EECS 678 – Spring 2014

Final Exam Preparation Questions

1 Chapter 6

1. (a) ____ is the amount of time to execute a particular process (submission time to completion time).
   (b) ____ is the amount of time a process is waiting in the ready queue.
   (c) ____ modelling uses a pre-determined workload and defines the performance of each algorithm for that workload.

2. Describe the difference between preemptive and non-preemptive scheduling. State why strict non-preemptive scheduling is unlikely to be used on machines supporting programs that provide interactive user services.

3. The processes listed below(P1, P2, P3, P4, and P5) are assume to all arrive at time 0. Perform the following analysis addressing how different scheduling algorithms would execute

Name	Burst Time	Priority
P1	3	1
P2	7	2
P3	2	2
P4	4	4
P5	1	3

   these processes, and how each would perform as measured by different metrics.

   - (a) Draw 4 Gantt charts illustrating the execution of these processes using FCFS, SJF, non-preemptive priority (a smaller priority number implies a higher priority), and RR (quantum=1) scheduling
   - (b) What is the turnaround time of each process for each of the scheduling algorithms specified in part (a)?
   - (c) What is the waiting time of each process for each of the scheduling algorithms given in part (a)?
   - (d) Which of the algorithms in part (a) results in the minimum average waiting time (over all processes)?

4. Most round-robin schedulers use a fixed size CPU time quantum for allocating CPU time. Large quantum sizes provide certain advantages to the system, while small quantum sizes provide other advantages. Assume that you are designing a system where throughput is more important than response time, while use of round-robin scheduling is required. Explain whether you would use a relatively large or relatively small quantum value for such a system, and why.

Name	Burst Time	Priority
P1	15	2
P2	5	8
P3	11	5
P4	9	1
P5	8	7

5. Five batch jobs, P1 through P5, arrive at a computer center at essentially the same time. Their burst times and priorities are defined by the table below. 0 is the best priority. For each of the following scheduling algorithms, determine the turnaround time for each process, and the average turnaround for all jobs. Ignore process switching overhead. Show how you arrived at your answers.

   Except for Round Robin, assume that only one job at a time runs, until it finishes. All jobs are completely CPU bound.

   - First Come First Serve (P1 first, P6 last)
   - Shortest Job First
   - Non-Preemptive Priority
   - Round Robin with a quantum of 1

6. Devise an approximation formula to estimate the length of the next CPU burst for the following criteria: (a) length of last CPU burst and past prediction histories are equally weighted. (b) past prediction history does not count.

   What would each formula *guess* for each of its next CPU bursts for a sequence of real CPU burst times that appear as following:
   CPU burst (t_i): 4 10 10 10 2 6 10 10 6 4 4
   Assume that the *initial* guess time is 6 time units.

7. Why is *aging* used during priority scheduling algorithms?

8. What is multi-level queue scheduling? Explain its main features.

9. During lottery scheduling each short job is assigned 5 tickets and each long job is given 2 tickets. How much CPU will be allocated to each short and long job is the system contains: (a) 5 short and 5 long jobs, (b) 1 short and 10 long jobs.

10. In ____ contention scope, all the threads in a single process are mapped to a single kernel thread.

11. What is load balancing on multi-threaded systems? How does the OS perform load balancing?

2 Chapter 8

1. Illustrate how to protect processes from one-another using the *base* and *limit* registers.
2. Explain compiler time, load time, and execution time address binding. What type of binding is generally used in current OS?
3. Discuss the differences between logical and physical addresses and address spaces as they relate to an executing program.
4. Explain static and dynamic loading.
5. What is dynamic linking ?
6. What is the need for swapping ?
7. What are the drawbacks of contiguous memory allocation ?
8. Explain, compare, and contrast the following resource allocation policies: (a) First Fit, (b) Best Fit, and (c) Worst Fit.
9. Compare and contrast internal and external fragmentation.
10. What is paging? How does it overcome the drawbacks of contiguous memory allocation ?
11. Assuming a page size of 2K and a 32-bit machine, how many bits are used for the page number, and how many are used for the offset within the page? Give the main reason why page sizes are usually powers of two.
12. Consider a system with memory mapping done on a page basis, and using a single level page table. Assume that the necessary page table is always in memory.
    (a) If a memory reference takes 100 nanoseconds, how long does a memory mapped reference take if there is no TLB within the MMU to cache recently used page addresses?
    (b)Now we add an MMU which imposes an overhead of 15 nanoseconds on a hit or a miss. If we assume that 90 percent of all memory references hit in the MMU TLB, what is the Effective Memory Access time?
13. When is a simple page table structure ineffective ? Explain the properties of (a) hierarchical page tables, (b) hashed page table, and (c) inverted page tables.
14. How is segmentation different from the paging memory model ? Illustrate the segmentation architecture.
15. Is there any advantage to supporting multiple page *sizes*, as is done on the Pentium ?
16. (Problem 8.11) Given five memory partitions of 100 KB, 500 KB, 200 KB, 300 KB, and 600 KB (in order), how would each of the first-fit, best-fit, and worst-fit algorithms place processes of 212 KB, 417 KB, 112 KB, and 426 KB (in order)?Which algorithm makes the most efficient use of memory?
17. Assuming a 1 KB page size, what are the page numbers and offsets for the following address references (provided as decimal numbers): a. 2375, b. 19366, c. 30000, d. 256, e. 16385

18. (problem 8.24) Consider a computer system with a 32-bit logical address and 4 KB page size. The system supports up to 512 MB of physical memory. How many entries are there in:
    a. A conventional single-level page table?
    b. An inverted page table?

19. (problem 8.28) Consider the following segment table: What are the physical addresses for

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

    the following logical addresses? a. 0,430, b. 1,10, c. 2,500, d. 3,400, e. 4,112

3 Chapter 9

1. Explain the need for Virtual Memory. Do you know any modern OS that use virtual memory concepts?

2. Explain demand paging.

3. Under what circumstances do page faults occur? Describe the actions taken by the operating system when a page fault occurs.

4. When would the OS invoke a page replacement algorithm ? Explain basic page replacement.

5. Assume that you have a page-reference string for a process with M frames (initially all empty). The page-reference string has length P; N distinct page numbers occur in it. Answer these questions for any page replacement algorithm:
    (a) What is a lower bound on the number of page faults?
    (b) What is an upper bound on the number of page faults?

6. What is the Belady's anomaly? Which page replacements suffer from it, and which are immune?

7. Explain two LRU implementation algorithms (counter and stack implementation).

8. How can *victim frames* optimize page fault handling ?

9. Should there be a minimum and/or maximum limit on the number of frames allocated to each process ? Explain.

10. Distinguish between: (a) Equal and proportional allocation, (b) Global and local page replacement algorithms.

11. What is thrashing? How may it be caused?

12. 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? Can you increase the degree of multiprogramming to increase the CPU utilization?
    (a) CPU utilization, 13 percent; disk utilization, 97 percent
    (b) CPU utilization, 87 percent; disk utilization, 3 percent
    (c) CPU utilization, 13 percent; disk utilization, 3 percent.

13. What are the benefits of the *slab allocation* policy over the buddy allocator.

14. Explain some issues in deciding the page *size* to use.

15. List costs and benefits of using virtual memory.

16. (Problem 9.8)

17. A certain computer provides its users with a virtual-memory space of 2^{32} bytes. The computer has 2^{18} bytes of physical memory. The virtual memory is implemented by paging, and the page size is 4096 bytes. A user process generates the virtual address 0x11123456. Explain how the system establishes the corresponding physical location. Distinguish between software and hardware operations.

18. Table 9.1 is a page table for a system with 12-bit virtual and physical addresses with 256-byte pages. The list of free page frames is D, E, F (that is, D is at the head of the list, E is second, and F is last.)

Page	Page Frame
0	-
1	2
2	C
3	A
4	
5	4
6	3
7	
8	B
9	0

    Given the following virtual addresses, convert them to their equivalent physical addresses in hexadecimal. All numbers are given in hexadecimal. (A dash for a page frame indicates the page is not in memory.) (a) 9EF, (b) 111, (c) 700, (d) 0FF

19. What is the cause of thrashing? How does the system detect thrashing? Once it detects thrashing, what can the system do to eliminate this problem?

20. Assume there is an initial 1024 KB segment where memory is allocated using the Buddy system. Using Figure 9.27 as a guide, draw the tree illustrating how the following memory requests are allocated: (a) request 240 bytes (b) request 120 bytes (c) request 60 bytes (d) request 130 bytes.

    Next, modify the tree for the following releases of memory. Perform coalescing whenever possible: (a) release 240 bytes (b) release 60 bytes (c) release 120 bytes.

4 Chapter 10

1. Explain disk terms: platter, head, track, cylinder, sector.
2. What is a *head crash*?
3. What is the disk *seek time* and *rotational delay*?
4. What is the *average* rotational latency of a disk rotating at 5,600 RPM ?
5. Serial ATA bus interface is faster than parallel ATA interface. Explain.
6. All the disk scheduling algorithms, except FCFS, are not truly fair (starvation may occur). Explain why this is true.
7. What are benefits of arranging disks in a RAID array ?
8. What intuition is used in RAID level 3 to improve cost over RAID level 2 ?
9. Suppose that a disk drive has 5000 cylinders, numbered 0 to 4999. The drive is currently serving a request at cylinder 143, and the previous request was at cylinder 125. The queue of pending requests, in FIFO order, is:

   86, 1470, 913, 1774, 948, 1509, 1022, 1750, 130.

   Starting from the current head position, what is the total distance (in cylinders) that the disk arm moves to satisfy all the pending requests, for each of the following disk-scheduling algorithms? a. FCFS b. SSTF c. SCAN d. LOOK e. C-SCAN
10. Could a RAID Level 1 organization achieve better performance for read requests than a RAID Level 0 organization (with nonredundant striping of data)? If so, how?

5 Chapter 11

1. The OS file system interface is said to provide an abstraction over reality. Explain.
2. What is a *raw* partition? When might it be useful?
3. What are the advantages and disadvantages of : (a) Single-level directory, (b) Tree structured directory.
4. How do acyclic graph directory structure overcome the limitation of tree structured directory structure ?
5. Explain soft and hard links, as used in Unix.
6. Explain in brief the file protection system in Linux.
7. Could you simulate a multilevel directory structure with a single-level directory structure in which arbitrary long names can be used? How? Would your answer change if the file names were limited to seven characters?
8. Consider a file system where a file can be deleted and its disk space reclaimed while links to that file still exist. What problems may occur if a new file is created in the same storage area or with the same absolute path name? How can these problems be avoided?