

Innopolis University, Academic Year 2016/2017

Written Exam of Operating Systems – Retake 2

27th January 2017

Questions and Solutions

Max 20 points + 6 points (30%) extra credit – Total time: 90 minutes

1. (3 points) What type of multiplexing (time, space, or both) can be used for sharing the following resources: CPU, memory, disk, network card, printer, keyboard, and display?

Time multiplexing: CPU, network card, printer, keyboard. Space multiplexing: memory, disk. Both: display.

2. (3 points) Multiple jobs can run in parallel and finish faster than if they had run sequentially. Suppose that two jobs, each needing 20 minutes of CPU time, start simultaneously. How long will the last one take to complete if they run sequentially? How long if they run in parallel? Assume 50% I/O wait.

If each job has 50% I/O wait, then it will take 40 minutes to complete in the absence of competition. If run sequentially, the second one will finish 80 minutes after the first one starts. With two jobs, the approximate CPU utilization is $1 - 0.52$. Thus, each one gets 0.375 CPU minute per minute of real time. To accumulate 20 minutes of CPU time, a job must run for $20/0.375$ minutes, or about 53.33 minutes. Thus running sequentially the jobs finish after 80 minutes, but running in parallel they finish after 53.33 minutes.

3. (3 points) Five batch jobs. A through E, arrive at a computer center at almost the same time. They have estimated running times of 10, 6, 2, 4, and 8 minutes. Their (externally determined) priorities are 3, 5, 2, 1, and 4, respectively, with 5 being the highest priority. For each of the following scheduling algorithms, determine the mean process turnaround time. Ignore process switching overhead.

- (a) Round robin.
- (b) Priority scheduling.
- (c) First-come, first-served (run in order 10, 6, 2, 4, 8).
- (d) Shortest job first.

For (a), assume that the system is multi-programmed, and that each job gets its fair share of the CPU. For (b) through (d), assume that only one job at a time runs, until it finishes. All jobs are completely CPU bound.

For round robin, during the first 10 minutes each job gets 1/5 of the CPU. At the end of 10 minutes, C finishes. During the next 8 minutes, each job gets 1/4 of the CPU, after which time D finishes. Then each of the three remaining jobs gets 1/3 of the CPU for 6 minutes, until B finishes, and so on. The finishing times for the five jobs are 10, 18, 24, 28, and 30, for an average of 22 minutes. For priority scheduling, B is run first. After 6 minutes it is finished. The other jobs finish at 14, 24, 26, and 30, for an average of 18.8 minutes. If the jobs run in the order A through E, they finish at 10, 16, 18, 22, and 30, for an average of 19.2 minutes. Finally, shortest job first yields finishing times of 2, 6, 12, 20, and 30, for an average of 14 minutes.

4. (3 points) A computer provides each process with 65,536 bytes of address space divided into pages of 4096 bytes each. A particular program has a text size of 32,768 bytes, a data size of 16,386 bytes, and a stack size of 15,870 bytes. Will this program fit in the machine's address space? Suppose that instead of 4096 bytes, the page size were 512 bytes, would it then fit? Each page must contain either text, data, or stack, not a mixture of two or three of them.

The text is eight pages, the data are five pages, and the stack is four pages. The program does not fit because it needs 17 4096-byte pages. With a 512-byte page, the situation is different. Here the text is 64 pages, the data are 33 pages, and the stack is 31 pages, for a total of 128 512-byte pages, which fits. With the small page size it is OK, but not with the large one.

5. (3 points) Consider a 4-TB disk that uses 4-KB blocks and the free-list method. How many block addresses can be stored in one block?

The number of blocks on the disk = $4 \text{ TB} / 4 \text{ KB} = 2^{30}$. Thus, each block address can be 32 bits (4 bytes), the nearest power of 2. Thus, each block can store $4 \text{ KB} / 4 = 1024$ addresses.

6. (3 points) The banker's algorithm is being run in a system with m resource classes and n processes. In the limit of large m and n , the number of operations that must be performed to check a state for safety is proportional to $m^a n^b$. What are the values of a and b ?

Comparing a row in the matrix to the vector of available resources takes m operations. This step must be repeated on the order of n times to find a process that can finish and be marked as done. Thus, marking a process as done takes on the order of mn steps. Repeating the algorithm for all n processes means that the number of steps is then mn^2 . Thus, $a = 1$ and $b = 2$.

7. (30% extra credit) Suppose that a system uses DMA for data transfer from disk controller to main memory. Further assume that it takes t_1 nsec on average to acquire the bus and t_2 nsec to transfer one word over the bus ($t_1 \gg t_2$). After the CPU has programmed the DMA controller, how long will it take to transfer 1000 words from the disk controller to main memory, if (a) word-at-a-time mode is used, (b) burst mode is used? Assume that commanding the disk controller requires acquiring the bus to send one word and acknowledging a transfer also requires acquiring the bus to send one word.

(a) Word-at-a-time mode: $1000 \times [(t_1 + t_2) + (t_1 + t_2) + (t_1 + t_2)]$

Where the first term is for acquiring the bus and sending the command to the disk controller, the second term is for transferring the word, and the third term is for the acknowledgement. All in all, a total of $3000 \times (t_1 + t_2)$ nsec.

(b) Burst mode: $(t_1 + t_2) + t_1 + 1000 \text{ times } t_2 + (t_1 + t_2)$

where the first term is for acquiring the bus and sending the command to the disk controller, the second term is for the disk controller to acquire the bus, the third term is for the burst transfer, and the fourth term is for acquiring the bus and doing the acknowledgement. All in all, a total of $3t_1 + 1002t_2$.

This is the end of the questions and solutions of the exam.