File Concept

Reading: Chapter10 of Textbook

How to store the large amount of data into the computer?
What happens, when process terminates or killed using some data?
How to assign the same data to the multiple processes?

The solution to all these problems is to store information on disks or on other external media called files.

Files

A file is named collection of related information normally resides on a secondary storage device such as disk or tape.

Commonly, files represent programs (both source and object forms) and data; data files may be numeric, alphanumeric, or binary.

Information stored in files must be persistent,- not be effected by power failures and system reboot.

The files are managed by OS.

The part of OS that is responsible to manage files is known as the file system.

File Attributes

A file is named, for the convenience of its human users, and it is referred to by its name. A name is string of characters.

The file attributes may very from system to system. Some common attributes are.

Name – only 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

OS provides system calls to perform operations on files. Some common calls are:

Create: If disk space is available it create new file without data.
Delete: Deletes files to free up disk space.
Open: Before using a file, a process must open it.
Close: When all access are finished, the file should be closed to free up the internal table space.
Read: Reads data from file.
Append: Adds data at the end of the file.
Seek: Repositions the file pointer to a specific place in the file.
Get attributes: Returns file attributes for processing.
Set attributes: To set the user settable attributes when file changed.
Rename: Rename file.

File Types

Many OS supports several types of files
To operate on file OS need to identify the file type
Common technique to implement the file type is to include the file type as part of file name, i.e file name is split into two parts: name and extension, separated by period (.)
E.g: *example.c*

But Unix like OS use a magic number to identify file types.

Common file types

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

Access Methods

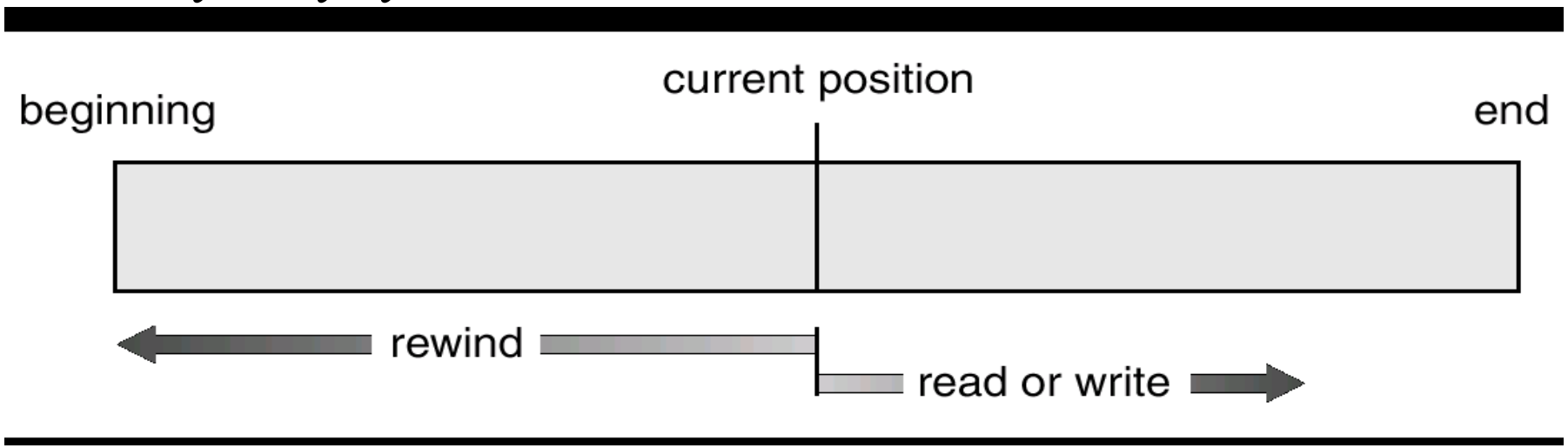
Sequential and Direct access

Sequential Access:

The simplest excess method; Information in the file is processed in order, one record after the other.

Convenient when the storage medium is magnetic tap.

Used in many early systems.



Sequential Access Operations:

read next - reads the next portion of the file and automatically advances a file pointer

write next - appends to the end of the file

reset – reset to the beginning

Access Methods

Direct Access:

Files whose bytes or records can be read in any order.

Based on disk model of file, since disks allow random access to any block.

Used for immediate access to large amounts of information.

When a query concerning a particular subject arrives, we compute which block contain the answer, and then read that block directly to provide desired information.

Operations:

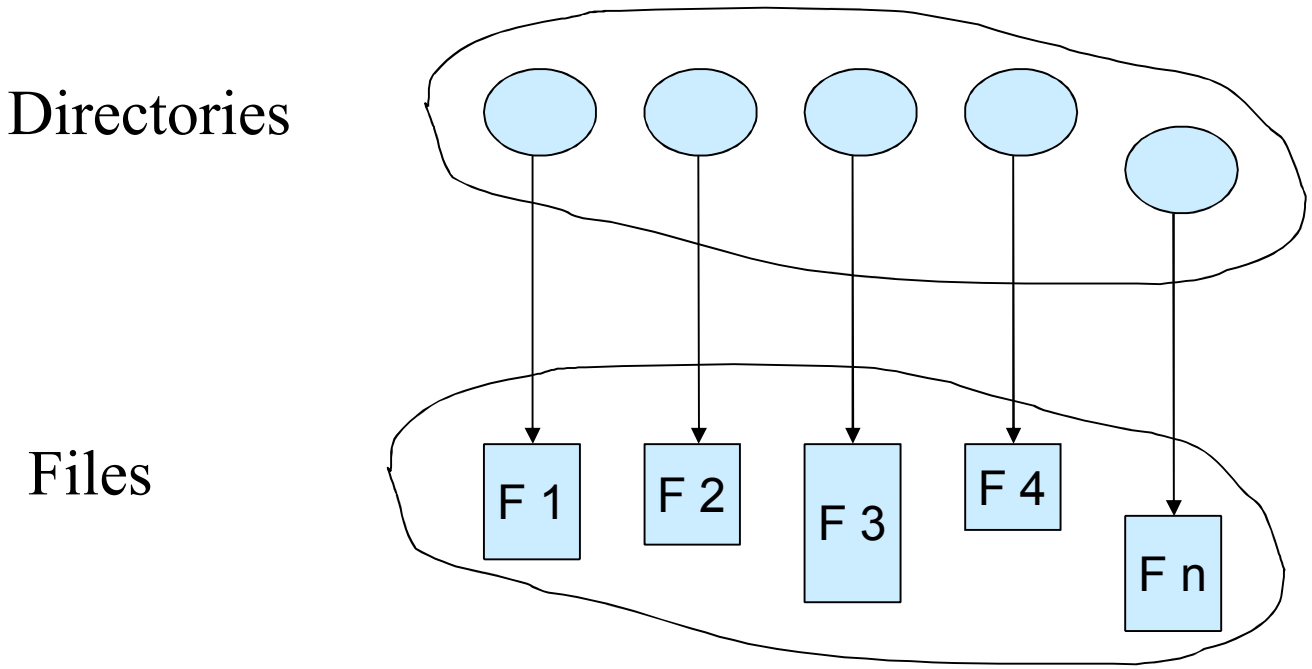
read *n*
write *n*
position to *n (seek)*
read next
write next
rewrite *n*

File can be access sequentially from the current position.

n = relative block number

Directory Structure

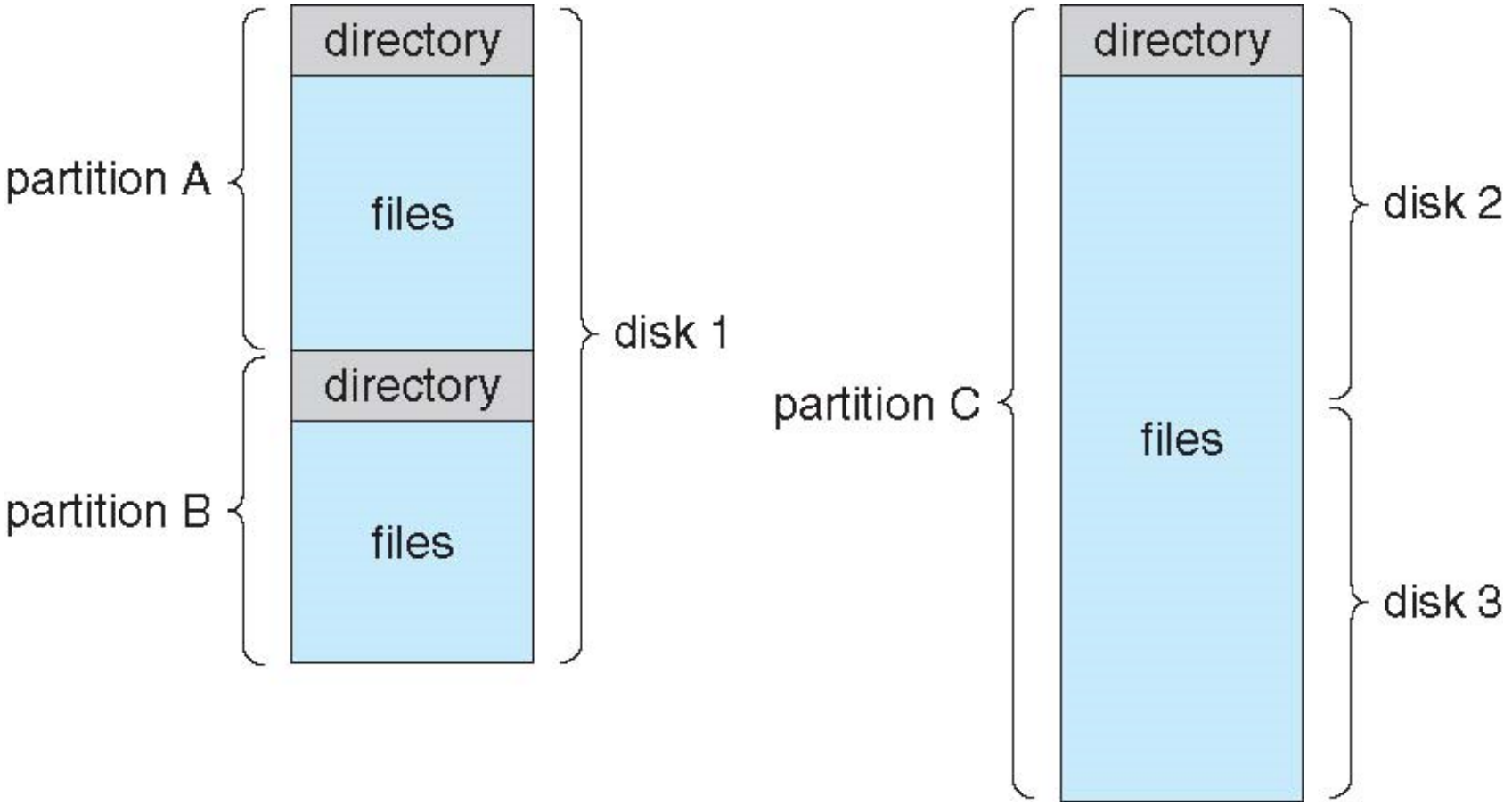
A directory is a node containing informations about files.



Both the directory structure and the files reside on disk
Directories can have different structures

Directory Structure

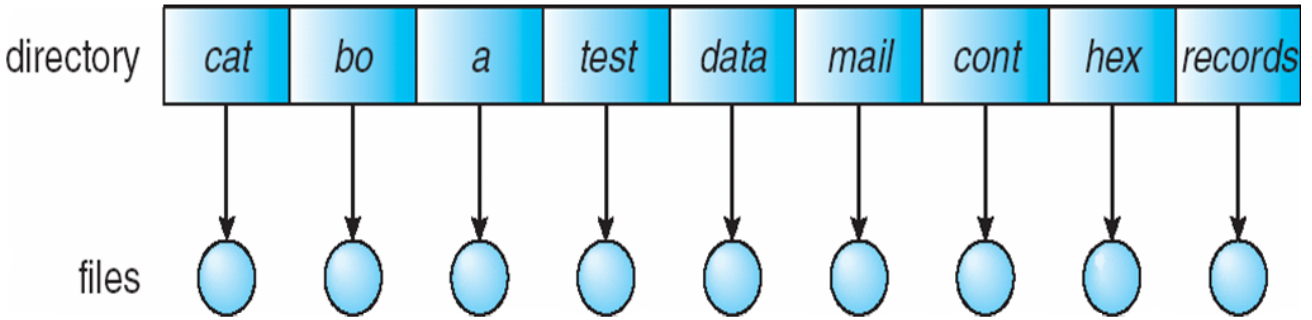
A Typical File-system Organization



Directory Structure

Single-Level-Directory:

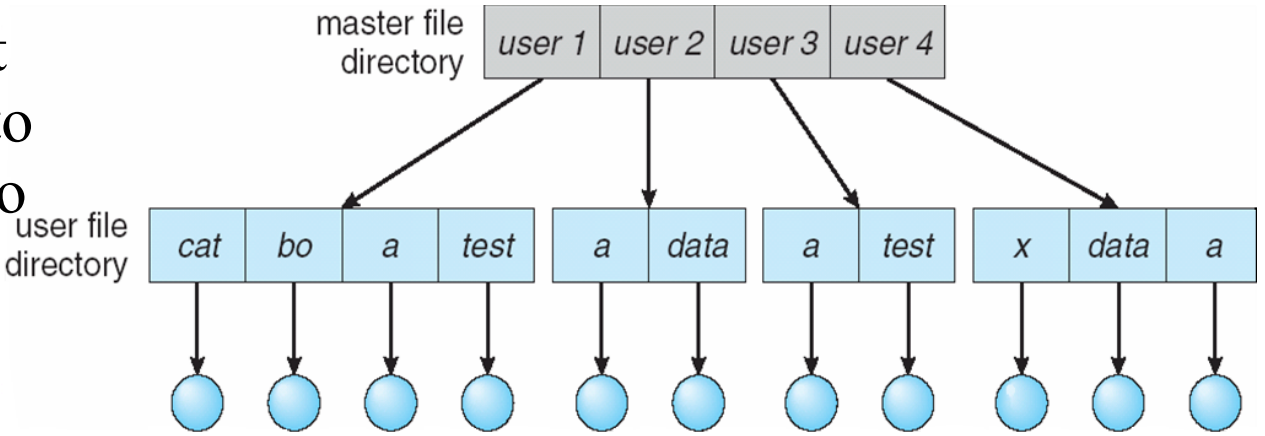
All files are contained in the same directory.
Easy to support and understand; but difficult to manage large amount of files and to manage different users.



Two-Level-Directory:

Separate directory for each user.
Used on a multiuser computer and on a simple network computers.

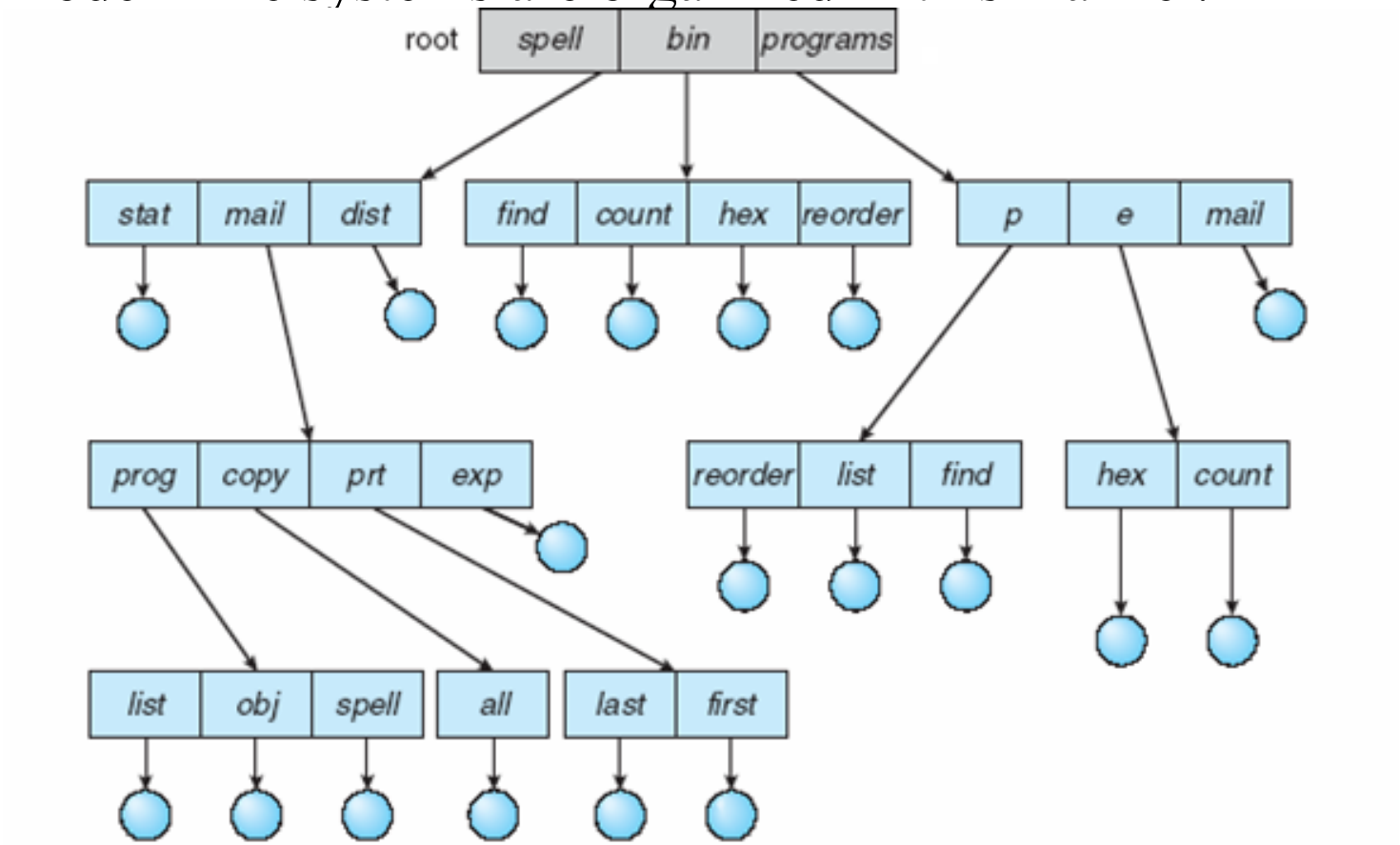
It has problem when users want to cooperate on some task and to access one another's files. It also cause problem when a single user has large number of files.



Directory Structure

Tree-Structured Directories

Generalization of two-level-structure to a tree of arbitrary height.
This allow the user to create their own subdirectories and to organize their files accordingly.
To allow to share the directory for different user acyclic-graph-structure is used.
Nearly all modern file systems are organized in this manner.



Home Works

HW # 9

Question No 10.1, 10.3, 10.5, 10.7 and 10.9 from book

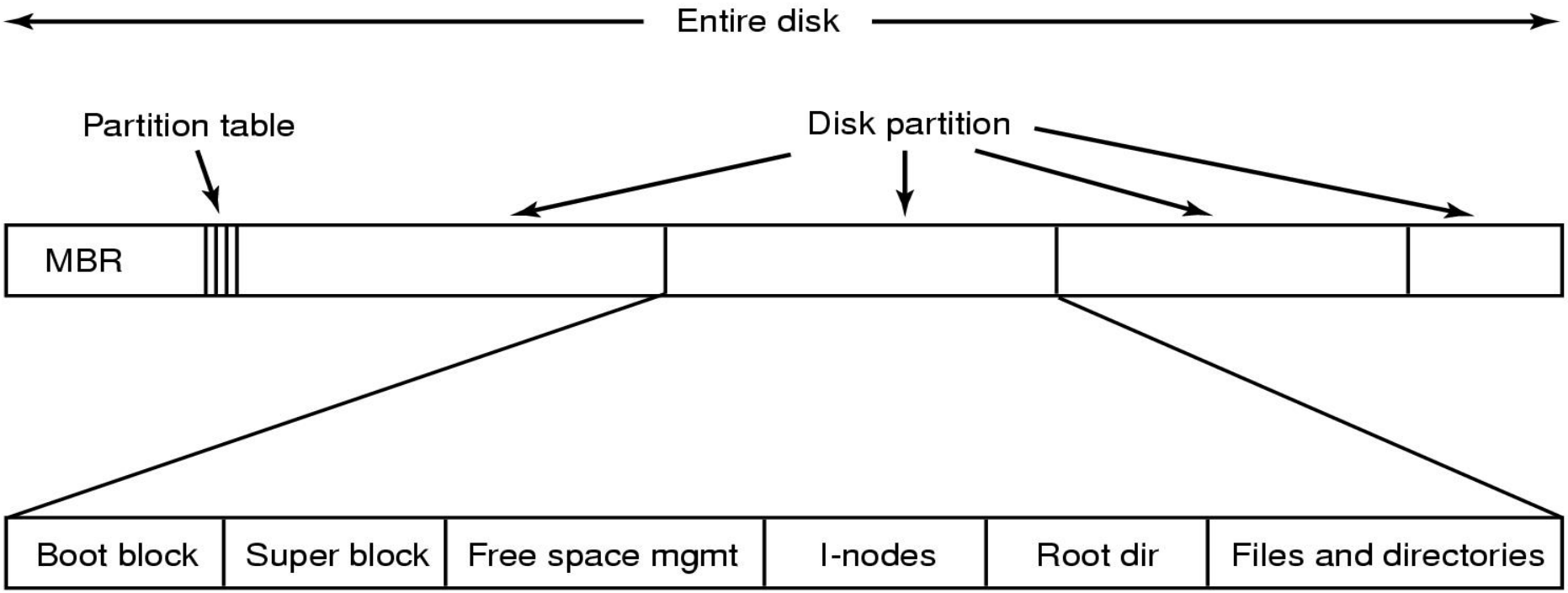
File-System Implementation

Reading: Chapter 11 of Textbook

How files and directories are stored?
How disk space is managed?
How to make every thing work efficiently and reliably?

File-System Layout

Disks are divided up into one or more partitions, with independent file system on each partition.

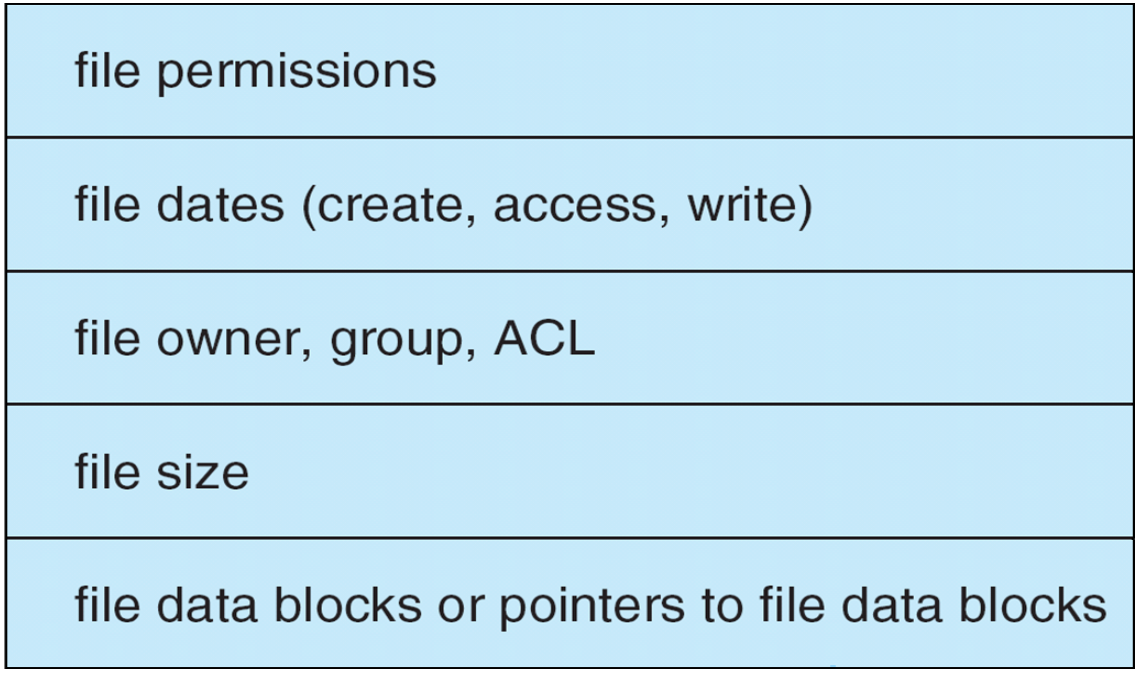


Boot Block: contain the information needed to boot OS from this partition, if this partition does not contain OS, this block will be empty.

Super block (with Free space mgmt.): Contain the partition detail, e.g number of block in the partition, size of block, free block count, free block pointer etc.

File Control Block (FCB)

FCB (also called inode) contains many details about the file, including the file permissions, ownership, size and location of data block.

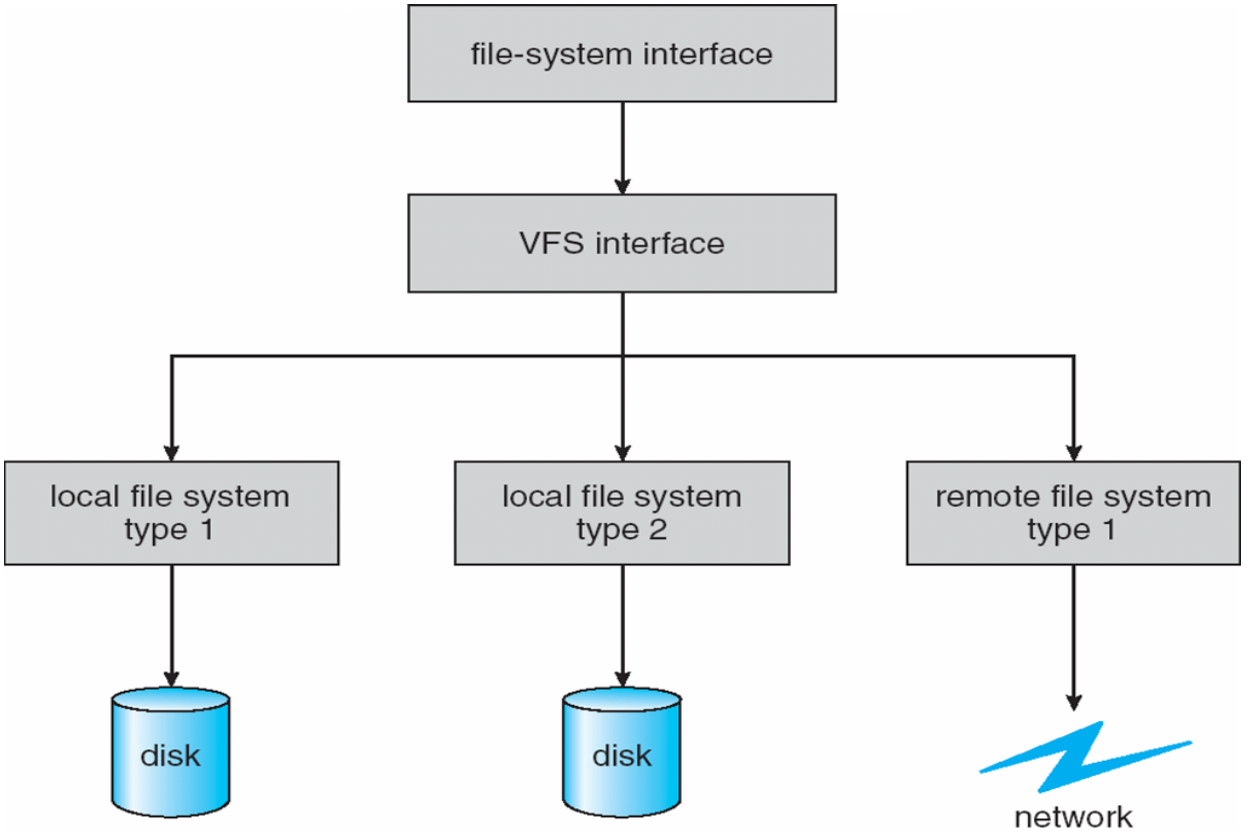


To create a new file, file system allocates a new FCB. The system then reads the appropriate directory into memory, update it with the new file name and FCB, and writes it back to the disk.

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.

Directory Implementation

The directory entry provides the information needed to find the disk block of the file. The file attributes are stored in the directory.

Linear List

Use the linear list of the file names with pointer to the data blocks.
It requires linear search to find a particular entry.
Advantages: Simple to implement.
Problem: linear search to find the file.

Hash Table

It consist linear list with hash table.
The hash table takes a value computed from the file name and returns a pointer to the file name in the linear list.
If that key is already in use, a linked list is constructed.
Advantages: greatly decrease the file search time.
Problem: It greatly fixed size and dependence of the hash function on that size.

Allocation Methods

Contiguous Allocation

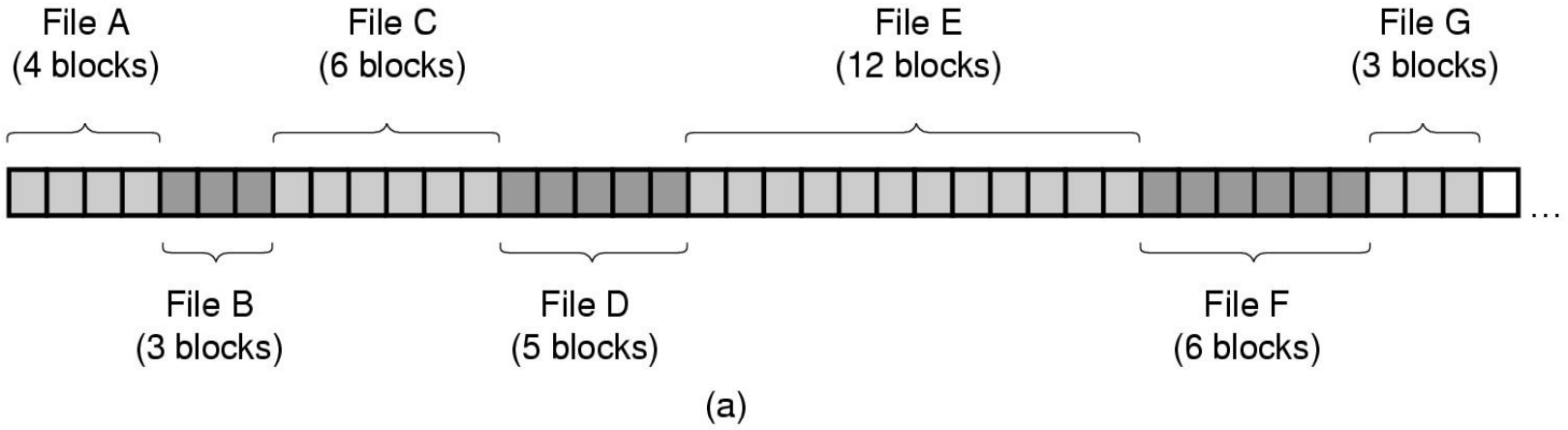
Each file occupy a set of contiguous block on the disk.

Disk addresses define a linear ordering on the disk.

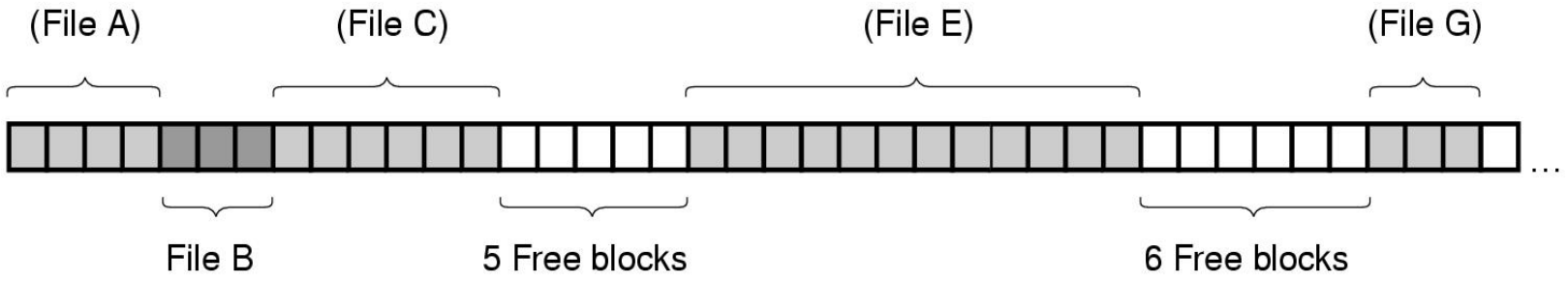
File is defined by the disk address and length in block units.

With 2-KB blocks, a 50-KB file would be allocated 25 consecutive blocks.

Both sequential and direct access can be supported by contiguous allocation.



(a)



(b)

Allocation Methods

Contiguous Allocation

Advantages:

Simple to implement; accessing a file that has been allocated contiguously is easy.

High performance; the entire file can be read in single operation i.e. decrease the seek time.

Problems:

fragmentation: when files are allocated and deleted the free disk space is broken into holes.

Dynamic-storage-allocation problem: searching of right holes.

Required pre-information of file size.

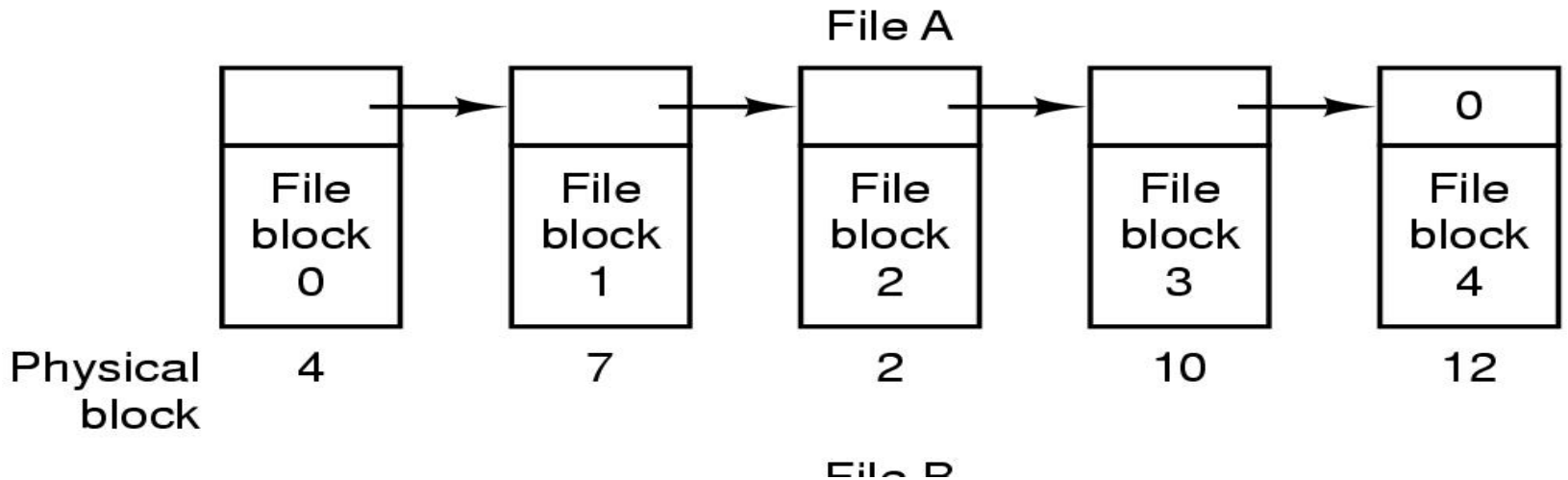
Due to its good performance it used in many system; it is widely used in CD-ROM file system.

Allocation Methods

Linked Allocation

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

Each block contain the pointer to the next block of same file. To create the new file, we simply create a new entry in the directory; with linked allocation, each directory entry has a pointer to the first disk block of the file.



Allocation Methods

Linked Allocation

Problems:

It solves all problems of contiguous allocation but it can used only for sequential access file; random access is extremely slow.

Each block access required disk seek.

It also required space for pointer.

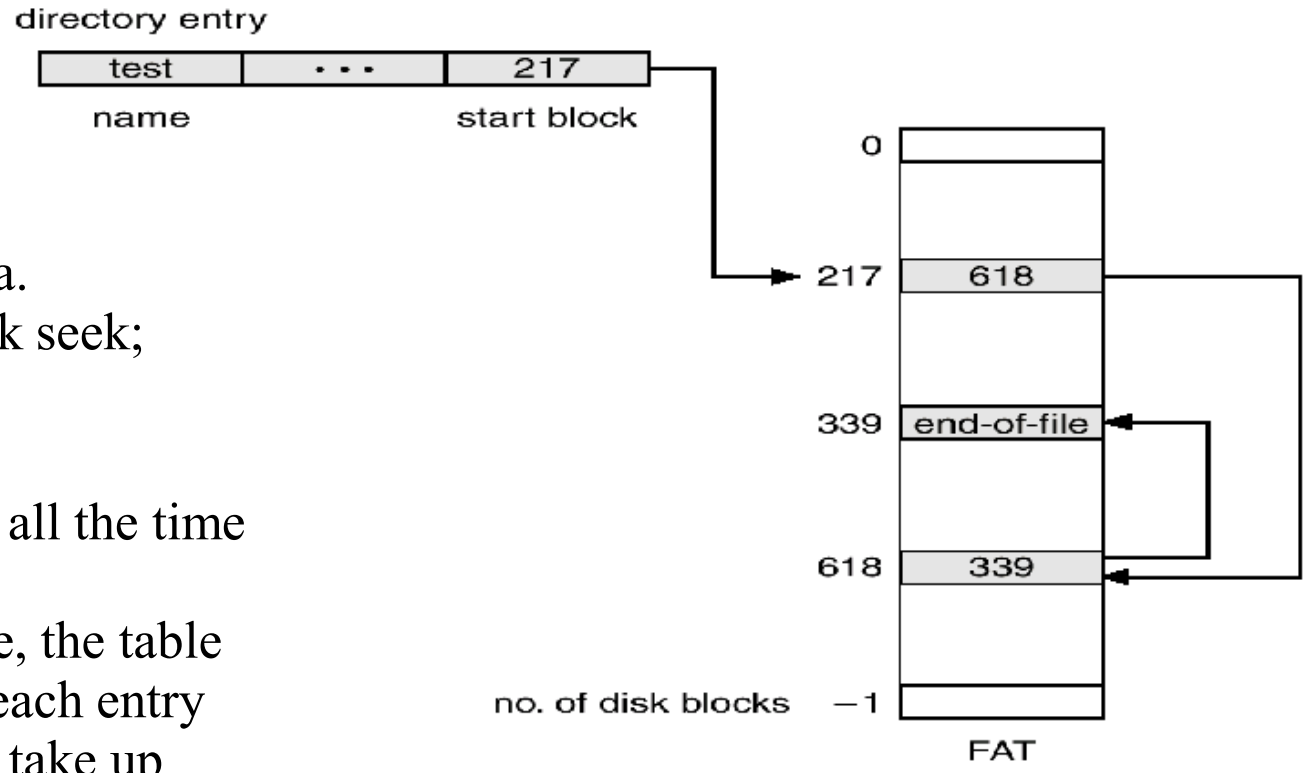
Solution: Using File Allocation Table (FAT).

The table has one entry for each disk block containing next block number for the file. This resides at the beginning of each disk partition.

Allocation Methods

Linked Allocation using FAT

The FAT is used as is a linked list. The directory entry contains the block number of the first block of the file. The FAT is looked to find next block until a special end-of-file value is reached.



Advantages:

The entire block is available for data.
Result the significant number of disk seek; random access time is improved.

Problems:

The entire table must be in memory all the time to make it work.
With 20GB disk and 1KB block size, the table needs 20 millions entries. Suppose each entry requires 3 bytes. Thus the table will take up 60MB of main memory all the time.

Allocation Methods

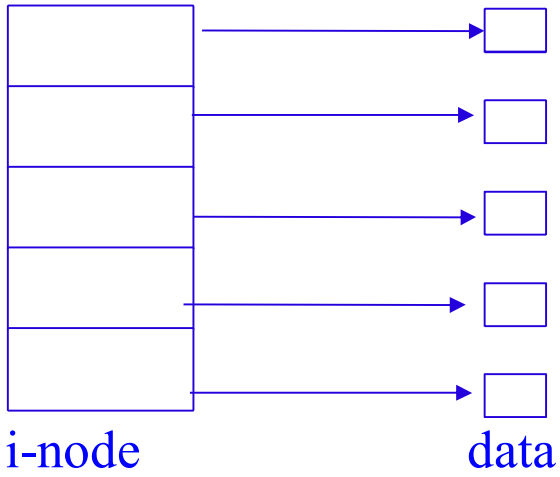
Index Allocation

To keep the track of which blocks belongs to which file, each file has data structure (i-node) that list the attributes and disk address of the disk block associate with the file.

Each i-node are stored in a disk block, if a disk block is not sufficient to hold i-node it can be multileveled.

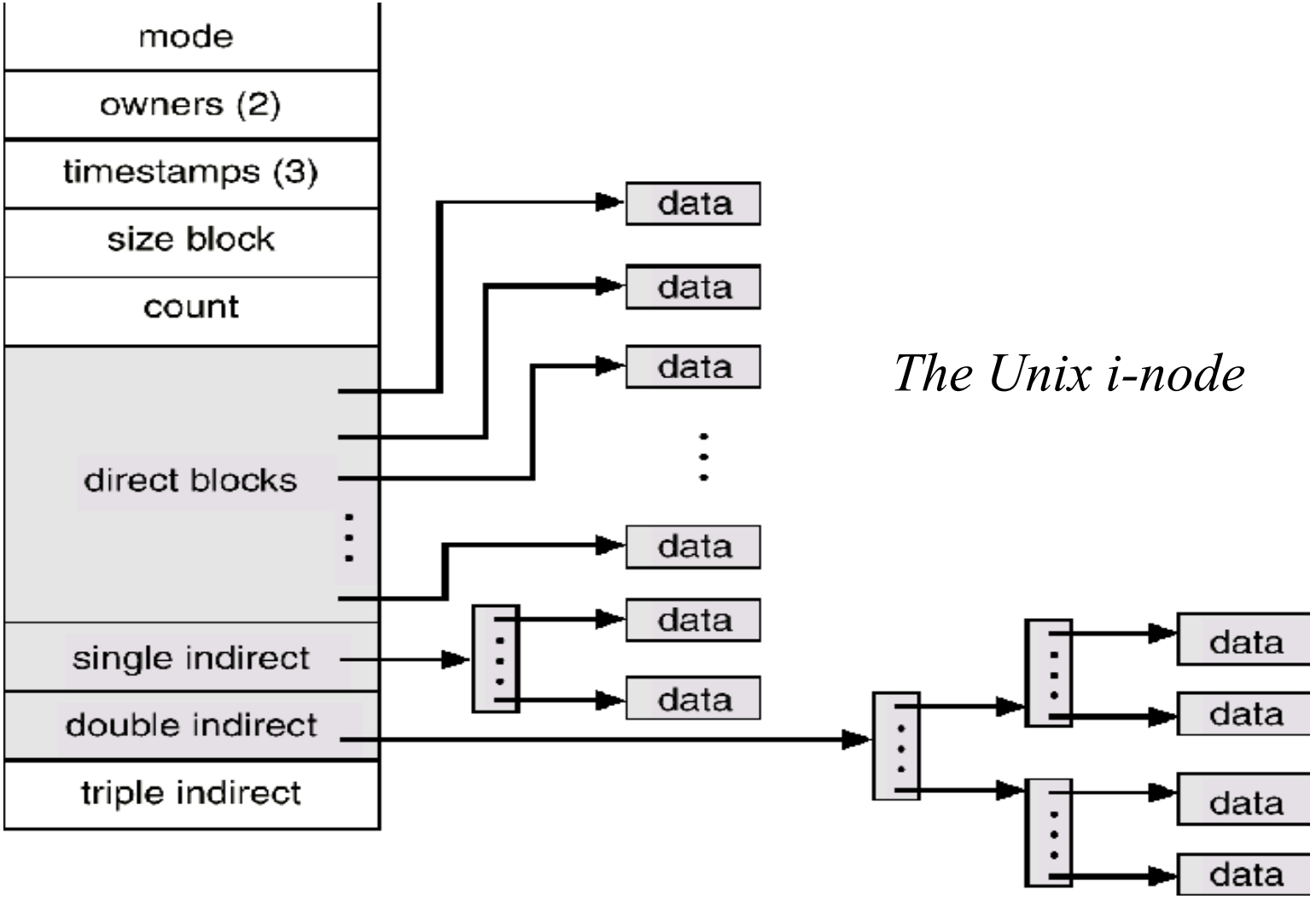
Independent to disk size.

If i-node occupies n bytes for each file and k files are opened, the total memory by i-nodes is kn bytes.



Allocation Methods

Index Allocation



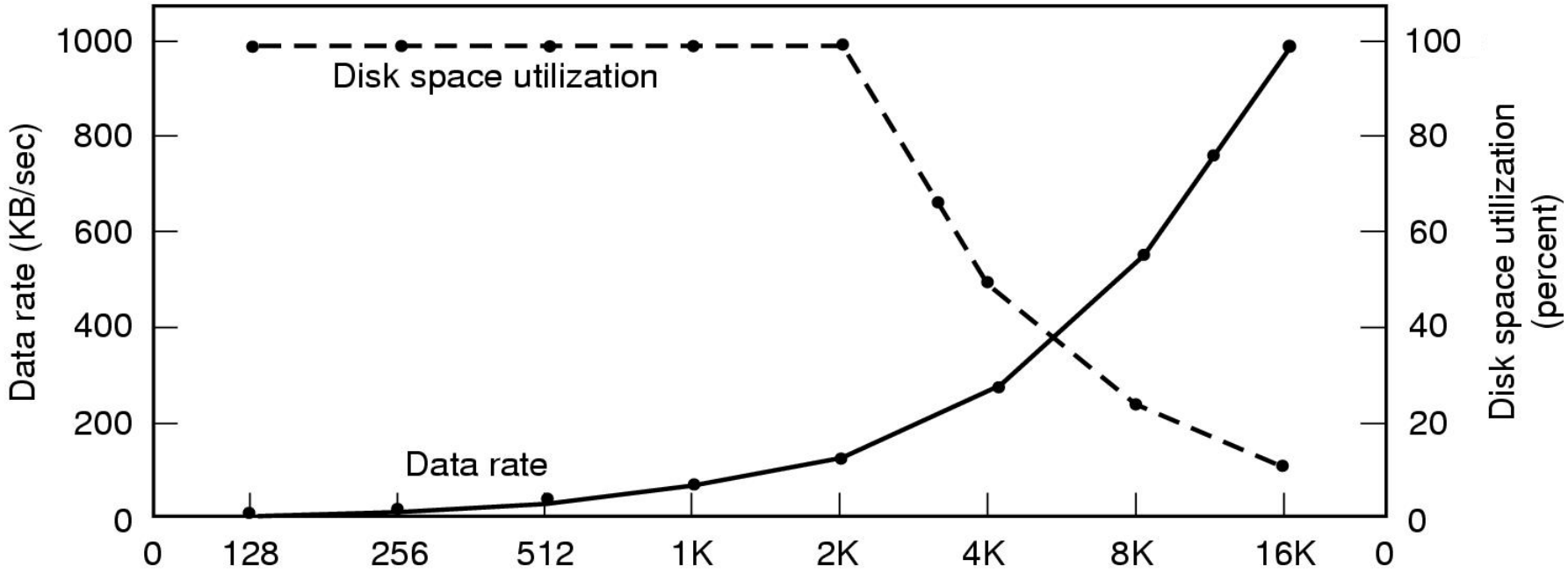
The Unix i-node

Advantages: Far smaller than space occupied by FAT.

Problems: This can be suffer from performance.

Block Size

Nearly all file systems chop files up into fixed-size block.
Using a large size result the wastage of disk space.
Using a small size increase the seek and rotational delay- low data access rate.



Dark line (left hand scale) gives data rate of a disk; Dotted line (right hand scale) gives disk space utilization. All files are 2KB

Free-Space Management

To keep track of free blocks, system maintains the free-space-list.
The free-space-list records all free blocks- those not allocated to some file or directory.
To create a file, system search the free-space-list for required amount of space, and allocate that space to the new file and removed from free-space-list.
When a file is deleted, its disk space is added to the free-space-list.
The free-space-list can be implemented in two ways:

- Bit Vector (bitmap)
- Linked list

Free-Space Management

Bit Vector

Each block is represented by a bit. If the block is free, the bit is 1; if the block is allocated the bit is 0.

A disk with n blocks requires a bit vector with n bits.

Ex: Consider a disk where blocks 2,3,4,5,8,9,10,11,12,13,17,18,..... are free, and rest are allocated. The free space bit vector would be

0011110011111100010.....

Advantages: Simple and efficient in finding first free block, or n consecutive free blocks.

Problems: Inefficient unless the entire bitmap is kept in main memory.

Keeping in main memory is possible only for small disk, when disk is large the bit vector would be large.

Many system use bit vector method.

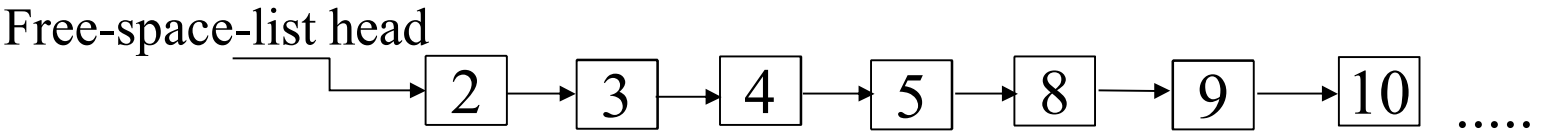
Free-Space Management

Linked List

Link together all the free disk blocks, keeping a pointer to the first free block in a special location on the disk and caching it in memory.

The first block contains a pointer to the next free block.

The previous example can be represented in linked as



Advantages: Only one block is kept in memory.

Problems: Not efficient; to traverse list, it must read each block.

File Recovery Systems

Files and directories are kept both in main memory and on disk, and care must be taken to ensure that a system failure does not result in loss of data or in data inconsistency

A system crash can cause inconsistencies among on-disk file-system data structures, such as directory structures, free-block pointers, and free FCB pointers

Consistency checking – compares data in directory structure with data blocks on disk, and tries to fix inconsistencies

Use system programs to **back up** data from disk to another storage device (magnetic tape, other magnetic disk, optical)

Recover lost file or disk by **restoring** data from backup

Log Structured File Systems

Consistency Checking algorithms - *log based transaction oriented file systems*

Log structured (or **journaling**) file systems record each metadata update to the file system as a **transaction**

All transactions are written to a log

- A transaction is considered committed once it is written to the log (sequentially)
- Sometimes to a separate device or section of disk
- However, the file system may not yet be updated

The transactions in the log are asynchronously written to the file system structures

- When the file system structures are modified, the transaction is removed from the log

If the file system crashes, all remaining transactions in the log must still be performed

Faster recovery from crash, removes chance of inconsistency of metadata

Home Works

HW #10

Q.1. How many bits would be needed to store the free-space-list under the following conditions if a bitmap were used to implement?

a) 500, 000 blocks total and 200, 000 free blocks.

b) 1,000,000 blocks total and 0 free blocks.

Also find how much space is required if it need to be stored in memory?

Question No 11.2, 11.3, 11.4, 11.5, 11.6, 11.9, 11.10, 11.12 & 11.14

Disk Management

Reading : Chapter 12 From Textbook

Disk is an I/O devices that is common to every computer.

Disk structure
Disk Scheduling
Disk Formatting & Error Handling

Disk Structure

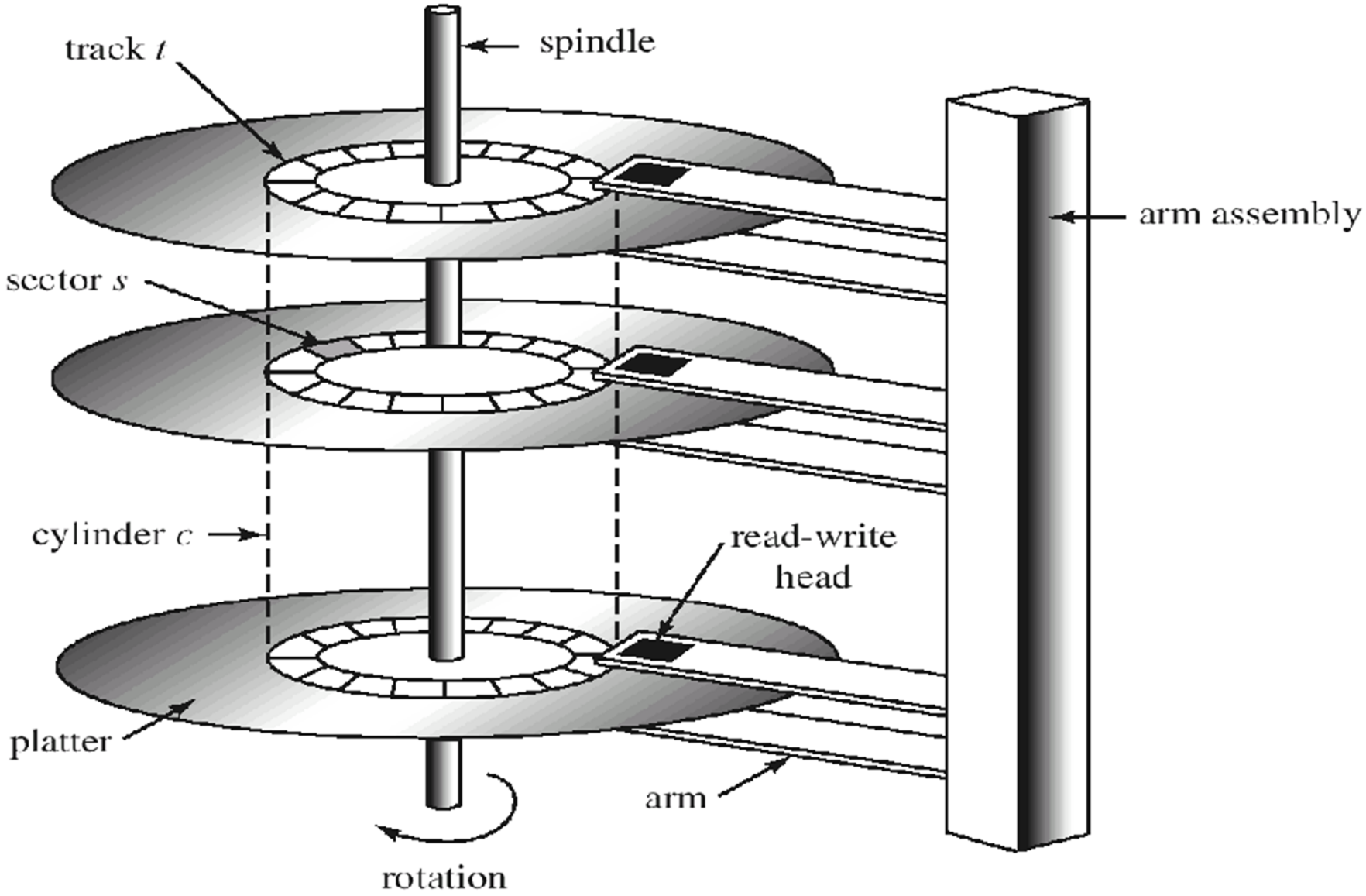
Disks comes in many sizes and speeds, and information may be stored optically or magnetically; however, all disks share a number of important features.

For Example: floppy disks, hard disks, CD-ROMs and DVDs.

Disk surface is divided into number of logical block called *sectors* and *tracks*.

The term *cylinder* refers to all the tracks at particular head position in hard disk.

Hard-Disk Structure



Disk Operations

Latency Time: The time taken to rotate from its current position to a position adjacent to the read-write head.

Seek: The processes of moving the arm assembly to new cylinder.

To access a particular record, first the arm assembly must be moved to the appropriate cylinder, and then rotate the disk until it is immediately under the read-write head.

The time taken to access the whole record is called *transmission time*.

Disk Scheduling

OS is responsible to use the hardware efficiently – for the disk drive this means fast seek, latency and transmission time.

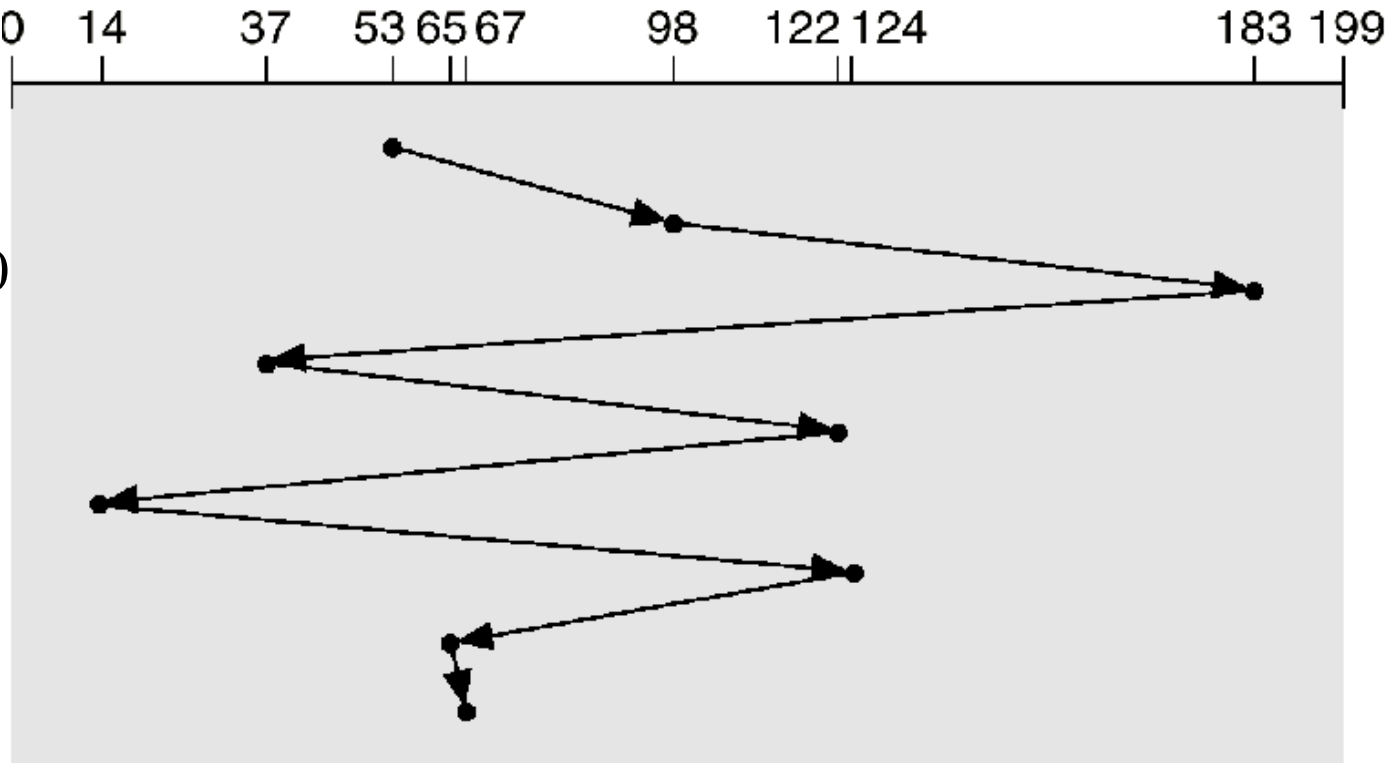
For most disks, the *seek time* dominates the other two times, so reducing the *mean seek time* can improve system performance substantially.

First-Come First-Served (FCFS)

The first request to arrive is the first one serviced.

Example:

queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53



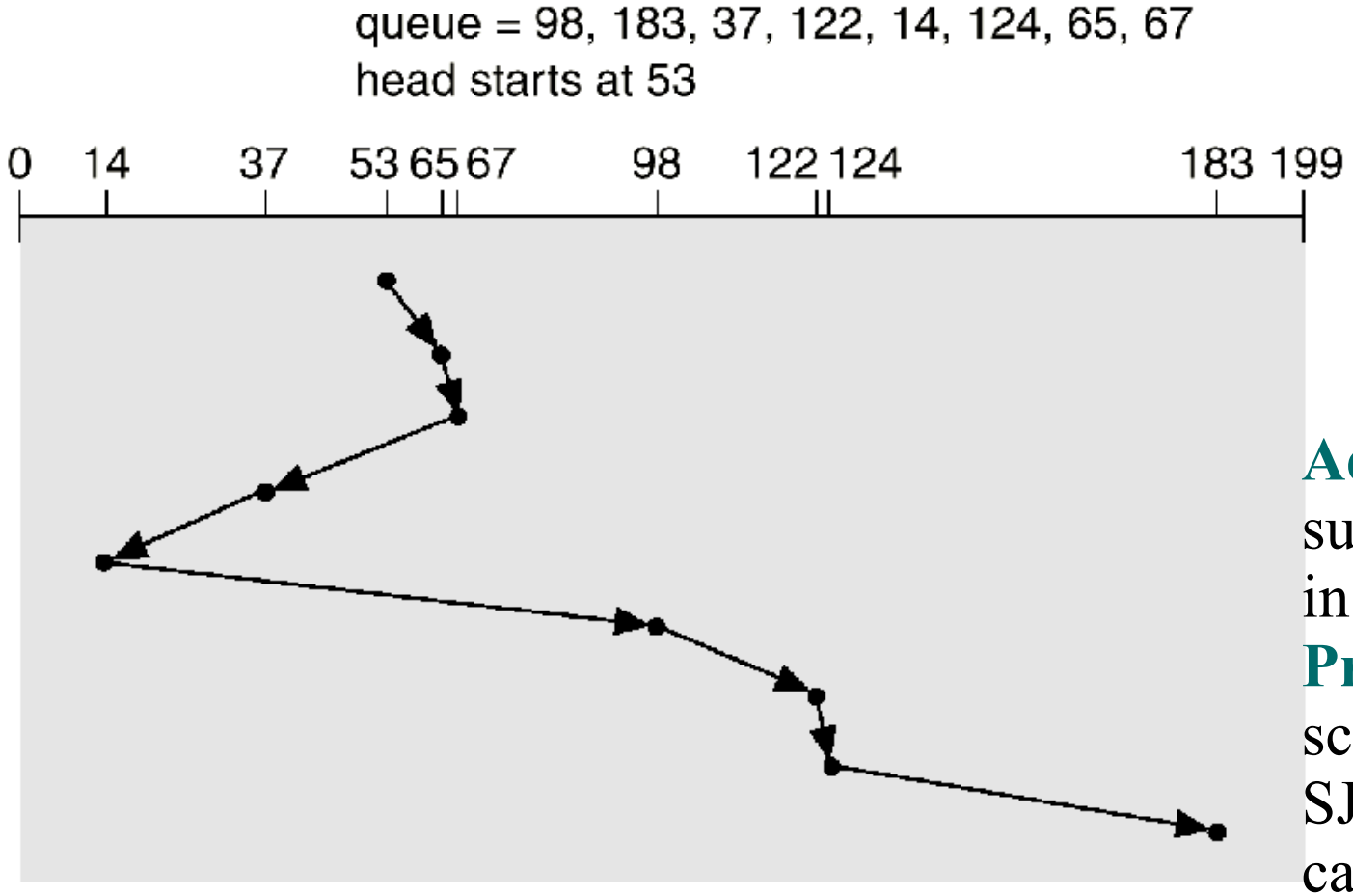
Total Head Movement = 640 cylinders

Advantages: Simple and Fair.

Problems: Does not provide fastest service.

Shortest-Seek-Time-First (SSTF)

Selects the request with the minimum seek time from the current head position.



Total Head Movement = 236 cylinders

Advantages: Gives a substantial improvement in performance.

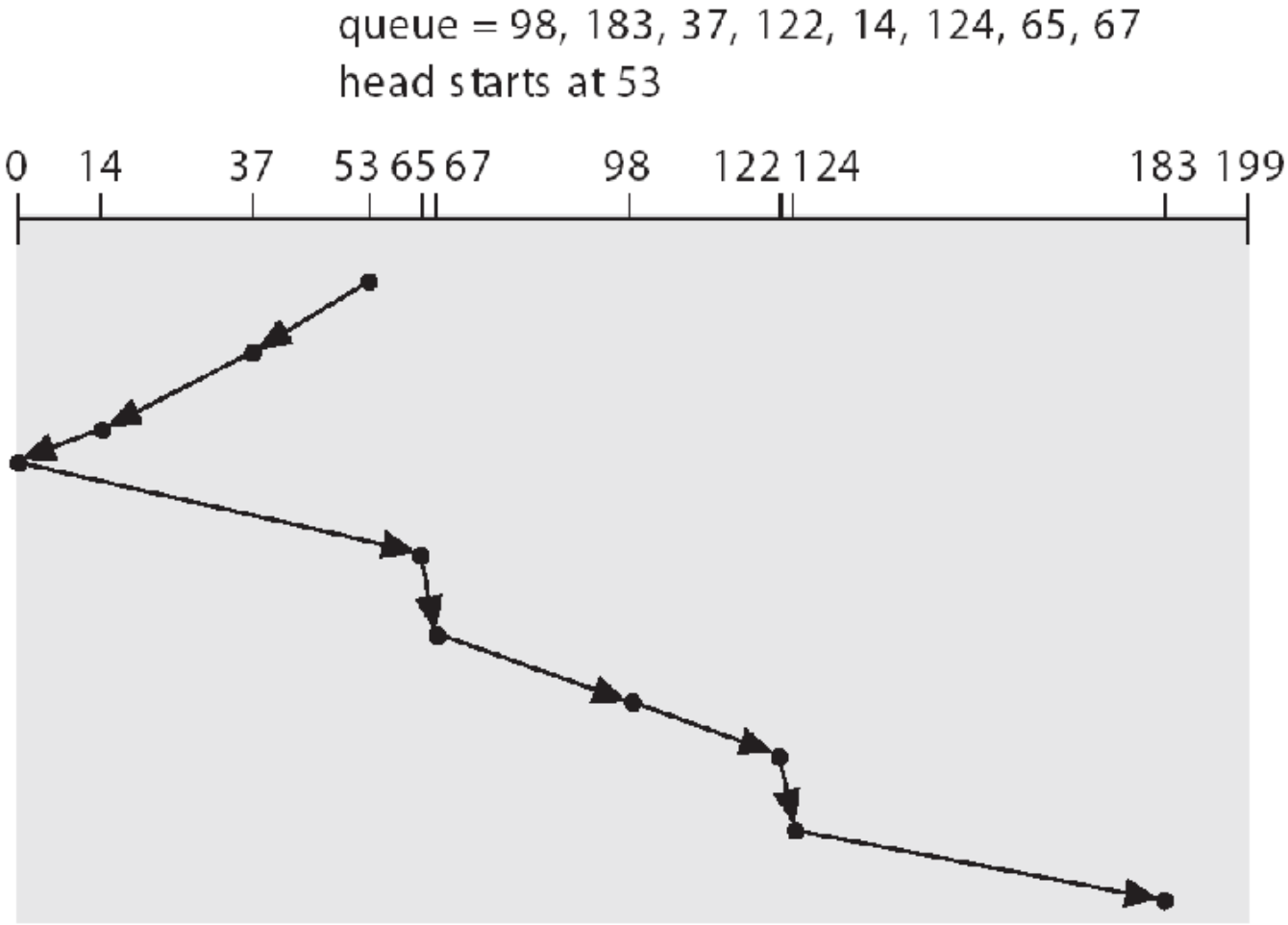
Problems: SSTF scheduling is a form of SJF scheduling; may cause starvation of some requests. Not optimal.

Used in batch system where throughput is the major consideration but unacceptable in interactive system.

SCAN

The disk arm starts at one end of the disk, and moves toward the other end, servicing requests until it gets to the other end of the disk, where the head movement is reversed and servicing continues.

Sometimes called the *elevator algorithm*.



Total Head Movement = 208 cylinders

Advantages: Decreases variances in seek and improve response time.
Problem: Starvation is possible if there are repeated request in current track.

C-SCAN

When a uniform distribution of request for cylinders, only the few request are in extreme cylinders, since these cylinders have recently been serviced.

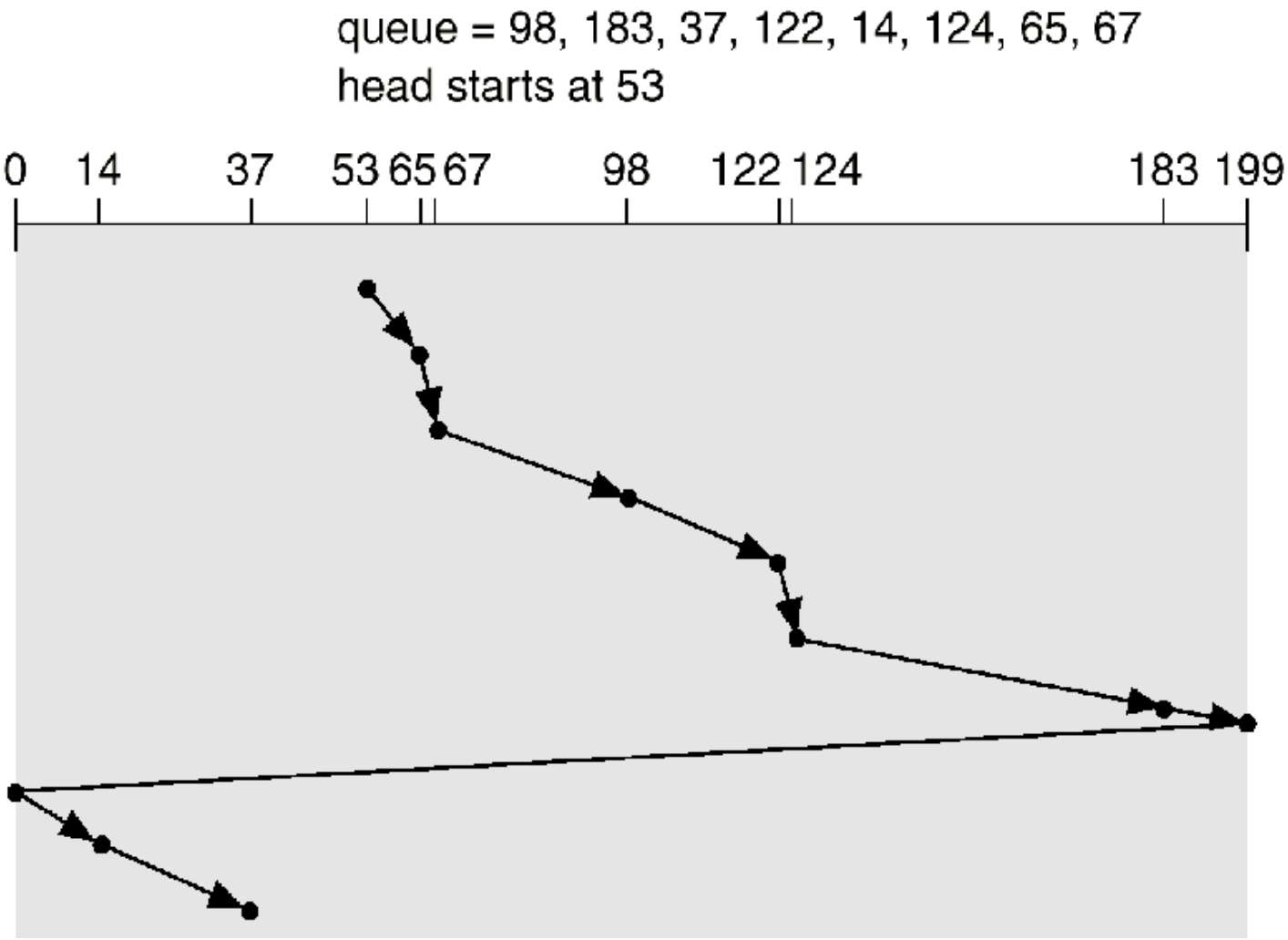
Why not go to the next extreme?

Circular SCAN is a variant of SCAN designed to provide a more uniform wait time.

The head moves from one end of the disk to the other. Servicing requests as it goes. When it reaches the other end, however, it immediately returns to the beginning of the disk, without servicing any requests on the return trip.

Treats the cylinders as a circular list that wraps around from the last cylinder to the first one.

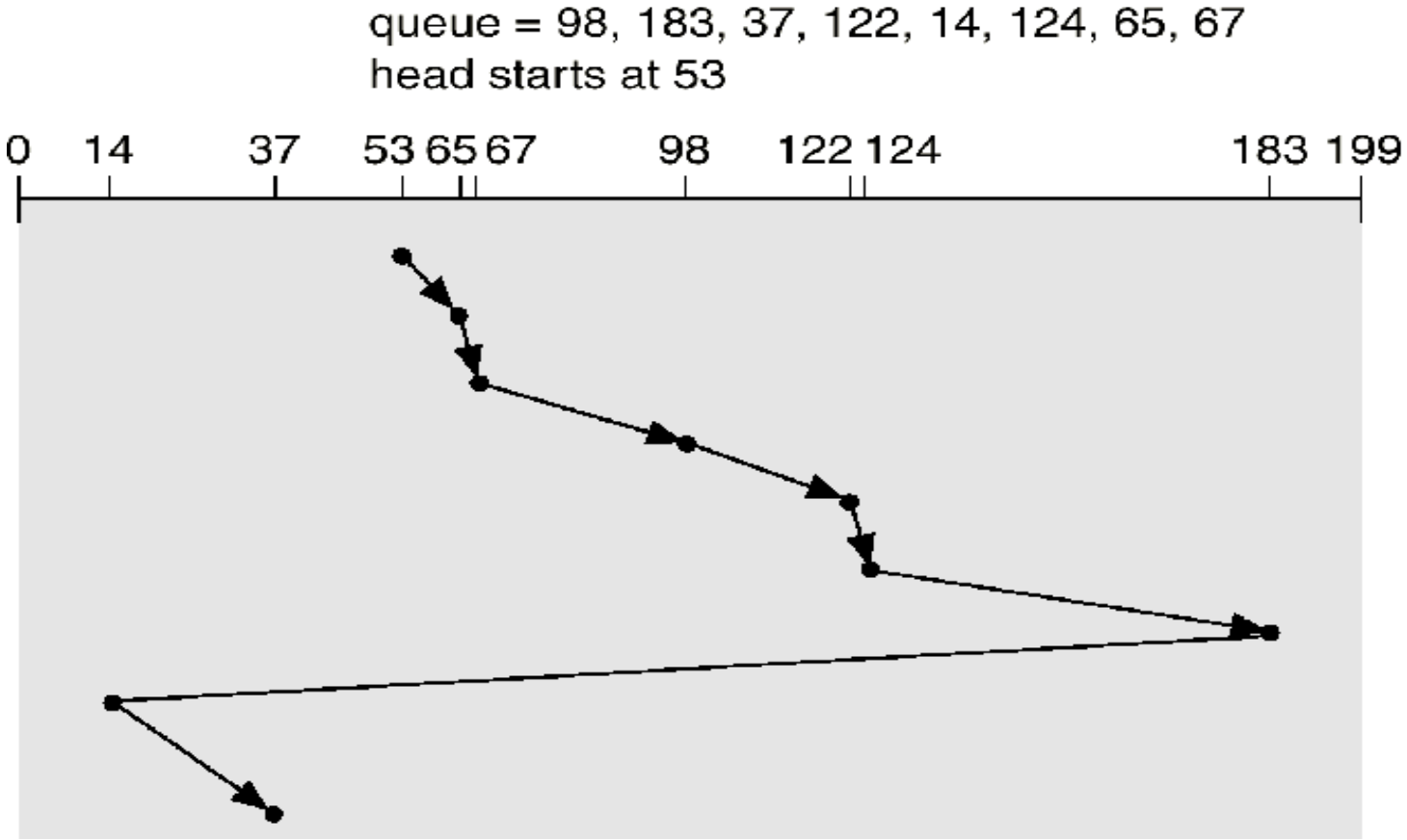
C-SCAN


queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53
0
14
37
53
65
67
98
122
124
183
199

C-LOOK

Version of C-SCAN

Arm only goes as far as the last request in each direction, then reverses direction immediately, without first going all the way to the end of the disk.

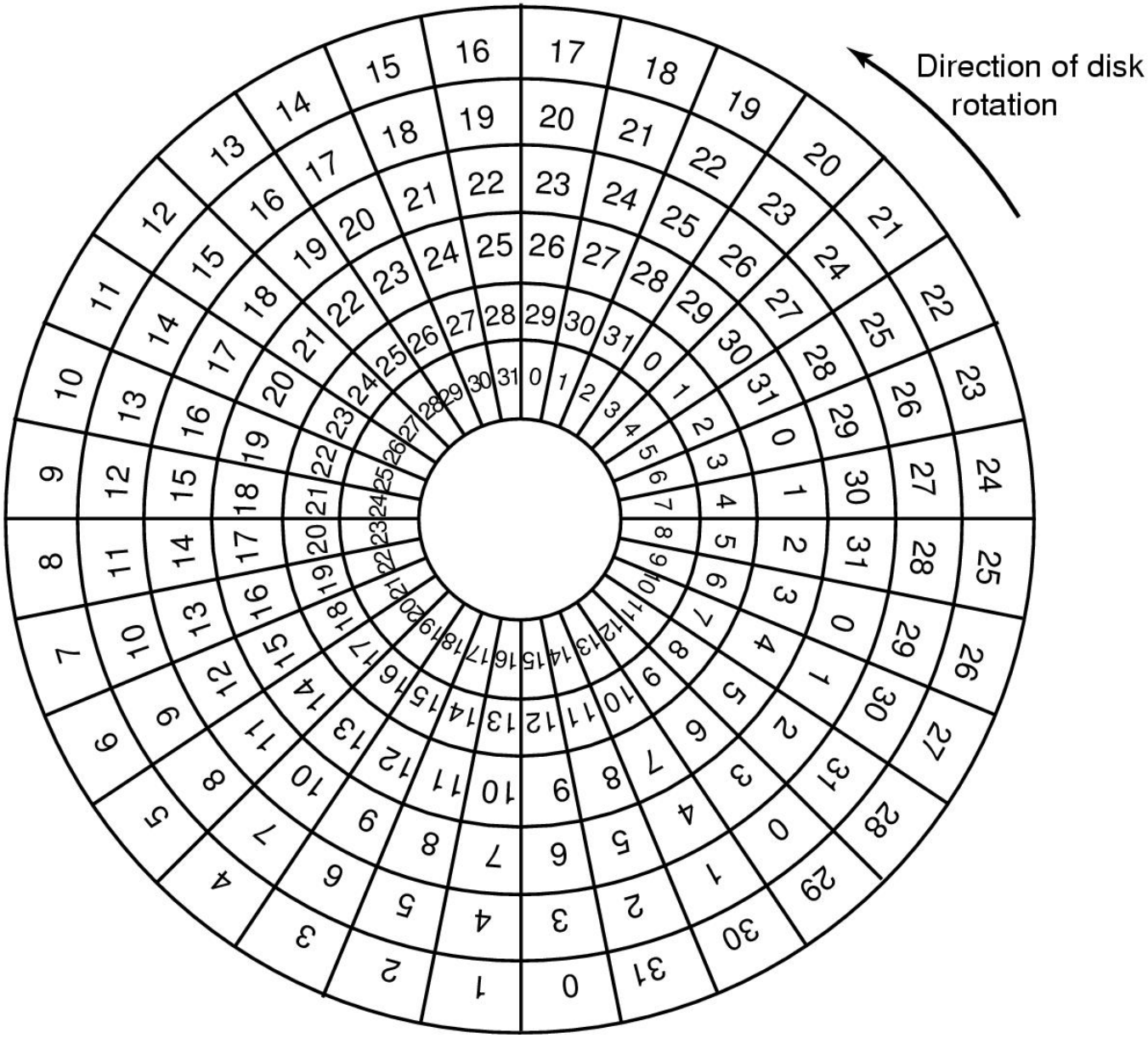


Disk Formatting

Before a disk can store data, it must be divided into sectors that the disk controller can read and write, called low-level formating.

The sector typically consists of preamble, data and ECC.
The preamble contains the cylinder and sector number and the ECC contains redundant information that can be used to recover from read error.
The size depends upon the manufacturer, depending on reliability.

Disk Formatting



Disk Formatting

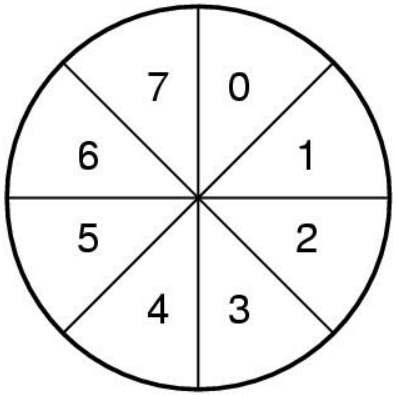
If disk I/O operations are limited to transferring a single sector at a time, it reads the first sector from the disk and doing the ECC calculation, and transfers to main memory, during this time the next sector will fly by the head.

When transferring completes the controller will have to wait almost an entire rotation for the second sector to come around again.

This problem can be eliminated by numbering the sectors in an interleaved fashion when formating the disk.

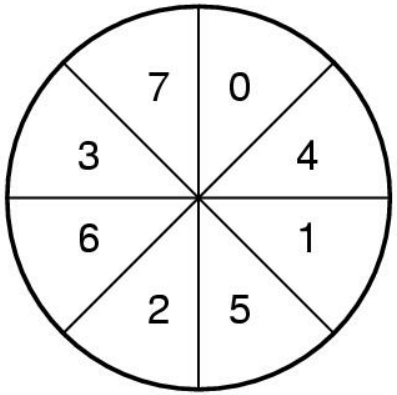
According to the copying rate, interleaving may be of single or double.

Disk Formatting



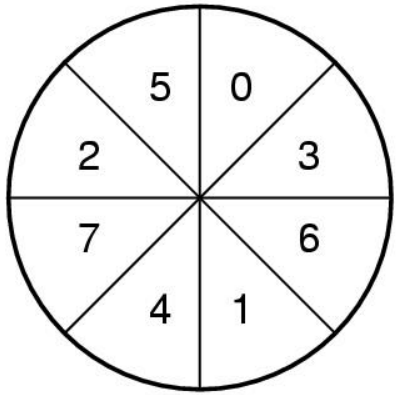
(a)

a) No interleaving



(b)

b) Single interleaving



(c)

c) Double interleaving

Error Handling

Bad Blocks

Most frequently, one or more sectors becomes defective or most disks even come from factory with bad blocks.

Depending on the disk and controller in use, these blocks handled in variety of ways.

1. Bad blocks are handled manually- For example run MS-DOS chkdsk command to find bad block, and format command to create new block – data resided on bad blocks usually are lost.
2. Using bad block recovery- The controller maintains the list of bad blocks on the disk and for each bad block, one of the spares is substituted.

Home Works

HW #11:

1. Question No 12.2, 12.8, 12.11, 12.14, 12.15, 12.17 & 12.18 from book
2. Suppose that a disk drive has 5,000 cylinders, numbered 0 to 4999. The drive is currently serving a request at cylinder 143, and the previous request was at cylinder 125. The queue of the pending requests, in FIFO order, is
   86, 1470, 913, 1774, 948, 1502, 1022, 1750, 130.
   Starting from the current head position, what is the total distance (in cylinders) that the disk arm moves to satisfy all the pending requests for each of the following disk-scheduling algorithms?
   a) FCFS b) SSTF c) SCAN d) C-SCAN e) C-LOOK
3. A disk has 8 sectors per track and spins at 600 rpm. It takes the controller 10ms from the end of one I/O operation before it can issue a subsequent one. How long does it take to read all 8 sectors using the following interleaving system?
   a) No interleaving b)Single interleaving c)Double interleaving