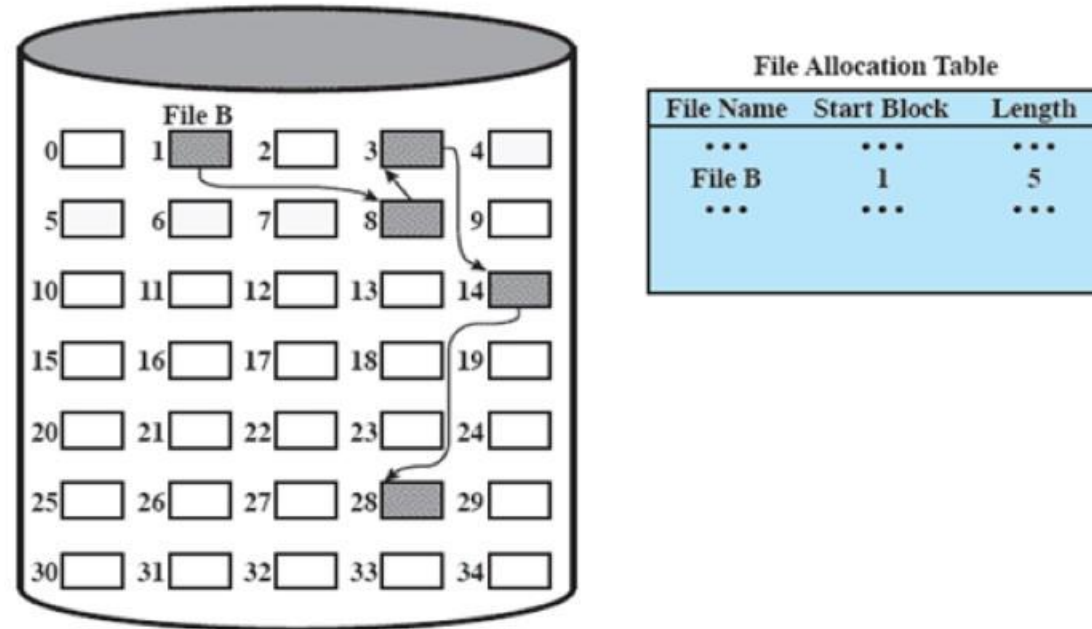
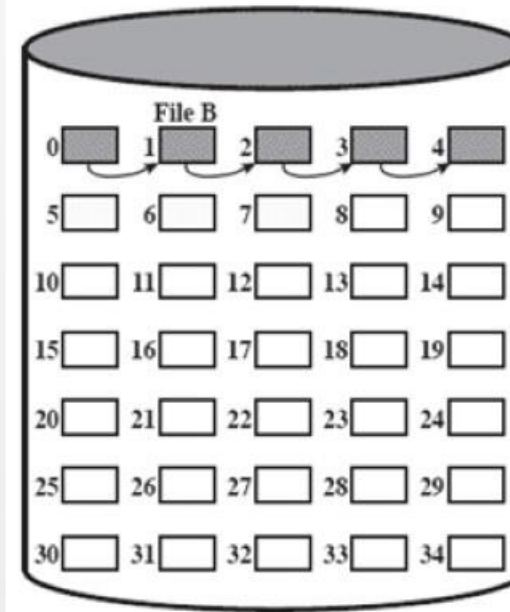


Figure 12.9 Chained Allocation

Chained Allocation

- If it is necessary to bring in several blocks of a file at a time, as in sequential processing, then a series of accesses to different parts of the disk are required.
- This is perhaps a more significant effect on a single-user system but may also be of concern on a shared system.
- To overcome this problem, some systems periodically consolidate files



File Allocation Table		
File Name	Start Block	Length
...	...	...
File B	0	5
...	...	...

Chained Allocation (After Consolidation)

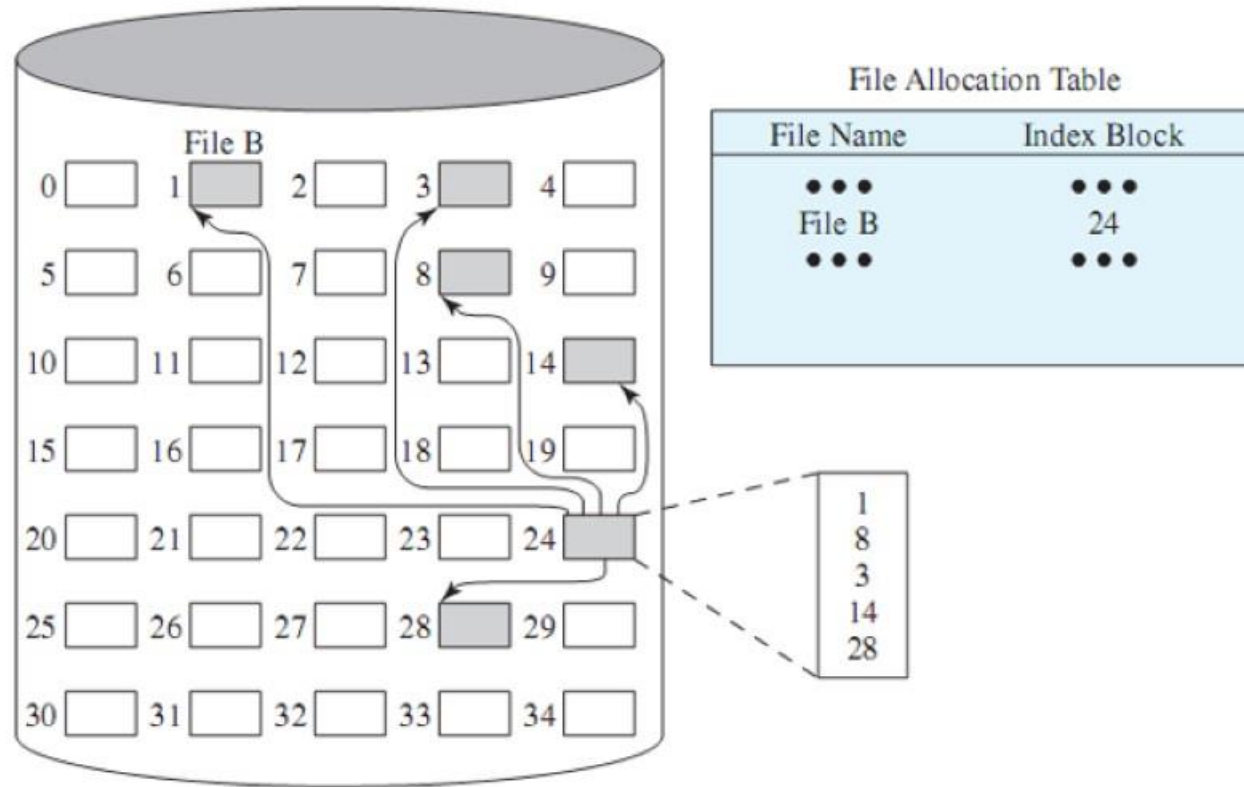
Indexed Allocation

- File allocation table contains a separate one-level index for each file
- The index has one entry for each portion allocated to the file
- The file allocation table contains block number for the index

Indexed Allocation Method

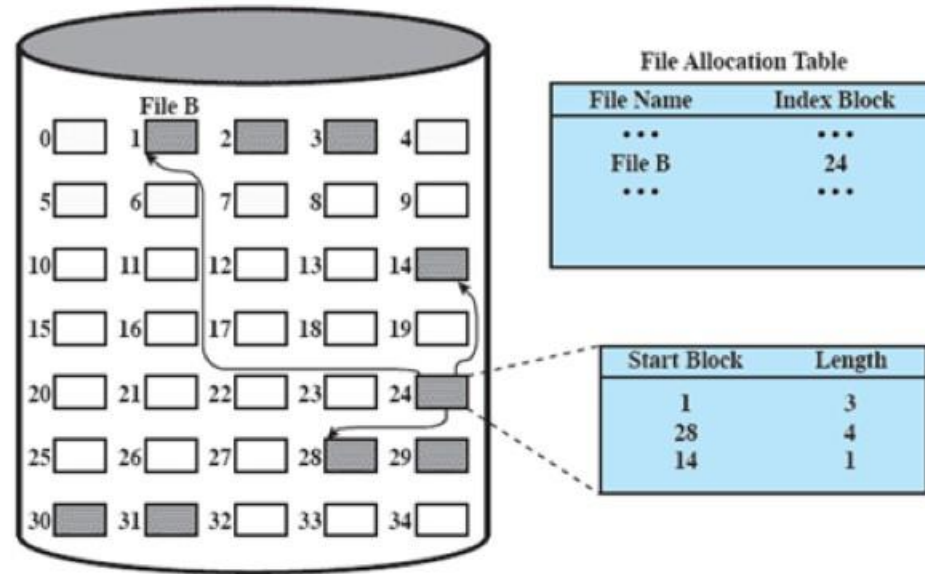
- Allocation may be either
 - Fixed size blocks or
 - Variable sized blocks
- Allocating by blocks eliminates external fragmentation
- Variable sized blocks improves locality
- Both cases require occasional consolidation

Indexed Allocation Method



lecture 09

Indexed allocation with Block Portions



Length

Indexed Allocation with Variable-Length Portions



Thank You