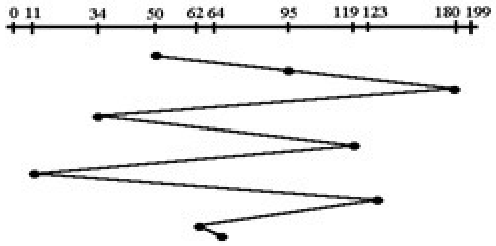
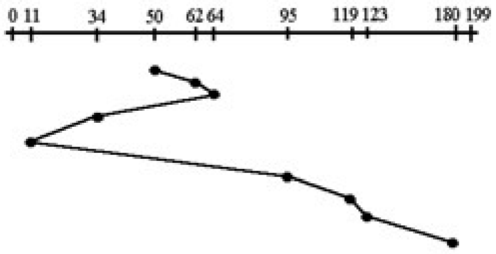
UNIT-5

Disk Scheduling Algorithms

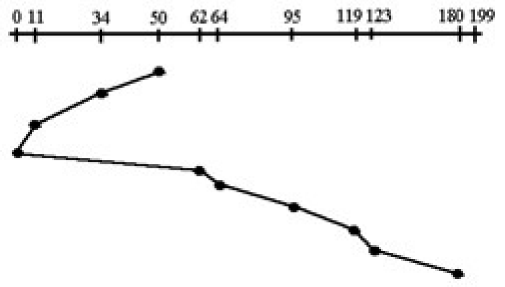
1. **FCFS:** FCFS is the simplest of all the Disk Scheduling Algorithms. In FCFS, the requests are addressed in the order they arrive in the disk queue.
Example: Given the following queue -- 95, 180, 34, 119, 11, 123, 62, 64 with the Read-write head initially at the track 50 and the tail track being at 199.



2. **SSTF:** In SSTF (Shortest Seek Time First), requests having shortest seek time are executed first. So, the seek time of every request is calculated in advance in the queue and then they are scheduled according to their calculated seek time. As a result, the request near the disk arm will get executed first. SSTF is certainly an improvement over FCFS as it decreases the average response time and increases the throughput of system.
Example:

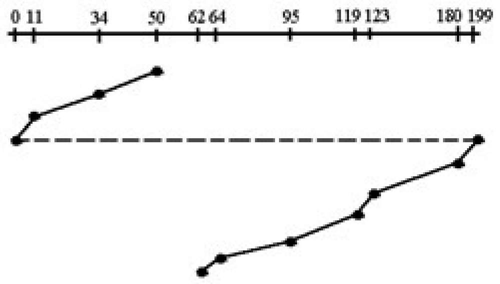


3. **SCAN:** In SCAN algorithm the disk arm moves into a particular direction and services the requests coming in its path and after reaching the end of disk, it reverses its direction and again services the request arriving in its path. So, this algorithm works as an elevator and hence also known as **elevator algorithm.** As a result, the requests at the midrange are serviced more and those arriving behind the disk arm will have to wait.
Example:



4. **CSCAN**: In SCAN algorithm, the disk arm again scans the path that has been scanned, after reversing its direction. So, it may be possible that too many requests are waiting at the other end or there may be zero or few requests pending at the scanned area.These situations are avoided in *CSAN* algorithm in which the disk arm instead of reversing its direction goes to the other end of the disk and starts servicing the requests from there. So, the 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).

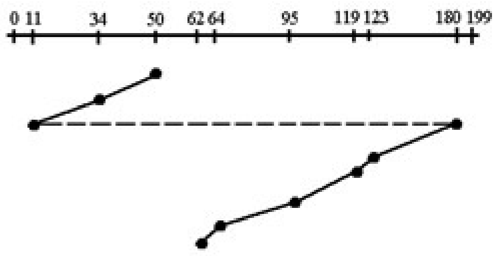
   Example:



5. **LOOK:** It is 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. Thus it prevents the extra delay which occurred due to unnecessary traversal to the end of the disk.

6. **CLOOK:** As LOOK is similar to SCAN algorithm, in similar way, CLOOK is similar to CSCAN disk scheduling algorithm. In 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. Thus, it also prevents the extra delay which occurred due to unnecessary traversal to the end of the disk.

   Example:



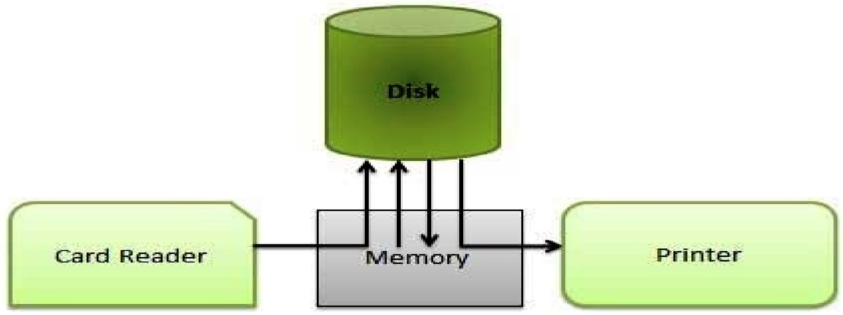
Kernel I/O Subsystem

Kernel I/O Subsystem is responsible to provide many services related to I/O. Following are some of the services provided.

- **Scheduling** − Kernel schedules a set of I/O requests to determine a good order in which to execute them. When an application issues a blocking I/O system call, the request is placed on the queue for that device. The Kernel I/O scheduler rearranges the order of the queue to improve the overall system efficiency and the average response time experienced by the applications.
- **Buffering** − Kernel I/O Subsystem maintains a memory area known as **buffer** that stores data while they are transferred between two devices or between a device with an application operation. Buffering is done to cope with a speed mismatch between the producer and consumer of a data stream or to adapt between devices that have different data transfer sizes.
- **Caching** − Kernel maintains cache memory which is region of fast memory that holds copies of data. Access to the cached copy is more efficient than access to the original.
- **Spooling and Device Reservation** − A spool is a buffer that holds output for a device, such as a printer, that cannot accept interleaved data streams. The spooling system copies the queued spool files to the printer one at a time.
- **Error Handling** − an operating system that uses protected memory can guard against many kinds of hardware and application errors.

Spooling

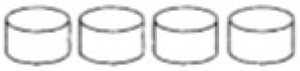
- ❖ Spooling refers to simultaneous peripheral operations online.
- ❖ A spool is a buffer that holds output for a device, such as a printer, that cannot accept interleaved data streams.
- ❖ In spooling data is first onto the disk and then CPU interacts with disk via main memory.
- ❖ Spooling is capable of overlapping I/O operations for one job with CPU operations of other jobs.
  Example: using printer



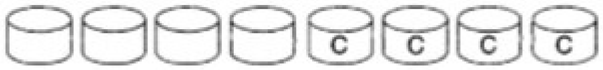
RAID/ Redundant Array of Independent Disks

RAID is the way of combining several independent and relatively small disks into a single storage of a large size. The disks included into the array are called array members. RAID storage uses multiple disks in order to provide fault tolerance, to improve overall performance, and to increase storage capacity in a system. This is in contrast with older storage devices that used only a single disk drive to store data. The disks can be combined into the array in different ways which are known as **RAID levels**. Each of RAID levels has its own characteristics of:

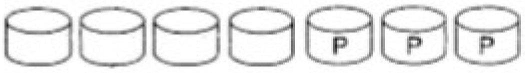
Standard RAID Levels RAID devices use much different architecture, called levels, depending on the desired balance between performance and fault tolerance. RAID levels describe how data is distributed across the drives. Standard RAID levels include the following:



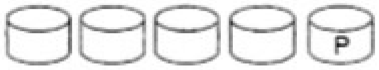
(a) RAID 0: non-redundant striping.



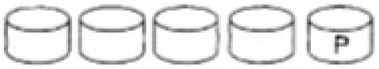
(b) RAID 1: mirrored disks.



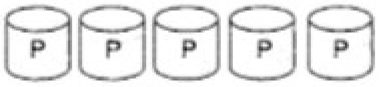
(c) RAID 2: memory-style error-correcting codes.



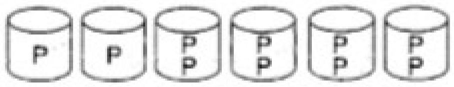
(d) RAID 3: bit-interleaved parity.



(e) RAID 4: block-interleaved parity.



(f) RAID 5: block-interleaved distributed parity.



(g) RAID 6: P + Q redundancy.

RAID levels.

Level 0: Striped disk array without fault tolerance

- Provides data striping (spreading out blocks of each file across multiple disk drives) but no redundancy. This improves performance but does not deliver fault tolerance. If one drive fails then all data in the array is lost.

Level 1: Mirroring and duplexing

- Provides disk mirroring. Level 1 provides twice the read transaction rate of single disks and the same write transaction rate as single disks.

Level 2: Error-correcting coding

- Not a typical implementation and rarely used, Level 2 stripes data at the bit level rather than the block level.

Level 3: Bit-interleaved parity

- Provides byte-level striping with a dedicated parity disk. Level 3, which cannot service simultaneous multiple requests, also is rarely used.

Level 4: Dedicated parity drive

- A commonly used implementation of RAID, Level 4 provides block-level striping (like Level 0) with a parity disk. If a data disk fails, the parity data is used to create a replacement disk. A disadvantage to Level 4 is that the parity disk can create write bottlenecks.

Level 5: Block interleaved distributed parity

- Provides data striping at the byte level and also stripe error correction information. This results in excellent performance and good fault tolerance. Level 5 is one of the most popular implementations of RAID.

Level 6: Independent data disks with double parity

- Provides block-level striping with parity data distributed across all disks.

Level 10: A stripe of mirrors

- Not one of the original RAID levels, multiple RAID 1 mirrors are created, and a RAID 0 stripe is created over these.

<u>Problems with RAID</u>: RAID does not always assure that data are available for the operating system and its users.

File Systems

File:A file is a collection of related information that is recorded on secondary storage. Or file is a collection of logically related entities. From user's perspective a file is the smallest allotment of logical secondary storage.

File Access Methods: The way that files are accessed and read into memory is determined by Access methods. Usually a single access method is supported by systems while there are OS's that support multiple access methods.

1. Sequential Access

- Data is accessed one record right after another is an order.
- Read command cause a pointer to be moved ahead by one.
- Write command allocate space for the record and move the pointer to the new End Of File.
- Such a method is reasonable for tape.

2. Direct Access

- This method is useful for disks.
- The file is viewed as a numbered sequence of blocks or records.
- There are no restrictions on which blocks are read/written; it can be done in any order.
- User now says "read n" rather than "read next".

***3.* Indexed Sequential Access**

- It is built on top of Sequential access.
- It uses an Index to control the pointer while accessing files.

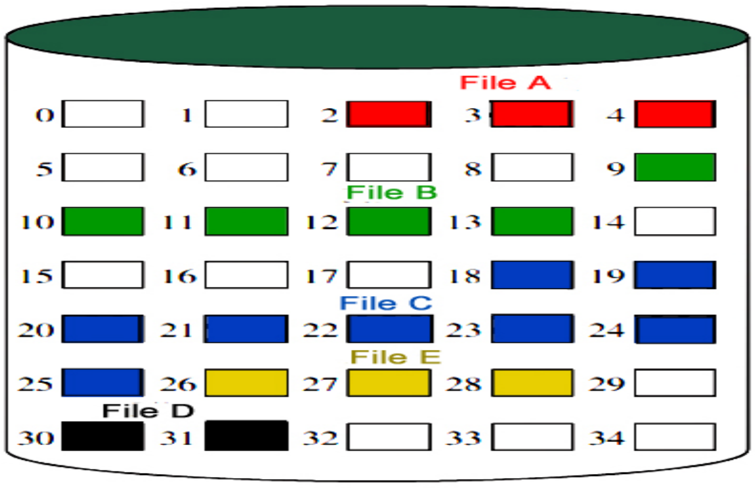
Directory: Information about files is maintained by Directories. A directory can contain multiple files. It can even have directories inside of them. In Windows we also call these directories as folders.

Following is the information maintained in a directory:

Name: The name visible to user.
Type: Type of the directory.
Location: Device and location on the device where the file header is located.
Size: Number of bytes/words/blocks in the file.
Position: Current next-read/next-write pointers.
Protection: Access control on read/write/execute/delete.
Usage: Time of creation, access, modification etc.
Mounting: When the root of one file system is "grafted" into the existing tree of another file system its called Mounting.

FILE Organization

1. Continuous Allocation: A single continuous set of blocks is allocated to a file at the time of file creation. Thus, this is a pre-allocation strategy, using variable size portions. The file allocation table needs just a single entry for each file, showing the starting block and the length of the file. This method is 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. For example, if a file starts at block b, and the ith block of the file is wanted, its location on secondary storage is simply b+i-1.



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

Disadvantage

- ❖ External fragmentation may occur.
- ❖ Also, with pre-allocation, it is necessary to declare the size of the file at the time of creation.

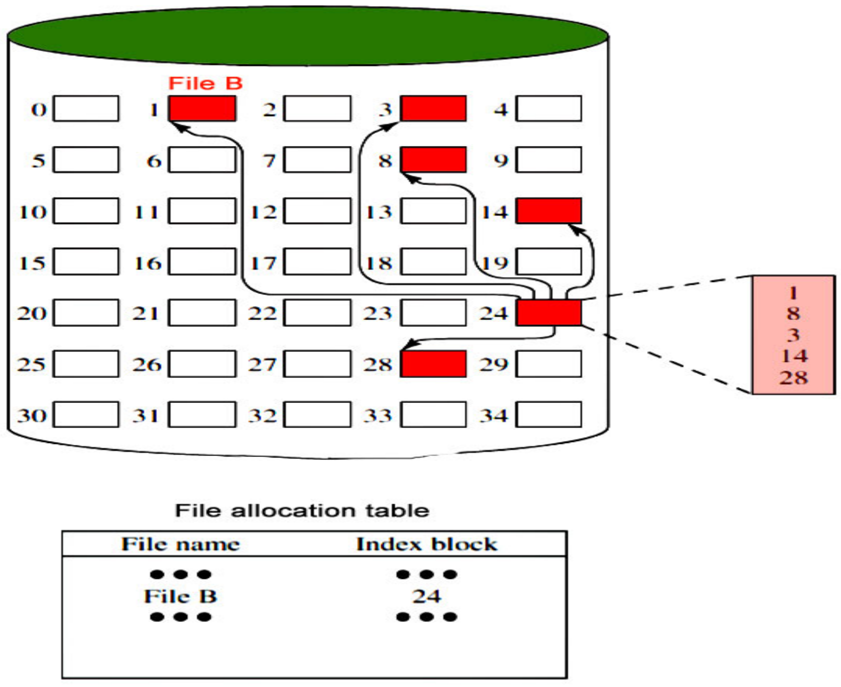
2. Linked Allocation (Non-contiguous allocation): Allocation is on an individual block basis. Each block contains a pointer to the next block in the chain. Again the file table needs just a single entry for each file, showing the starting block and the length of the file. Although pre-allocation is possible, it is more common simply to allocate blocks as needed. Any free block can be added to the chain. The blocks need not be continuous. Increase in file size is always possible if free disk block is available. There is no external fragmentation because only one block at a time is needed but there can be internal fragmentation but it exists only in the last disk block of file.

Disadvantage:

- ❖ Internal fragmentation exists in last disk block of file.
- ❖ There is an overhead of maintaining the pointer in every disk block.
- ❖ If the pointer of any disk block is lost, the file will be truncated.
- ❖ It supports only the sequential access of files.

3. Indexed Allocation:

It addresses many of the problems of contiguous and chained allocation. In this case, the file allocation table contains a separate one-level index for each file: The index has one entry for each block allocated to the file. Allocation may be on the basis of fixed-size blocks or variable-sized blocks. Allocation by blocks eliminates external fragmentation, whereas allocation by variable-size blocks improves locality. This allocation technique supports both sequential and direct access to the file and thus is the most popular form of file allocation.



File sharing

File sharing is the public or private sharing of computer data or space in a network with various levels of access privilege. While files can easily be shared outside a network (for example, simply by handing or mailing someone your file on a diskette), the term *file sharing* almost always means sharing files in a network, even if in a small local area network. File sharing allows a number of people to use the same file or file by some combination of being able to read or view it, write to or modify it, copy it, or print it. Typically, a file sharing system has one or more administrators. Users may all have the same or may have different levels of access privilege. File sharing can also mean having an allocated amount of personal file storage in a common file system.

Protection and Security Methods

The different methods that may provide protect and securities for different computer systems are:

Authentication

This deals with identifying each user in the system and making sure they are who they claim to be. The operating system makes sure that all the users are authenticated before they access the system. The different ways to make sure that the users are authentic are:

- **Username/ Password**
  Each user has a distinct username and password combination and they need to enter it correctly before they can access the system.
- **User Key/ User Card**
  The users need to punch a card into the card slot or use the individual key on a keypad to access the system.
- **User Attribute Identification**

Different user attribute identifications that can be used are fingerprint, eye retina etc. These are unique for each user and are compared with the existing samples in the database. The user can only access the system if there is a match.

One Time Password

These passwords provide a lot of security for authentication purposes. A one time password can be generated exclusively for a login every time a user wants to enter the system. It cannot be used more than once. The various ways a one time password can be implemented are:

- **Random Numbers**
  The system can ask for numbers that correspond to alphabets that are pre arranged. This combination can be changed each time a login is required.
- **Secret Key**
  A hardware device can create a secret key related to the user id for login. This key can change each time.

File system Implementation

Numerous on-disk and in-memory configurations and structures are being used for implementing a file system. These structures differ based on the operating system and the file system but applying following general principles.

- A **boot control block** usually contains the information required by the system for booting an operating system from that volume. When the disks do not contain any operating system, this block can be treated as empty. This is typically the first chunk of a volume.
- A **volume control block** holds volume or the partition details, such as the number of blocks in the partition, size of the blocks or chunks, free-block count along with free-block pointers.
- A **directory structure per file system** is required for organizing the files.
- The **FCB** contains many details regarding any file which includes file permissions, ownership; the size of file and location of data blocks.