OPERATING SYSTEM

UNIT-2

Memory Management

Memory management is a critical part of operating systems that involves handling computer memory, ensuring efficient and effective use of this resource. Let's break down the key concepts in easy-to-understand terms:

1. Logical and Physical Address Space

- Logical Address Space:
  - Also known as the virtual address space.
  - This is the set of addresses generated by a program's perspective.
  - These addresses are used by programs to access memory, but they are not the actual physical addresses.
- Physical Address Space:
  - The actual locations in the computer's memory hardware (RAM).
  - These addresses are where data is physically stored.

Translation: The operating system uses a mechanism (like the Memory Management Unit, MMU) to map logical addresses to physical addresses.

2. Swapping

- Swapping is a memory management technique used to increase the amount of available memory.
- When a computer runs out of RAM, the OS can move some inactive processes from RAM to a storage device (like a hard disk) to free up space.
- When these processes are needed again, they are swapped back into RAM.

Example: Think of a crowded room where some people are moved out temporarily to make room for others.

3. Contiguous Allocation

- Contiguous Allocation means that each process is allocated a single continuous block of memory.
- This method is simple and easy to manage but can lead to problems like fragmentation.

Example: Allocating seats in a row in a theater, where each person (process) gets seats (memory) next to each other.

4. Multiple Partitions

- In Multiple Partition Allocation, memory is divided into fixed or dynamic partitions.
- Each partition can hold exactly one process.
- Fixed partitions are pre-defined and do not change size, while dynamic partitions can vary based on the process requirements.

Example: Dividing a large plot of land into smaller plots where each plot is reserved for a specific type of crop (process).

5. Fragmentation

- Fragmentation refers to the inefficient use of memory that occurs when there are gaps or "fragments" of free memory space.
    - External Fragmentation: When there is enough total free memory space, but it is not contiguous (not in one block), making it difficult to allocate to a process.
    - Internal Fragmentation: When allocated memory blocks have wasted space inside them, often because of allocation size mismatches.

Example: A bookshelf with many small gaps between books (external fragmentation) or books with some pages blank (internal fragmentation).

6. Compaction

- Compaction is the process of eliminating fragmentation by moving processes around to create larger contiguous blocks of free memory.
- This can be time-consuming but helps in making memory usage more efficient.

Example: Rearranging books on a shelf to remove gaps and make space for new books.

7. Paging

- Paging is a memory management scheme that eliminates the need for contiguous allocation.
- Memory is divided into fixed-size blocks called pages (logical memory) and frames (physical memory).
- The logical address is divided into a page number and an offset, which are used to find the corresponding physical address.

Example: Think of a book where each page (logical) corresponds to a different drawer in a filing cabinet (physical).

Paging Benefits:

- Avoids external fragmentation.
- Allows the physical address space of a process to be non-contiguous.

Virtual Memory Management

Virtual Memory Overview:

- Virtual memory provides processes with an illusion of a larger memory space than physically available by utilizing secondary storage, such as a hard disk.
- Allows for efficient utilization of physical memory and supports multitasking.

Components of Virtual Memory:

1. Page Table:
    - Maintained by the operating system.
    - Maps logical addresses (used by processes) to physical addresses (actual memory locations).
    - Each entry in the page table contains information about the corresponding page's physical address, permissions, and status.
2. Page Fault Handler:
    - Handles page faults when a process accesses a page not currently in physical memory.
    - Triggers the loading of the required page from disk into memory.

Demand Paging

Demand Paging Overview:

- Demand paging is a virtual memory management technique where pages are loaded into memory only when needed.
- Reduces initial memory requirements and minimizes the need for extensive swapping.

How Demand Paging Works:

1. Page Fault:
    - Occurs when a process tries to access a page not in physical memory.
2. Page Fault Handling:

- Operating system identifies the missing page and allocates a free frame in physical memory.
- Page is loaded from disk into the allocated frame.
- Page table is updated to reflect the new page location.
- Process continues execution with the requested page now available in memory.

Page Replacement Algorithms

FIFO (First-In-First-Out)

Overview:

- FIFO is a straightforward page replacement algorithm based on the order of page arrival in memory.

How FIFO Works:

1. Page Replacement Decision:
    - When a page needs to be replaced due to memory full, the page that entered memory first (oldest) is selected for removal.
2. Replacement Process:
    - The oldest page is removed, making space for the new page to be loaded into memory.
    - The new page is placed at the end of the page queue.

Example:

- Imagine books placed on a shelf in the order they were added. When a new book arrives, the oldest book (first-in) is removed to make space for the new book (first-out).

LRU (Least Recently Used)

Overview:

- LRU is a page replacement algorithm that selects the least recently used page for replacement.

How LRU Works:

1. Page Usage Tracking:
    - The operating system keeps track of when each page was last accessed or used.
2. Page Replacement Decision:
    - When a page needs to be replaced, the page that was least recently accessed is selected for removal.

Example:

- Imagine a list of recently accessed books. When a new book arrives, the least recently accessed book is removed to make space for the new book.

Optimal Page Replacement

Overview:

- Optimal is an idealized page replacement algorithm that selects the page that will not be used for the longest time in the future.

How Optimal Works:

1. Future Access Prediction:
    - The algorithm predicts future page accesses based on the page reference string.
2. Page Replacement Decision:
    - Selects the page that will not be used for the longest duration in the future for replacement.

Example:

- Imagine having perfect knowledge of future book access. The book that will not be needed for the longest time is removed to make space for a new book.