DEPARTMENT OF AIML

23CST202- OPERATING SYSTEMS

II YEAR IV SEM AIML-B

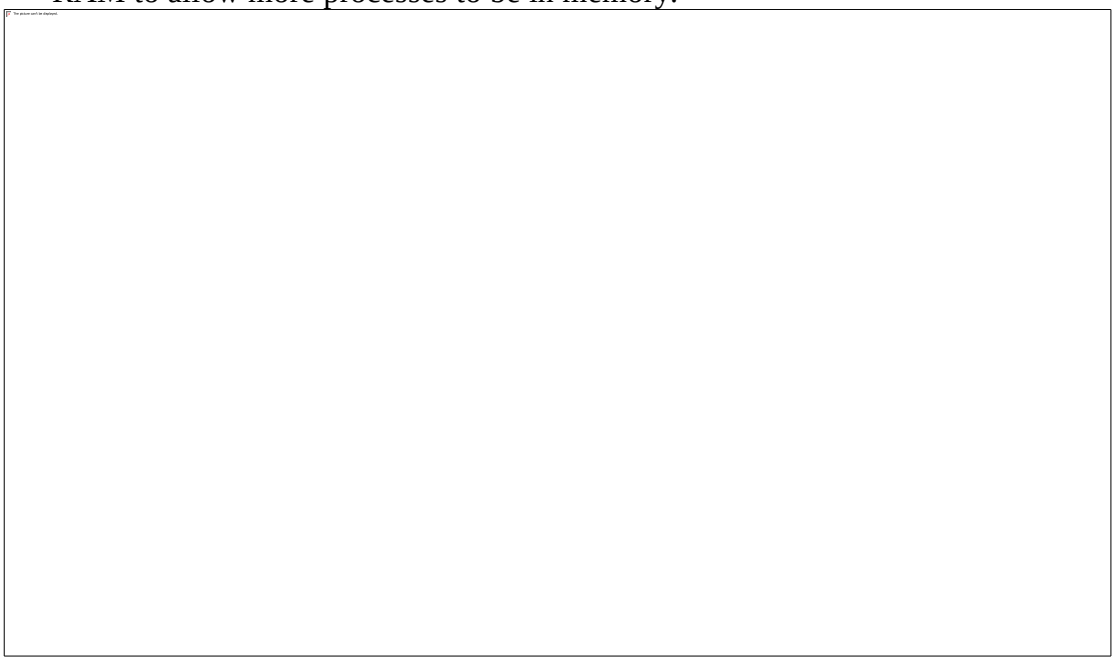
UNIT 3-MEMORY MANAGEMENT

TOPIC –VIRTUAL MEMORY BACKGROUND

VIRTUAL MEMORY BACKGROUND

Virtual memory is a memory management technique used by operating systems to give the appearance of a large, continuous block of memory to applications, even if the physical memory (RAM) is limited. It allows larger applications to run on systems with less RAM.

- The main objective of virtual memory is to support multiprogramming, The main advantage that virtual memory provides is, a running process does not need to be entirely in memory.
- Programs can be larger than the available physical memory. Virtual Memory provides an abstraction of main memory, eliminating concerns about storage limitations.
- A memory hierarchy, consisting of a computer system's memory and a disk, enables a process to operate with only some portions of its address space in RAM to allow more processes to be in memory.



A virtual memory is what its name indicates- it is an illusion of a memory that is larger than the real memory. We refer to the software component of virtual memory as a virtual memory manager. The basis of virtual memory is the noncontiguous memory allocation model. The virtual memory manager removes some components from memory to make room for other components.

The size of virtual storage is limited by the addressing scheme of the computer system and the amount of secondary memory available not by the actual number of main storage locations.

How Virtual Memory Works?

Virtual Memory is a technique that is implemented using both hardware and software. It maps memory addresses used by a program, called virtual addresses, into physical addresses in computer memory.

- All memory references within a process are logical addresses that are dynamically translated into physical addresses at run time. This means that a process can be swapped in and out of the main memory such that it occupies different places in the main memory at different times during the course of execution.
- A process may be broken into a number of pieces and these pieces need not be continuously located in the main memory during execution. The combination of dynamic run-time address translation and the use of a page or segment table permits this.

If these characteristics are present then, it is not necessary that all the pages or segments are present in the main memory during execution. This means that the required pages need to be loaded into memory whenever required. Virtual memory is implemented using Demand Paging or Demand Segmentation.

Types of Virtual Memory

In a computer, virtual memory is managed by the Memory Management Unit (MMU), which is often built into the CPU. The CPU generates virtual addresses that the MMU translates into physical addresses.

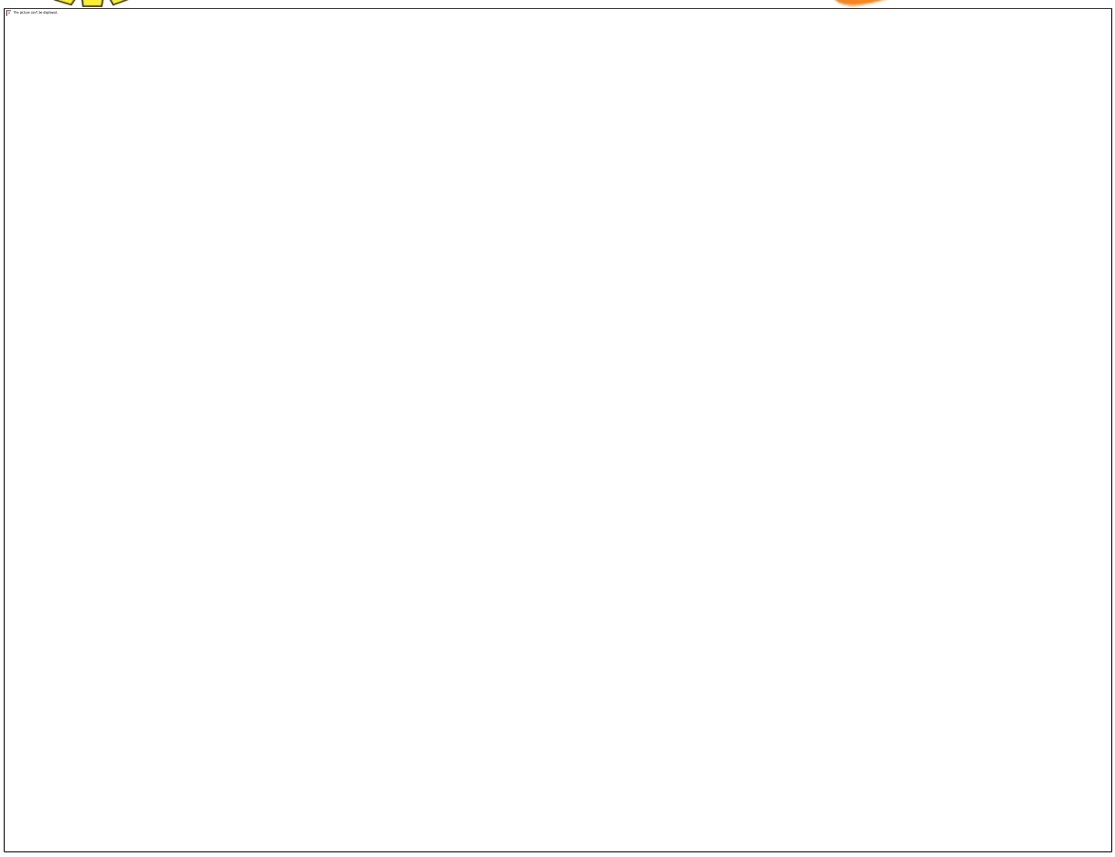
There are two main types of virtual memory:

- Paging
- Segmentation

Paging

Paging divides memory into small fixed-size blocks called pages. When the computer runs out of RAM, pages that aren't currently in use are moved to the hard drive, into an area called a swap file. The swap file acts as an extension of RAM. When a page is needed again, it is swapped back into RAM, a process known as page swapping. This ensures that the operating system (OS) and applications have enough memory to run.

Demand Paging: The process of loading the page into memory on demand (whenever a page fault occurs) is known as demand paging. The process includes the following steps are as follows:



- If the CPU tries to refer to a page that is currently not available in the main memory, it generates an interrupt indicating a memory access fault.
- The OS puts the interrupted process in a blocking state. For the execution to proceed the OS must bring the required page into the memory.
- The OS will search for the required page in the logical address space.
- The required page will be brought from logical address space to physical address space. The page replacement algorithms are used for the decision-making of replacing the page in physical address space.
- The page table will be updated accordingly.
- The signal will be sent to the CPU to continue the program execution and it will place the process back into the ready state.

Hence whenever a page fault occurs these steps are followed by the operating system and the required page is brought into memory.

What is Page Fault Service Time?

The time taken to service the page fault is called page fault service time. The page fault service time includes the time taken to perform all the above six steps.

Let Main memory access time is: m
Page fault service time is: s
Page fault rate is : p
Then, Effective memory access time = (p*s) + (1-p)*m

Page vs Frame

A page is a fixed size block of data in virtual memory and a frame is a fixed size block of physical memory in RAM where these pages are loaded. Think of a page as a piece of a puzzle (virtual memory) and a frame as the spot where it fits on the board (physical memory). When a program runs its pages are mapped to available frames so the program can run even if the program size is larger than physical memory.

Segmentation

Segmentation divides virtual memory into segments of different sizes. Segments that aren't currently needed can be moved to the hard drive. The system uses a segment table to keep track of each segment's status, including whether it's in memory, if it's been modified, and its physical address. Segments are mapped into a process's address space only when needed.

Combining Paging and Segmentation

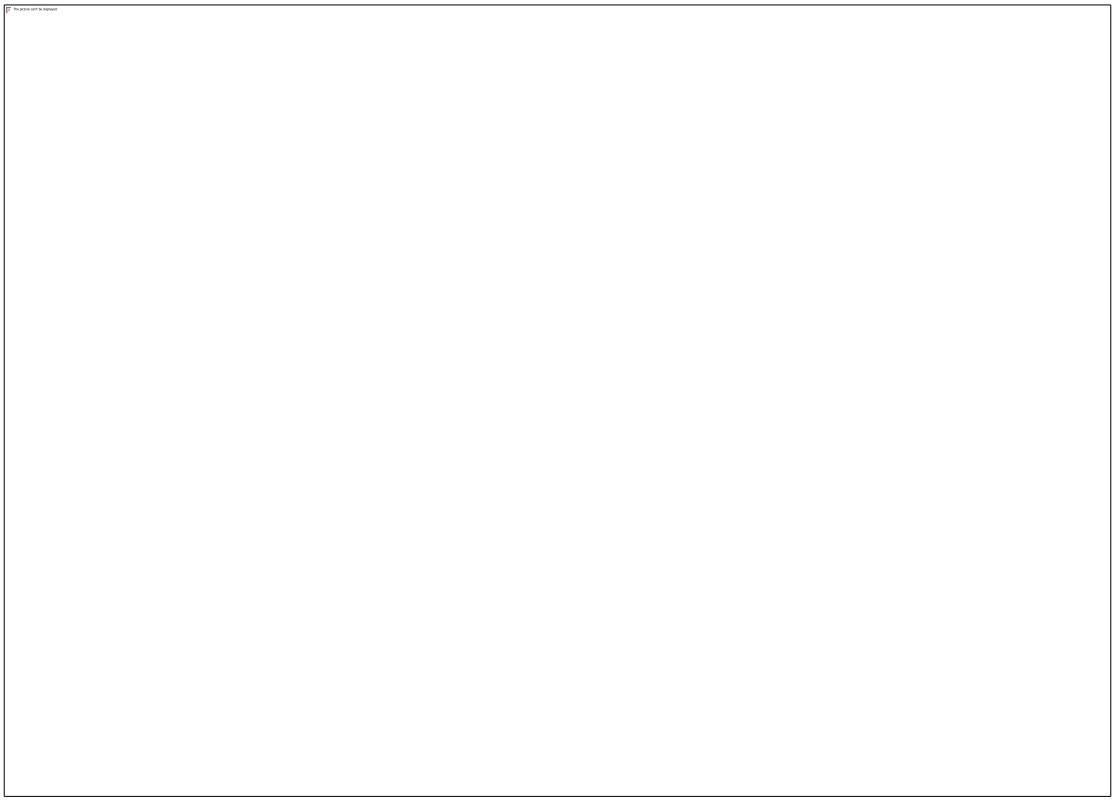
Sometimes, both paging and segmentation are used together. In this case, memory is divided into pages, and segments are made up of multiple pages. The virtual address includes both a segment number and a page number.

Virtual Memory vs Physical Memory

Feature	Virtual Memory	Physical Memory (RAM)
Definition	An abstraction that extends the available memory by using disk storage	The actual hardware (RAM) that stores data and instructions currently being used by the CPU
Location	On the hard drive or SSD	On the computer's motherboard
Speed	Slower (due to disk I/O operations)	Faster (accessed directly by the CPU)
Capacity	Larger, limited by disk space	Smaller, limited by the amount of RAM installed
Cost	Lower (cost of additional disk storage)	Higher (cost of RAM modules)
Data Access	Indirect (via paging and swapping)	Direct (CPU can access data directly)
Volatility	Non-volatile (data persists on disk)	Volatile (data is lost when power is off)

What is Swapping?

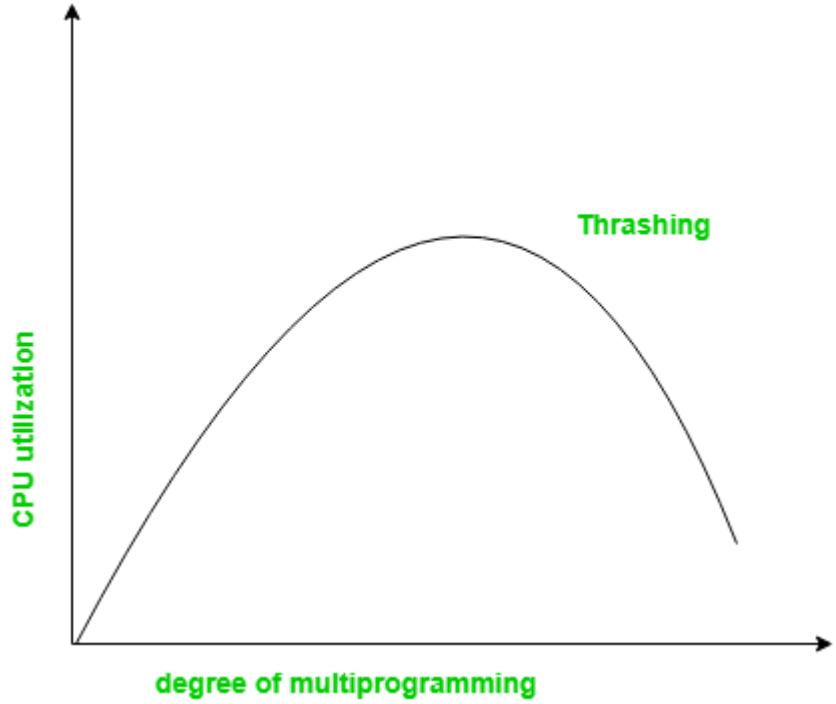
Swapping is a process out means removing all of its pages from memory, or marking them so that they will be removed by the normal page replacement process. Suspending a process ensures that it is not runnable while it is swapped out. At some later time, the system swaps back the process from the secondary storage to the main memory. When a process is busy swapping pages in and out then this situation is called thrashing.



Swappinghierar

What is Thrashing?

At any given time, only a few pages of any process are in the main memory, and therefore more processes can be maintained in memory. Furthermore, time is saved because unused pages are not swapped in and out of memory. However, the OS must be clever about how it manages this scheme. In the steady state practically, all of the main memory will be occupied with process pages, so that the processor and OS have direct access to as many processes as possible. Thus when the OS brings one page in, it must throw another out. If it throws out a page just before it is used, then it will just have to get that page again almost immediately. Too much of this leads to a condition called Thrashing. The system spends most of its time swapping pages rather than executing instructions. So a good page replacement algorithm is required.



In the given diagram, the initial degree of multiprogramming up to some extent of point(lambda), the CPU utilization is very high and the system resources are utilized 100%. But if we further increase the degree of multiprogramming the CPU utilization will drastically fall down and the system will spend more time only on the page replacement and the time taken to complete the execution of the process will increase. This situation in the system is called thrashing.

Causes of Thrashing

Thrashing occurs in a computer system when the CPU spends more time swapping pages in and out of memory than executing actual processes. This happens when there is insufficient physical memory, causing frequent page faults and excessive paging activity. Thrashing reduces system performance and makes processes run very slowly. There are many cause of thrashing as discussed below.

1. High Degree of Multiprogramming

If the number of processes keeps on increasing in the memory then the number of frames allocated to each process will be decreased. So, fewer frames will be available for each process. Due to this, a page fault will occur more frequently and more CPU time will be wasted in just swapping in and out of pages and the utilization will keep on decreasing.

For example:
Let free frames = 400
Case 1: Number of processes = 100
Then, each process will get 4 frames.

Case 2: Number of processes = 400
Each process will get 1 frame.

Case 2 is a condition of thrashing, as the number of processes is increased, frames per process are decreased. Hence CPU time will be consumed just by swapping pages.

2. Lacks of Frames

If a process has fewer frames then fewer pages of that process will be able to reside in memory and hence more frequent swapping in and out will be required. This may lead to thrashing. Hence a sufficient amount of frames must be allocated to each process in order to prevent thrashing.

Recovery of Thrashing

- Do not allow the system to go into thrashing by instructing the long-term scheduler not to bring the processes into memory after the threshold.
- If the system is already thrashing then instruct the mid-term scheduler to suspend some of the processes so that we can recover the system from thrashing.

Performance in Virtual Memory

- Let p be the page fault rate(0 <= p <= 1).
- if p = 0 no page faults
- if p =1, every reference is a fault.

Effective access time (EAT) = (1-p)* Memory Access Time + p * Page fault time.
Page fault time = page fault overhead + swap out + swap in +restart overhead

The performance of a virtual memory management system depends on the total number of page faults, which depend on "paging policies" and "frame allocation"

Frame Allocation

A number of frames allocated to each process in either static or dynamic.

- **Static Allocation:** The number of frame allocations to a process is fixed.
- **Dynamic Allocation:** The number of frames allocated to a process changes.

Paging Policies

- **Fetch Policy:** It decides when a page should be loaded into memory.
- **Replacement Policy:** It decides which page in memory should be replaced.
- **Placement Policy:** It decides where in memory should a page be loaded.

What are the Applications of Virtual memory?

Virtual memory has the following important characteristics that increase the capabilities of the computer system. The following are five significant characteristics of Lean.

- **Increased Effective Memory:** One major practical application of virtual memory is, virtual memory enables a computer to have more memory than the physical memory using the disk space. This allows for the running of larger applications and numerous programs at one time while not necessarily needing an equivalent amount of DRAM.
- **Memory Isolation:** Virtual memory allocates a unique address space to each process and that also plays a role in process segmentation. Such separation increases safety and reliability based on the fact that one process cannot interact with and or modify another's memory space through a mistake, or even a deliberate act of vandalism.

- **Efficient Memory Management:** Virtual memory also helps in better utilization of the physical memories through methods that include paging and segmentation. It can transfer some of the memory pages that are not frequently used to disk allowing RAM to be used by active processes when required in a way that assists in efficient use of memory as well as system performance.
- **Simplified Program Development:** For case of programmers, they don't have to consider physical memory available in a system in case of having virtual memory. They can program 'as if' there is one big block of memory and this makes the programming easier and more efficient in delivering more complex applications.

How to Manage Virtual Memory?

Here are 5 key points on how to manage virtual memory:

1. Adjust the Page File Size

- **Automatic Management**: All contemporary operating systems including Windows contain the auto-configuration option for the size of the empirical page file. But depending on the size of the RAM, they are set automatically, although the user can manually adjust the page file size if required.
- **Manual Configuration:** For tuned up users, the setting of the custom size can sometimes boost up the performance of the system. The initial size is usually advised to be set to the minimum value of 1. To set the size of the swap space equal to 5 times the amount of physical RAM and the maximum size 3 times the physical RAM.

2. Place the Page File on a Fast Drive

- **SSD Placement:** If this is feasible, the page file should be stored in the SSD instead of the HDD as a storage device. It has better read and write times, and the virtual memory may prove beneficial in an SSD.
- **Separate Drive:** Regarding systems having multiple drives involved, the page file needs to be placed on a different drive than the OS and that shall in turn improve its performance.

3. Monitor and Optimize Usage

- **Performance Monitoring:** Employ the software tools used in monitoring the performance of the system in tracking the amounts of virtual memory. High page file usage may signify that there is a lack of physical RAM or that virtual memory needs a change of settings or addition in physical RAM.
- **Regular Maintenance:** Make sure there is no toolbar or other application running in the background, take time and uninstall all the tool bars to free virtual memory.

4. Disable Virtual Memory for SSD

- **Sufficient RAM:** If for instance your system has a big physical memory, for example 16GB and above then it would be advised to freeze the page file in order to minimize SSD usage. But it should be done, in my opinion, carefully and only if the additional signals that one decides to feed into his applications should not likely use all the available RAM.

5. Optimize System Settings

- **System Configuration:** Change some general properties of the system concerning virtual memory efficiency. This also involves enabling additional control options in Windows such as adjusting additional system setting option on the operating system, or using other options in different operating systems such as Linux that provides different tools and commands to help in adjusting how virtual memory is utilized.
- **Regular Updates:** Ensure that your drivers are run in their newest version because new releases contain some enhancements and issues regarding memory management.

What are the Benefits of Using Virtual Memory?

- Many processes maintained in the main memory.
- A process larger than the main memory can be executed because of demand paging. The OS itself loads pages of a process in the main memory as required.
- It allows greater multiprogramming levels by using less of the available (primary) memory for each process.
- It has twice the capacity for addresses as main memory.
- It makes it possible to run more applications at once.
- Users are spared from having to add memory modules when RAM space runs out, and applications are liberated from shared memory management.
- When only a portion of a program is required for execution, speed has increased.
- Memory isolation has increased security.
- It makes it possible for several larger applications to run at once.
- Memory allocation is comparatively cheap.
- It doesn't require outside fragmentation.
- It is efficient to manage logical partition workloads using the CPU.
- Automatic data movement is possible.

What are the Limitation of Virtual Memory?

- It can slow down the system performance, as data needs to be constantly transferred between the physical memory and the hard disk.
- It can increase the risk of data loss or corruption, as data can be lost if the hard disk fails or if there is a power outage while data is being transferred to or from the hard disk.
- It can increase the complexity of the memory management system, as the operating system needs to manage both physical and virtual memory.