Operating Systems – Monsoon 2022

1 Linux threads and their scheduling

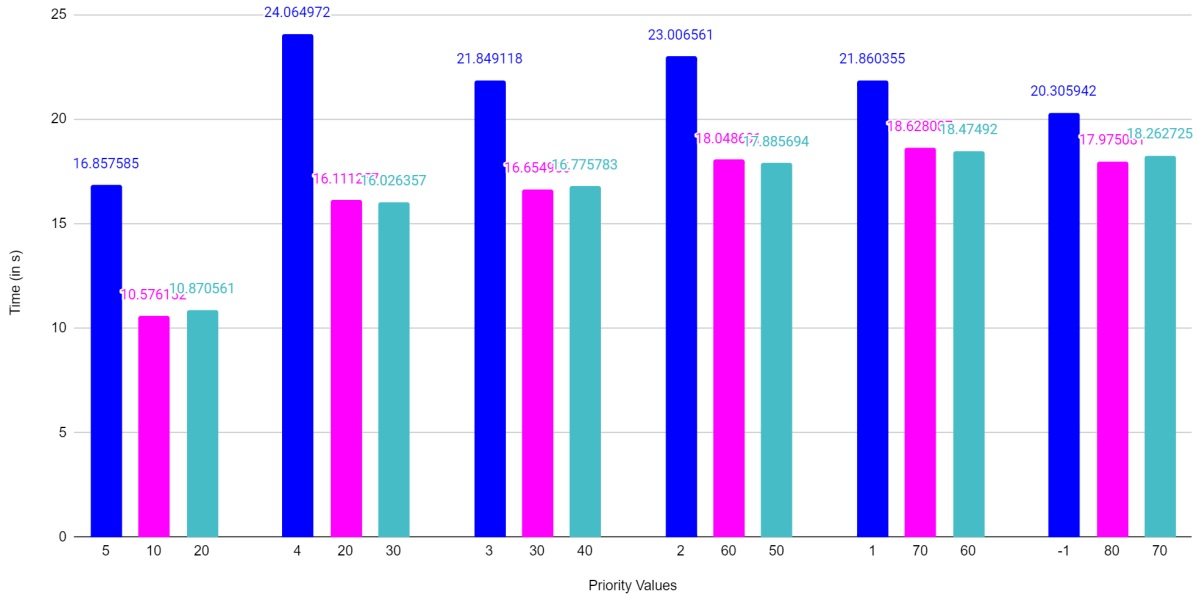
1.1 Thread scheduling

- To implement the Thread scheduling, first, initialize **3 thread variables using `pthread_t`** and then create **3 threads using `pthread_create()`** and assign them the function to be executed i.e. count functions to these threads.

- According to the question, these **3 threads(Thread A, Thread B, Thread C)**, will rely on **three different count functions (countA(), countB() and countC()) respectively.**

- Each count function does the same thing, i.e. counts from 1 – 2^32, using **a for loop.**

- We set the priority of these 3 threads by using sched_param and assign them their priority using `.sched_priority` , then these 3 threads are assigned the scheduling policy to be executed using `pthread_setschedparam()`. The scheduling policy assigned to each thread is the following:
    1. Thread 1 (call it Thr-A()): Uses SCHED_OTHER scheduling discipline with standard priority (nice:0).priority range(-19-20)
    2. Thread 2 (call it Thr-B()): Uses SCHED_RR scheduling discipline with default priority.priority range(1-90)
    3. Thread 3 (call it Thr-C()): Uses SCHED_FIFO scheduling discipline with default priority.priority range(1-90)
- Each of these threads time the process of counting from 1 – 2^32 using `clock_gettime()` in each function the `timespec structure` is used to get the starting and ending time.
- The time(in sec) is calculated using mathematical logic and printed as an output.
- Each of the threads are joined at the end of the main function to ensure the completion of each thread.
- The generated histograms show which scheduler completes the task when depending upon the scheduling policy. On the graph, X-axis has the thread priority(ThreadA-SCHED_OTHER →BLUE, ThreadB-SCHED_RR → PINK, ThreadC- SCHED_FIFO →GREEN) and the Y-axis has the process time.



1.2 Process scheduling

- We create three processes using fork() system calls and use three variables to store their process id.
- Before forking a parent process we use the `clock_gettime()` function to start counting the start time of the process.
- We use nested if-else to run the three processes simultaneously, using **execl** command, we call 3 bash script to run the commands for compiling 3 different kernels.
- We then use a for loop to get the process id of the process which is completed first and print their time according to the if condition they satisfy.