Thread Scheduling

Name - Kunal Sharma
Roll Number - 2021331
Branch - CSD

Code Description

I initially initialized the three thread IDs pid1, pid2, and pid3 once starting with the main function. Then I used pthread_create() to create three new threads, each of which accepts a thread ID and a function (CountA, CountB, and CountC, respectively), each of which does the same counting operation from 1 to 2^32. I use the clock_gettime() function in these routines to determine how long it takes the function to complete its duty by using the attribute CLOCK_REALTIME.

I'm then using pthread_setschedparam(), which accepts the thread ID of the current thread and the policy (SCHED_OTHER for CountA, SCHED_RR for CountB, and SCHED_FIFO for CountC) as an attribute, to set the priority of the presently executing thread. Last but not least, I'm using pthread_join(), which waits for each thread to finish before executing context switching between various threads according to their priority, with the exception of the thread with policy SCHED_OTHER. I have successfully created 6 graphs by altering the priorities of the threads.

The output of the code shows a tendency for SCHED_FIFO to execute in the shortest amount of time and SCHED_RR to execute in a little bit more time because SCHED_RR performs poorly when the same task is being performed by these two threads (given that SCHED_RR and SCHED_FIFO have the same priority), while SCHED_OTHER executes much more slowly than SCHED_RR because it has the lowest priority by default and is executed last.