Operating System Concepts

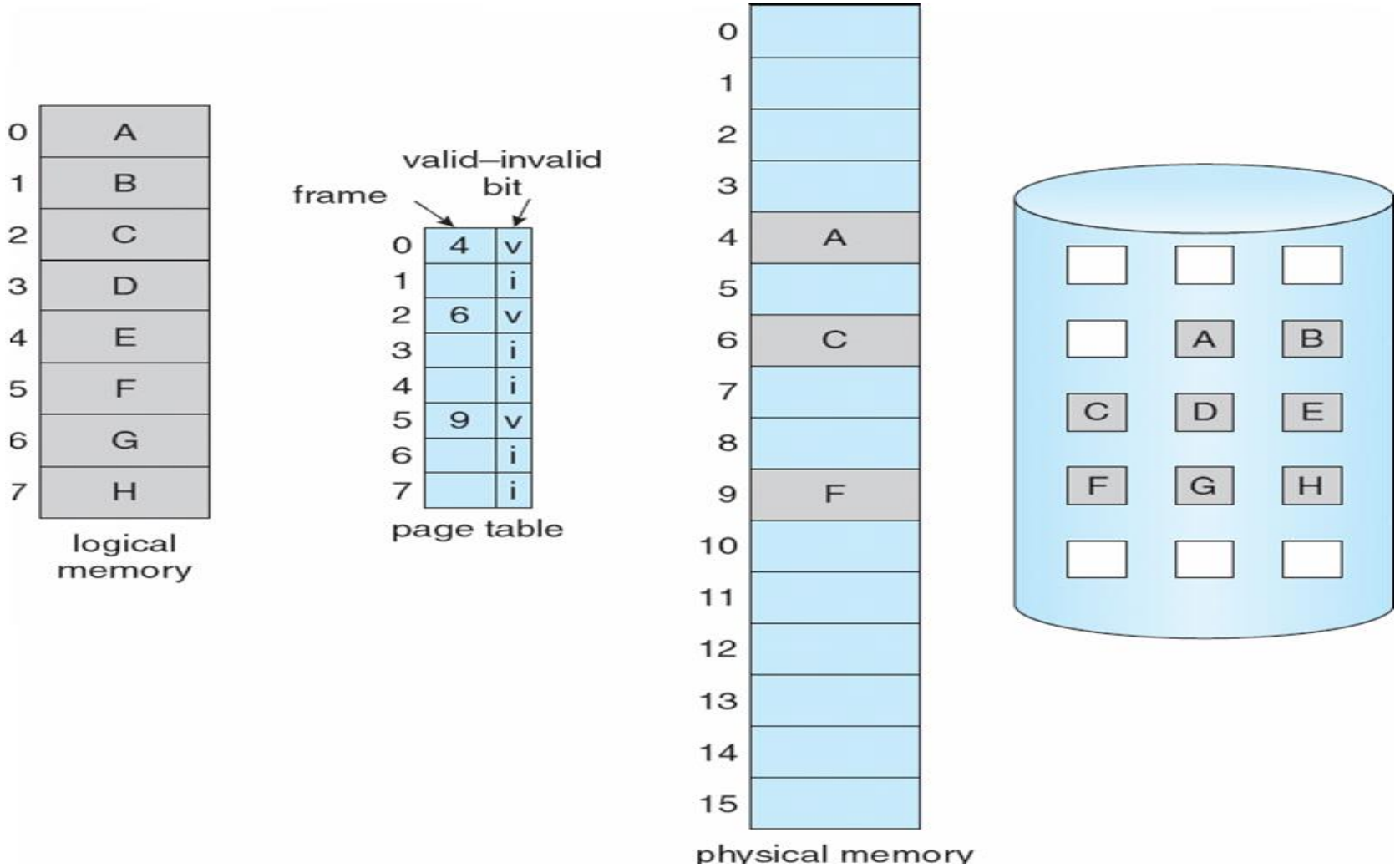
SunBeam Institute of Information & Technology, Hinjwadi, Pune & Karad.

Trainer: Akshita Chanchlani
Email: akshita.chanchlani@sunbeaminfo.com

Demand Paging

- The process of loading the page into memory on demand (whenever page fault occurs) is known as demand paging.
- Bring a page into memory only when it is needed:
  - Less I/O needed, no unnecessary I/O
  - Less memory needed
  - Faster response
  - More users
- Page is needed $\Rightarrow$ reference to it:
  - invalid reference $\Rightarrow$ abort
  - not-in-memory $\Rightarrow$ bring to memory
- Similar to paging system with swapping.
- **Lazy swapper – never swaps a page into memory unless page will be needed; Swapper that deals with pages is a pager.**

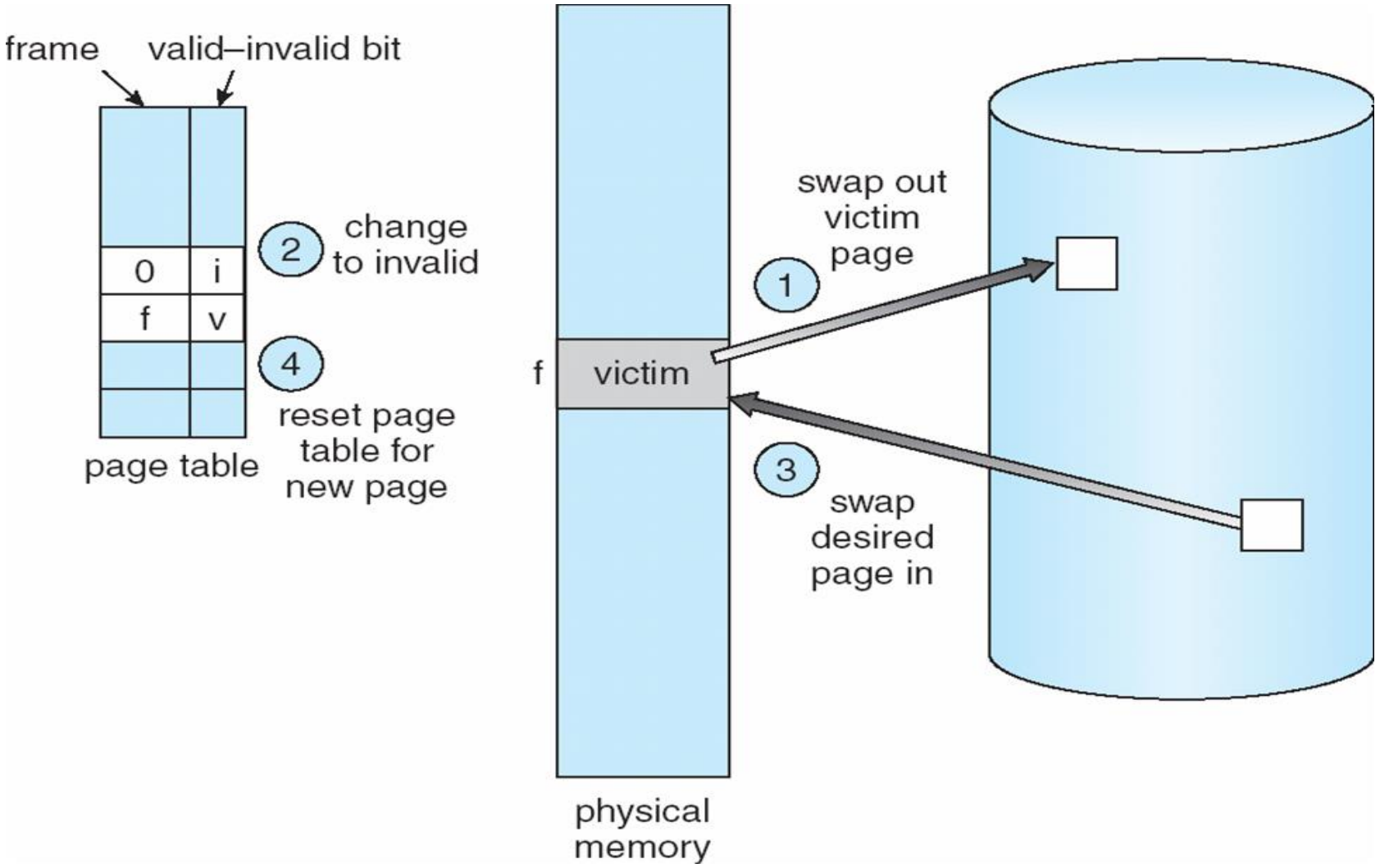
Page Table when some pages are not in Main Memory



What happens if there is no free frame?

- Page replacement – find some page in memory, but not really in use, swap it out.
- Need page replacement algorithm.
- Performance – want an algorithm which will result in minimum number of page faults.
- Same page may be brought into memory several times.

Steps in handling a Page Replacement



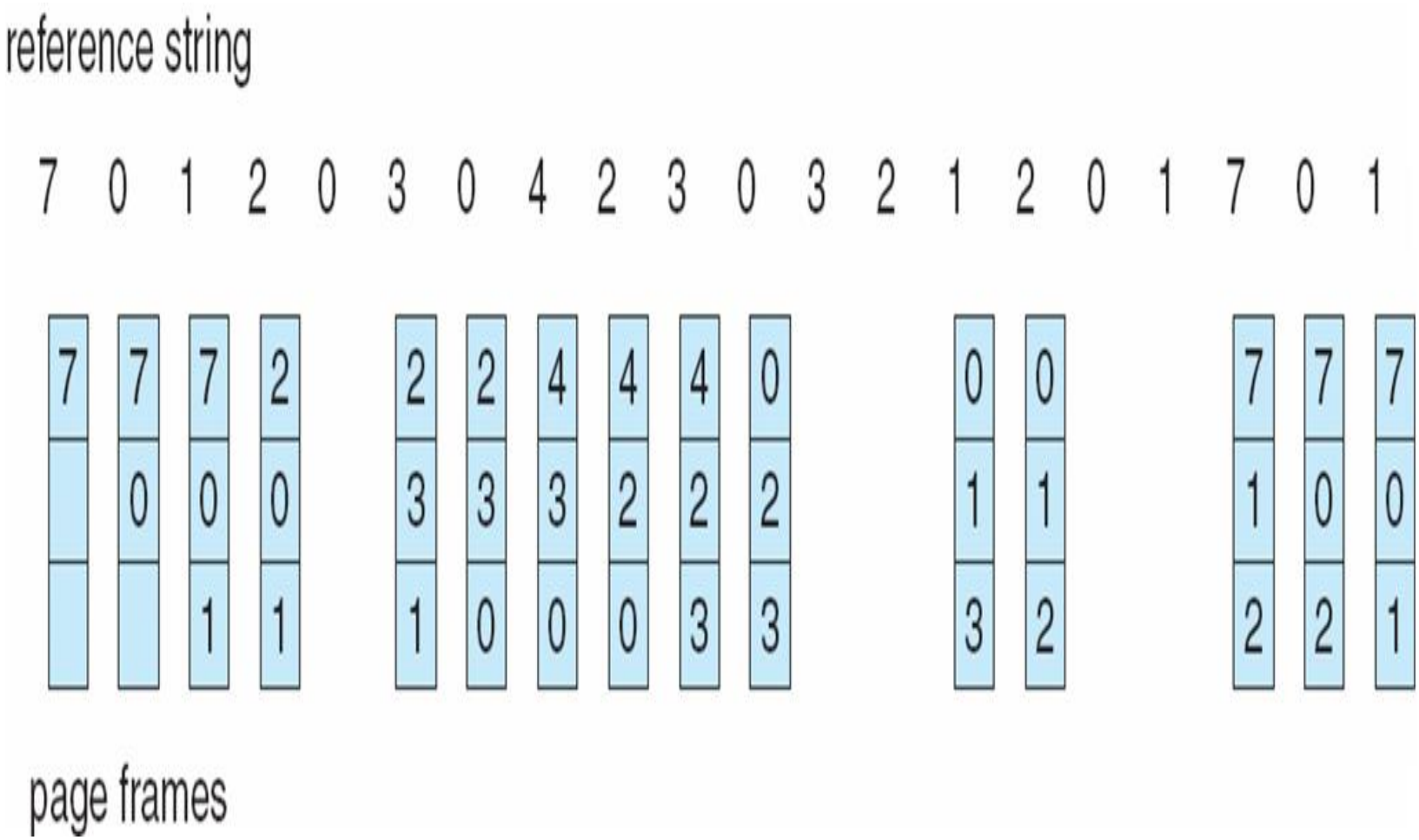
Steps in handling a Page Replacement

1. Find the location of the desired page on disk.
2. Find a free frame:
   - If there is a free frame, use it.
   - If there is no free frame, use a **page replacement algorithm to select a victim page.**
3. Bring the desired page into the (newly) free frame; update the page and frame tables.
4. Restart the process.

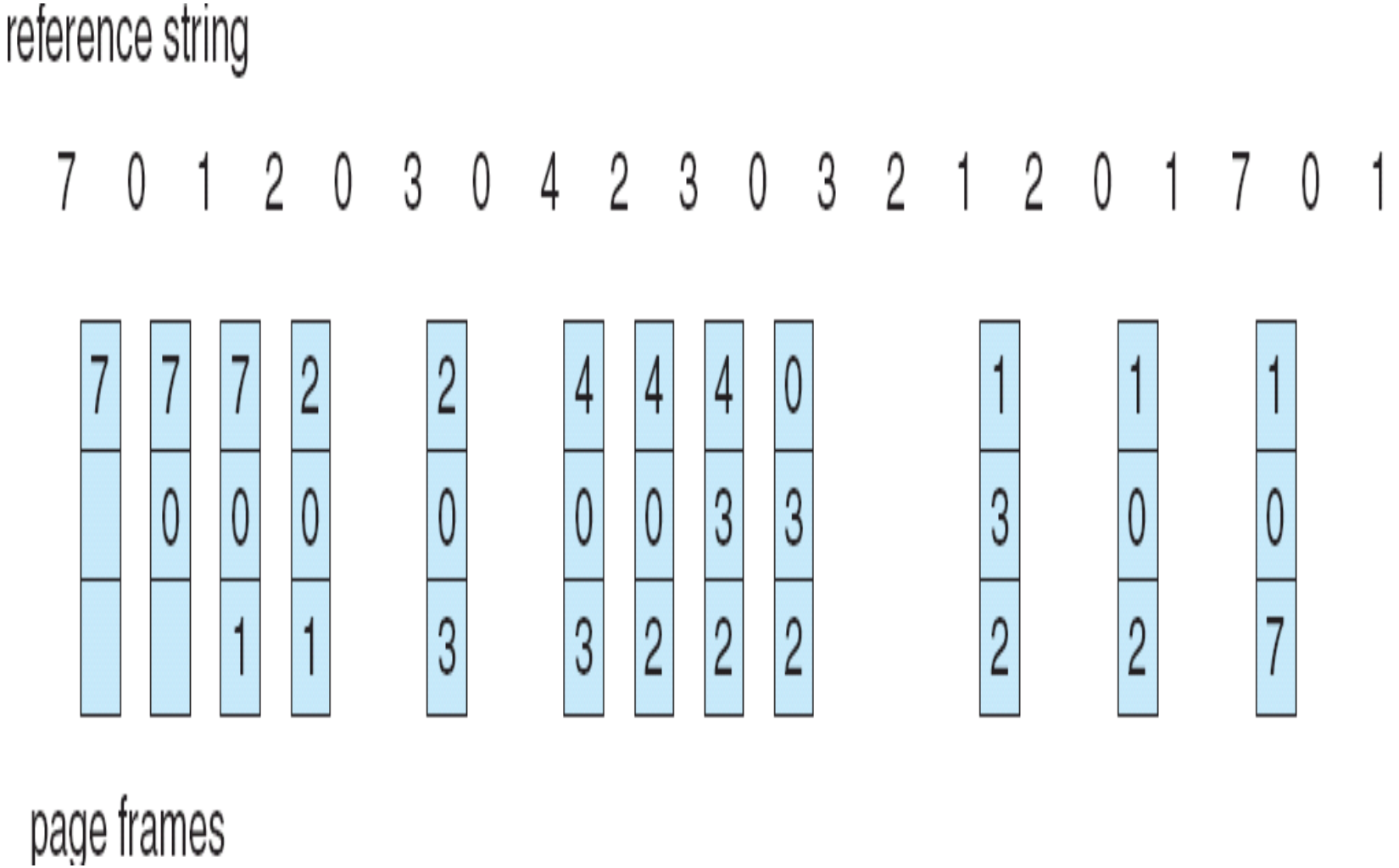
Page Replacement Algorithms

- FIFO,
- LRU,
- Optimal,

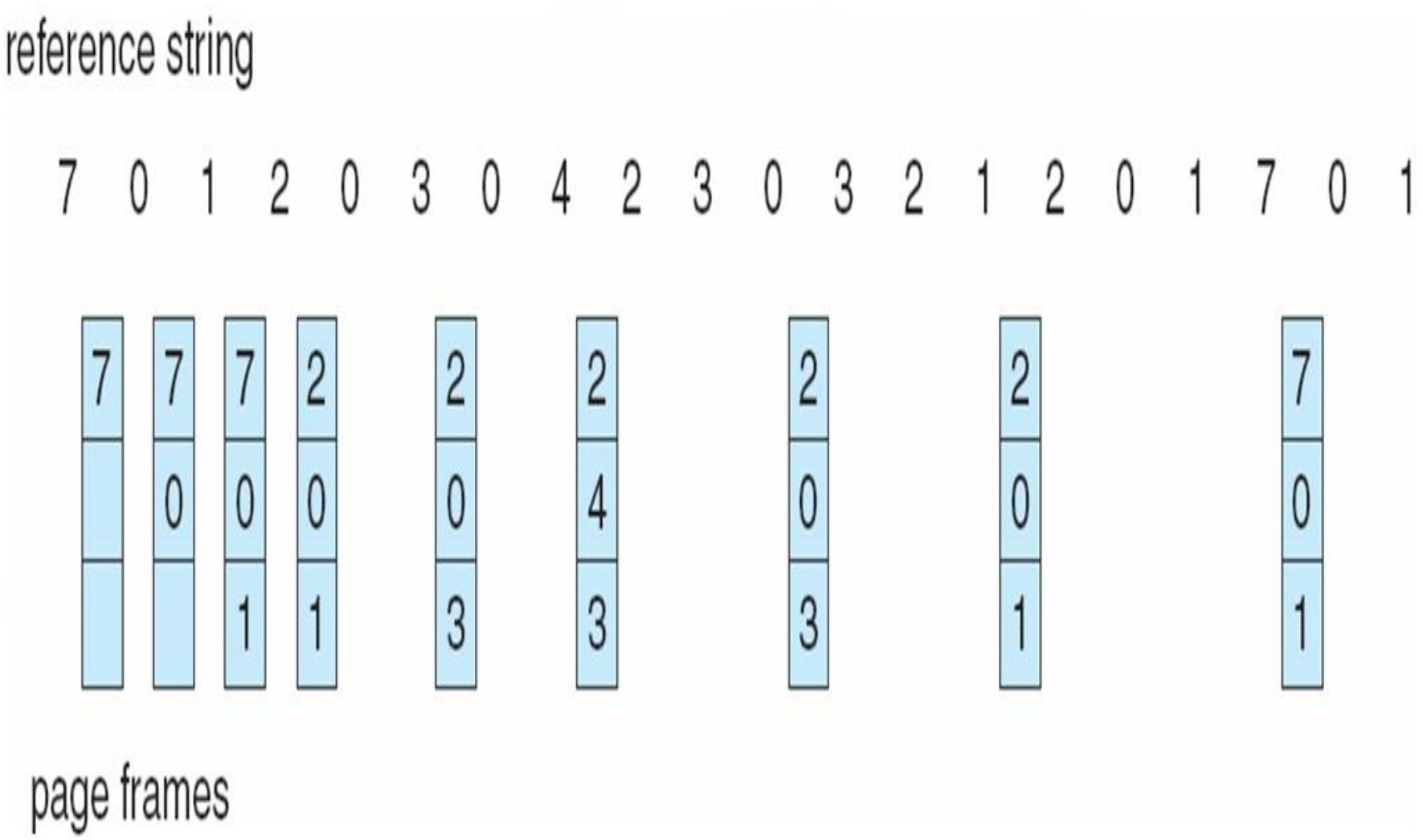
FIFO Page Replacement



LRU Page Replacement



Optimal Page Replacement



Thrashing

- If a process does not have “enough” pages, the page-fault rate is very high
  - low CPU utilization
  - OS thinks it needs increased multiprogramming
  - adds another process to system
- *Thrashing* is when a process is busy swapping pages in and out

File

- file is a collection of logically related data or information
- file is a stream of bytes/bits
- file is a basic storage unit
  - File = data+metadata
  - Data : Actual File Contents
  - Metadata : Information about file.
- File Attributes:
  - Name,type,location,size etc..
- File Operations
  - Create,delete,write,read

- **"file system“ is a way to store data onto the disk in an organized manner so that it can accessed efficiently**

- e.g. Each OS has its own filesystem like, UNIX: UFS(UNIX Filesystem), Linux: Extended filesystem ext2, ext3, ext4, Windows: FAT, NTFS etc..., MAC OSX: HFS(Hierarchical Filesystem) etc...

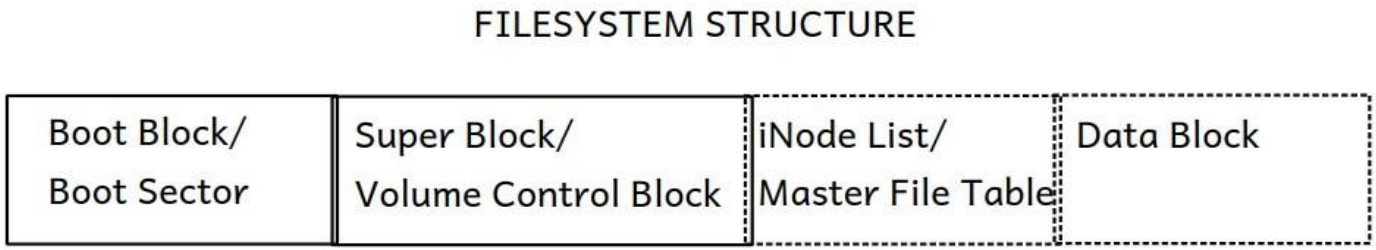
What is an inode / FCB

- An inode (index node) is a control structure that contains key information needed by the OS to access a particular file. Several file names may be associated with a single inode, but each file is controlled by exactly ONE inode.

- On the disk, there is an inode table that contains the inodes of all the files in the filesystem. When a file is opened, its inode is brought into main memory and stored in a memory-resident inode table.
- Information about the file can be kept in one structure referred as "FCB" i.e. File Control Block/iNode
  - inode/FCB contains info about the file like:
    - name of the file
    - type of the file
    - size of the file
    - parent folder location
    - access perms for user/owner, grp member and others etc

File System Structure

File system divides disk/partition logically into sectors/blocks, like **boot sector/boot block, volume control block/super block, master file table/iNode list block and data**

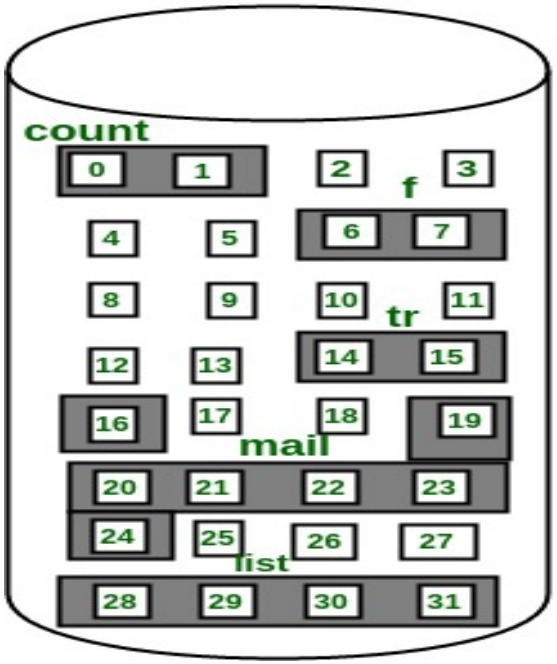


1. Boot Block: It contains information about booting the system like bootstrap program, bootloader etc...
2. Super Block: It contains information about remaining sections, like total no. of data blocks, no. of free data blocks, no. of allocated data blocks etc....
3. iNode List: It contains linked list of iNode's of all files exists on a disk.
4. Data Block: It contains actual data.

File Allocation on Disk

- Low level access methods for a file depend upon the disk allocation scheme used to store file data
  - Contiguous
  - Linked
  - Block or indexed

Contiguous Allocation



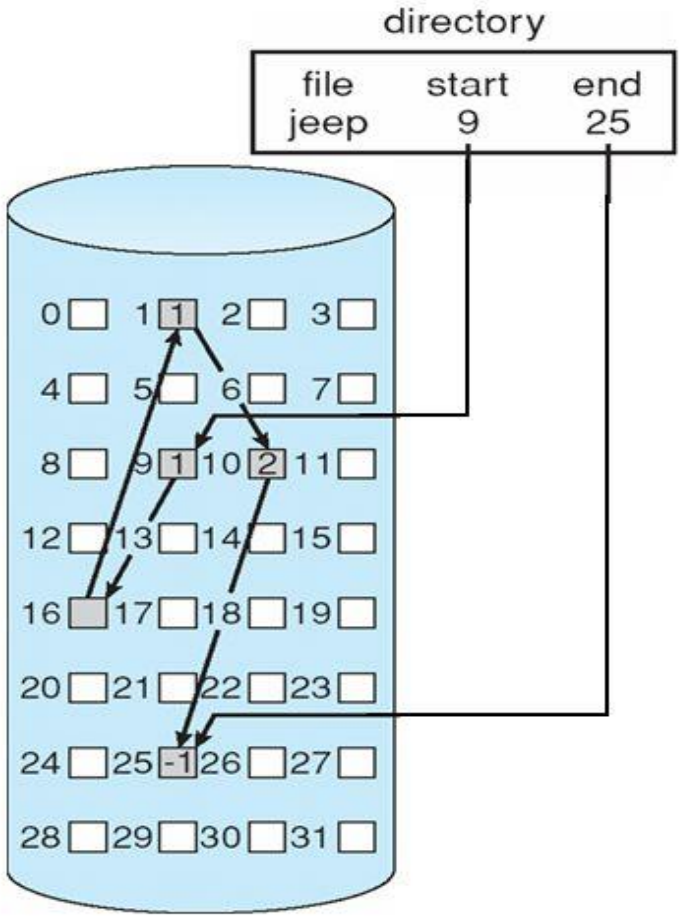
Directory

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

- File is allocated large contiguous chunks
- Expanding the file requires copying
- Dynamic storage allocation - first fit, best fit
- **External fragmentation occurs on disk**

Linked Allocation

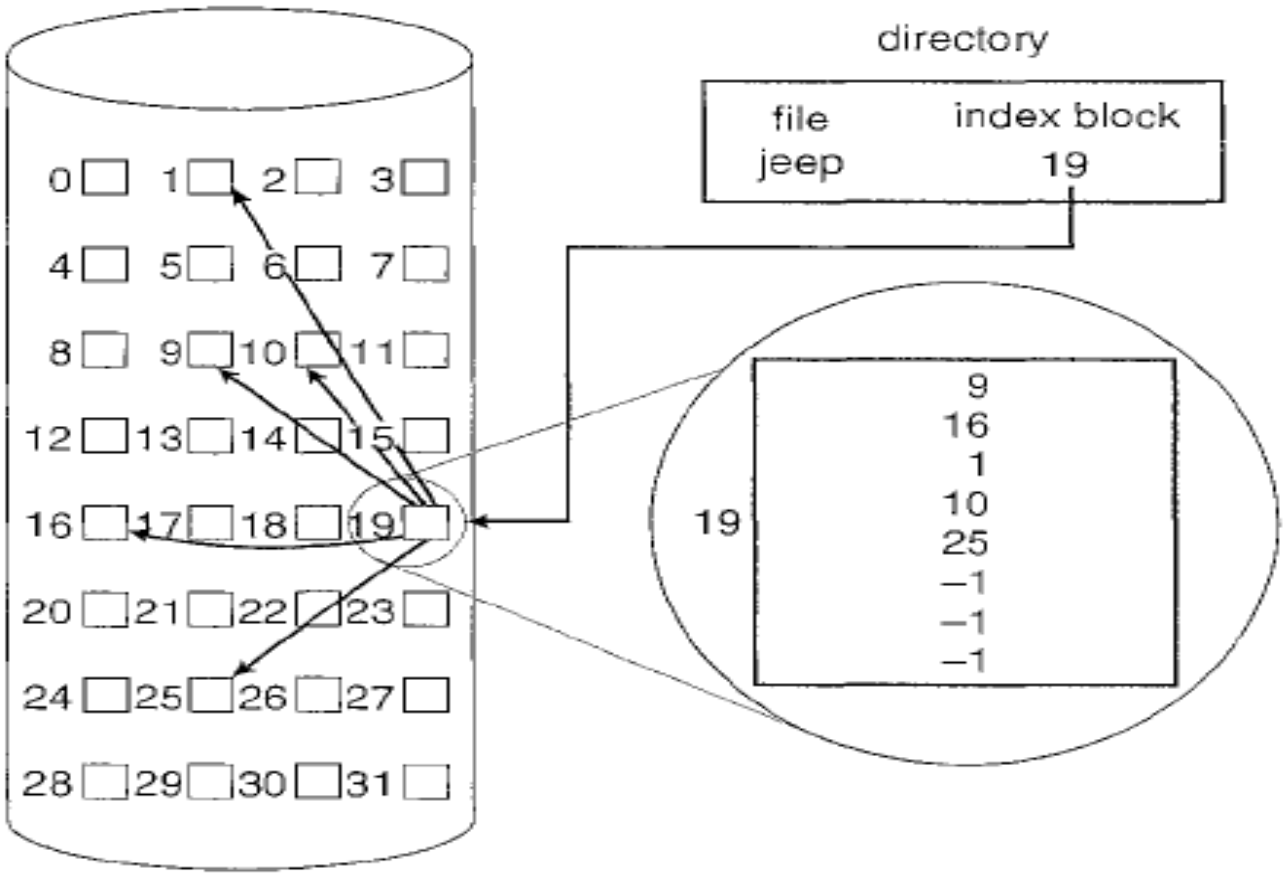
- Each file is a linked list of disk blocks, which may be scattered on the disk
- Directory contains a pointer to the first and last blocks, and each block contains a pointer to the next block
- **Advantages**:
  - No external fragmentation
  - Easy to expand the size of a file
- **Disadvantages**:
  - Not suitable for random access within a file
  - Pointers take up some disk space
  - Difficult to recover a file if a pointer is lost or damaged
- Blocks may be collected into **clusters** of several blocks
  - Fewer pointers are necessary
  - Fewer disk seeks to read an entire file
  - Greater internal fragmentation



Block / Indexed

- A special block known as the **Index block** contains the pointers to all the blocks occupied by a file.
- The ith entry in the index block contains the disk address of the ith file block.
- The directory entry contains the address of the index block.
- When the file is created, all pointers in the index block are set to *nil.*
- *When* the i^{th} block is first written, a block is obtained from the free-space manager and its address is put in the i^{th} index-block entry.

Indexed Allocation



Disk Scheduling

- **Disk scheduling** is done by operating systems to schedule I/O requests arriving for the disk.
- Disk scheduling is important because:
  - Multiple I/O requests may arrive by different processes and only one I/O request can be served at a time by the disk controller. Thus other I/O requests need to wait in the waiting queue and need to be scheduled.
  - Two or more request may be far from each other so can result in greater disk arm movement.
  - Hard drives are one of the slowest parts of the computer system and thus need to be accessed in an efficient manner.

Disk Scheduling Criteria

Seek Time

- Seek time is the time taken to locate the disk arm to a specified track where the data is to be read or write.
- minimum average seek time is better.

Rotational Latency

- The time taken by the desired sector of disk to rotate into a position so that it can access the read/write heads.
- minimum rotational latency is better.

Transfer Time

- The time to transfer the data.
- It depends on the rotating speed of the disk and number of bytes to be transferred.

Disk Access Time

- **SeekTime+Rotational Latency +Transfer Time**

Disk Response Time

- The average of time spent by a request waiting to perform its I/O operation.
- minimum variance response time is better.

Disk Scheduling Algorithms

First Come First Serve

- FCFS, the requests are addressed in the order they arrive in the disk queue.

Shortest Seek Time First

- SSTF (Shortest Seek Time First), requests having shortest seek time are executed first.
- it decreases the average response time and increases the throughput of system.

Scan/Elevator

- SCAN algorithm the disk arm moves into a particular direction and services the requests coming in its path and after reaching the end of disk, it reverses its direction and again services the request arriving in its path.

CSCAN

- disk arm moves in a circular fashion and this algorithm is also similar to SCAN algorithm and hence it is known as C-SCAN (Circular SCAN).

Look

- similar to the SCAN disk scheduling algorithm except for the difference that the disk arm in spite of going to the end of the disk goes only to the last request to be serviced in front of the head and then reverses its direction from there only.

Clook

- CLOOK, the disk arm in spite of going to the end goes only to the last request to be serviced in front of the head and then from there goes to the other end's last request.

FCFS(First Come First Serve)

a request queue (0-199)

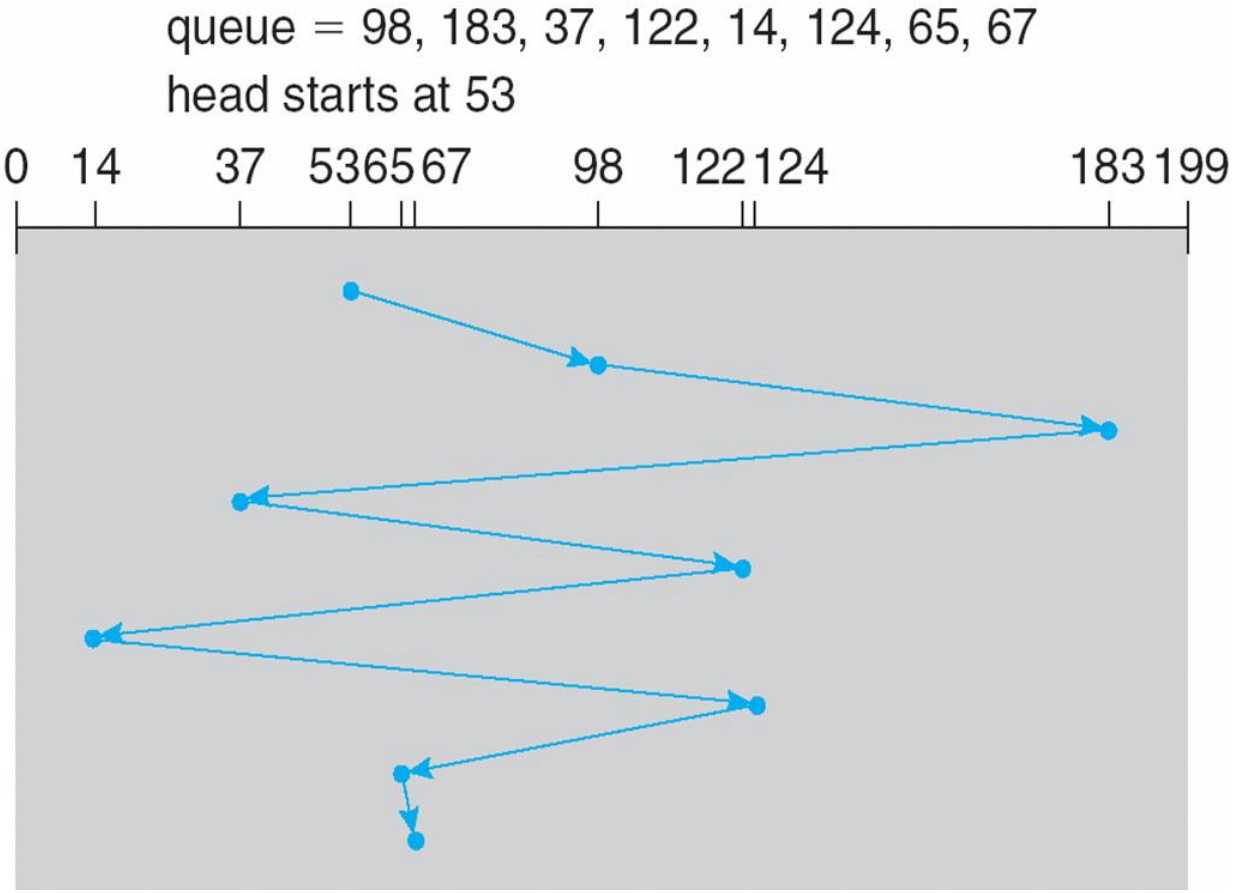


Illustration shows total head movement of 640 cylinders.

SSTF(Shortest Seek Time First)

- Selects the request with the minimum seek time from the current head position
- SSTF scheduling is a form of SJF scheduling; may cause starvation of some requests

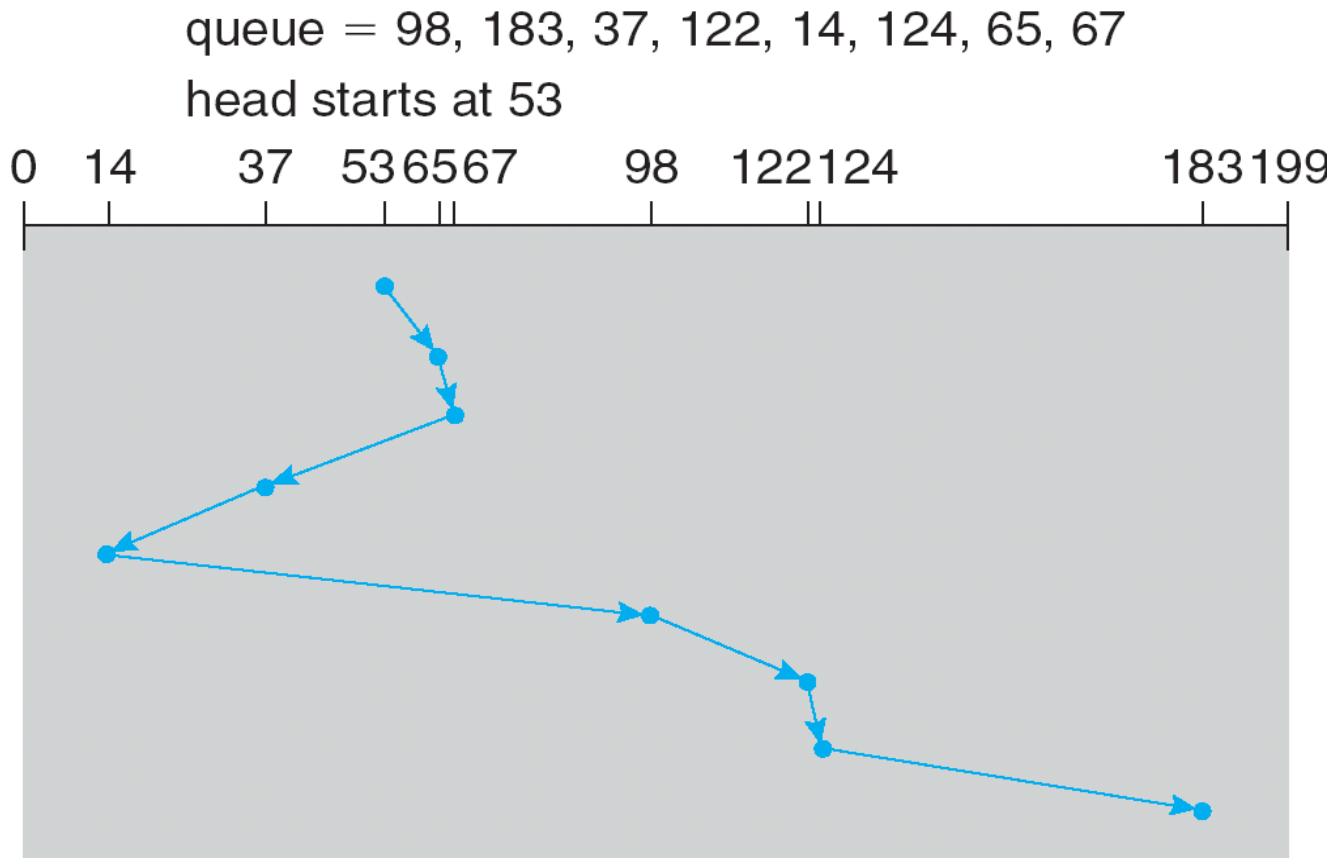


Illustration shows total head movement of 236 cylinders.

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.

•SCAN algorithm Sometimes called the elevator algorithm

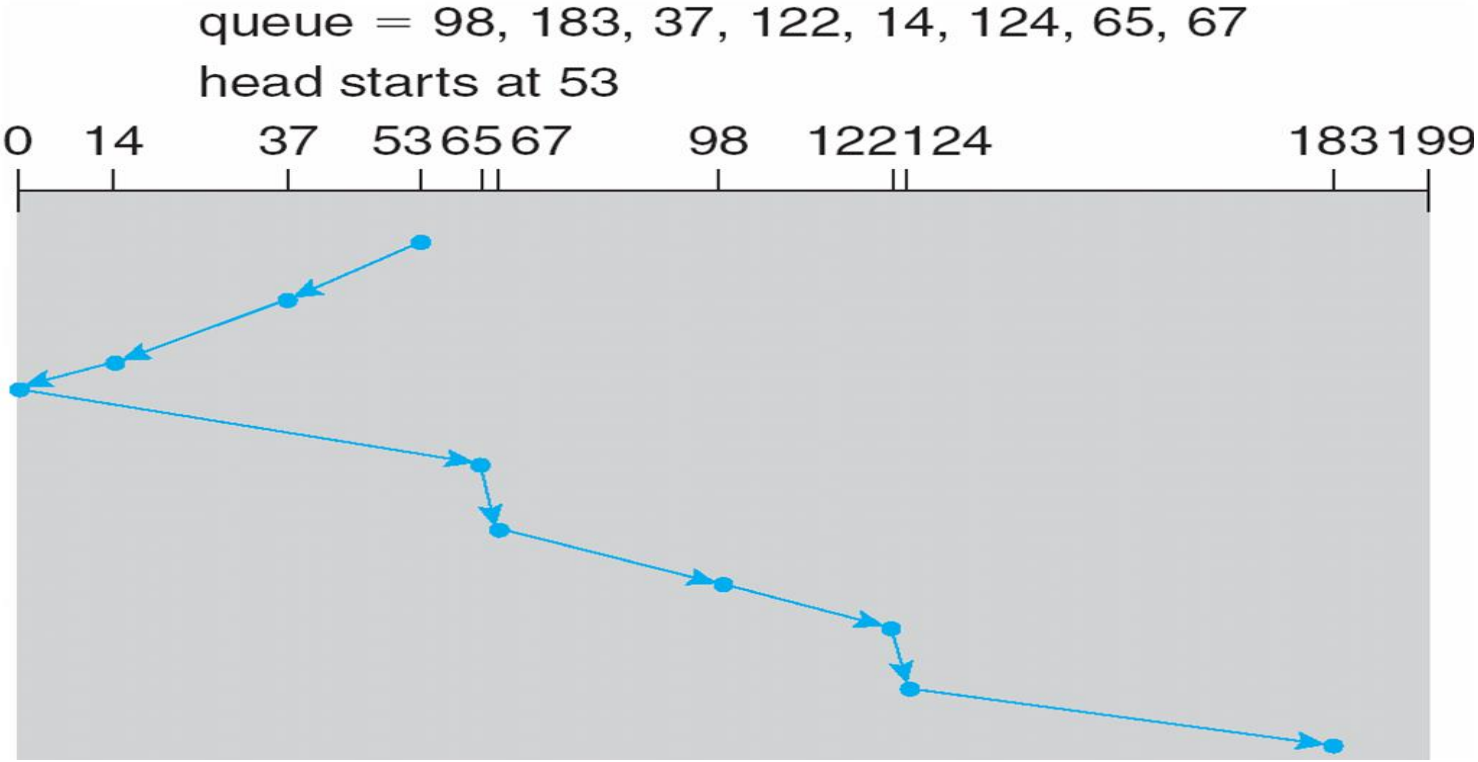
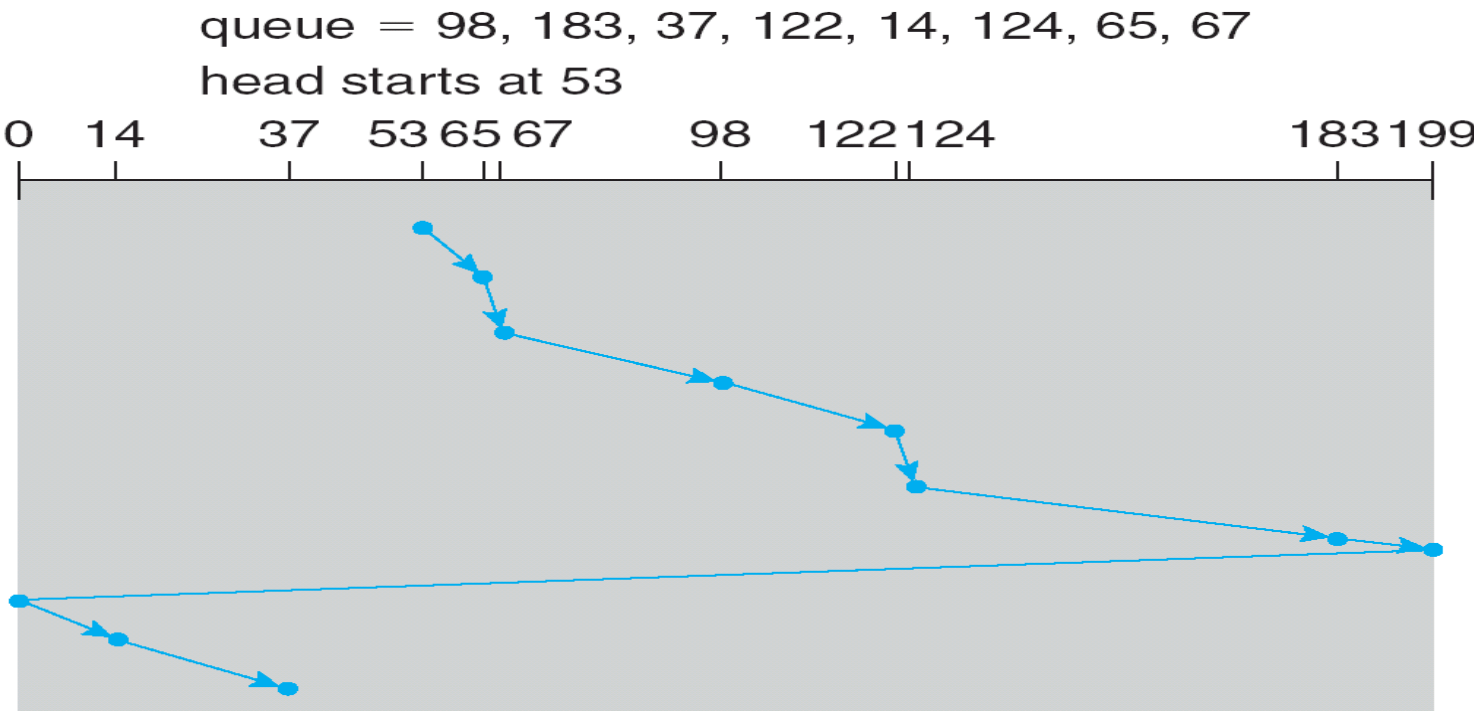


Illustration shows total head movement of 208 cylinders.

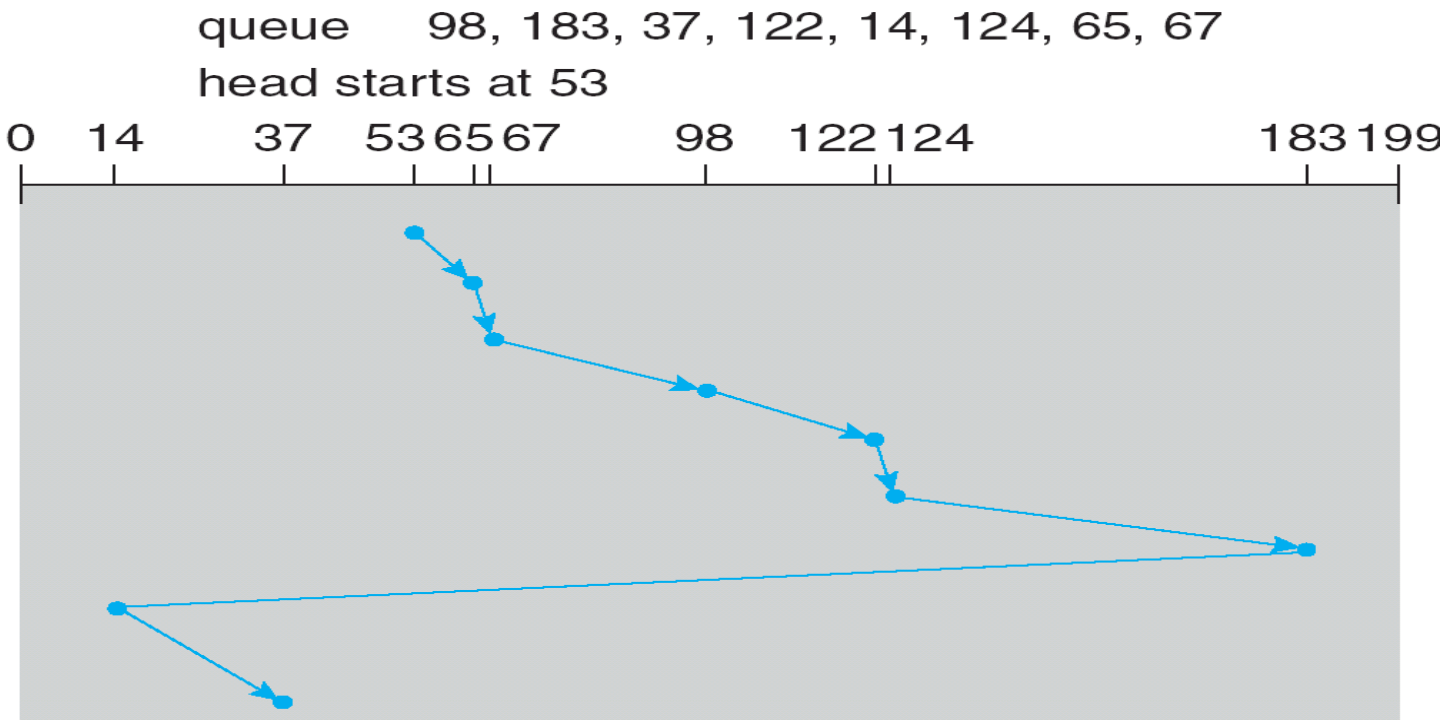
C-SCAN (Cont)

- Provides a more uniform wait time than SCAN
- 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-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



Thank You