Akshita Mittel
CS3030
10 October 2015

Linux Scheduling

Operating System: Assignment 7

Test case 1: *k = 2*

Process 1: Scheduler (FIFO) and Priority HIGHEST

Process 2: Scheduler (FIFO) and Priority LOWEST

Expected Result: The process that occupies the processor first will complete execution before the next process executes. No preemption takes place in a SCHED_FIFO.

Test case 2: *k = 2*

Process 1: Scheduler (FIFO) and Priority HIGHEST

Process 2: Scheduler (FIFO) and Priority HIGHEST

Expected Result: Like the first case, the process that occupies the processor first will complete execution before the next process executes. FIFO is non-premptive and has no context switching. The only way that preemption is possible is that if a higher priority process is waiting in queue; else the current process will continue to run till it decides to yield the processor. In this case both processors had the same priority.

Test case 3: *k = 2*

Process 1: Scheduler (RR) and Priority HIGHEST

Process 2: Scheduler (RR) and Priority LOWEST

Expected Result: SCHED_RR is an POSIX algorithm based on preemption, in which a specific time slice is given to each process in order to let the process complete some of its computations. The time slice in RR depends on the quantum. In my laptop, no preemption took place, maybe due to the large quantum.

Test case 4: *k = 2*

Process 1: Scheduler (RR) and Priority HIGHEST

Process 2: Scheduler (RR) and Priority HIGHEST

Expected Result: like test case 3, the processes completed before their time quantum got over.

Test case 5: *k = 3*

Process 1: Scheduler (SCHED_OTHER) and (static) Priority 0

Process 2: Scheduler (FIFO) and Priority LOWEST

Process 3: Scheduler (RR) and Priority LOWEST

Expected Result: SCHED_OTHER is another variation of the round robin time scheduling algorithm. However, it does not have a fixed time quantum, rather it checks the priority of other processes and sets its quantum accordingly. Unlike SCHED_FIFO and SCHED_RR, SCHED_OTHER is not a real time policy and hence is subjected to preemption by the real time policies. In this example the processes were executed in the order 2, 3, 1. Since both FIFO and RR had the same priority, RR was executed after FIFO.

Test case 6: *k = 3*

Process 1: Scheduler (SCHED_OTHER) and Priority 0

Process 2: Scheduler (RR) and Priority LOWEST

Process 3: Scheduler (RR) and Priority HIGHEST

Expected Result: As explained in the previous example, the processes are preempted in the following order. First process 3 (since it is real time with highest priority), then process 2 and then process 1 Since process 1 is not real time.