Exercise session
(Memory management / Virtual Memory)

Operating Systems – EDA092/DIT400

Vincenzo Gulisano
vincenzo.gulisano@chalmers.se

UNIVERSITY OF
GOTHENBURG

Exercise 1

Jake and Finn have a computer whose memory access time is 150 nanoseconds and whose page-fault service time is 5 milliseconds. A process previously run in an operating system without virtual memory was 2 times faster than the same process run by the actualized operating system (the latter having virtual memory). Every how many page accesses are Jake and Finn experiencing a page fault on average? Would buying a new disk able to bring the page-fault service time to 2 milliseconds make the process no more than 1.5 times slower in the new operating system (compared to the previous one)?

Exercise 1 - solution

EAT = (1 – p) x (150 nanoseconds) + p (5 milliseconds)
= (150 – p x 150 + p x 5,000,000)
= 150 + p x 4,999,850

Question 1

Without virtual memory p = 0 $\rightarrow$ EAT = 150

With virtual memory 150 + p x 4,999,850 = 2 x 150

p = 0.00003

1 page fault every 1/p (33333) on average

Question 2

150 + 0.00003 x 1,999,850 < 1.5 x 150 ?

209.9 < 225? $\rightarrow$ Yes

Exercise 2

- Given the following reference string

  7 2 1 2 0 1 1 4 2 3 0 7 0 1 2 7 7 3 0 0 3 0 3

  and assuming 4 frames are available, find:
  - The minimum number of page faults we can observe
  - The extra page faults caused by the FIFO replacement algorithm
  - The extra page faults caused by the LRU replacement algorithm

Exercise 2 - solution

OPTIMAL

7 2 1 2 0 1 1 4 2 3 0 7 0 1 2 7 7 3 0 0 3 0 3

7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	0	0	0	0	0
	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
		1	1	1	1	1	4	4	3	3	3	3	3	3	3	3	3	3	3	3	3	3
				0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1

Page faults

1	1	1	0	1	0	0	1	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0

8

Exercise 2 - solution

FIFO

7 2 1 2 0 1 1 4 2 3 0 7 0 1 2 7 7 3 0 0 3 0 3

7	7	7	7	7	7	7	4	4	4	4	4	4	4	2	2	2	2	2	2	2	2	2
	2	2	2	2	2	2	2	2	3	3	3	3	3	3	3	3	3	3	3	3	3	3
		1	1	1	1	1	1	1	1	1	7	7	7	7	7	7	7	0	0	0	0	0
				0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1

Page faults

1	1	1	0	1	0	0	1	0	1	0	1	0	1	1	0	0	1	1	0	0	0	0

11

Exercise 2 - solution

LRU

7 2 1 2 0 1 1 4 2 3 0 7 0 1 2 7 7 3 0 0 3 0 3

7	7	7	7	7	7	7	4	4	4	4	7	7	7	7	7	7	7	7	7	7	7	7
	2	2	2	2	2	2	2	2	2	2	2	2	1	1	1	1	1	0	0	0	0	0
		1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	3	3	3	3	3	3
				0	0	0	0	0	3	3	3	3	3	2	2	2	2	2	2	2	2	2

Page faults

1	1	1	0	1	0	0	1	0	1	1	1	0	1	1	0	0	1	1	0	0	0	0

12

Exercise 2 - solution

- Minimum is 8
- FIFO is +3
- LRU is +4

Exercise 3

- Given processes P_1,P_2,P_3 and P_4 having sizes of 20, 25, 50 and 4 pages and given a total number of frames equal to 80 (10% of which reserved to the OS), compute the number of frames allocated to each process if frames are allocated proportionally to the size of each program

Exercise 3 - solution

- Frames for P_1: 20/99*72 = 15
- Frames for P_2: 25/99*72 = 18
- Frames for P_3: 50/99*72 = 36
- Frames for P_4: 4/99*72 = 3

Notice that I’m rounding to the nearest whole number otherwise I will violate the assumption that 8 pages should be devoted to the OS!

Exercise 4

- Suppose a process P has size of 100 bytes. Compute (1) the number of wasted bytes caused by internal fragmentation if the page and frame sizes are set to 2^4 bytes and (2) the size in bits of the page table (assume each frame entry requires 1 byte and that dirty bits are also use to speed the swap out of pages).

Exercise 4 - solution

- Page size = 16 bytes
- Number of pages required = ceil(100/16) = 7
- Wasted bytes because of internal fragmentation 16 –100%16 = 12 bytes

- Page table = 7 * (8 + 1 + 1) = 70 bits
    - 8 bits → frame entry
    - 1 bit → valid/invalid bit
    - 1 bit → dirty bit