Fall 2017 COMP 3511 Homework Assignment #2 Solution

Handout Date: Oct. 9, 2017 Due Date: Oct. 23, 2017

Name: ____________________ ID: ____________________
E-Mail: ________________ Section: _________________

Please read the following instructions carefully before answering the questions:

- You should finish the homework assignment **individually**.
- There are a total of **4** questions.
- When you write your answers, please try to be precise and concise.
- Fill in your name, student ID, email at the top of each page.
- Please fill in your answers in the space provided, or you can type your answers in the *Microsoft Word* file.

- **Homework Collection: the hardcopy** is required and the homework is collected in collection box **#20**, located outside **Room 4210**, near **Lift 21** (there are labels on the boxes).

1. [20 points] Multiple choices

1) Which one of the following is not shared by threads?
a) program counter
b) stack
c) both (a) and (b)
d) none of the mentioned
Answer: c

2) Which of the following scheduling algorithms could result in starvation?
①First-come, first served ②Shortest job first (non-preemptive) ③Round Robin ④Priority
a) ①②
b) ②④
c) ①③
d) ③④
Answer: b

3) Bounded waiting implies that there exists a bound on the number of times a process is allowed to enter its critical section:
a) after a process has made a request to enter its critical section and before the request is granted
b) when another process is in its critical section

c) before a process has made a request to enter its critical section
d) None of these
Answer: a

4) The basic thread operation related to the change in thread state that occurs when a thread needs to wait for an event is referred to as the:
a. Unblock operation
b. Spawn operation
c. Block operation
d. None of the above
Answer: c

5) Which scheduling algorithm allocates the CPU first to the process that requests the CPU first?
a) first-come, first-served scheduling
b) shortest job scheduling
c) priority scheduling
d) none of the mentioned
Answer: a

6) The following program consists of 3 concurrent processes and 3 binary semaphores. The semaphores are initialized as S0 = 1, S1 = 0, S2 = 0.

```
Process P0
while(true)
{
wait(S0);
print '0';
release(S1);
release(S2);
}

Process P1
wait(S1);
release(S0);

Process P2
wait(S2);
release(S0);
```

How many times will P0 print '0' ?
a) At least twice
b) Exactly twice
c) Exactly thrice
d) Exactly once

Answer: a

7) A mutex__, while a semaphore__:
a) must be accessed from only one process, must be accessed from only one process
b) must be accessed from only one process, can be accessed from multiple processes
c) can be accessed from multiple processes, must be accessed from only one process
d) can be accessed from multiple processes, can be accessed from multiple processes
Answer: b

8) Time quantum is defined in
a) shortest job scheduling algorithm
b) round robin scheduling algorithm
c) priority scheduling algorithm
d) multilevel queue scheduling algorithm
Answer: b

9) Thread-local storage is data that:
A) is unique to each thread
B) has been modified by the thread, but not yet updated to the parent process
C) is generated by the thread independent of the thread's process
D) is not associated with any process
Answer: a

10) A process P1 has a period of 50 and a CPU burst of t1 = 25, P2 has a period of 80 and a CPU burst of 35. The total CPU utilization is:
a) 0.90
b) 0.74
c) 0.94
d) 0.80
Answer: c

2）[30 points] Please answer the following questions in 3-4 sentences

a. (5 points) Using Amdahl's Law, calculate the speedup gain of an application that has a 60 percent parallel component for (a) two processing cores and (b) four processing cores.

Answer:

Two processing cores = 1.43 speedup; four processing cores = 1.82 speedup.

b. (5 points) Discuss how the following pairs of scheduling criteria conflict in certain settings.

a. CPU utilization and response time

b. Average turnaround time and maximum waiting time

c. I/O device utilization and CPU utilization

Answer:

a. CPU utilization and response time: CPU utilization is increased if the overheads associated with context switching is minimized. The context switching overheads could be lowered by performing context switches infrequently. This could, however, result in increasing the response time for processes.

b. Average turnaround time and maximum waiting time: Average turnaround time is minimized by executing the shortest tasks first. Such a scheduling policy could, however, starve long-running tasks and thereby increase their waiting time.

c. I/O device utilization and CPU utilization: CPU utilization is maximized by running long-running CPU- bound tasks without performing context switches. I/O device utilization is maximized by scheduling I/O-bound jobs as soon as they become ready to run, thereby incurring the overheads of context switches.

c. (5 points) Consider the exponential average formula used to predict the length of the next CPU burst. What are the implications of assigning the following values to the parameters used by the algorithm?

a. $\alpha = 0$ and $\tau_0 = 100$ milliseconds

b. $\alpha = 0.99$ and $\tau_0 = 10$ milliseconds

Answer:

When $\alpha = 0$ and $\tau_0 = 100$ milliseconds, the formula always makes a prediction of 100 milliseconds for the next CPU burst. When $\alpha = 0.99$ and $\tau_0 = 10$ milliseconds, the most recent behavior of the process is given much higher weight than the past history associated with the process. Consequently, the scheduling algorithm is almost memoryless, and simply predicts the length of the previous burst for the next quantum of CPU execution.

d. (5 points) A variation of the round-robin scheduler is the regressive round-robin scheduler. This scheduler assigns each process a time quantum and a priority. The initial value of a time quantum is 50 milliseconds. However, every time a process has been allocated the CPU and uses its entire time quantum (does not block for I/O), 10 milliseconds is added to its time quantum, and its priority level is boosted. (The time quantum for a process can be increased to a maximum of 100 milliseconds.) When a process blocks before using its entire time quantum, its time quantum is reduced by 5 milliseconds, but its priority remains the same. What type of process (CPU-bound or I/O-bound) does the regressive round-robin scheduler favor? Explain.

Answer:

This scheduler would favor CPU-bound processes as they are rewarded with a longer time quantum as well as priority boost whenever they consume an entire time quantum. This scheduler does not penalize I/O-bound processes as they are likely to block for I/O before consuming their entire time quantum, but their priority remains the same.

e. (5 points) What is the meaning of the term busy waiting? What other kinds of waiting are there in an operating system? Can busy waiting be avoided altogether? Explain your answer.

Answer: Busy waiting means that a process is waiting for a condition to be satisfied in a tight loop without relinquish the processor. Alternatively, a process could wait by relinquishing the processor, and block on a condition and wait to be awakened at some appropriate time in the future. Busy waiting can be avoided but incurs the overhead associated with putting a process to sleep and having to wake it up when the appropriate program state is reached.

f. (5 points) Show that, if the wait() and signal() semaphore operations are not executed atomically, then mutual exclusion may be violated.

Answer: A wait operation atomically decrements the value associated with a semaphore. If two wait operations are executed on a semaphore when its value is 1, if the two operations are not performed atomically, then it is possible that both operations might proceed to decrement the semaphore value thereby violating mutual exclusion.

3） [40 points] CPU Scheduling

a. (25 points) Consider the following set of six processes, with the length of the CPU burst time given in milliseconds:

Process	Arrival Time (ms)	Burst Time (ms)

P1	0	3
P2	1	1
P3	4	6
P4	2	8
P5	12	7
P6	5	4

a) (12 points) Draw four Gantt charts that illustrate the execution of these processes using the scheduling algorithms listed below:

(i) First-Come-First-Serve (FCFS)
(ii) Shortest-Job-First (SJF)
(iii) Round-Robin (quantum = 3) (RR)

b) (6 points) What is the turnaround time of each process for each of the following scheduling algorithms?

Turnaround time	P1	P2	P3	P4	P5	P6
FCFS						
SJF						
RR						

c) (6 points) What is the waiting time of each process for each of the following scheduling algorithms?

Waiting time	P1	P2	P3	P4	P5	P6
FCFS						
SJF						
RR						

d) (1 points) Which of the algorithm results in the minimum average waiting time (over all processes) mentioned above?

Answer:

(i) FCFS

P1	P2	P4	P3	P6	P5

0 3 4 12 18 22 29

(ii) SJF

P1	P2	P3	P6	P5	P4

0 3 4 10 14 21 29

(iii) RR

P1	P2	P4	P3	P6	P4	P3	P5	P6	P4	P5

0 3 4 7 10 13 16 19 22 23 25 29

Turnaround time	P1	P2	P3	P4	P5	P6
FCFS	3	3	14	10	17	17
SJF	3	3	6	27	9	9
RR	3	3	15	23	17	18

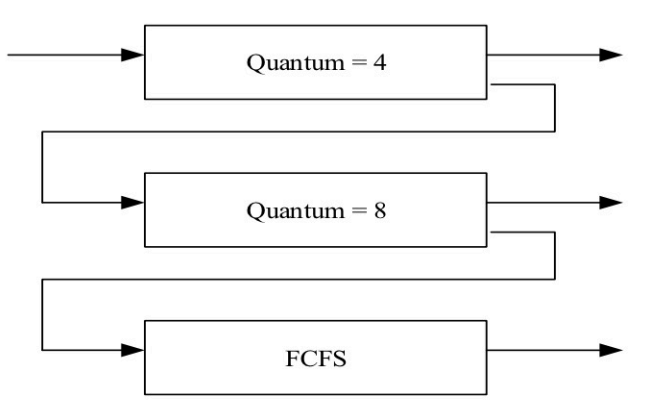
Waiting time	P1	P2	P3	P4	P5	P6
FCFS	0	2	8	2	10	13
SJF	0	2	0	19	2	5
RR	0	2	9	15	10	14

SJF has minimum average waiting time.

b. (15 Points) Given the arrival time and CPU-burst of 5 processes shown in the following diagram:

Process	Arrival Time (ms)	Burst Time (ms)
P1	0	25
P2	18	10
P3	2	42
P4	10	35
P5	21	6
P6	30	12

Suppose the OS uses a 3-level feedback queue to schedule the above 6 processes. Round-Robin scheduling strategy is used for the queue with the highest priority and the queue with the second highest priority, but the time quantum used in these two queues is different. First-come-first serve scheduling strategy is used for the queue with the lowest priority. The scheduling is **preemptive**.



Construct a Gantt chart depicting the scheduling for the set of processes specified in the above diagram using this 3-level feedback queue.

Answer:

P1	P3	P1	P4	P1	P2	P5	P1	P3	P6	P3	P4	P2	P5	P6	P1	P3	P4	
0	4	8	10	14	18	22	26	28	30	34	40	48	54	56	64	77	107	130

4）[10 points] Process Synchronization

Give a code example about how the Swap() instruction can be used to provide mutual exclusion that satisfies the bounded-waiting requirement.(Refer to Figure 6.7 Page 261 in your textbook)

Answer:

```
do {
   waiting[i] = TRUE;
   key = TRUE;
   while (waiting[i] && key)
      key = Swap(&lock, &key);
   waiting[i] = FALSE;
   /* critical section */
   j = (i+1) % n;
   while ((j != i) && !waiting[j])
      j = (j+1) % n;
   if (j == i)
      lock = FALSE;
   else
      waiting[j] = FALSE;
   /* remainder section */
} while (TRUE);
```