SYSC4001

Operating Systems

Assignment 3

DATE: 30/11/25

DUE DATE: 01/12/25

Bunyan Yorganci SN#101216821

Answer the following questions. Justify your answers. Show all your work.

1. [0.6 marks] Consider the following page reference string in an Operating System with a Demand Paging memory management strategy:

415, 305, 502, 417, 305, 415, 502, 518, 417, 305, 415, 502, 520, 518, 417, 305, 502, 415, 520, 518

(i) [0.3 marks] Assume we have 3 Frames Allocated. How many page faults will occur with 3 frames allocated to the program using the following page replacement algorithms?

(a) FIFO (First-In-First-Out)

Ref	Frame3	Frame2	Frame1	Note
415	-	-	415	F
305	-	415	305	F
502	415	305	502	F
417	305	502	417	F
305	305	502	417	H
415	305	502	417	F
502	415	417	502	H
518	417	502	518	F
417	417	502	518	H
305	502	518	305	F
415	518	305	415	F
502	305	415	502	F
520	415	502	520	F
518	502	520	518	F
417	520	518	417	F
305	518	417	305	F
502	417	305	502	F
415	305	502	415	F
520	502	415	520	F
518	415	520	518	F

Page faults = 16
Hits = 4
Hit ratio = 4/20 = 0.20

(b) LRU (Least Recently Used) [Student 1: explain the LRU algorithm]

Ref	Frame3	Frame2	Frame1	Note
415	-	-	415	F
305	-	415	305	F
502	415	305	502	F
417	305	502	417	F
305	502	417	305	F
415	417	305	415	F
502	305	415	502	F
518	415	502	518	F
417	502	518	417	F
305	518	417	305	F
415	417	305	415	F
502	305	415	502	F
520	415	502	520	F
518	502	520	518	F
417	520	518	417	F
305	518	417	305	F
502	417	305	502	F
415	305	502	415	F
520	305	415	520	F
518	415	520	518	F

Page faults = 19

Hits = 1

Hit ratio = 1/20 = 0.05

(c) Optimal

Ref	Frame3	Frame2	Frame1	Note
415	-	-	415	F
305	-	415	305	F
502	415	305	502	F
417	305	502	417	F
305	305	502	417	H
415	305	502	415	F
502	305	415	502	H
518	415	502	518	F
417	502	518	417	F
305	518	417	305	F

415	417	305	415	F
502	305	415	502	H
520	415	502	520	F
518	502	520	518	F
417	520	518	417	F
305	518	417	305	F
502	417	305	502	H
415	305	502	415	H
520	305	415	520	H
518	305	520	518	H

Page faults = 12
Hits = 8
Hit ratio = 8/20 = 0.40

Show your work for each of the algorithms. Calculate the hit ratio for each of the algorithms.

(ii) [0.1 marks] Assume that the case above is repeated with 4 Frames Allocated. Repeat all three algorithms from Part (i) with 4 frames allocated to the program. Show your work for each of the algorithms. Calculate the hit ratio for each of the algorithms.

Ref	F4	F3	F2	F1	Note
415	-	-	-	415	F
305	-	-	415	305	F
502	-	415	305	502	F
417	415	305	502	417	F
305	415	305	502	417	H
415	415	305	502	417	H
502	415	305	502	417	H
518	305	502	417	518	F
417	305	502	417	518	H
305	305	502	417	518	H
415	502	417	518	415	F
502	502	417	518	415	H
520	417	518	415	520	F
518	417	518	415	520	H
417	417	518	415	520	H
305	518	415	520	305	F
502	415	520	305	502	F

415	520	305	502	415	F
520	520	305	502	415	H
518	305	502	415	518	F

FIFO
Page faults = 10
Hits = 10
Hit ratio = 10/20 = 0.50

Ref	F4	F3	F2	F1	Note
415	-	-	-	415	F
305	-	-	415	305	F
502	-	415	305	502	F
417	415	305	502	417	F
305	305	502	417	415	H
415	502	417	415	305	F
502	417	415	305	502	H
518	415	305	502	518	F
417	305	502	518	417	F
305	502	518	417	305	H
415	518	417	305	415	H
502	417	305	415	502	H
520	305	415	502	520	F
518	415	502	520	518	F
417	502	520	518	417	F
305	520	518	417	305	F
502	518	417	305	502	H
415	417	305	502	415	H
520	305	502	415	520	H
518	502	415	520	518	H

LRU
Page faults = 17
Hits = 3
Hit ratio = 3/20 = 0.15

Ref	F4	F3	F2	F1	Note
415	-	-	-	415	F
305	-	-	415	305	F
502	-	415	305	502	F

417	415	305	502	417	F
305	415	305	502	417	H
415	415	305	502	417	H
502	415	305	502	417	H
518	305	502	417	518	F
417	305	502	417	518	H
305	305	502	417	518	H
415	502	417	518	415	F
502	417	518	415	502	H
520	518	415	502	520	F
518	415	502	520	518	H
417	502	520	518	417	F
305	520	518	417	305	F
502	518	417	305	502	H
415	417	305	502	415	H
520	305	502	415	520	H
518	502	415	520	518	H

Optimal
Page faults = 9
Hits = 11
Hit ratio = 11/20 = 0.55

(iii) [0.2 marks] Based on your results, answer the following questions: Which algorithm performs best with 3 frames and why? Which algorithm performs best with 4 frames and why? How do the results change when more frames are allocated? What is the relationship? Why is the Optimal algorithm impractical in real-world operating systems? Compare the performance of FIFO and LRU. When might FIFO be better or worse than LRU?

Optimal is best with both 3 frames and four frames with a hit ratio of 0.4 for 3 and 0.55 for four but realistically the optimal algorithm is impractical for real world systems because it requires complete knowledge of future references which is impossible at runtime, it's a useful benchmark for the system's maximum performance.

Among realistic options for both 3 frames and four frames the FIFO algorithm outperformed LRU for this set of data. FIFO is good for data sets that have standardized aging. A page that arrives serves its purpose ages and leaves, however if an old page is still heavily used this causes a problem because FIFO will get rid of it regardless of how frequently it is called this is where LRU would perform better.

Our exercise suggests that more frames results in fewer faults and thus a higher hit and hit ratio.

(practice exercise for the final: repeat with LFU) [Student 2: explain the LFU algorithm]

LFU (Least Frequently Used): remove the page with the fewest total references so far. It keeps pages that are used many times, even if they weren't used very recently. This can be good for pages used consistently over time, but can get stuck keeping pages that were heavily used in the past but are no longer needed.

2. [0.3 marks] Consider a system with memory mapping done on a page basis. Assume that the necessary page table is always in main memory. A single main memory access takes 120 nanoseconds (ns).

(a) [0.1 marks] How long does a paged memory reference take in this system without a TLB? Explain your answer.

We need one memory access to read the page table entry and one memory access to access the actual data. Time = 2 X 120ns = 240ns. 240ns per memory reference.

(b) [0.1 marks] If we add a Translation Lookaside Buffer (TLB) that imposes an overhead of 20 ns on a hit or a miss. If we assume a TLB hit ratio of 95%, what is the effective memory access time (EMAT)? Explain your answer.

If TLB hits then its the lookup (20ns) plus the memory access (120ns) for a total of 140ns. If TLB misses then its the lookup (20ns) plus memory access to fetch the page table entry (120ns) plus the memory access (120ns) for a total of 260ns.

Using the EMAT (Effective Memory Access) formula =

EMAT = h*140+(1-h)*260 = 0.95*140+(1-0.95)*260 = 146ns

(c) [0.1 marks] Why does adding an extra layer, the TLB, generally improve performance? Are there situations where the performance may be worse with a TLB than without one?Explain all cases.

Most programs will visit the same few pages overall so you don't frequently pay the double memory access fee and stay closer to the 140ns range however if the program for some reason never or very infrequently reuses the same pages you have to pay the double memory access fee more frequently pushing you back towards the 260ns range.

3. [0.3 marks] Consider a system with a paged logical address space composed of 128 pages of 4 Kbytes each, mapped into a 512 Kbytes physical memory space. Answer the following questions and justify your answers.

Converting everything to base 2:
4Kb = 4 x 1024 = 2^{12} bytes
128 = 2^7

(a) [0.1 marks] What is the format and size (in bits) of the processor's logical address?

Logical address = page number + offset = 7 + 12 = 19 bits

(b) [0.1 marks] What is the required length (number of entries) and width (size of each entry in bits, disregarding control bits) of the page table?

Length is one entry per logical page = 128, width size of each entry in bits 128 is 2^7 therefore 7 bits

(c) [0.1 marks] What is the effect on the page table width if now the physical memory space is reduced by half (from 512 Kbytes to 256 Kbytes)? Assume that the number of page entries and page size remain the same.

256Kb = 256 x 1024 = 262144 = 2^{18} bytes
$2^{18}/2^{12} = 2^6$ = 64 frames

[Student 1: explain what the physical memory space is]

Physical memory space refers to actual main memory in hardware. Physical addresses refer to real locations in RAM.

[Student 2: explain what the logical memory space is]

Logical memory refers to what each process sees as the set of virtual addresses the program uses. Virtual addresses need to be translated and do not refer to real locations on RAM; they just allow each program to act like it has its own memory while it's actually fragmented between all of them.

4. [0.1 marks] Explain, in detail, the sequence of operations and file system data structure accesses that occur when a process executes the lseek(fd, offset, SEEK_END) system call. Consider a system using a hierarchical directory structure and assume the file described by the file descriptor (fd) is not currently open by any other process.

When a process calls lseek(fd, offset, SEEK_END), the request enters the kernel through a system-call trap. The kernel validates the file descriptor using the process's file descriptor table, then retrieves the corresponding open-file table entry, which contains a pointer to the file's inode. The inode stores the file's metadata, including its size. For SEEK_END, the kernel reads the file size from the inode and computes the new offset as file_size + offset. It updates the offset field inside the open-file table entry and returns the new position to the process.

5. File System Organization

a) [0.1 marks] (from Silberschatz) Consider a file system that uses inodes to represent files. Disk blocks are 8Kb in size, and a pointer to a disk block requires 4 bytes. This file system has 12 direct disk blocks, as well as single, double, and triple indirect disk blocks. What is the maximum size of a file that can be stored in this file system?

Pointers per block = (8 x 1024)/4 = 2048

Blocks = 12 + 2048 + 2048^2 + 2048^3 = 8593186852

Max size = 8192 x 8593186852= 70403120791552 bytes or 70.4Tb

b) [0.1 marks] Explain what you can do in case (a) if you need to store a file that is larger than the maximum size computed. Give an example showing how you can define a larger file, and what the size of that file would be.

You could add a quadruple indirect pointer which would add another 2048^4 blocks, or add more triple indirect pointers, or increase block size. I'll double the amount of triple indirect pointers for this example which is the same calculation except we add 2 x 2048^3 instead of one.

Blocks = 12 + 2048 + 2048^2 + 2(2048^3) = 17184065548

Max size = 8192 x 17184065548 = 140771864969216 bytes or 140.8Tb

Approximately double the previous 70.4Tb

Github Link Part 1:

https://github.com/BunyanYorganci/SYSC4001_A3_P1

Github Link Part 2:

https://github.com/BunyanYorganci/SYSC4001_A3_P2