Operating Systems

Home Assignment IV

Problem I (Thread Scheduling with Pthreads, 30 Pts)

The file ***pth.c*** implements the thread creation and scheduling within Linux by using the Pthreads library. You are required to:

1. Understand the codes, and fill the comments from (1) to (11) except (6) to (9).
2. Complete the codes for (6), (7), and (8). You need to implement one scheduling policy in each of the three slots.
3. Complete the codes for (9). When SCHED_FIFO and SCHED_RR are used as the scheduling policies, you can assign different scheduling priorities for each thread you create. For example, you can assign priority 1 to thread 0, 2 for thread 1, and so and so forth.

In order to compile the program, you can use: **gcc –o pth pth.c –lpthread**

At the beginning of the program, you can use #define to select which scheduling policy you are going to use for the whole application.

Comment two #define statements and keep only one #define statement at one time for one compilation.

Also, you can define how many threads you want to create by changing the number after NUM_THRADS. Every time you change the codes, you have to re-compile the codes to make changes effective.

You are required to create 4, 8, and 16 threads. For each number of thread, you need to try every three scheduling policies. In other words, there are totally 9 scenarios and you need to run the application for 9 times.

Document the results (what the program outputs) and explain your observations. Try to use ./pth to run the program. If this does not work, try sudo ./pth and explain the reason.

<u>The following resource may help you with this homework:</u>

http://man7.org/linux/man-pages/man3/pthread_attr_setschedpolicy.3.html

http://man7.org/linux/man-pages/man3/pthread_attr_setschedparam.3.html

In Linux, you can also use chrt –m to check available scheduling policies and range of priorities in your current OS.

Problem 2 (Scheduling, optimization 15 pts)

Discuss how the following pairs of scheduling criteria conflict in certain settings.

1. CPU utilization and response time **(5 pt)**
2. Average turnaround time and maximum waiting time **(5 pt)**
3. I/O device utilization and CPU utilization **(5 pt)**

Problem 3 (Scheduling, computation 30 pts)

Consider the following set of processes, with the length of the CPU-burst time given in milliseconds:

Process	Burst Time	Priority
P1	10	3
P2	1	1
P3	2	3
P4	1	4
P5	5	2

The processes are assumed to have arrived in the order P1, P2, P3, P4, P5, all at time 0.

1- Draw four Gantt charts illustrating the execution of these processes using FCFS, SJF, a non preemptive priority (a smaller priority number implies a higher priority), and RR (quantum = 1) scheduling. **(12 Pt)**

2- What is the turnaround time of each process for each of the scheduling algorithms in part 1? **(8 Pt)**

3- What is the waiting time of each process for each of the scheduling algorithms in part 1? **(4 Pt)**

4- Which of the schedules in part a results in the minimal average waiting time (over all processes)? **(4 Pt)**

Problem 4 (Scheduling, Variable priorities 15 pts)

Consider a preemptive priority scheduling algorithm based on dynamically changing priorities. Larger priority numbers imply higher priority.

When a process is waiting for the CPU (in the ready queue, but not running), its priority changes at a rate α; when it is running, its priority changes at a rate β. All processes are given a priority of

0 when they enter the ready queue. The parameters α and β can be set to give many different scheduling algorithms.

1. What is the algorithm that results from $\beta > \alpha > 0$? (7 Pt)

2. What is the algorithm that results from $\alpha < \beta < 0$? (7 Pt)