ELC-467 Operating Systems
Spring 2020

Problem Set (1): Process Scheduling

Study Questions

[Q1] On a system with n processors, what is the maximum and minimum number of processes that can be in the ready, running and blocked states?

[Q2] Suggest one application that would benefit from the use of multithreading and one application that would not.

[Q3] Some operating systems specify a maximum number of threads that a process can create. What is the advantage of this policy? Can you suggest an alternative method?

[Q4] Given a system with n processes that arrive at the same time, what is the number of possible non-preemptive schedules that can be used for these processes? Which one will minimize the average waiting time? Can you answer these questions for **(a)** preemptive schedules, **(b)** processes that arrive at different times?

[Q5] Consider a multiple queue scheduling algorithm which has two queues with time slices of 10 ms and 100 ms respectively. A new process is placed in the first queue until it runs for one time slice, and then it is moved to the second queue. System runs processes in the first queue before running any process in the second queue. What advantages would this algorithm have?

[Q6] In the Completely Fair Scheduler (CFS) used in Linux a minimum time during which each task should run at least once (target latency) is specified. Scheduler then works as follows:
- System records the running time spent so far by each task.
- System selects for execution the task that ran for the least time.
- This task can run for the target latency divided by the number of ready tasks.

Why is this scheduler fair? How can priorities be introduced into this scheduler?

Problems:

[P1] A multitasking system is required to run four processes A, B, C, and D, with arrival times of 0, 2, 8 and 10 ms, and process times of 8, 5, 4 and 5 ms, respectively.

a) Calculate the average process waiting time if round-robin scheduling with q=1 ms is used with negligible switching time.
b) Repeat part (a) for the cases of FCFS, SPN and SRT scheduling.
c) Repeat the above cases if context switching takes a non-zero time of x.

[P2] On a system using round-robin scheduling, assume s is the time needed to perform a process switch, q is the time quantum, and r is the average time a process runs before blocking on I/O. Give a formula for CPU efficiency in the following cases:

a) q tending to infinity. **b)** q > r **c)** s < q < r
d) s=q <r **e)** q tending to 0.

[P3] Consider a multitasking system that uses round-robin scheduling with a quantum *q*. This system runs three processes *A*,*B*, and *C* that arrive at the same time, and have process times of p_a,p_b,and p_c. Assume that $p_a < p_b < p_c$ and that processes are placed in the ready list in the order ABC.

a) Find an expression for the waiting time of process *A* if p_a is an integer multiple of *q* (neglect switching times.)
b) Using the expression obtained above, discuss the effect of the length of *q* on the waiting time of process *A*.
c) Does the waiting time of process *C* depend on the length of *q*? If your answer is no, what assumptions did you make?

[P4] The traditional UNIX scheduler enforces an inverse relationship between priority numbers and priorities: the higher the number, the lower the priority. The scheduler recalculates process priorities once per second using the following function:

Priority = (recent CPU usage /2) + base

where base = 60 and recent CPU usage refers to a value indicating how often a process has used the CPU since priorities were last recalculated.

Assume that recent CPU usage for process P1 is 40, for process P2 is 18, and for process P3 is 10. What will be the new priorities for these three processes when priorities are recalculated? Based on this information, does the traditional UNIX scheduler raise or lower the relative priority of a CPU-bound process?

[P5] A multitasking system is required to run four processes A, B, C, and D, with arrival times of 0, 2 ms, 3 ms, and 5 ms, and process times of 3 ms, 7 ms, 4 ms, and 8 ms, respectively.

a) Calculate the average process waiting time if RR scheduling with q=1 ms is used.
b) Repeat if system has two priority levels, with 1st and 2nd processes having higher priority. Higher priority processes run first and RR with q=1 is used for processes with the same priority. If process takes more than 5 quanta, its priority decreases.
c) If *non-preemptive* priority scheduling is used, what are the conditions on priorities such that process D can terminate before time t=16 ms.

[P6] A multitasking system runs four processes A, B, C, and D that arrive at the same time, and have process times of $p_A, p_{B,}, p_{C,}$ and p_D. with $p_A < p_B < p_C < p_D$.

a) Find the minimum average waiting time for these processors on a single processor.
b) Find the average waiting time if processes run on two identical processors using FCFS with a single ready queue. Assume that processes are placed in the queue in the order ABCD. Is this the minimum average waiting time on two processors?
c) Find the average waiting time if processes run on two identical processors, assuming round-robin scheduling with quantum q on each processor, and assignment of processes to the least- loaded processor. Make any necessary additional assumptions.
