ABHINAV JHA
2301010211
B.Tech CSE Core - 4

Assignment - 2

1. Address translation in modern systems

- Each process generates logical (virtual) address
- MMU (Memory Managment Unit) translates these into physical address
- Translation steps :-

a) CPU generates logical address
b) MMU checks page table for corresponding frame number
c) Concatenates frame no + offset → physical address

2. Memory layout
→ Eg. layout

Process A (100 Kb of 120 Kb block) | Free 30 Kb | Process B (200 Kb)

- Internal fragmentation = 20 Kb wasted inside A's block
- External fragmentation = 30 Kb free, but too small for 40 Kb request

→ Mitigation techniques:
- Paging (eliminates external, but many cause small internal)
- Segmentation with paging hybrid.
- Buddy system allocation
- Slab allocators. (in Linux).

3. Paging-based allocation model for a hypothetical OS
- Memory divided into fixed-fram.
- Trade-offs:

* Overhead : Pages tables consume memory.
* Speed :
* Fragmentation

4. OS hardware interaction in virtual memory.
- Page-table in memory.
- MMU translates virtual
- TLB caches recent translations.
- Protection bits

5. 16-bit virtual address, 1KB page size
→ Virtual address = 16 bits = page no + offset
→ Page size = 1 KB = 2^{10} bytes → offset = 10 bits
→ Page no = 16 − 10 = 6 bits
→ * No. of virtual pages = 2^6 = 64
* Page table size = 64 entries × 2 bytes = 128 bytes

6.

Process	Size (KB)
P_1	212
P_2	417
P_3	112
P_4	426

- First-fit

P_1	P_2	P_3	P_4

0 212 629 741 1167

Unused memory = 259 Kb.

- Best-fit

P_1	P_2	P_3	P_4

212 417 112

P_4 still can't fit Unused = 259 KB

- Worst-fit
  P_1 (212) into 1000 → 788 left
  P_2 (417) into 788 → 371 left
  P_3 (112) into 371 → 259 left
  P_4 (426) can't fit

Unused = 259 KB.
All three give same unused memory, but Worst-Fit may delay fragmentation buildup.

7. Page replacement reference string:
7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 3

a)b) FIFO : 9 page faults
Optimal : 7 page faults
- LRU : 10 page faults

c) Best: Optimal (minimum). FIFO worse due to Belady's anomaly.

8. Disk write = 10 ms
Memory write = 100 ms
Dirty pages = 30% of 1000 = 300.

a) Overhead = 300 × 10 ms
= 3000 ms = 3 seconds

b) Optimization: Write-back caching with dirty bit tracking or pre-cleaning (background flush) reduces blocking time.

9.a) Working set model + replacement policy
- OS tracks recent active pages per task.
- For object detection: Allocate stable working set
- For infotainment: Allows flexible replacement so it adapts to available memory.

b) Memory allocation strategy.
- Use priority-based dynamic allocation.
- Real-time responsiveness ensured by working set + real time schedule