(ISLM-225)

Introduction
to
Operating System

Storage Management

Department of Information Science & Library Management (ISLM)
University of Rajshahi, Rajshahi-6205, Bangladesh

Unit-1: Operating System Overview: Introduction to OS. Operating system functions, evaluation of O.S., Different types of O.S.: batch, multi-programmed, time-sharing, real-time, distributed, parallel.

Unit-2: System Structure: Computer system operation, I/O structure, storage structure, storage hierarchy, different types of protections, operating system structure (simple, layered, virtual machine), O/S services, system calls.

Unit-3: Installing and Configuring OS: Introduction to Installation and Media Types, Performing a Custom OS Installation, Run Levels and the Startup/Shutdown Sequence, Logging In and Out of a Operating System.

Unit-4: Process Management: Processes- Concept of processes, process scheduling, operations on processes, co-operating processes, interprocess communication, Threads- overview, benefits of threads, user and kernel threads., CPU scheduling, process synchronization, deadlocks- system model, deadlock characterization, methods for handling deadlocks, deadlock prevention, deadlock avoidance, deadlock detection, recovery from deadlock.

Unit-5: Storage Management: Memory Management- background, logical vs. physical address space, swapping, contiguous memory allocation, paging, segmentation, segmentation with paging, Virtual Memory- background, demand paging, performance, page replacement, page replacement algorithms (FCFS, LRU), allocation of frames, thrashing, File Systems, I/O Management, Disk Management.

Unit-6: Distributed OS and File System: Motivation, Types of Network-based OS, Network structure, Distributed File System- Background, Naming and transparency, Remote File Access, State full and Stateless services. Distributed Synchronization: Event Ordering, Mutual Exclusion, Atomicity, Concurrency Control, Deadlock Handling, Election algorithm and Reaching agreement.

Memory

- Memory is the electronic holding place for instructions and data that the computer's microprocessor can reach quickly.
- When the computer is in normal operation, its memory usually contains the main parts of the operating system and some or all of the application programs and related data that are being used.
- Memory is often used as a shorter synonym for random access memory (RAM). This kind of memory is located on one or more microchips that are physically close to the microprocessor in the computer.

Memory

- Most desktop and notebook computers sold today include at least 2GB of RAM, and are upgradeable to include more.
- The more RAM you have, the less frequently the computer has to access instructions and data from the more slowly accessed hard disk form of storage.

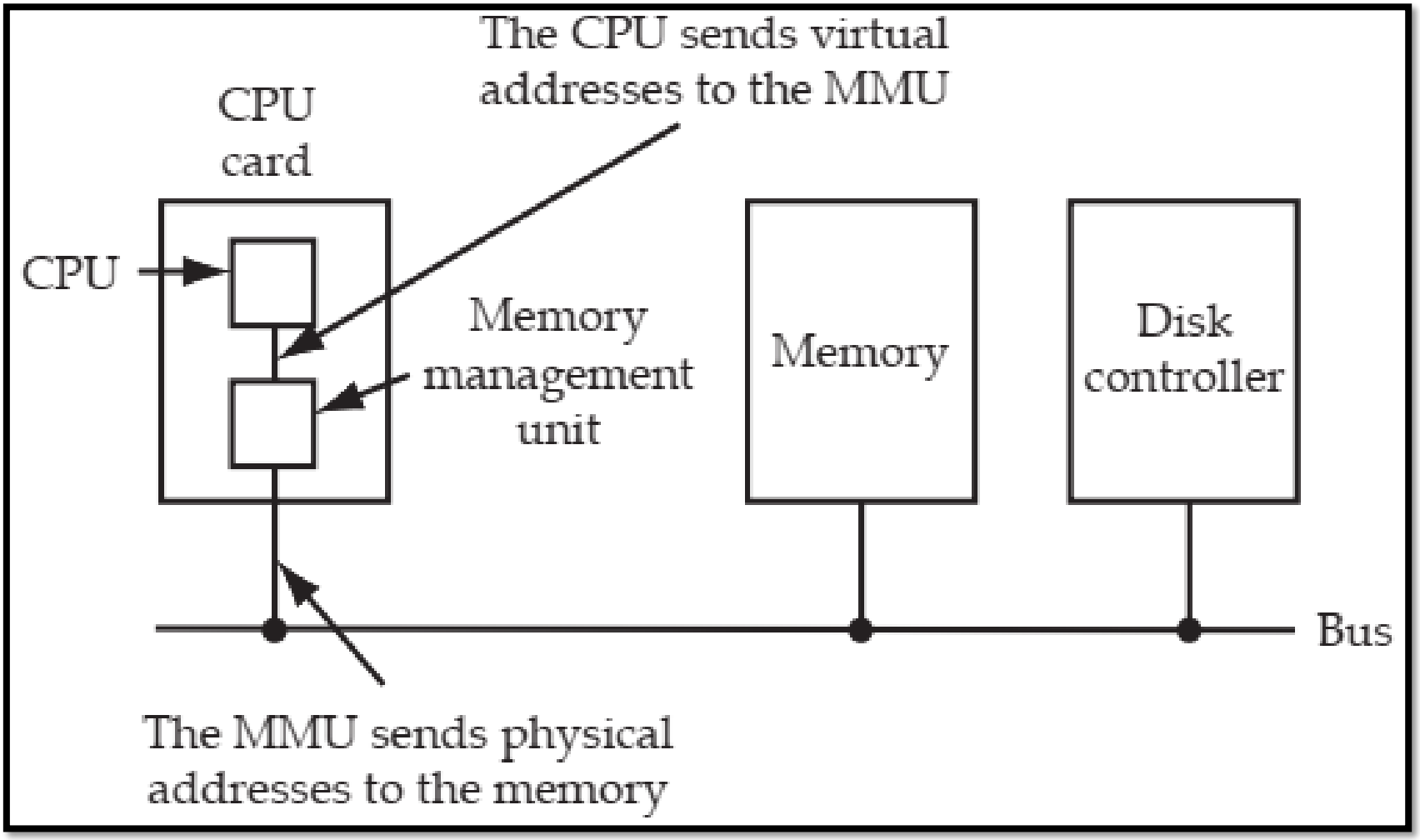
Memory

- Memory is sometimes distinguished from storage, or the physical medium that holds the much larger amounts of data that won't fit into RAM and may not be immediately needed there.
- Storage devices include hard disks, floppy disks, CD-ROM, and tape backup systems.
- The terms auxiliary storage, auxiliary memory, and secondary memory have also been used for this kind of data repository.
- Additional kinds of integrated and quickly accessible memory are read-only memory (ROM), programmable ROM (PROM), erasable programmable ROM (EPROM).

Memory Management

- In addition to the responsibility of managing processes, the OS must efficiently manage the primary memory of the computer.
- The part of the operating system which handles this responsibility is called the **memory manager**.
- Since every process must have some amount of primary memory in order to execute, the performance of the memory manager is crucial to the performance of the entire system.

Memory Management



Prepare it's description with your own word……

Accomplished Tasks of Memory Management

- When an operating system manages the computer's memory, there are two broad tasks to be accomplished:
  - *Each process must have enough memory in which to execute, and it can neither run into the memory space of another process nor be run into by another process.*
  - *The different types of memory in the system must be used properly so that each process can run most effectively.*
- The first task requires the OS to set up memory boundaries for types of software and for individual applications.

Virtual Memory Management

• In most PCs, it's possible to add memory beyond the original capacity.

•For example, you might expand RAM from 1 to 2 Gigabytes. This works fine, but tends to be relatively expensive.

•It also ignores a fundamental fact of computing - most of the information that an application stores in memory is not being used at any given moment.

•A processor can only access memory one location at a time, so the vast majority of RAM is unused at any moment.

•Since disk space is cheap compared to RAM, then moving information in RAM to hard disk can greatly expand RAM space at no cost.

•This technique is called virtual memory management.

Types of Memory in Computers

•Disk storage is only one of the memory types that must be managed by the operating system, and is the slowest.

•Ranked in order of speed, the types of memory in a computer system are:

- ***High-speed cache:*** This is fast, relatively small amounts of memory that are available to the CPU through the fastest connections. Cache controllers predict which pieces of data the CPU will need next and pull it from main memory into high-speed cache to speed up system performance.
- ***Main memory:*** This is the RAM that you see measured in megabytes when you buy a computer.
- ***Secondary memory:*** This is most often some sort of rotating magnetic storage that keeps applications and data available to be used, and serves as virtual RAM under the control of the operating system.

Physical Address

•A physical address, also real address, or binary address, is the memory address, that is electronically (in the form of binary number) presented on the computer address bus circuitry in order to enable the data bus to access a particular storage cell of main memory.

Logical Address

•The logical address is virtual address as it does not exist physically, therefore, it is also known as Virtual Address. This address is used as a reference to access the physical memory location by CPU.

•The term Logical Address Space is used for the set of all logical addresses generated by a program's perspective.

•Logical Address is generated by CPU while a program is running.

Logical Vs. Physical Address

Parameter	LOGICAL ADDRESS	PHYSICAL ADDRESS
Basic	generated by CPU	location in a memory unit
Address Space	Logical Address Space is set of all logical addresses generated by CPU in reference to a program.	Physical Address is set of all physical addresses mapped to the corresponding logical addresses.
Visibility	User can view the logical address of a program.	User can never view physical address of program.
Generation	generated by the CPU	Computed by MMU
Access	The user can use the logical address to access the physical address.	The user can indirectly access physical address but not directly.
Editable	Logical address can be change.	Physical address will not change.
Also called	virtual address.	real address.

Swapping

•Swapping is a memory management technique for swapping data between main memory and secondary memory for better memory utilization.

• It can be used to increase the operating system's performance.

Swapping-Example

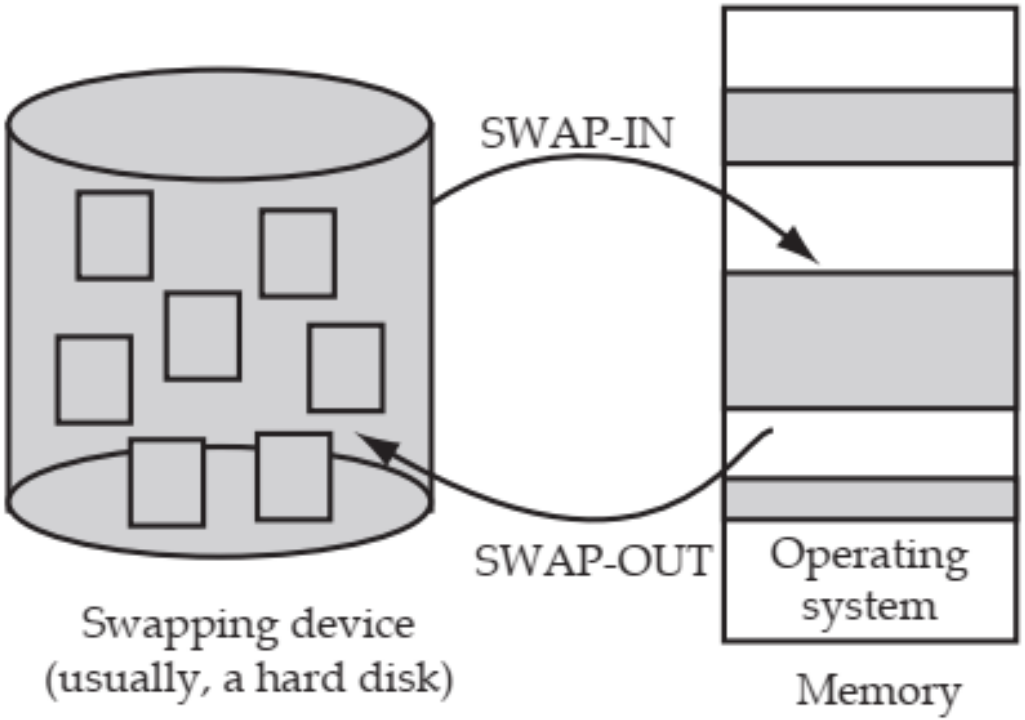
- When Windows operating system starts, many processes start running as the system starts the booting process. These processes do-
  - Checking for application updates.
  - Checking the incoming network, incoming mail, and so on.
- All these processes are brought into the main memory.
- Today's single-user applications require a lot of disk space to boot.
- Therefore, keeping all these processes in the main memory requires a lot of main memory.
- Also, as the main memory increases, the cost of the system increases.
- To deal with these kinds of memory overloads, we have come up with **swapping**.

Swapping

•To move a program from fast-access memory to a slow-access memory is known as "**swap out**", and

•The reverse operation is known as "**swap in**".

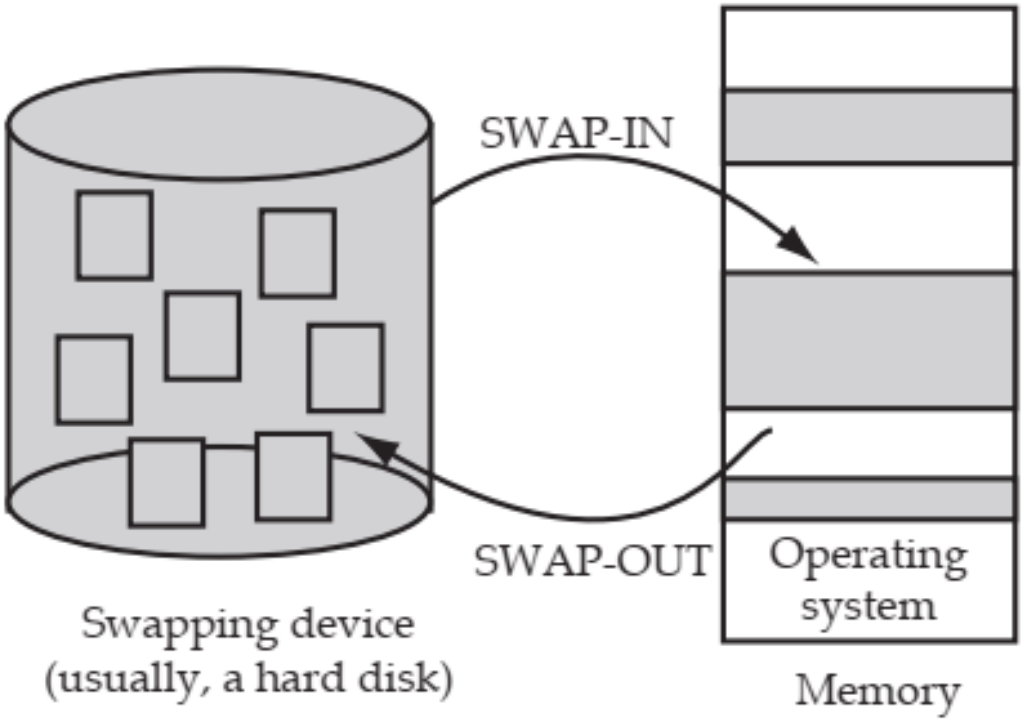
•The term often refers specifically to the use of a hard disk (or a swap file) as virtual memory or "swap space".



Swapping

•When a program is to be executed, possibly as determined by a scheduler, it is swapped into core for processing;

•when it can no longer continue executing for some reason, or the scheduler decides its time slice has expired, it is swapped out again.



Contiguous Memory Allocation

- The real challenge of efficiently managing memory is seen in the case of a system which has multiple processes running at the same time.
- Since primary memory can be space-multiplexed, the memory manager can allocate a portion of primary memory to each process for its own use.

Contiguous Memory Allocation

•However, the memory manager must keep track of which processes are running in which memory locations, and it must also determine how to allocate and deallocate available memory when new processes are created and when old processes complete execution.

Contiguous Memory Allocation

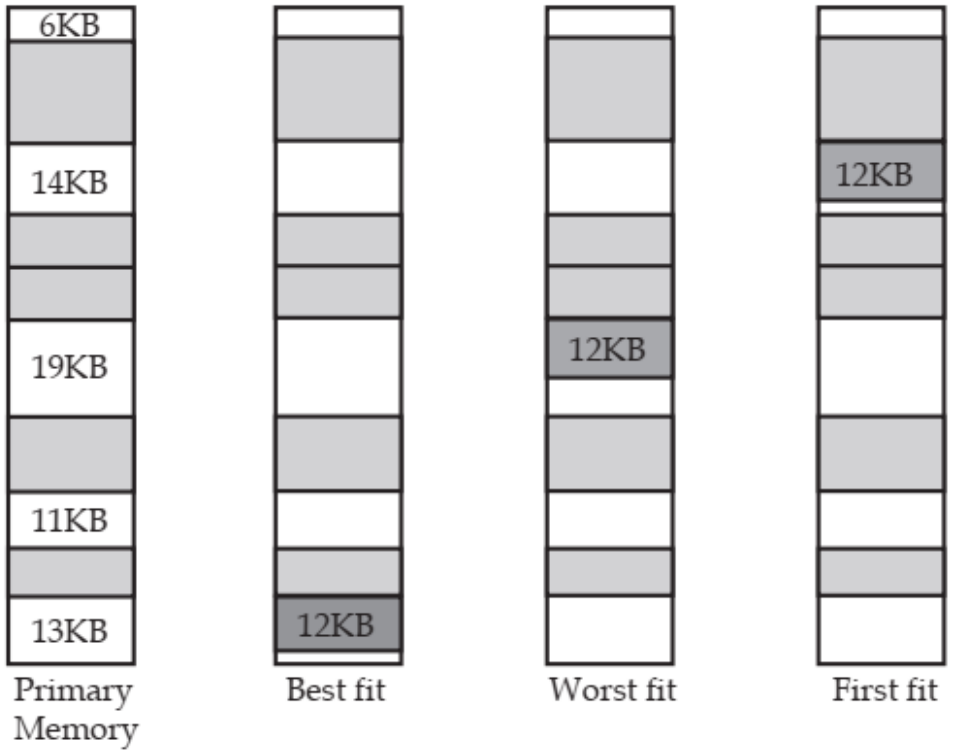
•While various different strategies are used to allocate space to processes competing for memory, three of the most popular are….

- Best fit,
- Worst fit, and
- First fit.

Contiguous Memory Allocation-*Best fit*

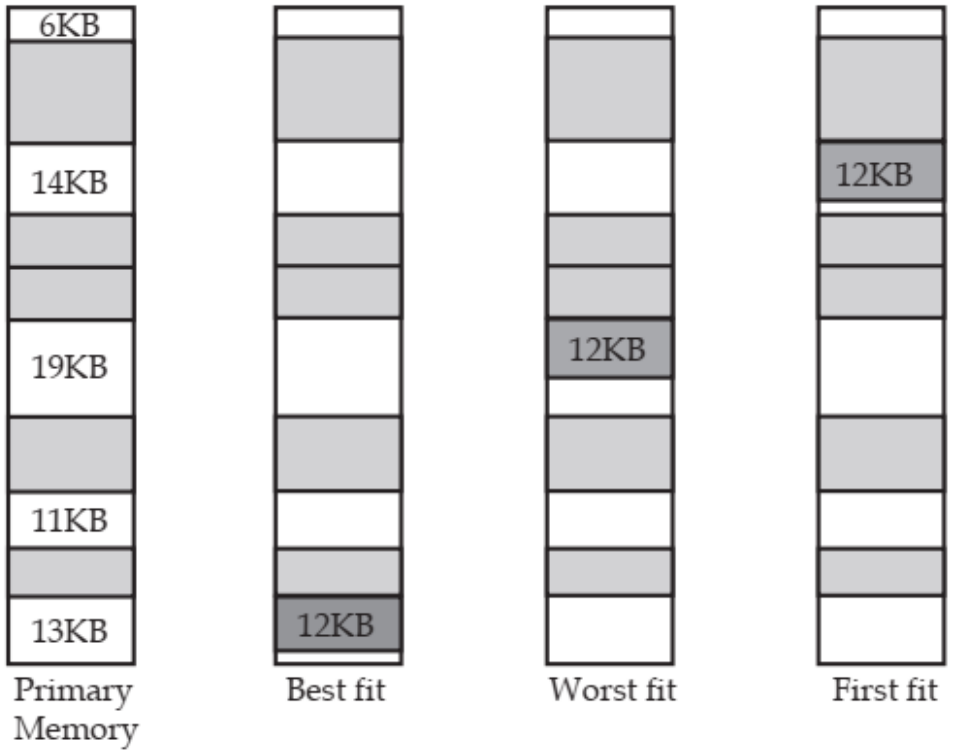
•The allocator places a process in the smallest block of unallocated memory in which it will fit.

•For example, suppose a process requests 12KB of memory and the memory manager currently has a list of unallocated blocks of 6KB, 14KB, 19KB, 11KB, and 13KB blocks.



•The best-fit strategy will allocate 12KB of the 13KB block to the process.

Contiguous Memory Allocation-*Worst fit*



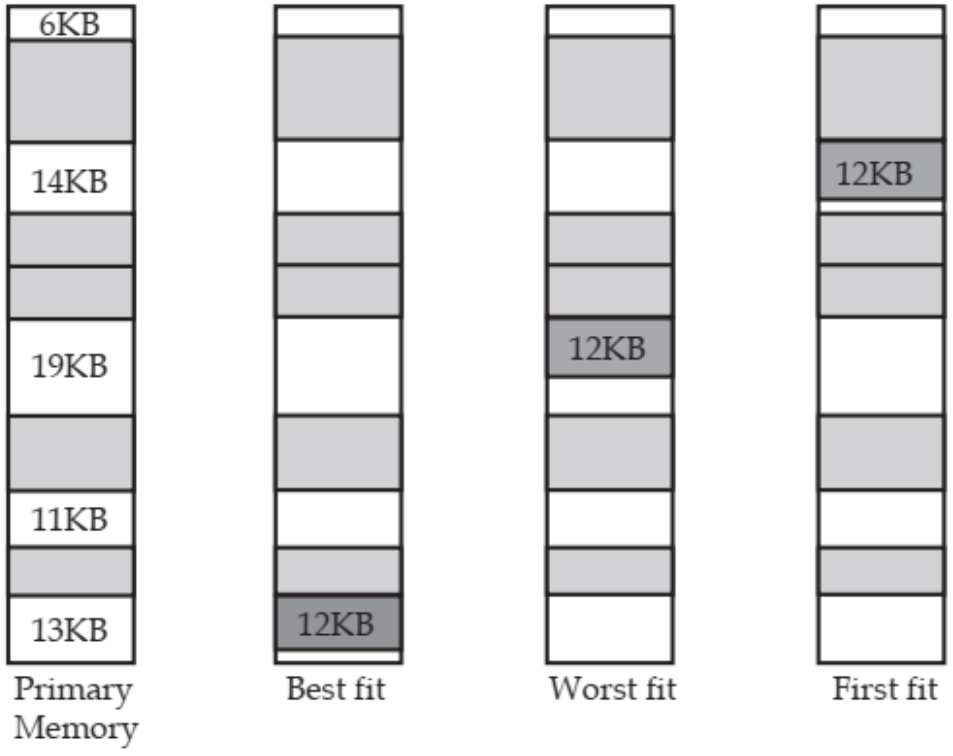
•The memory manager places a process in the largest block of unallocated memory available.

•The idea is that this placement will create the largest hold after the allocations, thus increasing the possibility that, compared to best fit, another process can use the remaining space.

•Using the same example as above, worst fit will allocate 12KB of the 19KB block to the process, leaving a 7KB block for future use.

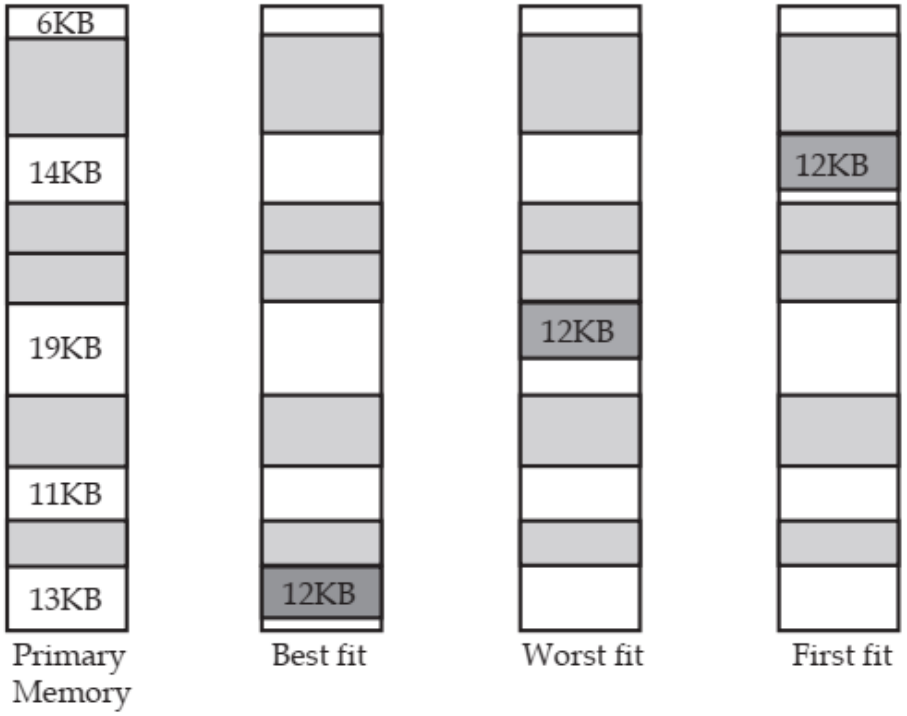
Contiguous Memory Allocation-*First fit*

•There may be many holes in the memory, so the OS, to reduce the amount of time it spends analyzing the available spaces, begins at the start of primary memory and allocates memory from the first hole it encounters large enough to satisfy the request.



•Using the same example as above, first fit will allocate 12KB of the 14KB block to the process.

Fragmentations

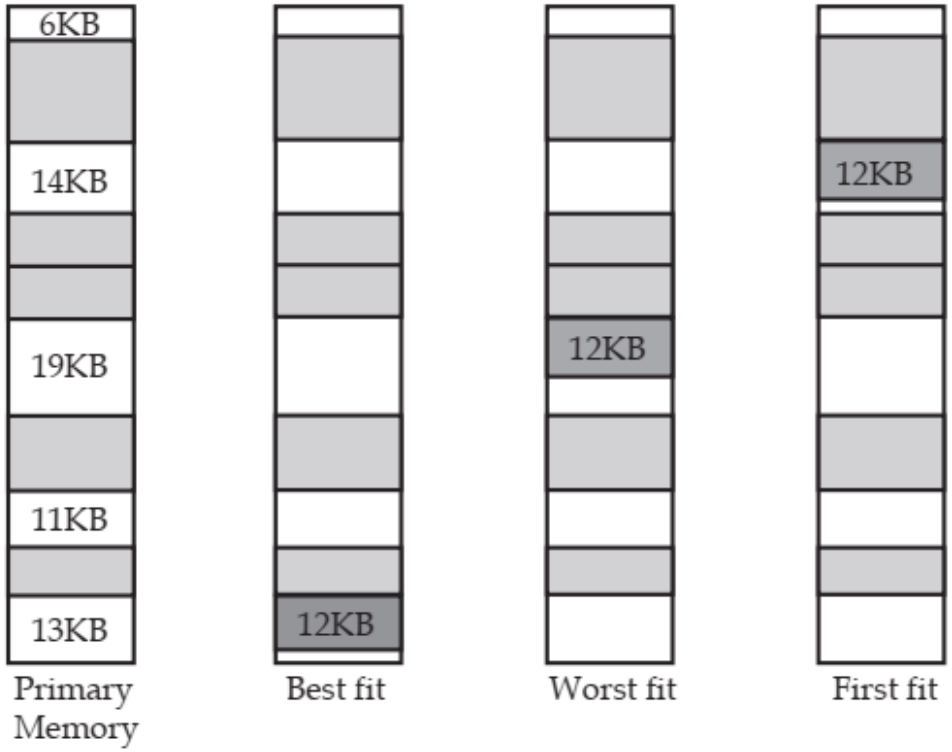


•Notice in this figure that the Best fit and First fit strategies both leave a tiny segment of memory unallocated just beyond the new process.

• Since the amount of memory is small, it is not likely that any new processes can be loaded here.

•This condition of splitting primary memory into segments as the memory is allocated and deallocated is known as fragmentation.

Fragmentations



•The Worst fit strategy attempts to reduce the problem of fragmentation by allocating the largest fragments to new processes.

•Thus, a larger amount of space will be left as seen in the Figure .

Paging

- It is a technique for increasing the memory space available by moving infrequently-used parts of a program's working memory from RAM to a secondary storage medium, usually hard disk.
- The unit of transfer is called a page.

Paging……

•Paging is a function of memory management where a computer will store and retrieve data from a device's secondary storage to the primary storage.

•Memory management is a crucial aspect of any computing device, and paging specifically is important to the implementation of virtual memory.

How Paging Works?

- Paging works by writing data to, and reading it from, secondary storage for use in primary storage.
- Paging is a basic function in memory management for a computer's operating system (OS) as well -- this includes Windows, Unix, Linux and macOSs.

How Paging Works?

- **In a memory management system that takes advantage of paging, the OS reads data from secondary storage in blocks called pages, all of which have identical size.**
- **The physical region of memory containing a single page is called a frame.**
- When paging is used, a frame does not have to comprise a single physically contiguous region in secondary storage.
- This approach offers an advantage over earlier memory management methods, because it facilitates more efficient and faster use of storage.

Demand Paging?

- Demand paging in OS is a memory management technique used by modern operating systems to efficiently manage memory usage.
- The basic idea behind demand paging is to allow the operating system to load only the parts of a program that are currently needed into memory, rather than loading the entire program all at once.
- When a program is launched, the OS does not load the entire program into memory at once. Instead, it loads only the parts of the program that are immediately required to start executing.
- These parts are typically the program's code and any data that is immediately needed. As the program runs and requires more memory, the operating system loads additional parts of the program into memory as needed.

Demand Paging

- The OS also keeps track of which parts of the program are currently being used and which parts are not. If a part of the program has not been used for a long time, the operating system can remove it from memory to free up space for other programs. This process is called "**swapping**."
- Demand paging in OS is an effective technique because it allows the OS to use memory more efficiently. Rather than allocating a large block of memory for a program, the operating system can allocate only the memory that is actually needed. This can help reduce the amount of memory that is wasted, which can be especially important in systems with limited memory resources.

Benefits Demand Paging

- One of the main benefits of demand paging is that it allows the OS to use memory more efficiently. Rather than allocating memory for a program's entire data and code segments, demand paging allows the system to allocate memory only when needed, freeing up memory that can be used for other processes.
- Another benefit of demand paging is that it allows multiple processes to run simultaneously on a single system without having to worry about running out of memory. The OS can simply swap out the pages that are not being used by a process and bring in the pages that are required for the currently running process, allowing multiple processes to run simultaneously without running out of memory.

Benefits Demand Paging

- Demand paging also helps to reduce the time it takes to load a program into memory. Instead of loading the entire program into memory at once, the OS can load only the required pages on demand, reducing the amount of time it takes to load the program and making the system more responsive.
- Demand paging provides several benefits for OSs, including efficient memory management, support for multiple processes, and faster program loading times.

Demand Paging Vs. Swapping

- Demand paging and swapping are two different memory management techniques used by operating systems to manage memory in a computer system.
- Demand paging is a memory management technique where the operating system loads only the required pages into memory when they are needed.
- The entire program is not loaded into memory at once. Instead, only those parts of the program that are required at a particular point in time are loaded into memory.
- This approach is very efficient in terms of memory usage since it does not require the entire program to be loaded into memory at once. However, there is a small delay in accessing pages that have not been loaded into memory, which can lead to slower performance.

Demand Paging Vs. Swapping

- Swapping, on the other hand, is a memory management technique where the entire process is moved to and from main memory to secondary storage (e.g., a hard disk) when needed.
- In swapping, the entire process is swapped out of memory and written to disk when the process is not currently being used, and then swapped back into memory when it is needed again. This approach is less efficient in terms of memory usage than demand paging since the entire process must be swapped in and out of memory, but it can be faster in terms of accessing pages since all the required pages are loaded into memory at once.

Common Algorithms Used in Demand Paging in OS

- **There are several algorithms used for, demand paging each with its own advantages and disadvantages.**

- **FIFO (First-In-First-Out):** This algorithm replaces the oldest page in memory when a new page is needed. It is simple to implement but can lead to thrashing when pages are repeatedly brought in and out of memory.
- **LRU (Least Recently Used):** This algorithm replaces the page that has not been used for the longest time. It is more effective than FIFO in reducing thrashing but can be computationally expensive to implement.
- **LFU (Least Frequently Used):** This algorithm replaces the page that has been used the least number of times. It is effective in reducing thrashing but requires additional bookkeeping to keep track of the usage count of each page.
- **MRU (Most Recently Used):** This algorithm replaces the page that has been used most recently. It is less effective than LRU in reducing thrashing but can be simpler to implement.
- **Random:** This algorithm randomly selects a page to replace. It is simple to implement but can be unpredictable in its effectiveness.

Pre-Paging in OS

- Pre-paging is a technique in which the OS loads multiple pages into main memory before they are actually demanded by the program.
- This technique is based on the assumption that if one page is needed, there is a good chance that the pages adjacent to it will also be needed soon.
- Pre-paging can improve the program's execution speed as it reduces the delay caused by demand paging.
- However, it can also result in unnecessary **memory allocation** and memory wastage.

Allocation of Frames in OS

- In a memory management system that takes advantage of paging, the OS reads data from secondary storage in blocks called pages, all of which have identical size.
- The physical region of memory containing a single page is called a frame.
- The main memory of the system is divided into frames.
- The OS has to allocate a sufficient number of frames for each process and to do so, the OS uses various algorithms.
- The five major ways to allocate frames are as follows:
  - **Proportional frame allocation**
  - **Priority frame allocation**
  - **Global replacement allocation**
  - **Local replacement allocation**
  - **Equal frame allocation**

Allocation of Frames in OS

- **Proportional frame allocation**

- The proportional frame allocation algorithm allocates frames based on the size that is necessary for the execution and the number of total frames the memory has.
- The only disadvantage of this algorithm is it does not allocate frames based on priority. This situation is solved by **Priority frame allocation.**

Allocation of Frames in OS

- **Priority frame allocation**

- Priority frame allocation allocates frames based on the priority of the processes and the number of frame allocations.
- If a process is of high priority and needs more frames then the process will be allocated that many frames.
- The allocation of lower priority processes occurs after it.

Allocation of Frames in OS

- **Global replacement allocation**

- When there is a page fault in the operating system, then the global replacement allocation takes care of it.
- The process with lower priority can give frames to the process with higher priority to avoid page faults.

Allocation of Frames in OS

- **Local replacement allocation**

- In local replacement allocation, the frames of pages can be stored on the same page.
- It doesn’t influence the behavior of the process as it did in **global replacement allocation.**

Allocation of Frames in OS

- **Equal frame allocation**

- In equal frame allocation, the processes are allocated equally among the processes in the operating system.
- The only disadvantage in equal frame allocation is that a process requires more frames for allocation for execution and there are only a set number of frames.