COMSW4118-25-1 12/7/2017

PROBLEM 1: System w/ 3 phys. frames and following page memory reference seq:
1, 3, 6, 7, 1, 3, 6, 7, 1, 3, 6, 7

What's the number of page faults that occurs w/ each of following page replacement algos:

a) optimal: f1 f3 f6; 1 3 f7; 1 f6 7; f3 6 7 = 6 faults

b) LRU: f1 f3 f6; f7 36; 7 f1 6; 7 1 f3; f6,1,3; 6 f7,3; 6 7 f1; f3 7 1; 3 f6 1; 3 6 f7 = 12 faults

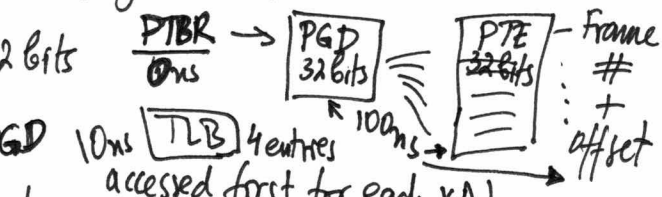
c) 2nd chance clock replacement:
1 f 3 f 6 f; f7 36; 7 f 1 3; .... same as LRU = 12 faults again

d) does an optimal page replacement algo exist that does not require future knowledge for cyclical memory references such as above (1, 3, 6, 7 ....)? Explain.
- Most recently used = will be the same as the optimal, if there are no repeating sequences like for example 1, 3, 6, 6

PROBLEM 2: System w/ 32-bit logical addr. space, 2-level paging scheme, 4-byte page table entries, 1 KB pages, and 4 entry TLB. Page-table base register access time is 0 ns, TLB access time is 10 ns and memory acess time is 100 ns.

↙ Typically, logical = physical unless noted otherwise

a) How many bits for page offset?
1 KB page = 2^{10} = 10 bits



b) How much memory to store the outer page ~~PGD~~ entirely in main memory?
accessed first for each VA!
~~2^{22} pages (from above $2^{32}/2^{10} = 2^{22}$) * 2^4 (each entry) = 226~~ ← that's for PTE, question is for PGD

NOTE: "outer page" means PGD!

we have the PTE and PGD in page-sized chunks, so each is 1 KB, so each index in them is 10 bytes max BUT a 1 KB page of 4 byte addresses can contain at most 256 entries (2^{10} page / 2^2 bytes = 2^8 entries = 256). and that requires 8 bits to access the PTE. Then, there are 14 bits to access the PGD! ↙ NOTE: 1st level (PGD) is not paged, it is just contiguous memory. The only reason for it to be > 1 page is to have bigger VA space!

So, PGD contains 2^{14} entries and each is 2^2 bytes = $2^{14} \cdot 2^2 = 2^{16}$ = 64 K

c) CPU context switches to a process w/ all its page tables entirely in main mem. If there is 1:1 mapping b/n logical & physical addresses, what's the avg. effective access time for sequentially accessing the following set of VAs: 2, 3, 4, 5, 5, 2010, 2022, 2009?

2 = 10 (TLB) + 100 (PGD) + 100 (PTE) + 100 (RAM) + 0 (PTBR) = 310; 3 = 10 (TLB) + 100 (PTE) = 110; 4 = 110; 5 = 110;
(context switch flushes TLB) (same page!)

2010 = 310; 2022 = 110; 2009 = 110 ⇒ (2·310 + 6·110)/8 = 1280/8 =

COMSW4118-25-2 12/7/2017

d) If TLB has 1 entry & PTBR access is 10ns, what's the answer for c? (PGD) ← no difference! ← add 10 to all 310s = (2·320 + 6·110)/8 = 1300/8

e) Suppose outer page table is stored in main mem. starting at frame 0 & inner page tables are stored in mem. sequentially immediately after it. If logical addr. 80386 translates to phys. addr. 1,073,742,338, what are the values stored in the corresponding outer page table & inner page table?

PA = 1,073,742,338 = $2^{30} + 2^9 + 2^1$

PA = [frame # | offset] ← 10 bits for offset

1 in 30th bit, rest 0 | 1 in 9th & 1st bit, rest 0

PTE contains the frame number, so 1 followed by 29 0's = 2^{30} or 2^{20} for the frame number since the frame offset is the last 10; both acceptable answers

VA = 80,386 = $2^{16} + 2^{15} + 2^{12} + 2^{11} + 2^9 + 2^1$

This is PGD + PTE

note: VA offset matches PA offset always!

PGD stores the base address of the PTEs. PGD starts from frame 0 and followed by PTEs:

Which entry is it in the PTE?

PGD	PTE	OFFSET
16 bits	8 bits	10 bits

This is 0 since the largest number in the VA is 2^{16} → so we hit the first PTE! ~~and its~~ which starts at 2^{16} in the 64th frame (each frame is 1k)

(2^{14} entries in PGD · 2 bytes each)

So, PGD stores 2^{16} = 64k which is the base address of the first PTE page following the PGD on the contiguous region.

PROBLEM 3: File on LOOK disk sched system. Disk is 1MB size w/ 1kB blocks. File stored in sequence in blocks: 20, 500, 10, and 900. Last disk access was to block 51 and directory entry for file is in block 50. 1st disk block is #0. A single disk block cache is present.

a) Suppose linked disk allocation is used. What's the total seek distance to read file from start to finish. ← note: we don't count the 51 → 50 move.

You start from 50 (directory entry), then 20, 500, 10, 900.

30 + 480 + 490 + 890 = total seek distance

b) Suppose FAT allocation w/ FAT table at beginning of disk & 2 byte entries. What's the total seek distance for appending and storing data in disk block 600 at the end of the file?

To find the file, need FAT = go to 0, so 50 → 0 (note: we assume getting to the dir entry 51 → 50 not counted)

We get to location 20 in FAT (index for first file block), then follow the links to last block = 900th entry.

COMSW4118-25-3 12/7/2017

This will not be in the same FAT block, since the entries are two bytes and we have 1k blocks, entry 900 will be in the second FAT Block, so:

50 → 0 → 1 → 600 (finally to get to the block to write) = 650