CSC 4103 - Operating Systems
Fall 2009

# Lecture - XXV
# Final Review

Tevfik Koşar

Louisiana State University
December 3rd, 2009

## Announcements

* You should have received your grades as well as graded papers for:
  - Homework 1-4
  - Quiz 1-3
  - Midterm

If not, please see me.

* Only two quizzes with highest grade out of three will be counted.

* Project 1 is still being graded, will be announced soon.
* Project 2 is due this Saturday @11:59pm.

# Final Exam

December 10th, Thursday
3:00pm-5:00pm
@1112 Patrick Taylor Hall

## Chapters included in Final

| | | |
|---|---|---|
| Ch. 3.2-3.4 | (Processes) | ~ 30% |
| Ch. 4.2-4.4 | (Threads) | |
| Ch. 5.2-5.3 | (CPU Scheduling) | |
| Ch. 6.5-6.7 | (Synchronization) | |
| Ch. 7-2-7.6 | (Deadlocks) | |
| Ch. 8.1-8.6 | (Main Memory) | ~ 70% |
| Ch. 9.1-9.6 | (Virtual Memory) | |
| Ch. 11.1-11.5 | (File Systems) | |
| Ch. 12.1-12.7 | (Mass Storage & IO) | |
| Ch. 15.1-15.5 | (Security) | |
| Ch. 18.1,18.2,18.5 | (Distr. Coordination) | |

## For Pre-Midterm topics

- Ch. 3.2-3.4 (Processes)
- Ch. 4.2-4.4 (Threads)
- Ch. 5.2-5.3 (CPU Scheduling)
- Ch. 6.5-6.7 (Synchronization)
- Ch. 7-2-7.6 (Deadlocks)

- Please revise Midterm Review class which is available at:
- http://www.cct.lsu.edu/~kosar/csc4103/slides/14_Midterm_Review.pdf

## 8. Main Memory

- Contiguous Allocation
- Dynamic Allocation Algorithms
- Fragmentation
- Address Binding
- Address Protection
- Paging
- Segmentation

# 9. Virtual Memory

- Demand Paging
- Page Faults
- Page Replacement
- Page Replacement Algorithms (FIFO, LRU, Optimal etc)
- Performance of Demand Paging

# 11. File Systems

- Directory structure & implementation
- File allocation methods
  - contiguous, linked, indexed
- Free space management
  - bit vectors, linked lists, grouping, counting

# 12. Mass Storage & I/O

- Disk Mechanism & Structure
- Disk Scheduling Algorithms
  - FCFS, SSTF, SCAN, LOOK, C-SCAN, C-LOOK
- Hierarchical Storage Management
- RAID Architectures
  - RAID 0-6, RAID 0+1, RAID 1+0

# 15. Security

- Security Violation Categories
- Security Violation Methods
- Program & Network Threats
- Cryptography
- Symmetric & Asymmetric Encryption
- Key distribution

# 16. Distributed Coordination

- Event Ordering
  - Happened before relationship
- Distributed Mutual Exclusion
  - Centralized & Fully Distributed Approaches
- Distributed Deadlock Prevention
  - Resource Ordering
  - Timestamp Ordering (Wait-die & Wound-wait)
- Distributed Deadlock Detection
  - Centralized & Fully Distributed Approaches

# Exercise Questions

# Question 1-a

- Consider the following page-reference string:

1, 2, 3, 4, 4, 3, 2, 1, 5, 6, 2, 1, 2, 3, 7, 8, 3, 2, 1, 5

How many page faults, page hits, and page replacements would occur for the following replacement algorithms, assuming 4 memory frames? Show your page assignments to frames.

(a) Assuming LRU page replacement algorithm is used.

| 1 | 2 | 3 | 4 | 4 | 3 | 2 | 1 | 5 | 6 | 2 | 1 | 2 | 3 | 7 | 8 | 3 | 2 | 1 | 5 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

# of page faults:
# of page hits:
# of page replacements:

# Question 1-b

(b) Assuming Optimal page replacement algorithm is used.

| 1 | 2 | 3 | 4 | 4 | 3 | 2 | 1 | 5 | 6 | 2 | 1 | 2 | 3 | 7 | 8 | 3 | 2 | 1 | 5 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

# of page faults:
# of page hits:
# of page replacements:

# Question 2

- Assume a disk with 500 cylinders is accessing cylinder 100 right now. Prior cylinder 100, the disk head accessed cylinder 101. Further assume that the FIFO queue of pending requests is 102, 20, 450, 60, 80, 220, 330, 250, 101, 190. What order will the pending requests be satisfied using the following scheduling algorithms?

(a) Circular Scan disk-scheduling policy?

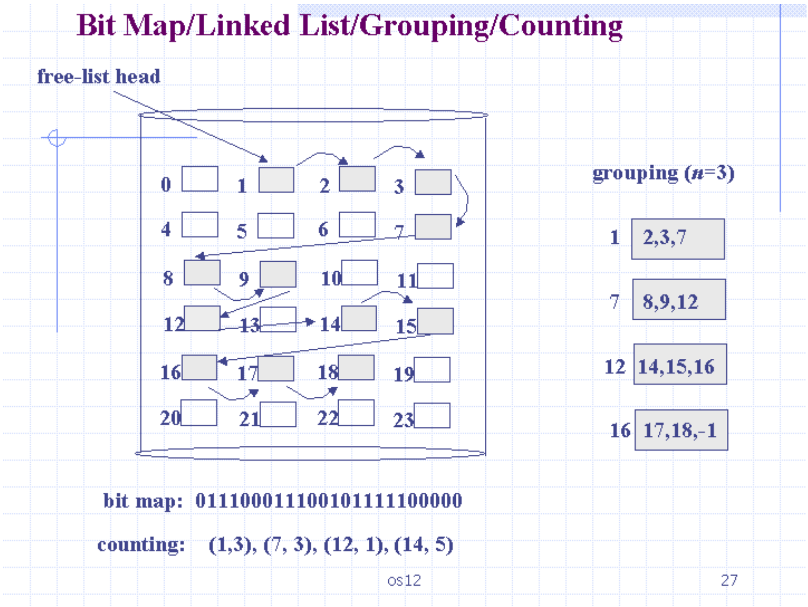
(b) SSTF disk-scheduling policy?

(c) Which of the above algorithms is more efficient in this particular case, and why? (Please show your work and justify your answer)

# Question 3

- Explain why a bit vector implementation of a free block list can provide increased reliability and performance compared with keeping a list of free blocks where the first few bytes of each free block provide the logical sector number of the next free block.

# Remember



## Bit Map/Linked List/Grouping/Counting

grouping ($n$=3)

| | |
|---|---|
| 1 | 2,3,7 |
| 7 | 8,9,12 |
| 12 | 14,15,16 |
| 16 | 17,18,-1 |

bit map: 011100011100101111100000

counting: (1,3), (7, 3), (12, 1), (14, 5)

os12 27

# Solution 3

- Performance: bit vectors provide fast access to find clusters of adjacent free blocks.
- Reliability: if an item in a linked list is lost, the rest of the list is lost. With a bit vectors only the items are lost. Also, it’s possible to have multiple copies of the bit vector since it is a more compact representation.

# Question 4

Consider a demand-paged computer system where the degree of multi-programming is currently fixed at four. The system was recently measured to determine utilization of CPU and the paging disk. The results are one of the following alternatives. For each case, what is happening (in one phrase)? Can you increase the degree of multiprogramming to increase the CPU utilization?

(a) CPU utilization 12 percent; disk utilization 2 percent.

b) CPU utilization 86 percent; disk utilization 4 percent.

c) CPU utilization 10 percent; disk utilization 95 percent.

# Solution 4

(a) CPU utilization 12 percent; disk utilization 2 percent.

Answer: both CPU and disk utilization are low. We can increase the degree of multiprogramming to increase CPU utilization.

b) CPU utilization 86 percent; disk utilization 4 percent.

Answer: CPU utilization is sufficiently high to leave things alone; increasing the degree of multiprogramming may decrease the CPU utilization.

c) CPU utilization 10 percent; disk utilization 95 percent.

Answer: thrashing is occurring. We cannot increase the CPU utilization

# Question 5

- Consider a demand-paging system with the following time-measured utilization:

CPU utilization 18%
Paging disk 96%
Other I/O devices 6%

For each of the following, say whether it will (or is likely to) improve CPU utilization. Answer with YES or NO or LIKELY, and justify your answers.

(a) Install a faster CPU.

(b) Install a bigger paging disk.

(c) Decrease the degree of multiprogramming.

(d) Install more main memory.

# Solution 5

- (a) Install a faster CPU.

NO. a faster CPU reduces the CPU utilization further since the CPU will spend more time waiting for a process to enter in the ready queue.

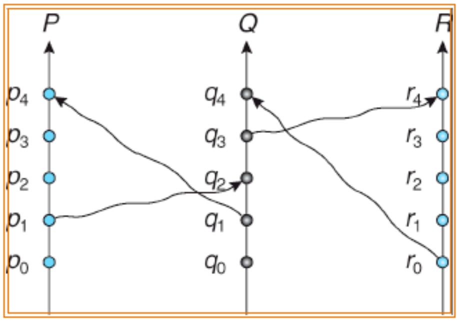
(b) Install a bigger paging disk.
NO. the size of the paging disk does not affect the amount of memory that is needed to reduce the page faults.

(c) Decrease the degree of multiprogramming.
YES. by suspending some of the processes, the other processes will have more frames in order to bring their pages in them, hence reducing the page faults.

(d) Install more main memory.
Likely. more pages can remain resident and do not require paging to or from the disks.

# Question 6

- Consider the following diagram which shows the relative time for three concurrent processes: P, Q, and R.



Which of the following event orderings are true?

(a) p4 and q4 are concurrent :
(b) q3 and r1 are concurrent :
(c) p0 happens before q1 :
(d) p1 happens before r4 :
(e) q0 happens before p3 :

# Question 7

- Given the following memory partitions (in kilobytes): 200, 600, 500, 800, 400, 300 (in order); how would each of the first-fit, best-fit, and worst-fit algorithms place processes of 292, 522, 138, 770, 162, 418 (in order).
- Which algorithm makes the most efficient usage of memory?

# Question 8

- Consider the following segment table:

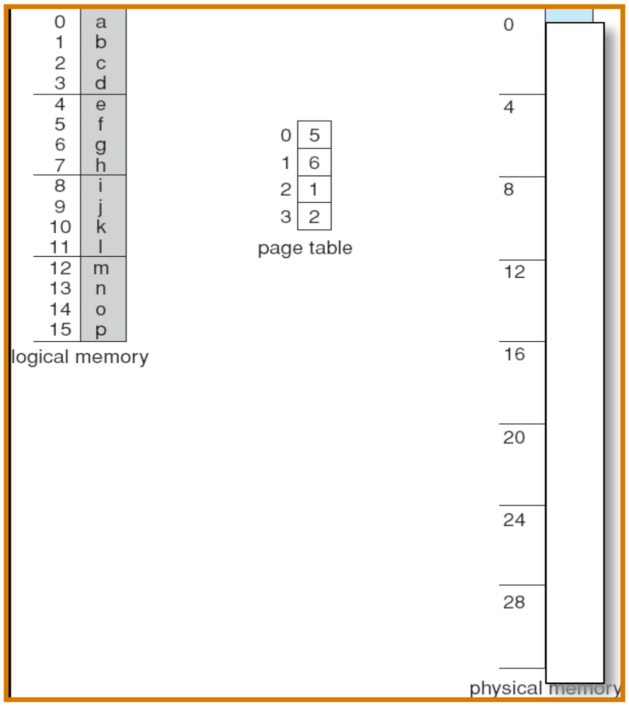
| Segment | Base | Length |
|---|---|---|
| 0 | 219 | 600 |
| 1 | 2300 | 14 |
| 2 | 90 | 100 |
| 3 | 1327 | 580 |
| 4 | 1952 | 96 |

What are the physical addresses for the following logical addresses?

a. 0, 100
b. 2, 100
c. 3, 0
d. 4, 10

# Question 9

- Consider the paging table on the right. What are the physical addresses of the following logical addresses [p,d] :
- a) 0,0
- b) 1,4
- c) 2,3



| | |
|---|---|
| 0 | a |
| 1 | b |
| 2 | c |
| 3 | d |
| 4 | e |
| 5 | f |
| 6 | g |
| 7 | h |
| 8 | i |
| 9 | j |
| 10 | k |
| 11 | l |
| 12 | m |
| 13 | n |
| 14 | o |
| 15 | p |

logical memory

| | |
|---|---|
| 0 | 5 |
| 1 | 6 |
| 2 | 1 |
| 3 | 2 |

page table

# Question 10

- Consider the following demand paging system:

Memory access time = 1 microsecond

40% of the time the page that is being replaced has been modified and therefore does need to be swapped out

Swap Page Time = 10 msec = 10,000 microsec

a) What is the Effective Access Time (EAT) if 1 out of 1000 memory accesses result in a page fault?

b) What if we only want 20% performance degradation?