PA-3: C++ Scheduler Simulation

Total points: 60 – 130 pts (60 Required, but must include Task 1 Required part)

Instructor: Joseph Waclawski

Logistics

I found a simple process scheduler simulation at the following GITHub url:

https://github.com/joedodson/cpu-scheduling-sim

I downloaded the buildable source and included it as a zip file on the assignment. I created a makefile so that you can easily make changes, add new files, etc., and rebuild. You will need to make sure that your Linux Environment has g++ installed. Note that this simulation does NOT account for I/O time, meaning all processes are swapping back and forth between Ready and Running, as there is nothing to block them. The GITHub location has documentation on how to use the scheduler simulation.

Task 1 (20 points)

OPTIONAL (10 Points Extra Credit) Update the scheduler so that the PP algorithm does Round Robin preemption for processes at the same priority. Currently, the simulation only does preemption when a higher priority process arrives. Given that there are no higher priority tasks arriving, the update will do Round Robin (i.e. with preemption) of tasks as the same priority.

You can use the information determined below to validate the output of your program.

REQUIRED (20 Points): Perform a manual analysis of how the updated PP algorithm (described above) would work, given the following input. This must be done regardless of whether you successfully updated the code above.

1	0	10	1
2	1	10	1
3	2	10	1
4	3	10	1
5	4	10	1
6	5	10	0

Below is a table that shows a format that can be used to do the manual analysis. From this data, you will determine Average Response Time, Average Turnaround Time, and the number of Context Switches each process goes through. Note that since the current simulation uses a clock tick of 0.5, I added a column here that will allow you to translate between the Clock Tick, and the actual time. Please note that for this exercise you should assume that the input file arrival time is in Clock Ticks, NOT actual time. This will certainly impact your answer, so be careful. If you have already written the program above and your output differs from what you get here, consider this when determining if your program is correct.

Clock Tick	Time	Arrivals	Priority	Actual Start	Status	CS	Remaining
0	0	Process 1 Arrives	1				
1	0.5	Process 2 Arrives	1	0.5	Process 1 is Running		
2	1	Process 3 Arrives	1		Process 1 is Running	Context Switch	8
3	1.5	Process 4 Arrives	1		Process 2 is Running		
4	2	Process 5 Arrives	1		Process 2 is Running	Context Switch	8
5	2.5	Process 6 Arrives	0		Process 3 is Running		
6	3						
7	3.5						

You will fill in the following table with the information determined from the worksheet above:

	P1	P2	P3	P4	P5	P6
Turnaround Time						
Response Time						
Context Switches						
Average Turnaround						
Average Response						

In the next section, you have two options: you can either complete Task 2 **OR** Task 3. Since Task 3 is more challenging than Task 2, those who choose to complete Task 3 will earn an additional 10 bonus points.

Task 2 (40 points)

The scheduler currently allows for the following algorithms.

Algorithm Numbers

0 - FCFS, First Come First Serve (FIFO)

1 - SRTF, Shortest Remaining Time First (preemptive)

2 - RR, Round Robin (must enter time quantum to execute)

3 - PP, Preemptive Priority (uses provided priorities in input file, same as RR otherwise)

In this task, you will add your own algorithm, as follows:

4 – PR – Preemptive Random - Randomly select a process in the Ready queue to run next, ignoring priority. You must take into account the time quantum parameter.

As part of this task you are required to provide a 1-2 page discussion (single spaced, 12 font) on the performance of each of the above algorithms as observed by you. You may NOT simply find something on the web and copy it. I expect to see the following:

1. The process information used as input (which will be the same for all algorithms)
2. Several (at least 5) runs of the scheduler with each of the above algorithms
3. Provide output showing processes status. You may need to add additional output to demonstrate your final product.
4. A discussion of your observations (e.g. discussion of average Turnaround Time and Response Time)

NOTE: If you have not finished the [now optional] program for Task 1 above, you should use the output of the manual analysis.

Task 3 (60 points)

You must do Task 1 first. Then you are to update the PP scheduling simulation to account for process blockage (i.e. Block queue) due to I/O. You must add a column to the input file for total
I/O Burst time needed. If a process never needs I/O, then the I/O Burst for that process would be 0. For ease of implementation, the following approach is suggested:

1. All processes go into the Ready Queue at their arrival time.
2. Processes that have a non-zero IO Burst time, upon entering the running queue for the first time, will be placed into the Blocked queue after 50% of their FIRST CPU time quantum has been used. This will simulate a process doing some work, then needing to do I/O so it is blocked.
3. Upon completion of a process' I/O Burst time in the Blocked queue, it will be moved back to the Ready queue. At this time, the process will continue to be scheduled until it's CPU Bust is complete (that is, continue to get time quantums, based upon its priority, until the remainder of its CPU Burst is exhausted).
4. To simplify odd cases, it is highly suggested that processes with non-zero I/O Burst time should have a CPU Burst time that is twice the time quantum.
5. Provide output showing processes status. You may need to add additional output to demonstrate your final product.

When all changes are made and tested, tar/zip the directory tree and submit it to the Blackboard.