UNIT-4

File System

File Concept :

Computers can store information on various storage media such as, magnetic disks, magnetic tapes, optical disks. The physical storage is converted into a logical storage unit by operating system. The logical storage unit is called FILE. A file is a collection of similar records. A record is a collection of related fields that can be treated as a unit by some application program. A field is some basic element of data. Any individual field contains a single value. A data base is collection of related data.

Student Name	Marks in sub1	Marks in sub2	Fail/Pass
KUMAR	85	86	P
LAKSHMI	93	92	P

DATA FILE

Student name, Marks in sub1, sub2, Fail/Pass are fields.

The collection of fields is called a RECORD.

RECORD:

LAKSHMI	93	92	P

Collection of these records is called a data file.

FILE ATTRIBUTES :

1. Name : A file is named for the convenience of the user and is referred by its name. A name is usually a string of characters.
2. Identifier : This unique tag, usually a number ,identifies the file within the file system.
3. Type : Files are of so many types. The type depends on the extension of the file.

   Example: .exe→Executable file

   .obj→Object file

   .src→Source file

4. Location : This information is a pointer to a device and to the location of the file on that device.
5. Size : The current size of the file (in bytes, words, blocks).
6. Protection : Access control information determines who can do reading, writing, executing and so on.
7. Time, Date, User identification : This information may be kept for creation, last modification, last use.

FILE OPERATIONS

1. Creating a file : Two steps are needed to create a file. They are :
   - *Check whether the space is available or not.*
   - *If the space is available then made an entry for the new file in the directory. The entry includes name of the file, path of the file, etc…*
2. Writing a file : To write a file, we have to know 2 things. One is name of the file and second is the information or data to be written on the file, the system searches the entired given location for the file. If the file is found, the system must keep a write pointer to the location in the file where the next write is to take place.
3. Reading a file : To read a file, first of all we search the directories for the file, if the file is found, the system needs to keep a read pointer to the location in the file where the next read is to take place. Once the read has taken place, the read pointer is updated.
4. Repositioning within a file : The directory is searched for the appropriate entry and the current file position pointer is repositioned to a given value. This operation is also called file seek.
5. Deleting a file : To delete a file, first of all search the directory for named file, then released the file space and erase the directory entry.
6. Truncating a file : To truncate a file, remove the file contents only but, the attributes are as it is.

FILE TYPES: The name of the file split into 2 parts. One is name and second is Extension. The file type is depending on extension of the file.

File Type	Extension	Purpose
Executable	.exe .com .bin	Ready to run (or) ready to run machine language program.
Source code	.c .cpp .asm	Source code in various languages.
Object	.obj .o	Compiled, machine language, not linked.
Batch	.bat .sh	Commands to the command interpreter
Text	.txt .doc	Textual data, documents
Word processor	.doc .wp .rtf	Various word processor formats
Library	.lib .dll	Libraries of routines for programmers.
Print or View	.pdf .jpg	Binary file in a format for printing or viewing.
Archive	.arc .zip	Related files grouped into a file.
Multimedia	.mpeg .mp3 .avi	Binary file containing audio or audio/video information

FILE STRUCTURE

File types also can be used to indicate the internal structure of the file. The operating system requires that an executable file have a specific structure so that it can determine where in memory to load the file and what the location of the first instruction is. If OS supports multiple file structures, the resulting size of OS is large. If the OS defines 5 different file structures, it needs to contain the code to support these file structures. All OS must support at least one structure that of an executable file so that the system is able to load and run programs.

INTERNAL FILE STRUCTURE

In UNIX OS, defines all files to be simply stream of bytes. Each byte is individually addressable by its offset from the beginning or end of the file. In this case, the logical record size is 1 byte. The file system automatically packs and unpacks bytes into physical disk blocks, say 512 bytes per block.

The logical record size, physical block size, packing determine how many logical records are in each physical block. The packing can be done by the user's application program or OS. A file may be considered a sequence of blocks. If each block were 512 bytes, a file of 1949 bytes would be allocated 4 blocks(2048 bytes). The last 99 bytes would be wasted. It is called internal fragmentation all file systems suffer from internal fragmentation,the larger the block size, the greater the internal fragmentation.

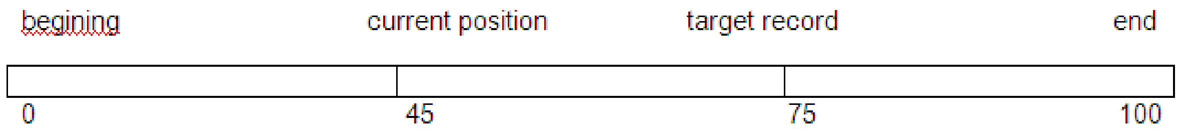
FILE ACCESS METHODS

Files stores information, this information must be accessed and read into computer memory. There are so many ways that the information in the file can be accessed.

1. Sequential file access :

Information in the file is processed in order i.e. one record after the other. Magnetic tapes are supporting this type of file accessing.

Eg : A file consisting of 100 records, the current position of read/write head is 45th record, suppose we want to read the 75th record then, it access sequentially from 45, 46, 47 …….. 74, 75. So the read/write head traverse all the records between 45 to 75.



2. Direct access :

Direct access is also called relative access. Here records can read/write randomly without any order. The direct access method is based on a disk model of a file, because disks allow random access to any file block.

Eg : A disk containing of 256 blocks, the position of read/write head is at 95th block. The block is to be read or write is 250th block. Then we can access the 250th block directly without any restrictions.

Eg : CD consists of 10 songs, at present we are listening song 3, If we want to listen song 10, we can shift to 10.

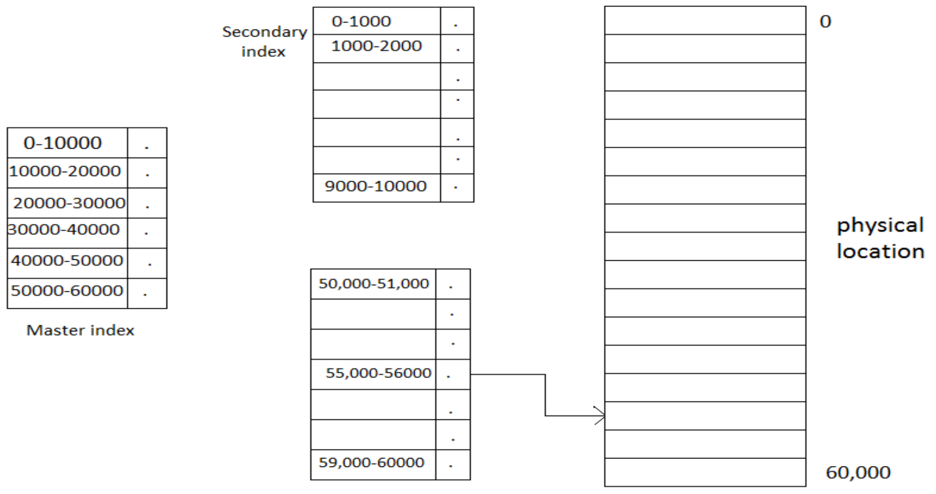
3.INDEXED SEQUENTIAL FILE ACCESS

The main disadvantage in the sequential file is, it takes more time to access a Record .Records are organized in sequence based on a key field.

Eg :

A file consisting of 60000 records,the master index divide the total records into 6 blocks, each block consisiting of a pointer to secondary index.The secondary index divide the 10,000 records into 10 indexes.Each index consisting of a pointer to its orginal location.Each record in the index file consisting of 2 field, A key field and a pointer field.

suppose we want to access the 55,550th record,access index (50,000-60,000), this block consisting of a pointer ,this pointer points to the 6th index in the secondary index.This index points to the original location of the record from (55000-56000). from this it follows the sequential method.Thats why this method is said to be indexed sequential file. It is used in airline reservation systems,payroll system.



DIRECTORY STRUCTURE

Some times the file system consisting of millions of files,at that situation it is very hard to manage the files.To mange these files grouped these files and load one group into one partition.

Each partition is called a directory .a directory structure provides a mechanism for organizing many files in the file system.

OPERATION ON THE DIRECTORIES :

1. Search for a file : Search a directory structure for required file.

2.create a file : New files need to be created, added to the directory.

3.Delete a file : When a file is no longer needed,we want to remove it from the directory.

4.List a directory : We can know the list of files in the directory.

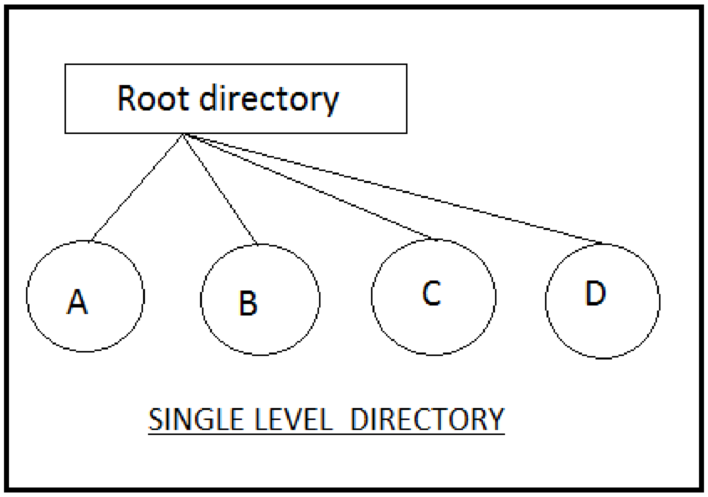
5.Rename a file : When ever we need to change the name of the file, we can change the name.

6.Traverse the file system : We need to access every directory and every file with in a directory structure we can traverse the file system.

The various directory structures

1. Single level directory :

The directory system having only one directory,it consisting of all files some times it is said to be root directory.



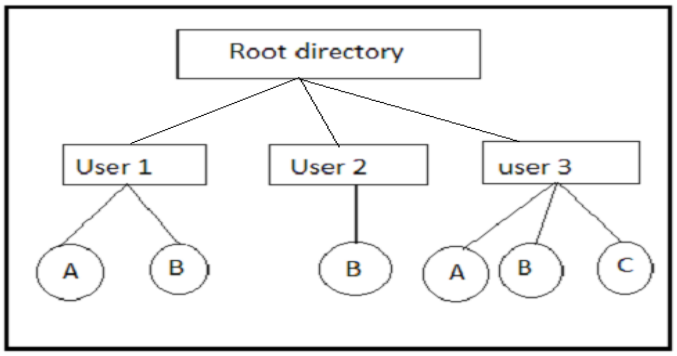
E.g :- Here directory containing 4 files (A,B.C,D).the advantage of the scheme is its simplicity and the ability to locate files quickly.The problem is different users may accidentally use the same names for their files.

E.g :- If user 1 creates a files caled sample and then later user 2 to creates a file called sample,then user2's file will overwrite user 1 file.Thats why it is not used in the multi user system.

2.Two level directory :

The problem in single level directory is different user may be accidentally use the same name for their files. To avoid this problem each user need a private directory,

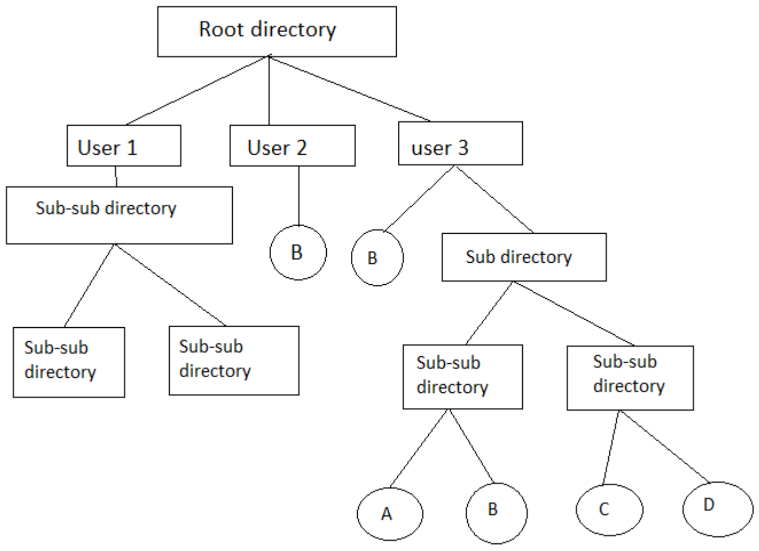
Names chosen by one user don't interfere with names chosen by a different user.



Root directory is the first level directory.user 1,user2,user3 are user level of directory A,B,C are files.

3.Tree structured directory :

Two level directory eliminates name conflicts among users but it is not satisfactory for users with a large number of files.To avoid this create the sub-directory and load the same type of files into the sub-directory.so, here each can have as many directories are needed.



There are 2 types of path

1. Absoulte path

2.Relative path

Absoulte path : Beging with root and follows a path down to specified files giving directory,directory name on the path.

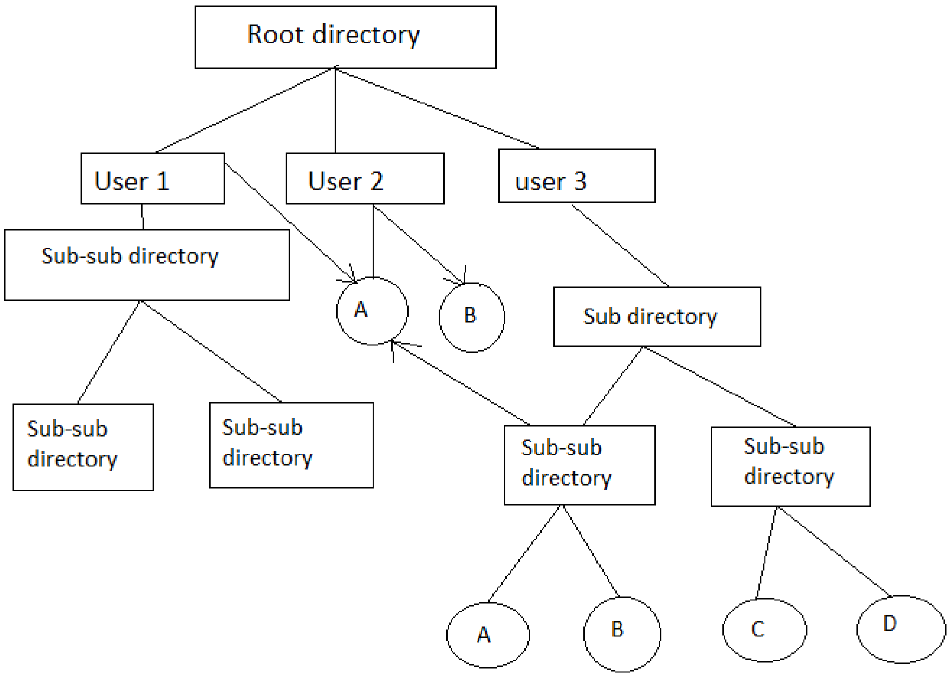
Relative path : A path from current directory.

4. Acyclic graph directory

Multiple users are working on a project, the project files can be stored in a comman sub-directory of the multiple users.This type of directory is called acyclic graph directory .The common directory will be declared a shared directory.The graph contain no cycles with shared files,changes made by one user are made visible to other users.A file may now have multiple absolute paths.when shared directory/file is deleted,all pointers to the directory/ files also to be removed.

5.General graph directory :

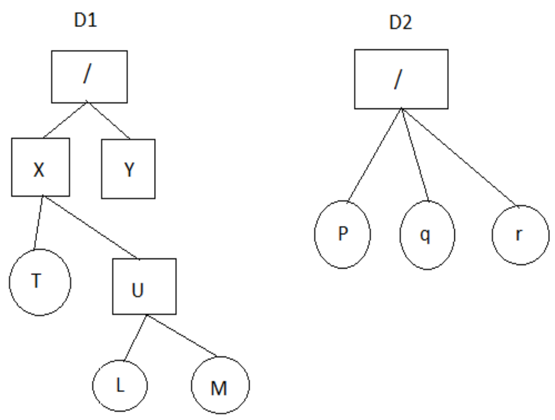
When we add links to an existing tree structured directory,the tree structure is destroyed,resulting is a simple graph structure.



Advantages :- Traversing is easy.easy sharing is possible.

FILE SYSTEM MOUNTING

Assume there are 2 disks D1 and D2 connected to a system .D1 could be a hard disk and D2 could be a floppy disk (or) both could be hard disk (or) floppy disk.



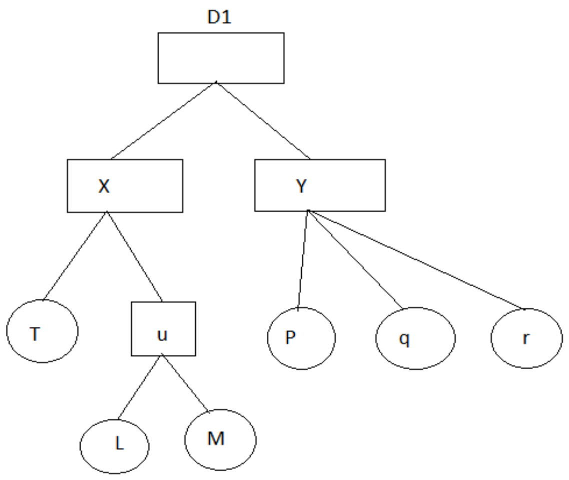
<u>Two file system on 2 Devices</u>

assume to copy a file r from D2 to a directory u under D1.

In MS-DOS

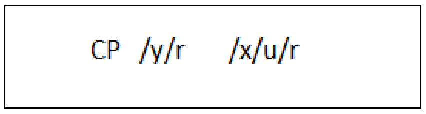
```
CP   D2 : r        D1 : /x/u/r
              (or)
CP   D2 : r        /x/u/r
```

UNIX adopts a different approach .UNIX allows to uproot the whole directory structure under D2 : mount (or) graft it on D1 : under a specific directory -say Y by using a mount command. after mounting ,the directory structure looks as



A Directory structure after mounting

now copying of the file r.



```
CP  /y/r    /x/u/r
```

Similarly, a file system can be unmounted and separated into another independent file system with a separate root directory

File Sharing

In a Single User System, the concept of File Sharing is not needed. Only one user needs files stored on a computer. In Multi user scenario, there are multiple users accessing Multiple Computers. In Such a case, the need for accessing the files stored on other computers is felt many times. Here file sharing comes into picture.

Managing Access Rights

In Multi user scenario, it is very important to decide which user can access which files, specify specific actions that a user can perform on a particular file.

Access Right	**Meaning**
None	The user would not be able to perform any operation on the file. The user may not even be aware about the existence of the file
Read-only	The user can read the file but can't perform insert/delete/update operations.
Read – Write	The user can perform all kinds of read and update operations (insert/delete/update) on file.
Execute	User can Execute the file

Who can have access rights?	
Access right to whom?	Meaning
User	The specified access rights are Provided to a particular user
Group	The specified access rights are provided to a group of users
All	The specified access rights apply to all the users

Remote File Systems

Today, communication among remote computers is possible. Networking allows the sharing of resources spread across a campus (or) even around the world.

Example: Share data in the form of files.

Remote file sharing methods have changed. The first method involves manually transferring files between machines via FTP. The second method uses a distributed file system (DFS) in which remote directories are visible from a local machine. The third method, the WWW. A Browser is needed to gain access to remote files and separate operations are used to transfer files.

FTP is used for both anonymous and authentication access. Anonymous access allows a user to transfer files without having an account on the remote system. WWW uses Anonymous File exchange. DFS involves a much tighter integration between the machine that is accessing the remote files and the mutual providing the files.

Client & Server Model

Remote file systems, the machine containing the files is the server, and the machine seeking access to the files is the client. The server declares that a resource is available to clients and specifies exactly which files and exactly which clients. A server can serve multiple clients and a client can use multiple servers. The server usually specifies the available files on directory level. A client can be specified by an IP address. But these can be imitated as a result of imitating; an unauthorized client could be allowed to access the server. More secure Solutions include secure Authentication of the client via encrypted Keys. Once remote file system mounted, file operation requests are sent to server. Server checks if the user have credentials to access the file. The request is either allowed/denied. If allowed the file is return to client, client can read, write, other operations. The client closes the file when access is completed.

Distributed information systems

To make client- server systems easier to manage, distributed information systems also known as distributed naming services, provide uniform access to the information needed for remote computing.

DNS (Domain name system), we can visit a website by typing in the domain name rather than the IP address (67.43.14.98). DNS translates domain names into IP address, allowing to access an internet location by its domain name. Before DNS became wide spread ,files containing the same information were sent via emails (or) FTP between all networked hosts.

In the case of Microsoft common Internet File System (CIFS) Network information is used in conjunction with user authentication (User name, password) to create a network login that the server uses to decide whether to allow (or) deny access to a requested file system. In windows XP, windows 2000, new distributed naming structure is active directory, once established, the distributing facility is used by all clients and servers to authenticate users .The industry is moving towards use of the light weight directory access protocol (LDAP) active directory is based on LDAP. Sun micro systems include LDAP with the OS and allows it to the employed for user authentication as well as system wide retrieval of information, (availability of printers). One distributed LDAP directory could be used by an organization to store all users and resource information for all the organization computers. The result would be secure single sign on for users who would enter their authentication information once for access to all computers within the organization.

Failure Models

Local file systems can fail for a variety of reasons, failure of disk corruption of the directory Structure, cable failure……User(or) System administrator failure can also cause files to be lost. Human intervention will be required to repair the damage. remote file system have even more failure modes because of the complexity of the network systems and the required interaction between remote machines.

In the case of networks, the network can be interrupted between 2 hosts, such interruptions can result from h/w failure, poor h/w configuration etc.

Consider, a crash of the server, suddenly the remote file system is no longer reachable. The system can either terminate all operations to the lost server (or) delay operations until the server is again reachable.

To implement recovery from failure, state information may be maintained on both client, Server. If both server, client maintain knowledge of the current activities, then they can recover from failure.

Consistency Semantics

Consistency semantics represent an important criterion for evaluating any file system that supports file sharing. These semantics specify how multiple users of a system are to access a shared file simultaneously. They specify when modifications of data by one user will be observable by other users. These semantics are typically implemented as code with the file system.

UNIX Semantics

- Writes to an open file by a user are visible immediately to other users who have this file open.
- One mode of sharing allows users to share the pointer of current location into the file. Thus, advancing of the pointer by one user affects all sharing users.

Session Semantics

- Writes to an open file by a user are not visible immediately to other users who have this same file open.
- Once a file is closed, the changes made to it are visible only in sessions starting later, already open instances of the file do not reflect these changes.

Immutable Shared Files Semantics

Once a file is declared as shared by its creator, it cannot be modified. An immutable file has 2 key properties; its name may not be reused, its data may not altered.So, name of an immutable file signifies that the data of file are fixed.

Protection

When information stored in a computer system, we want to keep it safe from physical damage and improper access. Reliability is generally provided by duplicate copies of files. Many computers have systems programs that automatically copy disk files to tape at regular intervals(once per day (or) week) to maintain a copy should a file system be accidentally destroyed. File systems can be damaged by hardware problems, power failures, and temperature extremes. Files may be deleted accidentally. Bugs in the file system software also cause file contents to be lost. Protection can be provided in many ways. For a small single user system, we might provide protection by physically removing the floppy disks and locking them in a disk drawer. In multilayer system, other mechanisms are needed.

Types of Access

Protection mechanisms provide controlled access by limiting types of file access that can be made. Access is permitted/denied depending on several factors, one of which is the type of access requested.

Read: Read from the file
Write: Write/rewrite the file
Execute: load the file into memory & execute it
Append: Write new information at the end of the file
Delete: delete the file and free its space for reuse
List: list the name and attributes of the file.
Renaming, copying, editing the file may also be controlled

Access Control

Most common approach to the protection problem is to make access dependent on the identity of the user different users may need different types of access to a file. An access control list (ACL) specifying user names and types of access file, OS checks the list (ACL) associated with that file. If that user is listed for the requested access, the access is allowed. Otherwise protection violation occurs, and user process is denied access to the file.

Access can be provided to the following class of users:

1) Owner: The user who created the file is the owner.
2) Group: A set of users who are sharing the file.
3) Universe: All the other users in the system constitute the universe.

Other Protection approaches : Maintain password for each file.

Disadvantages

a) No of passwords that a user needs to remember may become large.
b) If only one password is used for all files, then once it is discovered, all files are accessible.
c) Some systems allow a user to associate a password with a sub directory rather than individual file.

File system structure:

Disk provides the bulk of secondary storage on which a file system is maintained. They have 2 characteristics that make them a convenient medium for storing multiple files.

1. A disk can be rewritten in place. It is possible to read a block from the disk, modify the block, and write it back into same place.
2. A disk can access directly any block of information it contains.

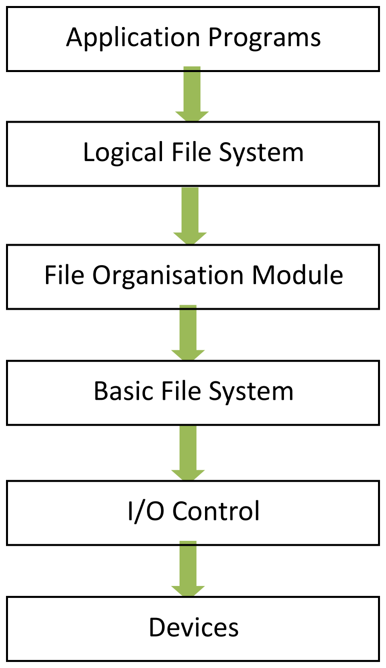


Fig: layered file system

I/O Control:_consists of device drivers and interrupt handlers to transfer information between the main memory and the disk system. The device driver writes specific bit patterns to special locations in the I/O controller's memory to tell the controller which device location to act on and what actions to take.

The Basic File System_needs only to issue commands to the appropriate device driver to read and write physical blocks on the disk. Each physical block is identified by its numeric disk address (Eg. Drive 1, cylinder 73, track2, sector 10).

The File Organization Module knows about files and their logical blocks and physical blocks. By knowing the type of file allocation used and the location of the file, file organization module can translate logical block address to physical addresses for the basic file system to transfer. Each file's logical blocks are numbered from 0 to n. so, physical blocks containing the data usually do not match the logical numbers. A translation is needed to locate each block.

The Logical File System manages all file system structure except the actual data (contents of file). It maintains file structure via file control blocks. A file control block (inode in Unix file systems) contains information about the file, ownership, permissions, location of the file contents.

File System Implementation:

Overview:

A Boot Control Block (per volume) can contain information needed by the system to boot an OS from that volume. If the disk does not contain an OS, this block can be empty.

A Volume Control Block (per volume) contains volume (or partition) details, such as number of blocks in the partition, size of the blocks, a free block, count and free block pointers, free FCB count, FCB pointers.

A Directory Structure (per file system) is used to organize the files.

A PER-FILE FCB contains many details about the file.

A file has been created; it can be used for I/O. First, it must be opened. The open( ) call passes a file name to the logical file system. The open( ) system call First searches the system wide open file table to see if the file is already in use by another process. If it is ,a *per process open file table* entry is created pointing to the existing *system wide open file table*. If the file is not already open, the directory structure is searched for the given file name. Once the file is found, FCB is copied into a system wide open file table in memory. This table not only stores the FCB but also tracks the number of processes that have the file open.

Next, an entry is made in the per – process open file table, with the pointer to the entry in the system wide open file table and some other fields. These are the fields include a pointer to the current location in the file (for the next read/write operation) and the access mode in which the file is open. The open () call returns a pointer to the appropriate entry in the per-process file system table. All file operations are preformed via this pointer. When a process closes the file the per-process table entry is removed. And the system wide entry open count is decremented. When all users that have opened the file close it, any updated metadata is copied back to the disk base directory structure. System wide open file table entry is removed.

System wide open file table contains a copy of the FCB of each open file, other information. Per process open file table, contains a pointer to the appropriate entry in the system wide open file
table, other information.

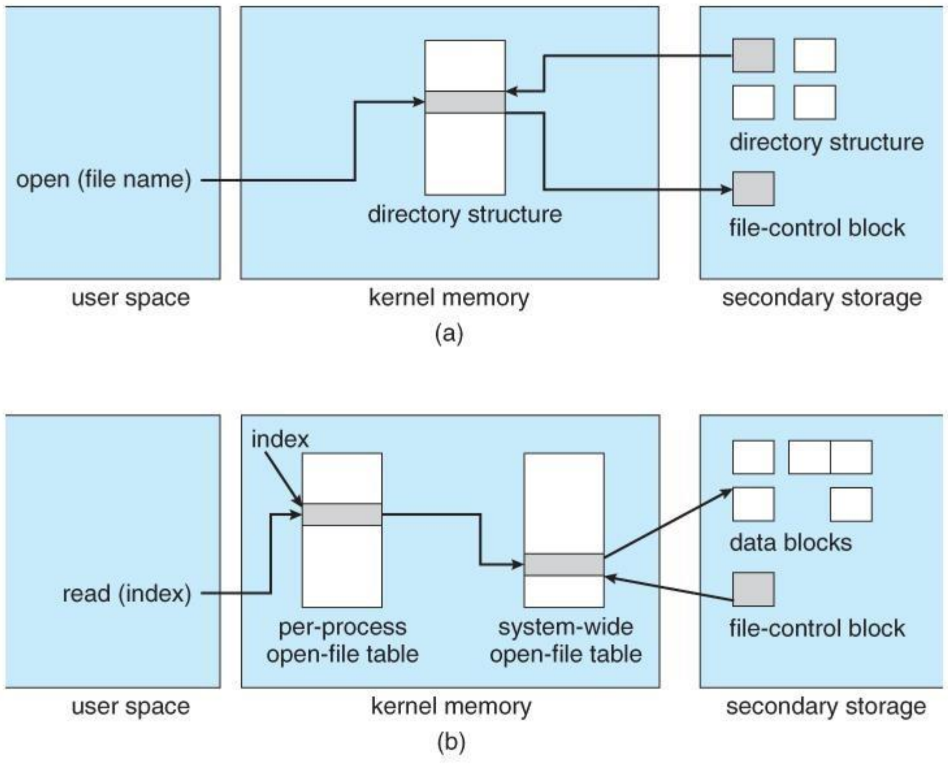
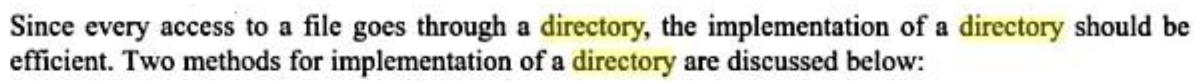


Fig: a)File open b) File read

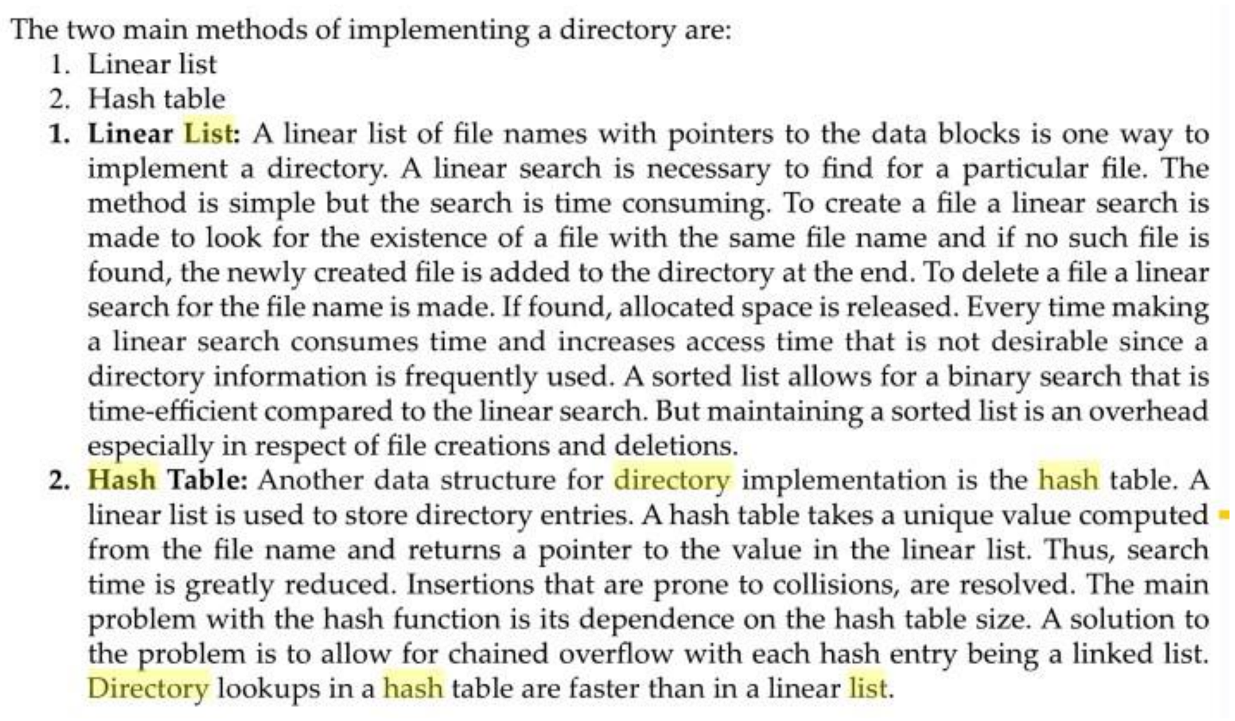
Partitioning and Mounting:

PCs can be dual booted. Multiple OS can be installed on such systems. How does a system know which one to boot ?. A boot loader that understands multiple file systems and multiple OS can occupy the boot space. Once loaded, it can boot one of the OS available on the disk. The disk can have multiple partitions, each containing a different type of file system and different OS. The root partition which contains the OS kernel, other system file is mounted at boot time. Other volumes can be automatically mounted at boot. As part of successful mount operation, OS verify that the device contains a valid file system.

Directory implementation



Since every access to a file goes through a directory, the implementation of a directory should be efficient. Two methods for implementation of a directory are discussed below:



The two main methods of implementing a directory are:

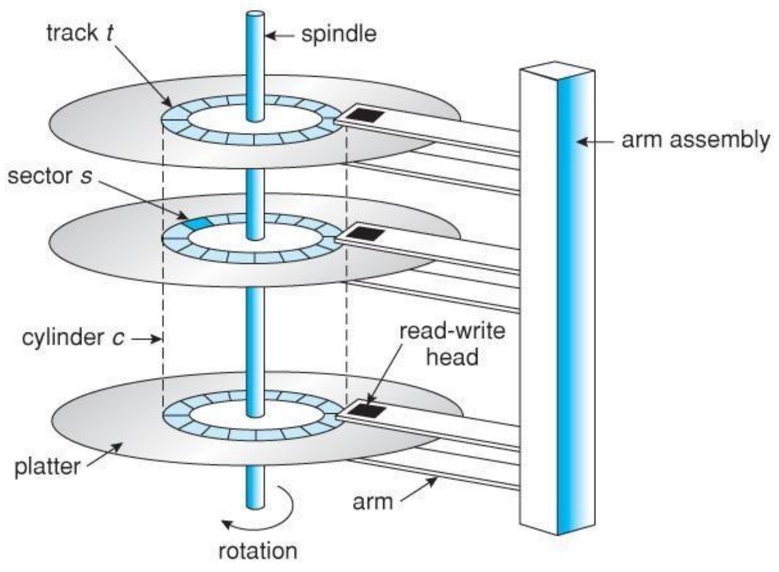
1. Linear list
2. Hash table

1. **Linear List:** A linear list of file names with pointers to the data blocks is one way to implement a directory. A linear search is necessary to find for a particular file. The method is simple but the search is time consuming. To create a file a linear search is made to look for the existence of a file with the same file name and if no such file is found, the newly created file is added to the directory at the end. To delete a file a linear search for the file name is made. If found, allocated space is released. Every time making a linear search consumes time and increases access time that is not desirable since a directory information is frequently used. A sorted list allows for a binary search that is time-efficient compared to the linear search. But maintaining a sorted list is an overhead especially in respect of file creations and deletions.
2. **Hash Table:** Another data structure for directory implementation is the hash table. A linear list is used to store directory entries. A hash table takes a unique value computed from the file name and returns a pointer to the value in the linear list. Thus, search time is greatly reduced. Insertions that are prone to collisions, are resolved. The main problem with the hash function is its dependence on the hash table size. A solution to the problem is to allow for chained overflow with each hash entry being a linked list. Directory lookups in a hash table are faster than in a linear list.

Secondary storage structure:

Overview of mass storage structure

Magnetic disks: Magnetic disks provide the bulk of secondary storage for modern computer system . each disk platter has a flat circular shape, like a CD. Common platter diameters range from 1.8 to 5.25 inches. The two surfaces of a platter are covered with a magnetic material. We store information by it magnetically on the platters.



Moving head disk mechanism

A read /write head files just above each surface of every platter. The heads are attached to a disk arm that moves all the heads as a unit. The surface of a platter is logically divided into circular tracks, which are sub divided into sectors. The set of tracks that are at one arm position makes up a cylinder. There may be thousands of concentric cylinders in a disk drive, and each track may contain hundreds of sectors.

When the disk in use, a driver motor spins it at high speed. Most drivers rotate 60 to 200 times per second. Disk speed has 2 parts. The transfer rate is the at which data flow between the drive and the computer. To read/write, the head must be positioned at the desired track and at the beginning of the desired sector on the track, the time it takes to position the head at the desired track is called seek time. Once the track is selected the disk controller waits until desired sector reaches the read/write head. The time it takes to reach the desired sector is called ***latency time or rotational dealy-access time***. When the desired sector reached the read/write head, then the real data transferring starts.

A disk can be removable. Removable magnetic disks consist of one platter, held in a plastic case to prevent damage while not in the disk drive. Floppy disks are in expensive removable magnetic disks that have a soft plastic case containing a flexible platter. The storage capacity of a floppy disk is 1.44MB.

A disk drive is attached to a computer by a set of wires called an I/O bus. The data transfer on a bus are carried out by special processors called controllers. The host controller is the controller at the computer end of the bus. A disk controller is built into each disk drive . to perform i/o operation, the host controller operates the disk drive hardware to carry out the command. Disk controllers have built in cache, data transfer at the disk drive happens b/w cache and disk surface. Data transfer at the host, occurs b/w cache and host controller.

Magnetic Tapes: magnetic tapes was used as an early secondary storage medium. It is permanent and can hold large amount of data. It access time is slow compared to main memory and magnetic disks. Tapes are mainly used for back up, for storage of infrequently used information. Typically they store 20GB to 200GB.

Disk Structure: most disks drives are addressed as large one dimensional arrays of logical blocks. The one dimensional array of logical blocks is mapped onto the sectors of the disk sequentially. sector 0 is the fist sector of the first track on the outermost cylinder. The mapping proceeds in order through that track, then through the rest of the tracks in that cylinder, and then through the rest of the cylinder from outermost to inner most. As we move from outer zones to inner zones, the number of sectors per track decreases. Tracks in outermost zone hold 40% more sectors then innermost zone. The number of sectors per track has been increasing as disks technology improves, and the outer zone of a disk usually has several hundred sectors per track. Similarly, the number of cylinders per disk has been increasing; large disks have tens of thousands of cylinders.

Disk attachment

Computer access disk storage is 2 ways.

1. Via I/O ports(host attached storage)
2. Via a remote host in a distributed file system(network attached storage).

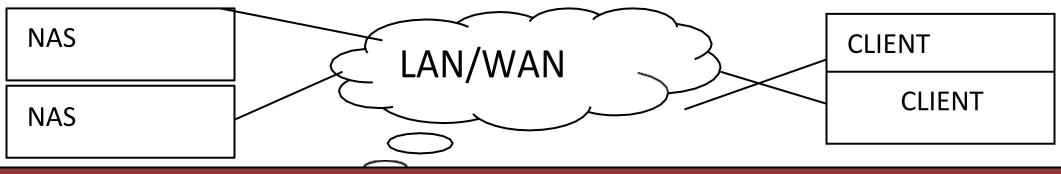
1 .Host attached storage : host attached storage are accessed via local I/O ports. The desktop pc uses an I/O bus architecture called IDE. This architecture supports maximum of 2 drives per I/O bus. High end work station and servers use SCSI and FC.

SCSI is an bus architecture which have large number of conductor's in a ribbon cable (50 or 68) scsi protocol supports maximum of 16 drives an bus. Host consists of a controller card (SCSI Initiator) and upto 15 storage device called SCSI targets.

Fc(fiber channel) is the high speed serial architecture. It operates mostly on optical fiber (or) over 4 conductor copper cable. It has 2 variants. One is a large switched fabric having a 24-bit address space. The other is an (FC-AL) arbitrated loop that can address 126 devices.

A wide variety of storage devices are suitable for use as host attached.(hard disk,cd ,dvd,tape devices)

2.Network-attached storage: A(NAS) is accessed remotely over a data network .clients access network attached storage via remote procedure calls. The rpc are carried via tcp/udp over an ip network-usually the same LAN that carries all data traffic to the clients.



NAS provides a convenient way for all the computers on a LAN to share a pool of storage with the same ease of naming and access enjoyed with local host attached storage .but it tends to be less efficient and have lower performance than direct attached storage.

3.Storage area network: The drawback of network attached storage(NAS) is storage I/O operations consume bandwidth on the data network. The communication b/w servers and clients competes for bandwidth with the communication among servers and storage devices.

A storage area network(SAN) is a private network using storage protocols connecting servers and storage units. The power of a SAN is its flexibility. multiple hosts and multiple storage arrays can attach to the same SAN, and storage can be dynamically allocated to hosts. SANs make it possible for clusters of server to share the same storage.

Disk Scheduling Algorithms

Disk scheduling algorithms are used to allocate the services to the I/O requests on the disk . Since seeking disk requests is time consuming, disk scheduling algorithms try to minimize this latency. If desired disk drive or controller is available, request is served immediately. If busy, new request for service will be placed in the queue of pending requests. When one request is completed, the Operating System has to choose which pending request to service next. The OS relies on the type of algorithm it needs when dealing and choosing what particular disk request is to be processed next. The objective of using these algorithms is keeping Head movements to the amount as possible. The less the head to move, the faster the seek time will be. To see how it works, the different disk scheduling algorithms will be discussed and examples are also provided for better understanding on these different algorithms.

1.First Come First Serve (FCFS)

It is the simplest form of disk scheduling algorithms. The I/O requests are served or processes according to their arrival. The request arrives first will be accessed and served first. Since it follows the order of arrival, it causes the wild swings from the innermost to the outermost tracks of the disk and vice versa . The farther the location of the request being serviced by the read/write head from its current location, the higher the seek time will be.

Example: Given the following track requests in the disk queue, compute for the Total Head Movement (THM) of the read/write head :

95, 180, 34, 119, 11, 123, 62, 64

Consider that the read/write head is positioned at location 50. Prior to this track location 199 was serviced. Show the total head movement for a 200 track disk (0-199).

Solution:

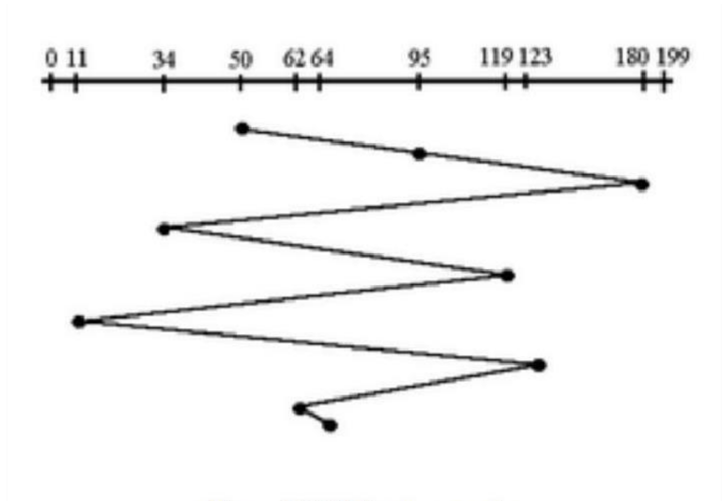


Fig. FCFS Representation

Total Head Movement Computation: (THM) =

(180 - 50) + (180-34) + (119-34) + (119-11) + (123-11) + (123-62) + (64-62) =

130 + 146 + 85 + 108 + 112 + 61 + 2 (THM) = 644 tracks

Assuming a seek rate of 5 milliseconds is given, we compute for the seek time using the formula:

Seek Time = THM * Seek rate

= 644 * 5 ms

Seek Time = 3,220 ms.

2.Shortest Seek Time First (SSTF):

This algorithm is based on the idea that that he R/W head should proceed to the track that is closest to its current position . The process would continue until all the track requests are taken care of. Using the same sets of example in FCFS the solution are as follows:

Solution:

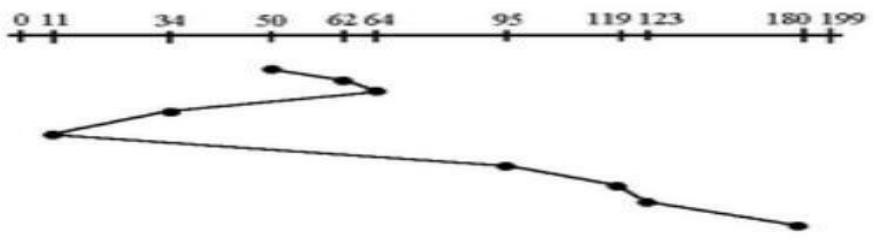


Fig. SSTF Representation

(THM) = (64-50) + (64-11) + (180-11) =

14 + 53 + 169 (THM) = 236 tracks

Seek Time = THM * Seek rate

= 236 * 5ms

Seek Time = 1,180 ms

In this algorithm, request is serviced according to the next shortest distance. Starting at 50, the next shortest distance would be 62 instead of 34 since it is only 12 tracks away from 62 and 16 tracks away from 34 . The process would continue up to the last track request. There are a total of 236 tracks and a seek time of 1,180 ms, which seems to be a better service compared with FCFS which there is a chance that starvation3 would take place. The reason for this is if there were lots of requests closed to each other, the other requests will never be handled since the distance will always be greater .

3.SCAN Scheduling Algorithm

This algorithm is performed by moving the R/W head back-and-forth to the innermost and outermost track. As it scans the tracks from end to end, it process all the requests found in the direction it is headed. This will ensure that all track requests, whether in the outermost, middle or innermost location, will be traversed by the access arm thereby finding all the requests. This is also known as the Elevator algorithm. Using the same sets of example in FCFS the solution are as follows:

Solution:

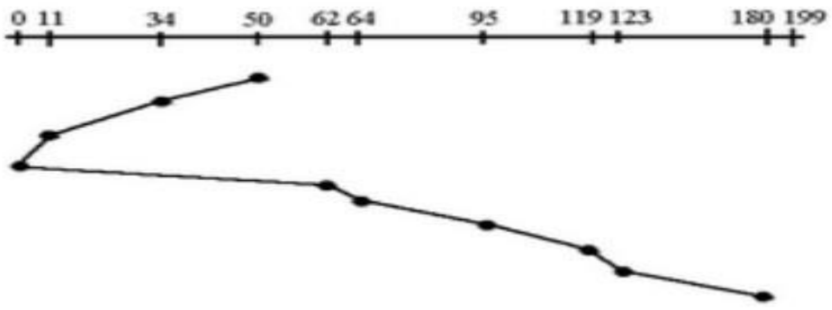


Fig. SCAN Representation

(THM) = (50-0) + (180-0)
= 50 + 180
(THM) = 230

*Seek Time = THM * Seek rate*
*= 230 * 5ms*
Seek Time = 1,150 ms

This algorithm works like an elevator does. In the algorithm example, it scans down towards the nearest end and when it reached the bottom it scans up servicing the requests that it did not get going down. If a request comes in after it has been scanned, it will not be serviced until the process comes back down or moves back up. This process moved a total of 230 tracks and a seek time of 1,150. This is optimal than the previous algorithm.

4 .LOOK Scheduling Algorithm

This algorithm is similar to SCAN algorithm except for the end-to-end reach of each sweep. The R/W head is only tasked to go the farthest location in need of servicing. This is also a directional algorithm, as soon as it is done with the last request in one direction it then sweeps in the other direction. Using the same sets of example in FCFS the solution are as follows:

Solution:

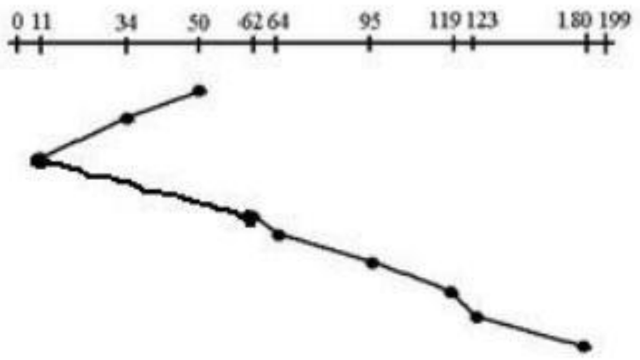


Fig. LOOK Representation

$$(THM) = (50-11) + (180-11)$$
$$= 39 + 169$$
$$\textbf{(THM) = 208 tracks}$$

Seek Time = THM * Seek rate

= 208 * 5ms

Seek Time = 1,040 ms .

This algorithm has a result of 208 tracks and a seek rate of 1,040 milliseconds. This algorithm is better than the previous algorithm.

4.Circular SCAN (C-SCAN) Algorithm

This algorithm is a modified version of the SCAN algorithm. C-SCAN sweeps the disk from end-to-end, but as soon it reaches one of the end tracks it then moves to the other end track without servicing any requesting location. As soon as it reaches the other end track it then starts servicing and grants requests headed to its direction. This algorithm improves the unfair situation of the end tracks against the middle tracks. Using the same sets of example in FCFS the solution are as follows:

Solution:

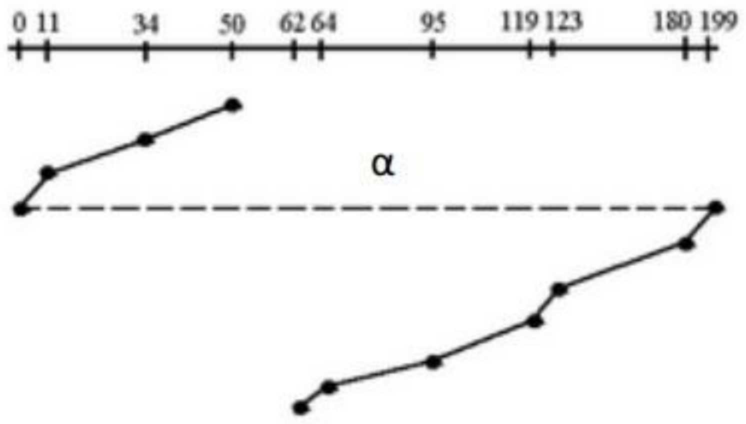


Fig. C-SCAN Representation

Notice that in this example an alpha3 symbol (α) was used to represent the dash line. This return sweeps is sometimes given a numerical value which is included in the computation of the THM . As analogy, this can be compared with the carriage return lever of a typewriter. Once it is pulled to the right most direction, it resets the typing point to the leftmost margin of the paper . A typist is not supposed to type during the movement of the carriage return lever because the line spacing is being adjusted . The frequent use of this lever consumes time, same with the time consumed when the R/W head is reset to its starting position.

Assume that in this example, α has a value of 20ms, the computation would be as follows:

(THM) = (50-0) + (199-62) + α

= 50 + 137 + 20 (THM)

= 207 tracks

Seek Time = THM * Seek rate

= 187 * 5ms Seek Time = 935 ms .

The computation of the seek time excluded the alpha value because it is not an actual seek or search of a disk request but a reset of the access arm to the starting position .

Disk management

Disk formatting*:* A magnetic disk is a blank slate. It is just a platter of **a magnetic recording material.**before a disk can store data , it must be divided into sectors that the disk controller can read and write. This process is called low level formatting (or)physical formatting. low level formatting fills the disk with a special data structure for each sector .the Data structure **for a sector** typically consists of a header, a data **area**, a trailer . the header and trailer contain information used by the disk controller ,such as a sector number and a**n** error correcting code(ECC). When the controller writes a sector of data during normal I/O, the ECC is updated with a value calculated from all the bytes in the data area . when the sector is read ,the EC**C** is recalculated and compared with the stored value. If the stored and calculated **numbers** are different, this mismatch indicates that the data area of this sector has become corrupted, and that the disk **sector** may be bad. ECC contains enough information, **if** only few bits of data have been corrupted, to enable the controller to identify which bits have changed and calculate what their correct values should be. The controller automatically does the ECC processing what ever a sector is read/written for many hard disks, when the disk controller is instructed to low level format the disk, it can also be told how many bytes of data space to leave between the header and trailer of all sectors. Before it can use a disk to hold files , OS still needs to record its own data structures on the disk. It does in 2 steps. The first step is to partition the disk in to one/more groups of cylinders. OS can treat each partition as a separate disk. The second step is logical formatting (or)creation of file system. In this step, OS stores the initial File system **d**ata structures on to the disk. These data structures include maps of free and allocate space and initial empty director**y**.

***Boot block*:-**

When a computer is powered up -it must have an initial program to run. This initial bootstrap program initializes all aspects of the system, from CPU registers to device controllers, and the contents of main memory, and then starts the OS. To do its job, the bootstrap program finds the OS kernel on disk, loads that kernel into memory and jumps to an initial address to begin the OS execution. For most computers, the bootstrap is stored in ROM. This location is convenient, because ROM needs no initialization and is at a fixed location that the CPU can start executing when powered up, ROM is read only, it cannot be infected by computer virus. The problem is that changing this bootstrap code requires changing the ROM hardware chips. For this reason, most systems store a tiny bootstrap loader program in the boot ROM whose job is to bring in a full bootstrap program from disk. The full bootstrap program is stored in the boot blocks at a fixed location on the disk. A disk that has a boot partition is called a boot disk or system disk. The code in the boot ROM instructs the disk controller to read the boot blocks into memory and then starts executing that code.

Bad blocks:-

A Block in the disk damaged due to the manufacturing defect or virus or physical damage. This defector block is called Bad block. MS-DOS format command, scans the disk to find bad blocks. If format finds a bad block, it tells the allocation methods not to use that block. Chkdsk program search for the bad blocks and to lock them away. Data that resided on the bad blocks usually are lost. The OS tries to read logical block 87.
The controller calculates ECC and finds that the sector is bad. It reports this finding to the OS. The next time the system is rebooted, a special command is run to tell the SCS controller to replace the bad sector

with a spare.
After that, whenever the system requests logical block 87, the request is translated into the replacement sectors address by the controller.

Sector slipping:-

Logical block 17 becomes defective and the first available spare follows sector 202. Then, sector slipping remaps all the sectors from 17 to 202, sector 202 is copied into the spare, then sector 201 to 202, 200 to 201 and so on. Until sector 18 is copied into sector 19. Slipping the sectors in this way frees up the space of sector 18.

Swap space management:-

System that implements swapping may use swap space to hold an entire process image, including the code and data segments. Paging systems may simply store pages that have been pushed out of main memory. Note that it may be safer to overestimate than to underestimate the amount of swap space required, because if a system runs out of swap space it may be forced to abort processes. Overestimation wastes disk space that could otherwise be used for files, but it does no other harm. Some systems recommend the amount to be set aside for swap space. Linux has suggested setting swap space to double the amount of physical memory. Some OS allow the use of multiple swap spaces. These swap spaces as put on separate disks so that load placed on the (I/O) system by paging and swapping can be spread over the systems I/O devices.

Swap space location:-

A Swap space can reside in one of two places. It can be carved out of normal file system (or) it can be in a separate disk partition. If the swap space is simply a large file, within the file system, normal file system methods used to create it, name it, allocate its space. It is easy to implement but inefficient. External fragmentation can greatly increase swapping times by forcing multiple seeks during reading/writing of a process image. We can improve performance by caching the block location information in main memory and by using special tools to allocate physically contiguous blocks for the swap file. Alternatively,swap space can be created in a separate raw partition.a separate swap space storage manager is used to allocate /deallocate the blocks from the raw partition.this manager uses algorithms optimized for speed rather than storage efficiency.internal fragmentation may increase but it is acceptable because life of data in swap space is shorter than files.since swap space is reinitialized at boot time,any fragmentation is short lived.the raw partition approach creates a fixed amount of swap space during disk partitioning adding more swap space requires either repartitioning the disk (or) adding another swap space elsewhere.