FACULTY OF ENGINEERING AND TECHNOLOGY
UNIVERSITY OF LUCKNOW
LUCKNOW

Operating System
AI-602

Dr. Zeeshan Ali Siddiqui
Assistant Professor
Deptt. of C.S.E.

VIRTUAL MEMORY

Virtual Memory

- Virtual memory involves the separation of *logical memory* as perceived by users from physical memory.

- This separation allows an *extremely large virtual memory* to be provided for programmers when only a smaller physical memory is available.

Virtual Memory

- Virtual Memory:
  - Allows *execution* of program that may not be completely in memory.
  - *Logical address space* can therefore be much larger than physical address space.
  - Allows address spaces to be shared by several *processes*.
  - Each user take *less physical memory* so more program can be run at the same time.
  - CPU utilization and throughput *increased* but response time and turnaround time do not increase.
  - Less I/O needed to load and swap user program so each user program run *faster*.

Virtual Memory

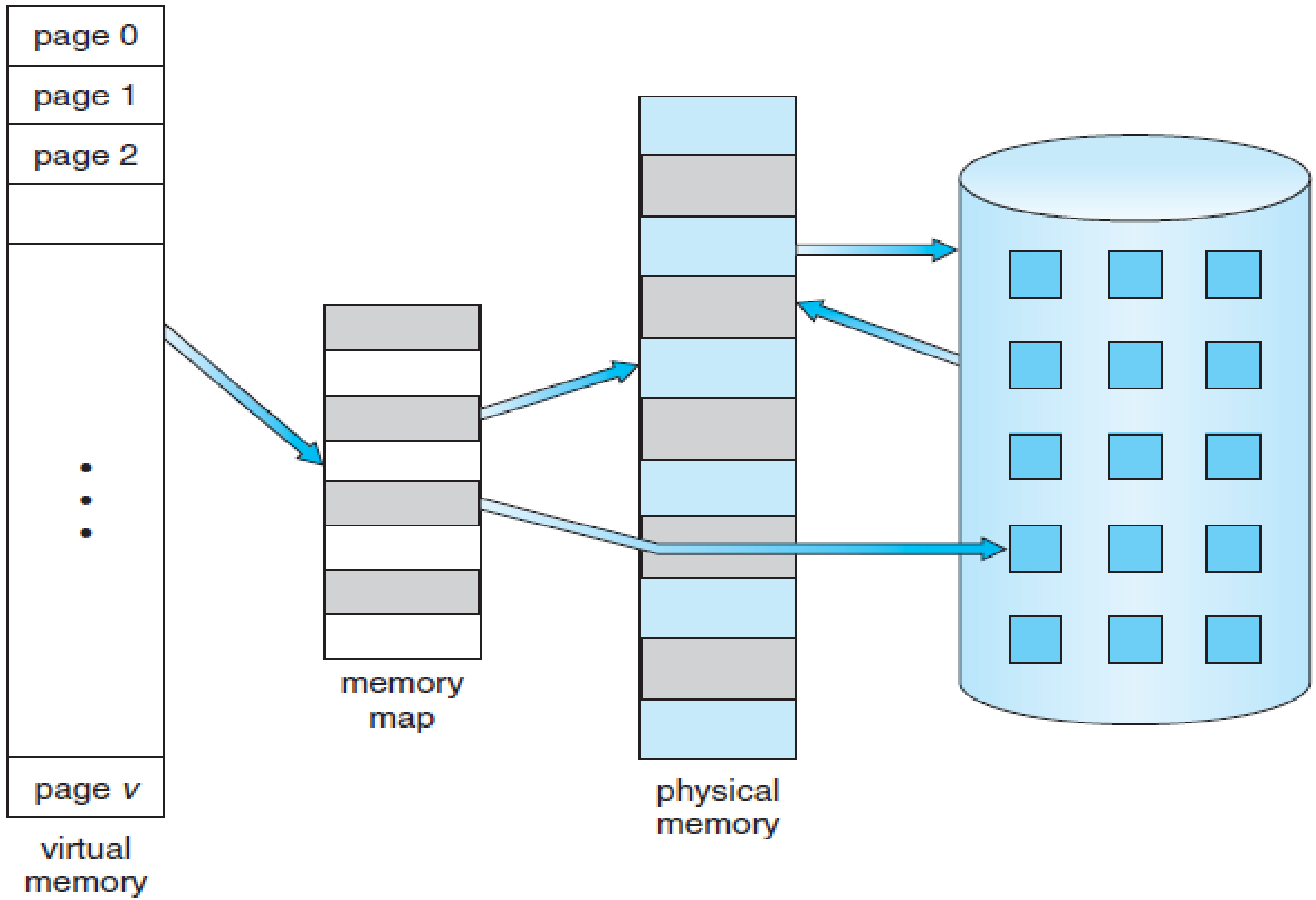


Diagram showing virtual memory that is larger than physical memory

Virtual Memory

- Virtual memory can be implemented via:

  - Demand paging

  - Demand segmentation

Demand Paging

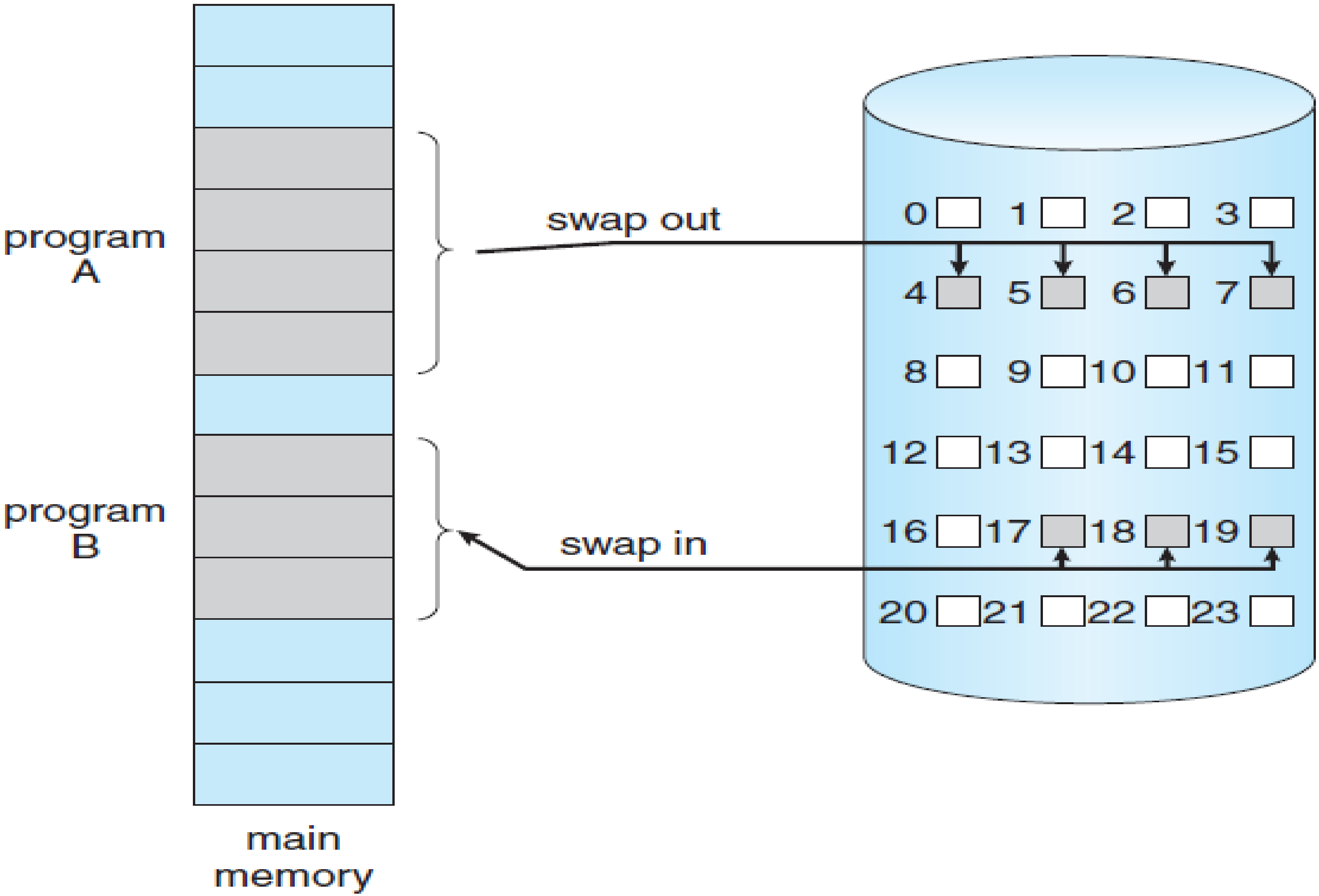
Demand Paging

- In demand paging technique, pages are loaded only when they are *demanded* during program execution.

- *Pages* that are never accessed are thus never loaded into physical memory.

- **Advantages**:
  - Less I/O needed
  - Less memory needed
  - Faster response
  - More users

Demand Paging



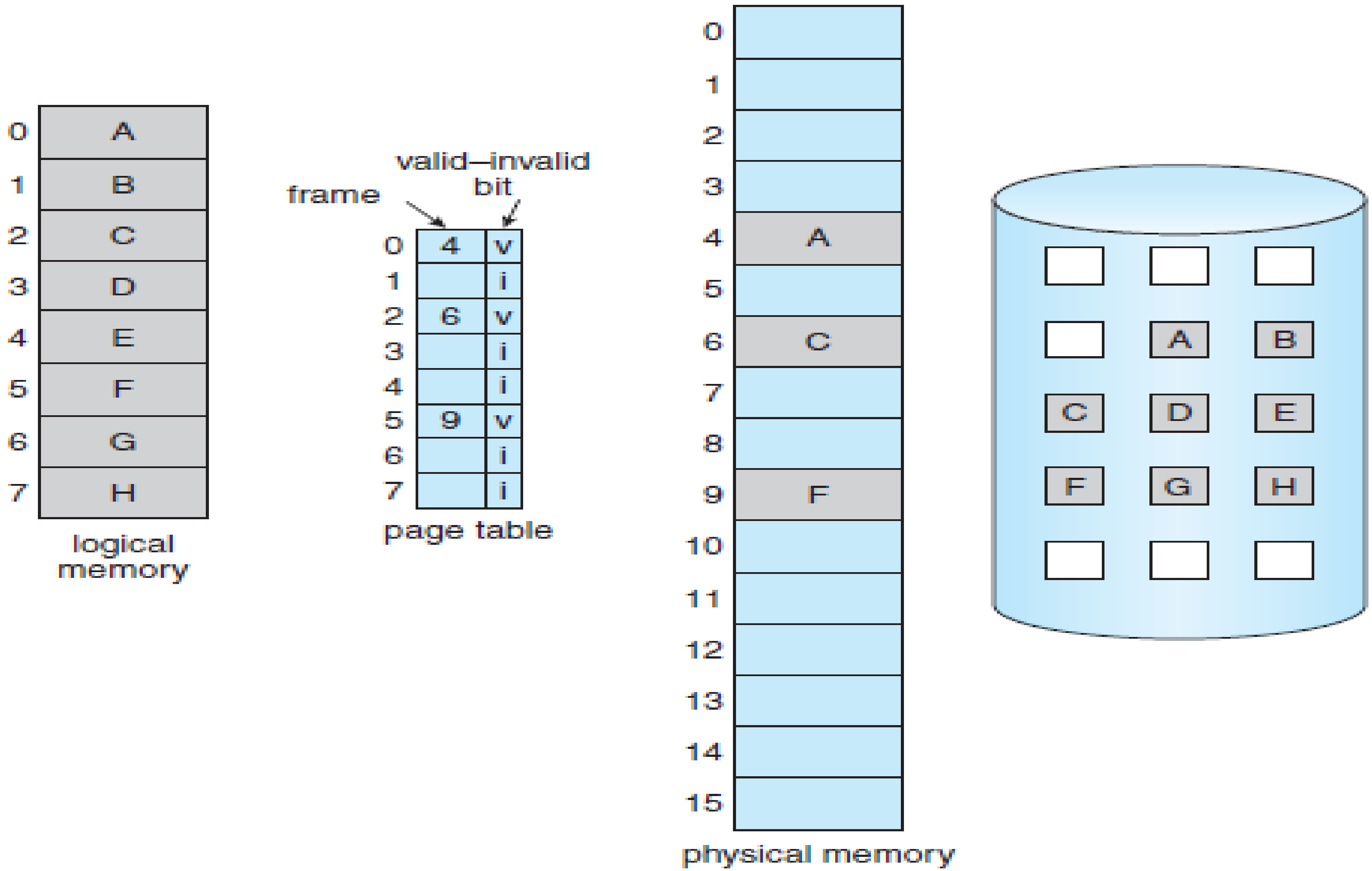
Transfer of a paged memory to contiguous disk space

Demand Paging: Basic Concepts

- **Pager**:
  - When a process is to be swapped in, the pager *guesses* which pages will be used before the process is swapped out again.
  - Instead of swapping in a whole process, the *pager* brings only those pages into memory.
  - Thus, it *avoids* reading into memory pages that will not be used anyway, decreasing the swap time and the amount of physical memory needed.

Demand Paging: Basic Concepts

- With each page table entry a *valid–invalid bit* is associated (v -> in-memory, i -> not-in-memory means page fault).



Page table when some pages are not in main memory

Performance of Demand Paging

Performance of Demand Paging

- Let memory-access time is ma, p be the probability of a page fault ($0 \le p \le 1$) Then:

Effective access time* = *(1 − p)* × *ma* + *p* × *page fault time

- Three major components of the page-fault service time:
  - Service the page-fault interrupt.
  - Read in the page.
  - Restart the process.

Example

With an average page-fault service time of 8 milliseconds and a memory-access time of 200 nanoseconds, the effective access time in nanoseconds is

$$\begin{aligned}\text{effective access time} &= (1 - p) \times (200) + p \text{ (8 milliseconds)} \\ &= (1 - p) \times 200 + p \times 8{,}000{,}000 \\ &= 200 + 7{,}999{,}800 \times p.\end{aligned}$$

Homework

- Lazy swapper

- Pure demand paging

- Locality of reference.

References

1. Silberschatz, Galvin and Gagne, “Operating Systems Concepts”, Wiley.

2. William Stallings, “Operating Systems: Internals and Design Principles”, 6^{th} Edition, Pearson Education.

3. D M Dhamdhere, “Operating Systems: A Concept based Approach”, 2^{nd} Edition, TMH.

Thank You.