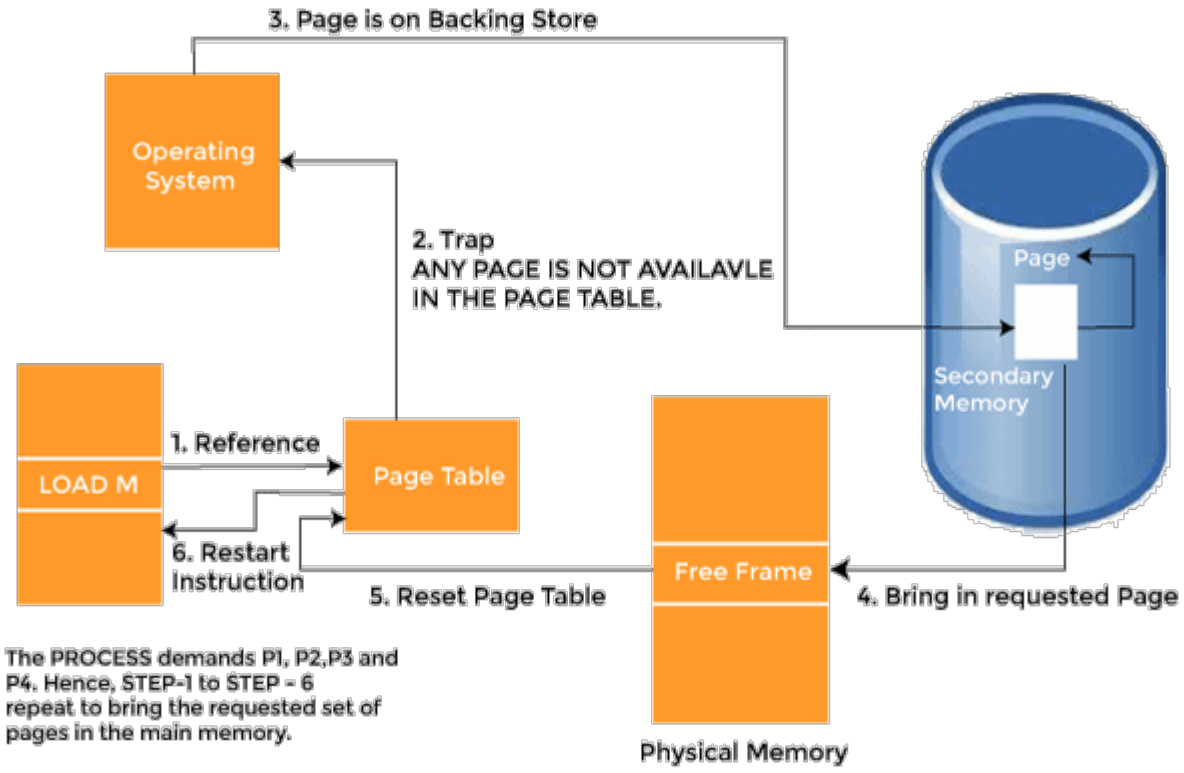
UNIT IV

Demand Paging

Demand paging is a technique used in virtual memory systems where the pages are brought in the main memory only when required or demanded by the CPU. Hence, it is also called lazy swapper because the swapping of pages is done only when the CPU requires it. Virtual memory is commonly implemented in demand paging.

How does Demand Paging Work?



The demand paging system depends on the page table implementation because the page table helps map logical memory to physical memory. Bitwise operators are implemented in the page table to indicate that pages are ok or not (valid or invalid). All valid pages exist in primary memory, and invalid pages exist in

secondary memory. Now all processes go-ahead to access all pages, and then the following things will be happened, such as:

Attempt to currently access page.

If the page is ok (Valid), then all processing instructions work as normal.

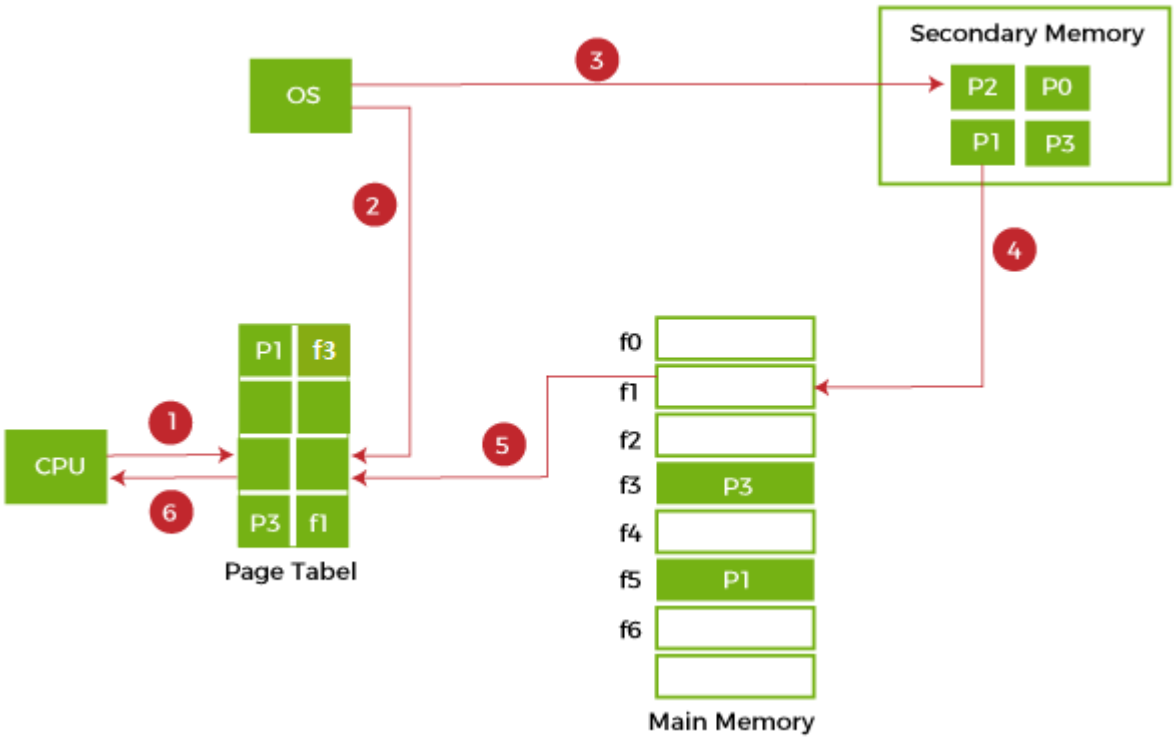
If anyone's page is found invalid, then a page fault issue arises.

Now memory reference is determined whether a valid reference exists on the auxiliary memory or not. If not exist, then the process is terminated, otherwise needed pages are paged in.

Now disk operations are implemented to fetch the required page into primary memory.

Examples of Demand Paging

Suppose we have to execute a process P having four pages P0, P1, P2, and P3. Currently, in the page table, we have pages P1 and P3.



What are Demand Paging and Pre-paging

If the CPU wants to access page P2 of a process P, it will first search the page in the page table.

As the page table does not contain this page so it will be a trapor page fault. The control goes to the operating system as soon as the trap is generated and context switching happens.

The OS system will put the process in a waiting/ blocked state. The OS system will now search that page in the backing store or secondary memory.

The OS will then read the page from the backing store and load it to the main memory.

Next, the OS system will update the page table entry accordingly.

Finally, the control is taken back from the OS, and the execution of the process is resumed.

Hence, the operating system follows these steps whenever a page fault occurs and the required page is brought into memory.

So whenever a page fault occurs, as shown in all the above steps 2 to 6. This time taken to service the page fault is called the Page fault service time.

Effective Memory Access Time: When the page fault rate is 'p' while executing any process, then the effective memory access time is calculated as follows:

Effective Memory Access time = (p)*(s) + (1-p)*(m)

Advantages of Demand Paging

Here are the following advantages of demand paging in the operating system, such as:

It increases the degree of multiprogramming as many processes can be present in the main memory simultaneously.

There is a more efficient use of memory as processes having a size more than the size of the main memory can also be executed using this mechanism because we are not loading the whole page at a time.

We have to the right for scaling of virtual memory.

If any program is larger than physical memory, it helps run this program without compaction.

Partition management is simpler.

It is more useful in a time-sharing system.

It has no limitations on the level of multi-programming.

Discards external fragmentation.

Easy to swap all pages.

Copy on Write

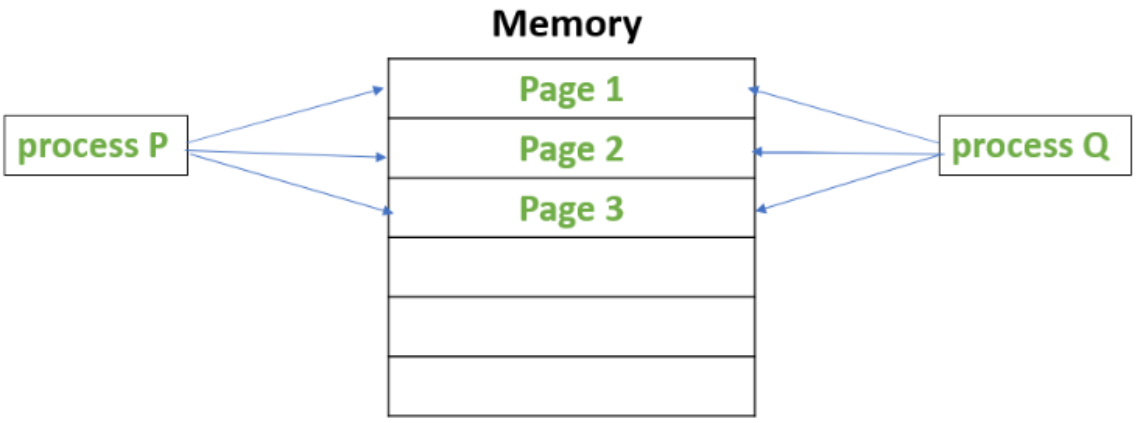
Copy on Write or simply COW is a resource management technique. One of its main use is in the implementation of the fork system call in which it shares the virtual memory(pages) of the OS.

In UNIX like OS, fork() system call creates a duplicate process of the parent process which is called as the child process.

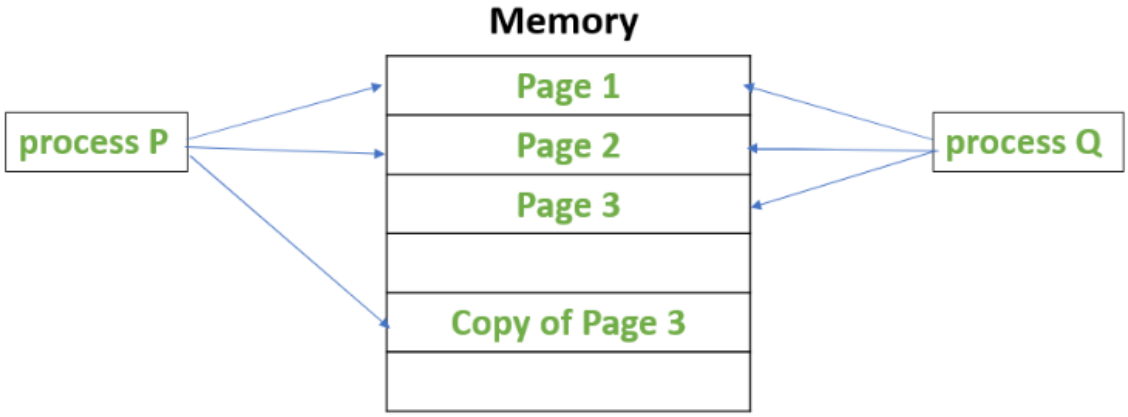
The idea behind a copy-on-write is that when a parent process creates a child process then both of these processes initially will share the same pages in memory and these shared pages will be marked as copy-on-write which means that if any of these processes will try to modify the shared pages then only a copy of these pages will be created and the modifications will be done on the copy of pages by that process and thus not affecting the other process.

Suppose, there is a process P that creates a new process Q and then process P modifies page 3.

The below figures shows what happens before and after process P modifies page 3.



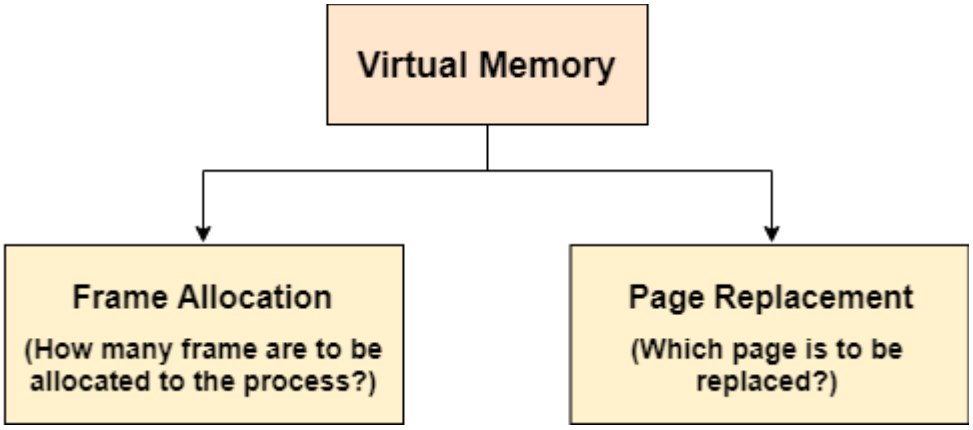
Before process P modifies Page 3



After process P modifies Page 3

Page Replacement Algorithms

The page replacement algorithm decides which memory page is to be replaced. The process of replacement is sometimes called swap out or write to disk. Page replacement is done when the requested page is not found in the main memory (page fault).



There are two main aspects of virtual memory, Frame allocation and Page Replacement. It is very important to have the optimal frame allocation and page replacement algorithm. Frame allocation is all about how many frames are to be allocated to the process while the page replacement is all about determining the page number which needs to be replaced in order to make space for the requested page.

What If the algorithm is not optimal?

1. if the number of frames which are allocated to a process is not sufficient or accurate then there can be a problem of thrashing. Due to the lack of frames, most of the pages will be residing in the main memory and therefore more page faults will occur.

However, if OS allocates more frames to the process then there can be internal fragmentation.

2. If the page replacement algorithm is not optimal then there will also be the problem of thrashing. If the number of pages that are replaced by the requested pages will be referred in the near future then there will be more number of swap-in and swap-out and therefore the OS has to perform more replacements then usual which causes performance deficiency.

Therefore, the task of an optimal page replacement algorithm is to choose the page which can limit the thrashing.

Types of Page Replacement Algorithms

There are various page replacement algorithms. Each algorithm has a different method by which the pages can be replaced.

Optimal Page Replacement algorithm → this algorithms replaces the page which will not be referred for so long in future. Although it can not be practically implementable but it can be used as a benchmark. Other algorithms are compared to this in terms of optimality.

Least recent used (LRU) page replacement algorithm → this algorithm replaces the page which has not been referred for a long time. This algorithm is just opposite to the optimal page replacement algorithm. In this, we look at the past instead of staring at future.

FIFO → in this algorithm, a queue is maintained. The page which is assigned the frame first will be replaced first. In other words, the page which resides at the rare end of the queue will be replaced on the every page fault.

Allocation of frames in Operating System

An important aspect of operating systems, virtual memory is implemented using demand paging. Demand paging necessitates the development of a page-replacement algorithm and a frame allocation algorithm. Frame allocation algorithms are used if you have multiple processes; it helps decide how many frames to allocate to each process.

There are various constraints to the strategies for the allocation of frames:

You cannot allocate more than the total number of available frames.

At least a minimum number of frames should be allocated to each process. This constraint is supported by two reasons. The first reason is, as less number of frames are allocated, there is an increase in the page fault ratio, decreasing the performance of the execution of the process. Secondly, there should be enough frames to hold all the different pages that any single instruction can reference.

Frame allocation algorithms –

The two algorithms commonly used to allocate frames to a process are:

Equal allocation: In a system with x frames and y processes, each process gets equal number of frames, i.e. x/y. For instance, if the system has 48 frames and 9 processes, each process will get 5 frames. The three frames which are not allocated to any process can be used as a free-frame buffer pool.

Disadvantage: In systems with processes of varying sizes, it does not make much sense to give each process equal frames. Allocation of a large number of frames to a small process will eventually lead to the wastage of a large number of allocated unused frames.

Proportional allocation: Frames are allocated to each process according to the process size.

For a process pi of size si, the number of allocated frames is ai = (si/S)*m, where S is the sum of the sizes of all the processes and m is the number of frames in the system. For instance, in a system with 62 frames, if there is a process of 10KB and another process of 127KB, then the first process will be allocated (10/137)*62 = 4 frames and the other process will get (127/137)*62 = 57 frames.

Advantage: All the processes share the available frames according to their needs, rather than equally.

Global vs Local Allocation –

The number of frames allocated to a process can also dynamically change depending on whether you have used global replacement or local replacement for replacing pages in case of a page fault.

Local replacement: When a process needs a page which is not in the memory, it can bring in the new page and allocate it a frame from its own set of allocated frames only.

Advantage: The pages in memory for a particular process and the page fault ratio is affected by the paging behavior of only that process.

Disadvantage: A low priority process may hinder a high priority process by not making its frames available to the high priority process.

Global replacement: When a process needs a page which is not in the memory, it can bring in the new page and allocate it a frame from the set of all frames, even if that frame is currently allocated to some other process; that is, one process can take a frame from another.

Advantage: Does not hinder the performance of processes and hence results in greater system throughput.

Disadvantage: The page fault ratio of a process can not be solely controlled by the process itself. The pages in memory for a process depends on the paging behavior of other processes as well.

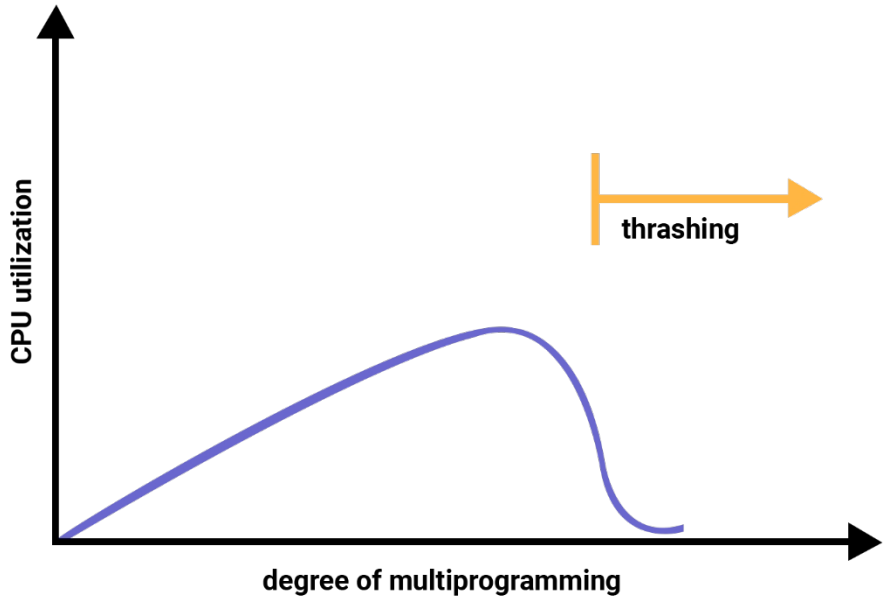
Thrashing in Operating System

In this tutorial, we will be covering the concept of thrashing in the Operating system.

In case, if the page fault and swapping happens very frequently at a higher rate, then the operating system has to spend more time swapping these pages. This state in the operating system is termed thrashing. Because of thrashing the CPU utilization is going to be reduced.

Let's understand by an example, if any process does not have the number of frames that it needs to support pages in active use then it will quickly page fault. And at this point, the process must replace some pages. As all the pages of the process are actively in use, it must replace a page that will be needed again right away. Consequently, the process will quickly fault again, and again, and again, replacing pages that it must bring back in immediately. This high paging activity by a process is called thrashing.

During thrashing, the CPU spends less time on some actual productive work spend more time swapping.



Causes of Thrashing

Thrashing affects the performance of execution in the Operating system. Also, thrashing results in severe performance problems in the Operating system.

When the utilization of CPU is low, then the process scheduling mechanism tries to load many processes into the memory at the same time due to which degree of Multiprogramming can be increased. Now in this situation, there are more processes in the memory as compared to the available number of frames in the memory. Allocation of the limited amount of frames to each process.

Whenever any process with high priority arrives in the memory and if the frame is not freely available at that time then the other process that has occupied the frame is residing in the frame will move to secondary storage and after that this free frame will be allocated to higher priority process.

We can also say that as soon as the memory fills up, the process starts spending a lot of time for the required pages to be swapped in. Again the utilization of the CPU becomes low because most of the processes are waiting for pages.

Thus a high degree of multiprogramming and lack of frames are two main causes of thrashing in the Operating system

Effect of Thrashing

At the time, when thrashing starts then the operating system tries to apply either the Global page replacement Algorithm or the Local page replacement algorithm.

Global Page Replacement

The Global Page replacement has access to bring any page, whenever thrashing found it tries to bring more pages. Actually, due to this, no process can get enough frames and as a result, the thrashing will increase more and more. Thus the global page replacement algorithm is not suitable whenever thrashing happens.

Local Page Replacement

Unlike the Global Page replacement, the local page replacement will select pages which only belongs to that process. Due to this, there is a chance of a reduction in the thrashing. As it is also proved that there are many disadvantages of Local Page replacement. Thus local page replacement is simply an alternative to Global Page replacement.

Mapping from page table to main memory

In operating systems, there is always a requirement of mapping from logical address to the physical address. However, this process involves various steps which are defined as follows.

1. Generation of logical address

CPU generates logical address for each page of the process. This contains two parts: page number and offset.

2. Scaling

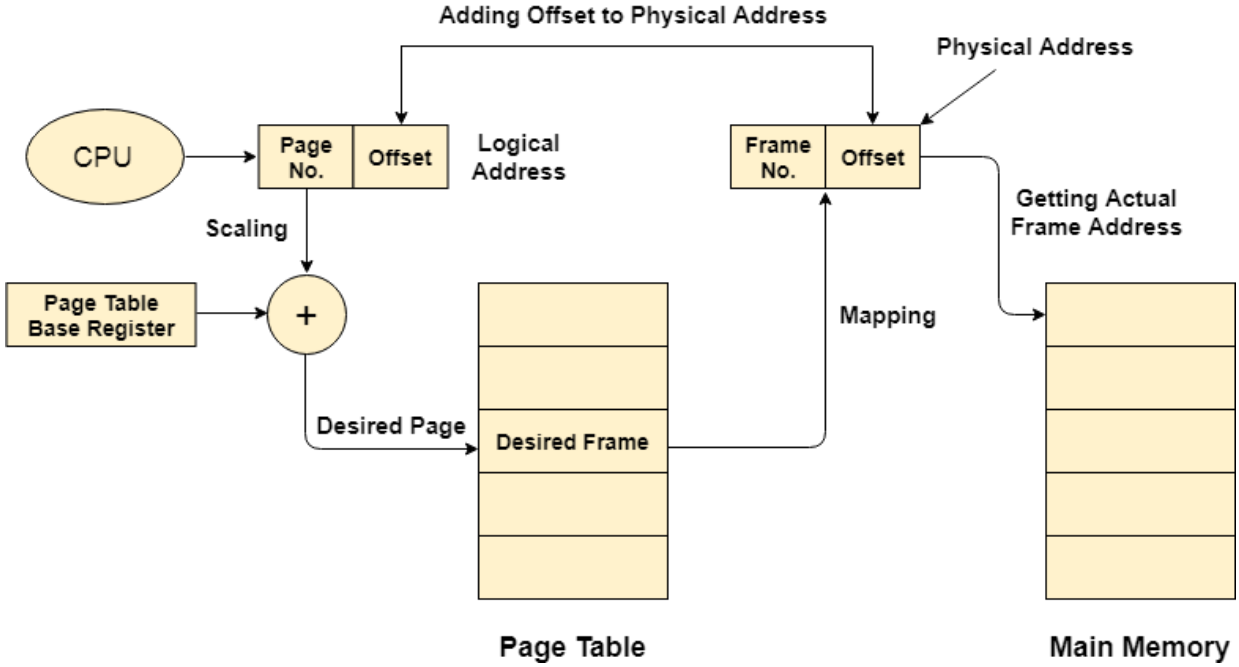
To determine the actual page number of the process, CPU stores the page table base in a special register. Each time the address is generated, the value of the page table base is added to the page number to get the actual location of the page entry in the table. This process is called scaling.

3. Generation of physical Address

The frame number of the desired page is determined by its entry in the page table. A physical address is generated which also contains two parts : frame number and offset. The Offset will be similar to the offset of the logical address therefore it will be copied from the logical address.

4. Getting Actual Frame Number

The frame number and the offset from the physical address is mapped to the main memory in order to get the actual word address.



Allocating Kernel Memory

What is Kernel?

In computer science, Kernel is a computer program that is a core or heart of an operating system. Before discussing kernel in detail, let's first understand its basic, i.e., Operating system in a computer.

Operating System

An operating system

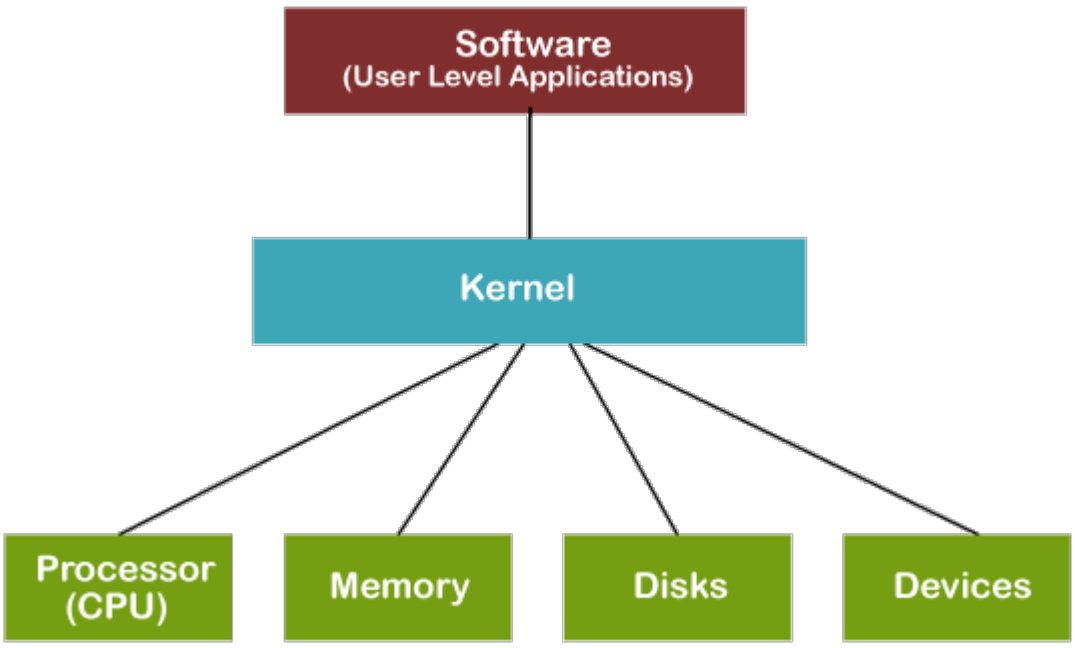
or OS is system software that works as an interface between hardware components and end-user. It enables other programs to run. Each computer system, whether it is desktop, laptop, tablet, or smartphone, all must have an OS to provide basic functionalities for the device. Some widely used operating systems are Windows

, Linux

, MacOS, Android

, iOS, etc.

What is Kernel in Operating System?



As discussed above, Kernel is the core part of an OS(Operating system); hence it has full control over everything in the system. Each operation of hardware and software is managed and administrated by the kernel.

It acts as a bridge between applications and data processing done at the hardware level. It is the central component of an OS.

It is the part of the OS that always resides in computer memory and enables the communication between software and hardware components.

It is the computer program that first loaded on start-up the system (After the bootloader). Once it is loaded, it manages the remaining start-ups. It also manages memory, peripheral, and I/O requests from software. Moreover, it translates all I/O requests into data processing instructions for the CPU. It manages other tasks also such as memory management, task management, and disk management.

A kernel is kept and usually loaded into separate memory space, known as protected Kernel space. It is protected from being accessed by application programs or less important parts of OS.

Other application programs such as browser, word processor, audio & video player use separate memory space known as user-space.

Due to these two separate spaces, user data and kernel data don't interfere with each other and do not cause any instability and slowness.

Other considerations

Functions of a Kernel

A kernel of an OS is responsible for performing various functions and has control over the system. Some main responsibilities of Kernel are given below:

Device Management

To perform various actions, processes require access to peripheral devices such as a mouse, keyboard, etc., that are connected to the computer. A kernel is responsible for controlling these devices using device drivers. Here, a device

driver is a computer program that helps or enables the OS to communicate with any hardware device.

A kernel maintains a list of all the available devices, and this list may be already known, configured by the user, or detected by OS at runtime.

Memory Management

The kernel has full control for accessing the computer's memory. Each process requires some memory to work, and the kernel enables the processes to safely access the memory. To allocate the memory, the first step is known as virtual addressing, which is done by paging or segmentation. Virtual addressing is a process of providing virtual address spaces to the processes. This prevents the application from crashing into each other.

Resource Management

One of the important functionalities of Kernel is to share the resources between various processes. It must share the resources in a way that each process uniformly accesses the resource.

The kernel also provides a way for synchronization and inter-process communication (IPC). It is responsible for context switching between processes.

Accessing Computer Resources

A kernel is responsible for accessing computer resources such as RAM and I/O devices. RAM or Random-Access Memory is used to contain both data and instructions. Each program needs to access the memory to execute and mostly wants more memory than the available. For such a case, Kernel plays its role and decides which memory each process will use and what to do if the required memory is not available.

The kernel also allocates the request from applications to use I/O devices such as keyboards, microphones, printers, etc.

Storage Management

Storage Management is defined as it refers to the management of the data storage equipment’s that are used to store the user/computer generated data. Hence it is a tool or set of processes used by an administrator to keep your data and storage equipment’s safe.

Storage management is a process for users to optimize the use of storage devices and to protect the integrity of data for any media on which it resides and the category of storage management generally contain the different type of subcategories covering aspects such as security, virtualization and more, as well as different types of provisioning or automation, which is generally made up the entire storage management software market.

Storage management key attributes:

Storage management has some key attribute which is generally used to manage the storage capacity of the system. These are given below:

1. Performance
2. Reliability
3. Recoverability
4. Capacity

Feature of Storage management:

There is some feature of storage management which is provided for storage capacity. These are given below:

Storage management is a process that is used to optimize the use of storage devices.

Storage management must be allocated and managed as a resource in order to truly benefit a corporation.

Storage management is generally a basic system component of information systems.

It is used to improve the performance of their data storage resources.

Advantage of storage management:

There are some advantage of storage management which are given below:

It is very simple to managed a storage capacity.

It is generally take a less time.

It is improve the performance of system.

In virtualization and automation technologies can help an organization improve its agility.

File Systems in Operating System

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.

The name of the file is divided into two parts as shown below:

name

extension, separated by a period.

FILE DIRECTORIES:

Collection of files is a file directory. The directory contains information about the files, including attributes, location and ownership. Much of this information, especially that is concerned with storage, is managed by the operating system. The directory is itself a file, accessible by various file management routines.

Information contained in a device directory are:

Name

Type

Address

Current length

Maximum length

Date last accessed

Date last updated

Owner id

Protection information

Operation performed on directory are:

Search for a file

Create a file

Delete a file

List a directory

Rename a file

Traverse the file system

Advantages of maintaining directories are:

Efficiency: A file can be located more quickly.

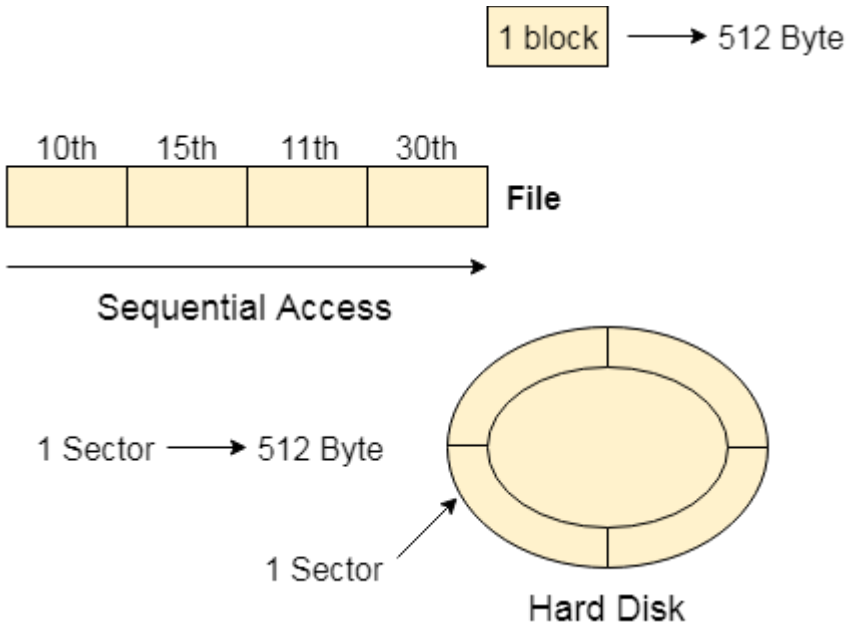
Naming: It becomes convenient for users as two users can have same name for different files or may have different name for same file.

Grouping: Logical grouping of files can be done by properties e.g. all java programs, all games etc.

File Access Methods

Let's look at various ways to access files stored in secondary memory.

Sequential Access



Most of the operating systems access the file sequentially. In other words, we can say that most of the files need to be accessed sequentially by the operating system.

In sequential access, the OS read the file word by word. A pointer is maintained which initially points to the base address of the file. If the user wants to read first word of the file then the pointer provides that word to the user and increases its value by 1 word. This process continues till the end of the file.

Modern word systems do provide the concept of direct access and indexed access but the most used method is sequential access due to the fact that most of the files such as text files, audio files, video files, etc need to be sequentially accessed.

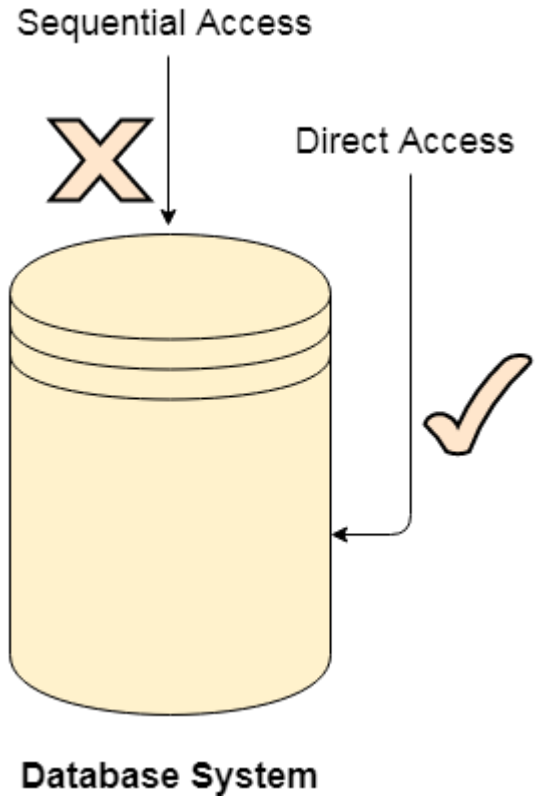
Direct Access

The Direct Access is mostly required in the case of database systems. In most of the cases, we need filtered information from the database. The sequential access can be very slow and inefficient in such cases.

Suppose every block of the storage stores 4 records and we know that the record we needed is stored in 10th block. In that case, the sequential access will not be

implemented because it will traverse all the blocks in order to access the needed record.

Direct access will give the required result despite of the fact that the operating system has to perform some complex tasks such as determining the desired block number. However, that is generally implemented in database applications.



Indexed Access

If a file can be sorted on any of the filed then an index can be assigned to a group of certain records. However, A particular record can be accessed by its index. The index is nothing but the address of a record in the file.

In index accessing, searching in a large database became very quick and easy but we need to have some extra space in the memory to store the index value.

Directory structure file system system mounting

File system is the part of the operating system which is responsible for file management. It provides a mechanism to store the data and access to the file contents including data and programs. Some Operating systems treats everything as a file for example Ubuntu.

The File system takes care of the following issues

File Structure

We have seen various data structures in which the file can be stored. The task of the file system is to maintain an optimal file structure.

Recovering Free space

Whenever a file gets deleted from the hard disk, there is a free space created in the disk. There can be many such spaces which need to be recovered in order to reallocate them to other files.

disk space assignment to the files

The major concern about the file is deciding where to store the files on the hard disk. There are various disks scheduling algorithm which will be covered later in this tutorial.

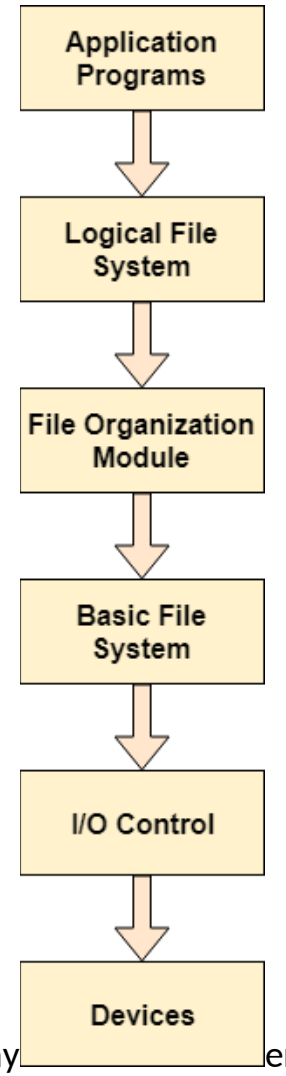
tracking data location

A File may or may not be stored within only one block. It can be stored in the non contiguous blocks on the disk. We need to keep track of all the blocks on which the part of the files reside.

File System Structure

File System provide efficient access to the disk by allowing data to be stored, located and retrieved in a convenient way. A file System must be able to store the file, locate the file and retrieve the file.

Most of the Operating Systems use layering approach for every task including file systems. Every layer of the file system is responsible for some activities.

The image shown below, elaborates how the file system is divided in different layers, and also the functionality of each lay

er.

File System Structure

File System provide efficient access to the disk by allowing data to be stored, located and retrieved in a convenient way. A file System must be able to store the file, locate the file and retrieve the file.

Most of the Operating Systems use layering approach for every task including file systems. Every layer of the file system is responsible for some activities.

The image shown below, elaborates how the file system is divided in different layers, and also the functionality of each layer.

When an application program asks for a file, the first request is directed to the logical file system. The logical file system contains the Meta data of the file and directory structure. If the application program doesn't have the required permissions of the file then this layer will throw an error. Logical file systems also verify the path to the file.

Generally, files are divided into various logical blocks. Files are to be stored in the hard disk and to be retrieved from the hard disk. Hard disk is divided into various tracks and sectors. Therefore, in order to store and retrieve the files, the logical blocks need to be mapped to physical blocks. This mapping is done by File organization module. It is also responsible for free space management.

Once File organization module decided which physical block the application program needs, it passes this information to basic file system. The basic file system is responsible for issuing the commands to I/O control in order to fetch those blocks.

I/O controls contain the codes by using which it can access hard disk. These codes are known as device drivers. I/O controls are also responsible for handling interrupts.

File Sharing in Operating System

File sharing, also known as file-swapping is the accessing or sharing of files by one or more users. It is performed on computer networks as a quick way to transmit data. Generally, a file-sharing system usually has more than one administrator, where the users may have the same or different access privileges.

It also implies having an allocated amount of personal files in the common storage.

File sharing has been used in mainframe and multi-user computer systems for many years, and now with widespread access to the internet, a file transfer system known as the File-Transfer Protocol or FTP is widely used.

The world wide web in itself can be considered as a large-scale file-sharing system where files are constantly downloaded or viewed by the web user.

However, in the file-sharing system, the user can read and write in the files, and if the entire system is not available for access, at least the common file that is shared by the owner can be read and written onto.

Certain issues arise when multiple users are allowed to access the files. The system can either grant access to the users by default or requires the user to specifically grant access to the files. This causes problems of control and protection, and to implement sharing and protection, the system needs to maintain more file and directory attributes than it would have done in the case of a single-user system.

System Protection in Operating System

Protection refers to a mechanism which controls the access of programs, processes, or users to the resources defined by a computer system. We can take protection as a helper to multi programming operating system, so that many users might safely share a common logical name space such as directory or files.

Need of Protection:

To prevent the access of unauthorized users and

To ensure that each active programs or processes in the system uses resources only as the stated policy,

To improve reliability by detecting latent errors.

Role of Protection:

The role of protection is to provide a mechanism that implement policies which defines the uses of resources in the computer system.Some policies are defined at the time of design of the system, some are designed by management of the system and some are defined by the users of the system to protect their own files and programs.

Every application has different policies for use of the resources and they may change over time so protection of the system is not only concern of the designer of the operating system. Application programmer should also design the protection mechanism to protect their system against misuse.

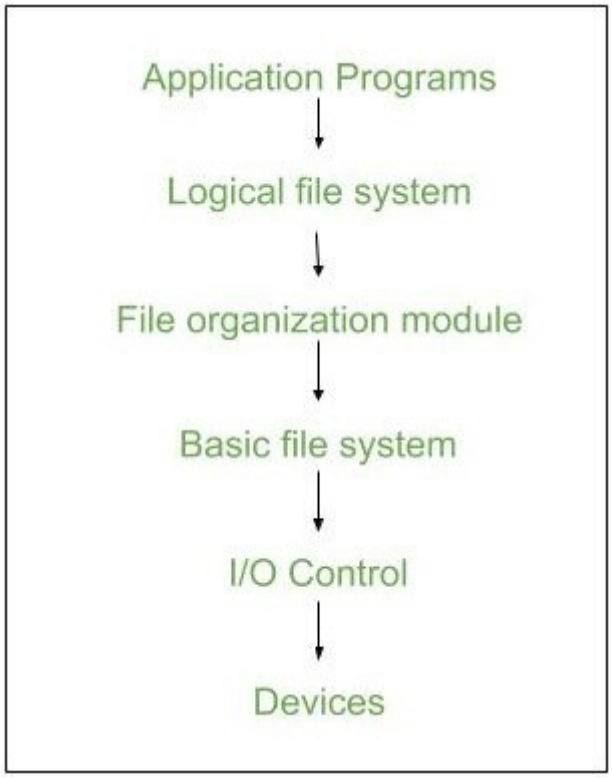
Policy is different from mechanism. Mechanisms determine how something will be done and policies determine what will be done.Policies are changed over time and place to place. Separation of mechanism and policy is important for the flexibility of the system.

File System Implementation in Operating System

Prerequisite – File Systems in Operating Systems

A file is a collection of related information. The file system resides on secondary storage and provides efficient and convenient access to the disk by allowing data to be stored, located, and retrieved.

File system organized in many layers :



I/O Control level –

Device drivers acts as interface between devices and Os, they help to transfer data between disk and main memory. It takes block number a input and as output it gives low level hardware specific instruction.

Basic file system –

It Issues general commands to device driver to read and write physical blocks on disk.It manages the memory buffers and caches. A block in buffer can hold the contents of the disk block and cache stores frequently used file system metadata.

File organization Module –

It has information about files, location of files and their logical and physical blocks.Physical blocks do not match with logical numbers of logical block numbered from 0 to N. It also has a free space which tracks unallocated blocks.

Logical file system –

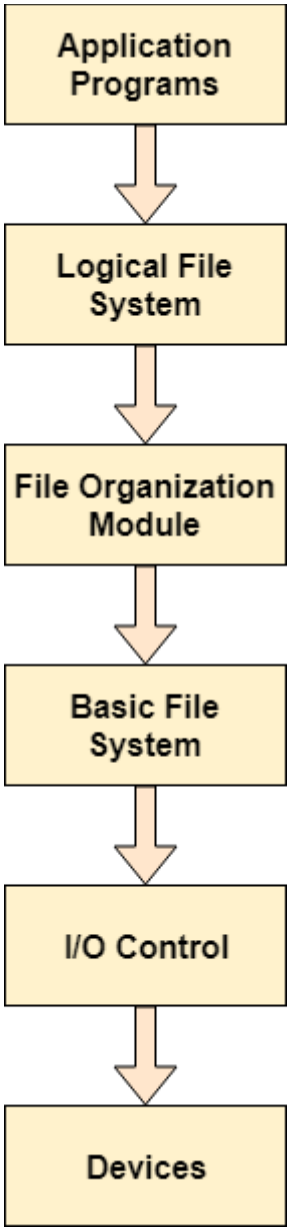
It manages metadata information about a file i.e includes all details about a file except the actual contents of file. It also maintains via file control blocks. File control block (FCB) has information about a file – owner, size, permissions, location of file contents.\

File System Structure

File System provide efficient access to the disk by allowing data to be stored, located and retrieved in a convenient way. A file System must be able to store the file, locate the file and retrieve the file.

Most of the Operating Systems use layering approach for every task including file systems. Every layer of the file system is responsible for some activities.

The image shown below, elaborates how the file system is divided in different layers, and also the functionality of each layer.



When an application program asks for a file, the first request is directed to the logical file system. The logical file system contains the Meta data of the file and directory structure. If the application program doesn't have the required permissions of the file then this layer will throw an error. Logical file systems also verify the path to the file.

Generally, files are divided into various logical blocks. Files are to be stored in the hard disk and to be retrieved from the hard disk. Hard disk is divided into various tracks and sectors. Therefore, in order to store and retrieve the files, the logical blocks need to be mapped to physical blocks. This mapping is done by File organization module. It is also responsible for free space management.

Once File organization module decided which physical block the application program needs, it passes this information to basic file system. The basic file system is responsible for issuing the commands to I/O control in order to fetch those blocks.

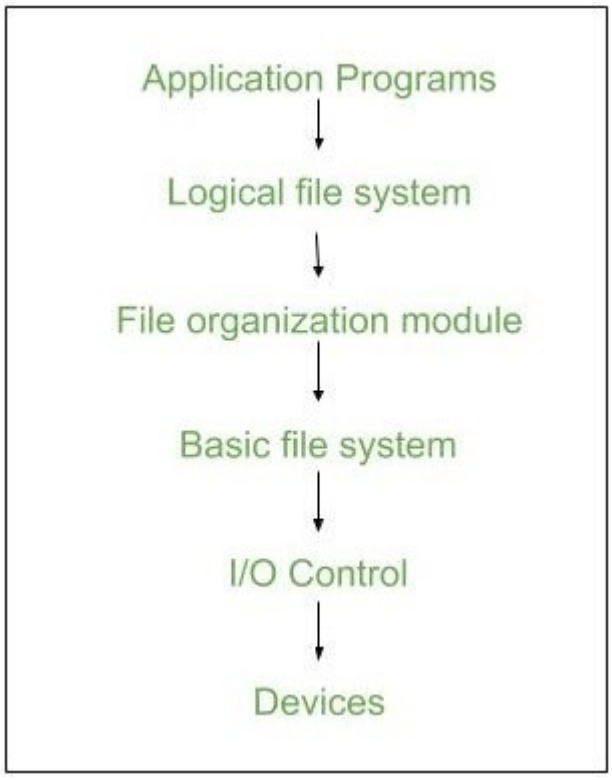
I/O controls contain the codes by using which it can access hard disk. These codes are known as device drivers. I/O controls are also responsible for handling interrupts.

File System Implementation in Operating System

Prerequisite – File Systems in Operating Systems

A file is a collection of related information. The file system resides on secondary storage and provides efficient and convenient access to the disk by allowing data to be stored, located, and retrieved.

File system organized in many layers :



I/O Control level –

Device drivers acts as interface between devices and Os, they help to transfer data between disk and main memory. It takes block number a input and as output it gives low level hardware specific instruction.

Basic file system –

It Issues general commands to device driver to read and write physical blocks on disk.It manages the memory buffers and caches. A block in buffer can hold the contents of the disk block and cache stores frequently used file system metadata.

File organization Module –

It has information about files, location of files and their logical and physical blocks.Physical blocks do not match with logical numbers of logical block numbered from 0 to N. It also has a free space which tracks unallocated blocks.

Logical file system –

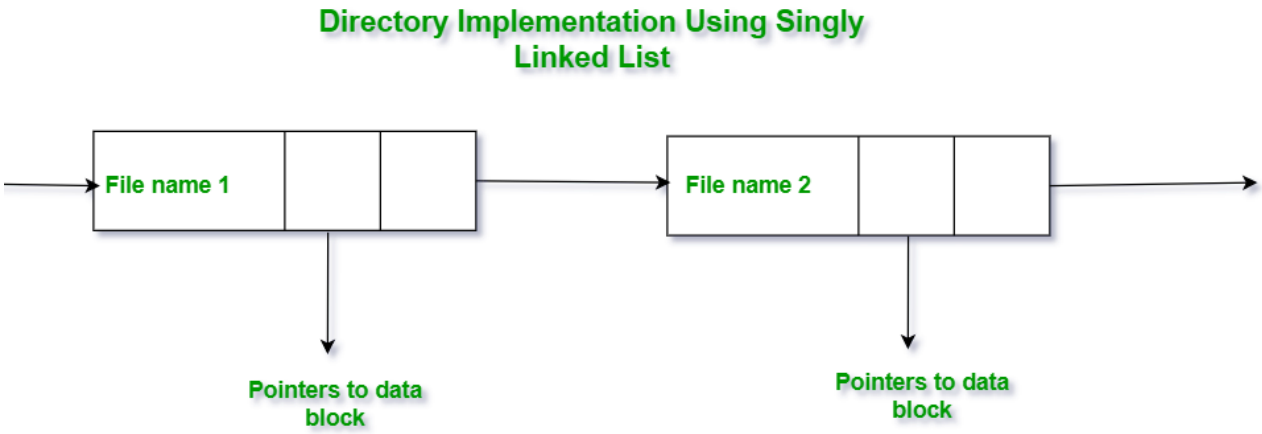
It manages metadata information about a file i.e includes all details about a file except the actual contents of file. It also maintains via file control blocks. File control block (FCB) has information about a file – owner, size, permissions, location of file contents.

Directory Implementation in Operating System

Directory implementation in the operating system can be done using Singly Linked List and Hash table. The efficiency, reliability, and performance of a file system are greatly affected by the selection of directory-allocation and directory-management algorithms. There are numerous ways in which the directories can be implemented. But we need to choose an appropriate directory implementation algorithm that enhances the performance of the system.

Directory Implementation using Singly Linked List

The implementation of directories using a singly linked list is easy to program but is time-consuming to execute. Here we implement a directory by using a linear list of filenames with pointers to the data blocks.



To create a new file the entire list has to be checked such that the new directory does not exist previously.

The new directory then can be added to the end of the list or at the beginning of the list.

In order to delete a file, we first search the directory with the name of the file to be deleted. After searching we can delete that file by releasing the space allocated to it.

To reuse the directory entry we can mark that entry as unused or we can append it to the list of free directories.

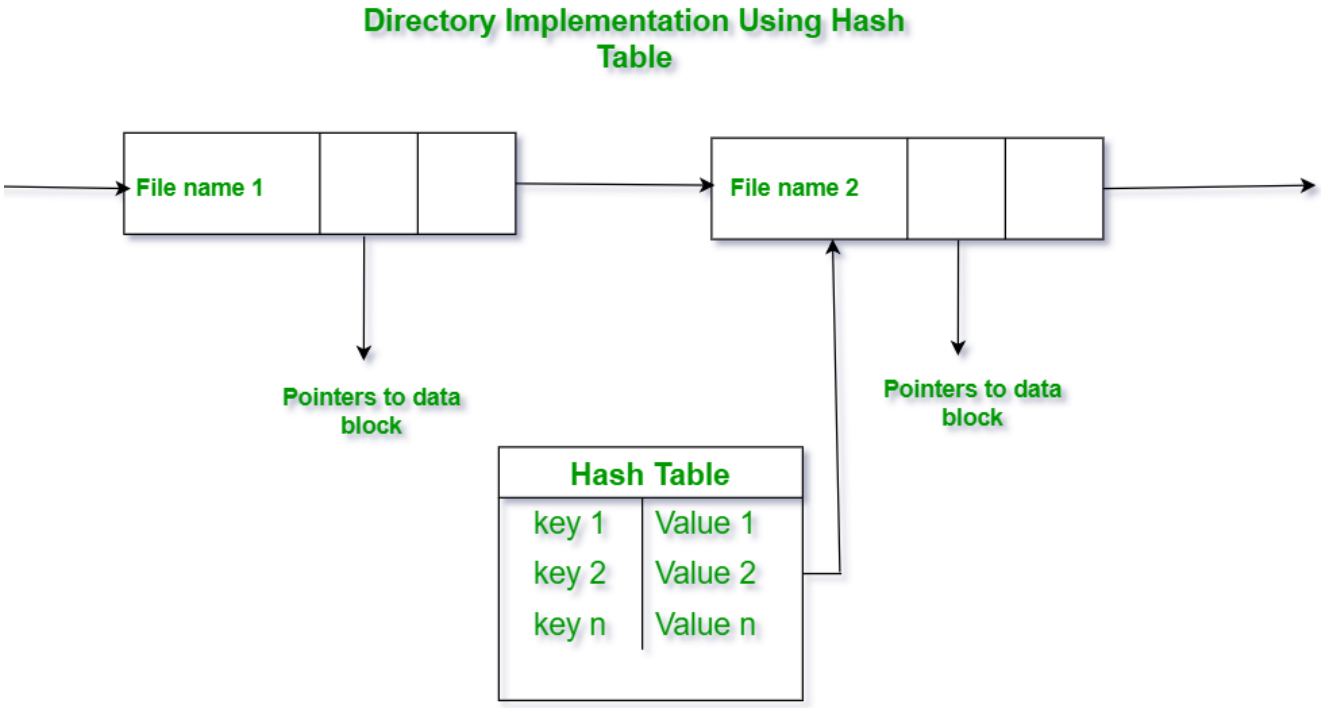
To delete a file linked list is the best choice as it takes less time.

Directory Implementation using Hash Table

An alternative data structure that can be used for directory implementation is a hash table. It overcomes the major drawbacks of directory implementation using a linked list. In this method, we use a hash table along with the linked list. Here the linked list stores the directory entries, but a hash data structure is used in combination with the linked list.

In the hash table for each pair in the directory key-value pair is generated. The hash function on the file name determines the key and this key points to the corresponding file stored in the directory. This method efficiently decreases the directory search time as the entire list will not be searched on every operation.

Using the keys the hash table entries are checked and when the file is found it is fetched.



File Allocation Methods

he allocation methods define how the files are stored in the disk blocks. There are three main disk space or file allocation methods.

Contiguous Allocation

Linked Allocation

Indexed Allocation

The main idea behind these methods is to provide:

Efficient disk space utilization.

Fast access to the file blocks.

All the three methods have their own advantages and disadvantages as discussed below:

1. Contiguous Allocation

In this scheme, each file occupies a contiguous set of blocks on the disk. For example, if a file requires n blocks and is given a block b as the starting location, then the blocks assigned to the file will be: b, b+1, b+2,……b+n-1. This means that given the starting block address and the length of the file (in terms of blocks required), we can determine the blocks occupied by the file.

The directory entry for a file with contiguous allocation contains

Address of starting block

Length of the allocated portion.

The file ‘mail’ in the following figure starts from the block 19 with length = 6 blocks. Therefore, it occupies 19, 20, 21, 22, 23, 24 blocks.



Directory

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

Advantages:

Both the Sequential and Direct Accesses are supported by this. For direct access, the address of the kth block of the file which starts at block b can easily be obtained as (b+k).

This is extremely fast since the number of seeks are minimal because of contiguous allocation of file blocks.

Disadvantages:

This method suffers from both internal and external fragmentation. This makes it inefficient in terms of memory utilization.

Increasing file size is difficult because it depends on the availability of contiguous memory at a particular instance.

Free space management in Operating System

The system keeps tracks of the free disk blocks for allocating space to files when they are created. Also, to reuse the space released from deleting the files, free space management becomes crucial. The system maintains a free space list which keeps track of the disk blocks that are not allocated to some file or directory. The free space list can be implemented mainly as:

Bitmap or Bit vector –

A Bitmap or Bit Vector is series or collection of bits where each bit corresponds to a disk block. The bit can take two values: 0 and 1: 0 indicates that the block is allocated and 1 indicates a free block.

The given instance of disk blocks on the disk in Figure 1 (where green blocks are allocated) can be represented by a bitmap of 16 bits as: 0000111000000110.

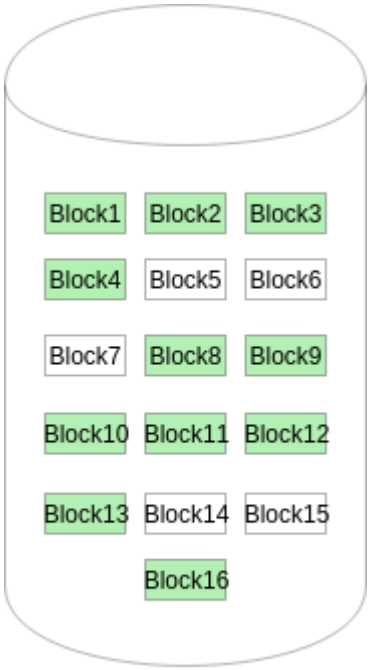


Figure - 1

Linked List –

In this approach, the free disk blocks are linked together i.e. a free block contains a pointer to the next free block. The block number of the very first disk block is stored at a separate location on disk and is also cached in memory.

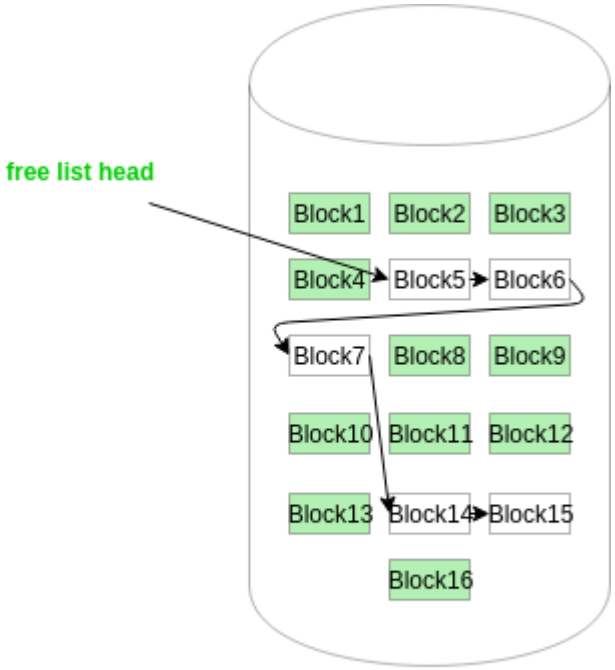


Figure - 2

In Figure-2, the free space list head points to Block 5 which points to Block 6, the next free block and so on. The last free block would contain a null pointer indicating the end of free list.

A drawback of this method is the I/O required for free space list traversal.

Grouping –

This approach stores the address of the free blocks in the first free block. The first free block stores the address of some, say n free blocks. Out of these n blocks, the first n-1 blocks are actually free and the last block contains the address of next free n blocks.

An advantage of this approach is that the addresses of a group of free disk blocks can be found easily.

Counting –

This approach stores the address of the first free disk block and a number n of free contiguous disk blocks that follow the first block.

Every entry in the list would contain:

Address of first free disk block

A number n

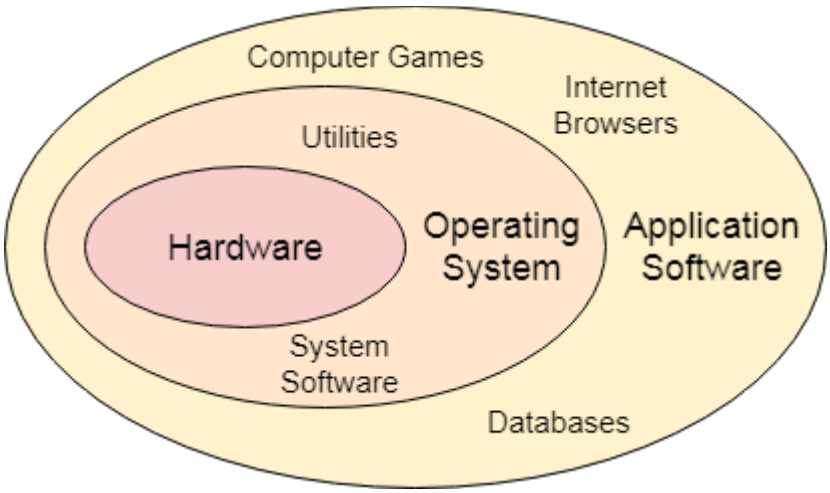
For example, in Figure-1, the first entry of the free space list would be: ([Address of Block 5], 2), because 2 contiguous free blocks follow block 5.

Efficiency and performance

Operating System Definition and Function

In the Computer System (comprises of Hardware and software), Hardware can only understand machine code (in the form of 0 and 1) which doesn't make any sense to a naive user.

We need a system which can act as an intermediary and manage all the processes and resources present in the system.



An Operating System can be defined as an interface between user and hardware. It is responsible for the execution of all the processes, Resource Allocation, CPU management, File Management and many other tasks.

The purpose of an operating system is to provide an environment in which a user can execute programs in convenient and efficient manner.

Structure of a Computer System

A Computer System consists of:

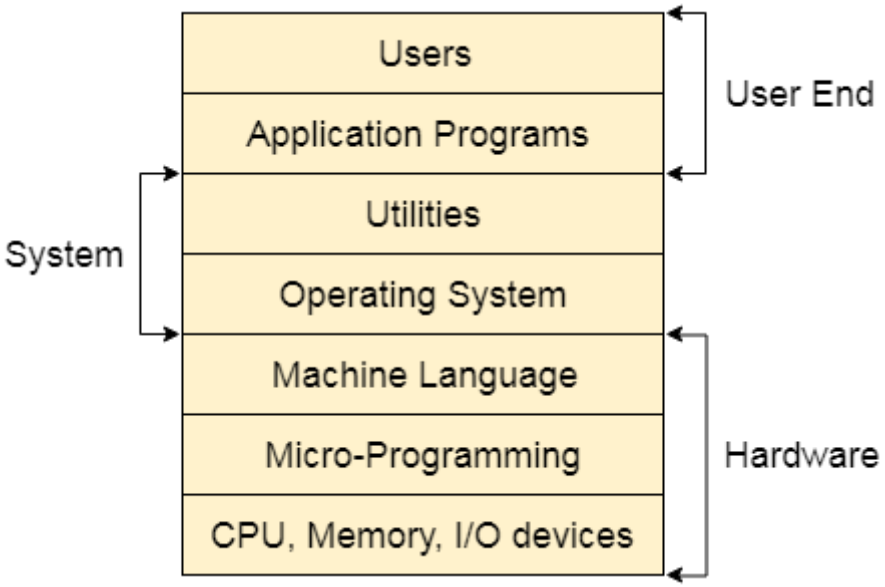
Users (people who are using the computer)

Application Programs (Compilers, Databases, Games, Video player, Browsers, etc.)

System Programs (Shells, Editors, Compilers, etc.)

Operating System (A special program which acts as an interface between user and hardware)

Hardware (CPU, Disks, Memory, etc)



What does an Operating system do?

Process Management

Process Synchronization

Memory Management

CPU Scheduling

File Management

Security

Recovery in Operating System

Prerequisite – Deadlock Detection And Recovery

When a Deadlock Detection Algorithm determines that a deadlock has occurred in the system, the system must recover from that deadlock. There are two approaches of breaking a Deadlock:

1. Process Termination:

To eliminate the deadlock, we can simply kill one or more processes. For this, we use two methods:

(a). Abort all the Deadlocked Processes:

Aborting all the processes will certainly break the deadlock, but with a great expense. The deadlocked processes may have computed for a long time and the result of those partial computations must be discarded and there is a probability to recalculate them later.

(b). Abort one process at a time until deadlock is eliminated:

Abort one deadlocked process at a time, until deadlock cycle is eliminated from the system. Due to this method, there may be considerable overhead, because after aborting each process, we have to run deadlock detection algorithm to check whether any processes are still deadlocked.

2. Resource Preemption:

To eliminate deadlocks using resource preemption, we preempt some resources from processes and give those resources to other processes. This method will raise three issues –

(a). Selecting a victim:

We must determine which resources and which processes are to be preempted and also the order to minimize the cost.

(b). Rollback:

We must determine what should be done with the process from which resources are preempted. One simple idea is total rollback. That means abort the process and restart it.

(c). Starvation:

In a system, it may happen that same process is always picked as a victim. As a result, that process will never complete its designated task. This situation is called Starvation and must be avoided. One solution is that a process must be picked as a victim only a finite number of times.