CS & IT ENGINEERING

Operating System

Virtual Memory

Lecture - 02

By- Vishvadeep Gothi sir

Recap of Previous Lecture

Topic Virtual Memory

Topic Demand Paging

Topic Page Fault

Topics to be Covered

1 ⇒ TLB h

p ⇒ page fault in mm

(1-p) page hit in mm

Topic Effective Memory Access Time

Topic Page Replacement Algorithm

Topic FIFO, Optimal, LRU Algorithms

E.M.A.T. with page fault

———— " ———— " ———— + TLB

Topic : Dirty Bit Included (Without TLB)

P.T. search with L.A. in mm

Page hit

P.A.

⇓

content accessed

$= t_{mm} + t_{mm}$

Page fault

Replaced page dirty or not

non-dirty

P.f. service without copying replaced page to disk

t_{mm} + P.f. service time for non-dirty page

dirty

P.f. service with replaced page copied to disk

t_{mm} + P.f. service time for dirty page

$$EMAT = (1-p)(2t_{mm}) + p\left[(1-M)\left(t_{mm} + \text{p.f.s.t. for non-dirty pages}\right) + M * \left(t_{mm} + \text{p.f.s.t. for dirty pages}\right)\right]$$

M = % of replaced page which are dirty / modified

Topic : Question

$P = 2\%$

$M = 5\%$

$T_{mm} = 100$ n sec

P. F .S.T. when page dirty = 200μ sec

P. F .S.T. when not dirty = 100μ sec

E. M. A. T. = 2298 n sec.

$$= 0.98 * (2 * 100) + 0.02 * [0.95 * (100 + 100000) + 0.05 * (100 + 20000)]$$

$$= 2298 \text{ ns}$$

Topic : Dirty Bit with TLB

$$EMAT = H * (t_{TLB} + t_{mm})$$

$$(1-H)\left[t_{TLB} + (1-p) * 2t_{mm} + p\left[(1-M)\left(t_{mm} + \text{p.f.s.T. for non-dirty}\right) + M\left(t_{mm} + \text{p.f.s.T. for dirty}\right)\right]\right]$$

Topic : Question

$H = 85\%$

$t_{TLB} = 20$ n sec

$t_{mm} = 200$ n sec

$P = 1\%$

$M = 2\%$

P.F.S.T. with dirty page = 5000 µ sec

—||— without —||— = 2000 µ sec

EMAT = ______ ns

$$E.M.A.T. = 0.85 * (20 + 200) + 0.15 \left[20 + 0.99 * 400 + 0.01 \left(0.98 * (200 + 2000000) + 0.02 * (200 + 5000000) \right) \right]$$

$$= 3339.7 \text{ ns}$$

P.F. service time when replaced page is not dirty :-

$$= \underset{\text{from disk to mm}}{\text{page transfer time}} + \underline{t_{mm}}$$

for updating page table

P.F.S.T. for dirty page :-

$$= \underset{\text{from mm to disk for dirty page}}{\text{Page transfer time}} + \underset{\text{from disk to mm for faulted page}}{\text{Page transfer time}} + \underline{t_{mm}}$$

Topic : Question

TLB Hit = 90%

t_{TLB} = 15 n sec

T_{mm} = 120 n sec

P = 2%

M = 10%

Time taken to transfer a page b/w mm & sm = 150 μ sec

E.M.A.T. = 477 ?

$$= 0.9 * (15 + 120)$$
$$+ 0.1 * [15 + 0.98 * 240 + 0.02 [0.9 * (120 + 150000 + 120) + 0.1 * (120 + 2 * 150000 + 120)]]$$

$$= 477 \text{ ns}$$

Topic : Question

[GATE-2020]

#Q. Consider a paging system that uses 1-level page table residing in main memory and a TLB for address translation. Each main memory access takes 100 ns and TLB lookup takes 20 ns. Each page transfer to/from the disk takes 5000 ns. Assume that the TLB hit ratio is 95%, page fault rate is 10%. Assume that for 20% of the total page faults, a dirty page has to be written back to disk before the required page is read from disk. TLB update time is negligible. The average memory access time in ns (round off to 1 decimal places) is 155.

Given $\Rightarrow$ 154.5 to 155.5

$$= 0.95\,(20 + 100)$$

$$+ 0.05\Big[20 + 0.9 * (2 * 100)$$

$$+ 0.1 * \Big[0.8\,(100 + 5000 + 100)$$

$$+ 0.2\,(100 + 2 * 5000 + 100)\Big]\Big]$$

$$= 155 \text{ ns}$$

Cache with TLB :-

physically address cache ⇒ cache is accessed using P.A. only.

Assume:-

H_{cm} = cache hit ratio

t_{cm} = cache access time

Content access time with P.A. $= H_{cm} * t_{cm} + (1 - H_{cm})(t_{cm} + t_{mm})$

CPU generates L.A.

⇓

search in TLB

- Hit
  - P.A.
  - ⇓
  - search in cache with P.A.
    - Hit
      - access content from cm
    - Miss
      - Access content from mm
- Miss
  - search in P.T. with L.A.
  - ⇓
  - P.A.
  - ⇓
  - search in cache with P.A.
    - Hit
      - Access content from c.m.
    - Miss
      - Access content from mm

$$EMAT = H_{TLB} * \left[t_{TLB} + \text{content access time} \right]$$

$$+ (1 - H_{TLB}) * \left[t_{TLB} + t_{mm} + \text{Content access time} \right]$$

$$\text{content access time} = H_{cm} * t_{cm} + (1 - H_{cm})(t_{cm} + t_{mm})$$

Topic : Page Replacement Policies

1. First In First Out (FIFO)
2. Optimal Policy
3. Least Recently Used (LRU)
4. Least Frequently Used (LFU)
5. Most Frequently Used (MFU)
6. Last In First Out (LIFO)
7. Second Chance
8. Most Recently Used (MRU)

Topic : First In First Out (FIFO)

Replace the page which comes first in mm among all mm pages.

Assume:

- Number of frames = 3 (All empty initially)
- Page reference sequence: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

no. of page faults = 9

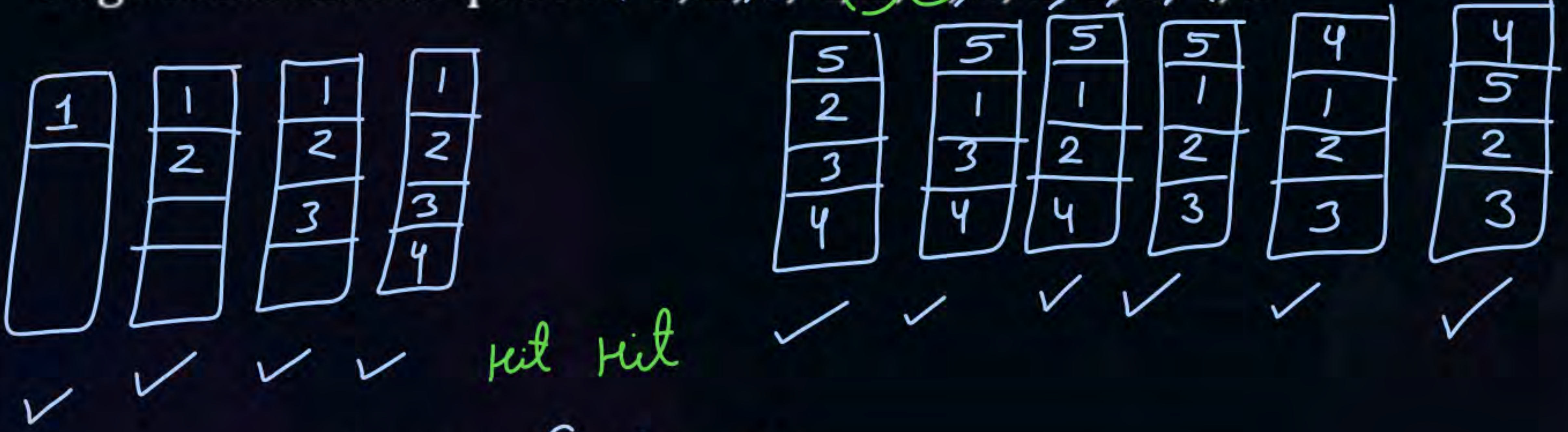
Page fault rate = $\frac{9}{12}$

Page hit rate = $\frac{3}{12}$

Topic : First In First Out (FIFO)

Assume:

- Number of frames = 4 (All empty initially)
- Page reference sequence: ~~1~~, ~~2~~, ~~3~~, ~~4~~, 1, 2, ~~5~~, ~~1~~, ~~2~~, ~~3~~, ~~4~~, ~~5~~



no. of page faults = 10

Topic : Belady's Anomaly

→ It occurs only in FIFO

→ For some page reference sequences, increasing number of frames may increase number of page faults.

Topic : First In First Out (FIFO)

Advantages:

1. Simple and easy to implement.
2. Low overhead. → To check which page to replace

Disadvantages:

1. Poor performance. → High no. of page faults
2. Doesn't consider the frequency of use or last used time, simply replaces the oldest page.
3. Suffers from Be lady's Anomaly

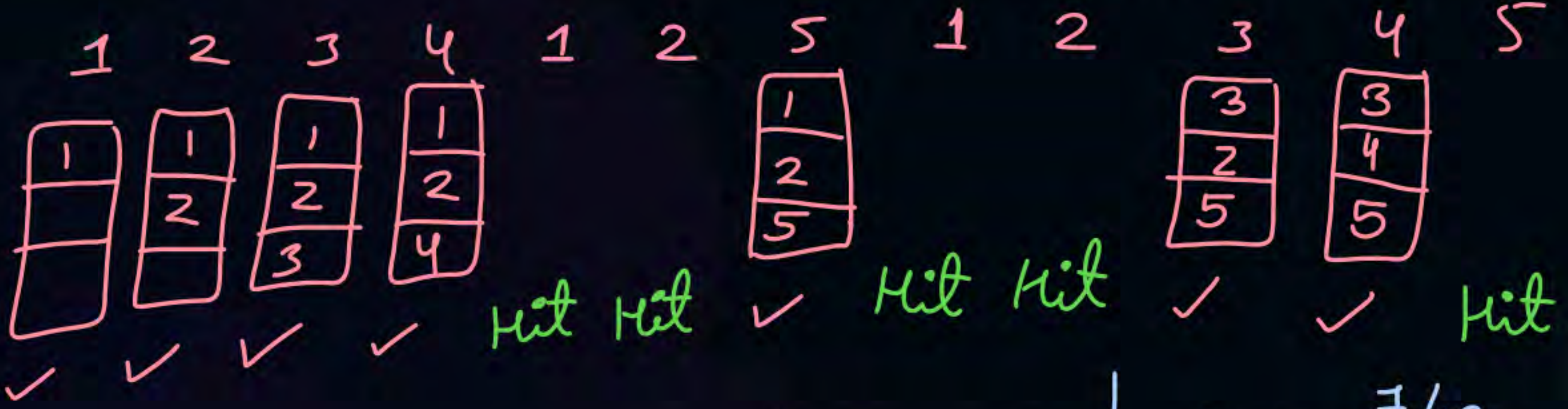
Topic : Optimal Policy

Replace the page which is not going to be used for longest period of time.

↳ Tie breaker ⇒ FIFO

Assume:

- Number of frames = 3 (All empty initially)
- Page reference sequence: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5



no. of page faults = 7

$p = 7/12$

$(1-p) = 5/12$

Topic : Optimal Policy

Advantages:

1. Easy to Implement
2. Simple data structures are used
3. Highly efficient $\Rightarrow$ min. no. of page faults

Disadvantages:

1. Requires future knowledge of the program (which is not practical)
2. Time-consuming

2 mins Summary

Topic Effective Memory Access Time

Topic Page Replacement Algorithm

Topic FIFO, Optimal, LRU Algorithms

Happy Learning

THANK - YOU