Enhancing xv6 OS

To Run

make clean
make qemu SCHEDULER=<SCHEDULER> (The input 'SCHEDULER' is the scheduling algorithm we would like to use to run the processes. The options are listed below)

System Calls

waitx - The waitx system call waits for a process to terminate and gives the wait time and run time of that process

ps - The ps system call gives information about the currently running processes, such as its pid, priority value, state, run time, wait time, number of times it was picked up by the scheduler (n_run), etc...

set_priority - The set_priority system call is used to change the priority of a process. This is required only in Priority Based Scheduling (PBS). It takes 2 inputs, the pid of the process that we would like to change the priority of and the new priority that we would like to give to said process

User Programs

time - "time <command>" will run the inputted command and display the number of clock ticks that the relevant process waited for and the number of clock ticks that it ran for. It uses the waitx system call to implement this

ps - the ps user program displays information related to the current processes such as its pid, priority value, state, run time, wait time, etc... It uses the ps system call to implement this

setPriority - the setPriority user program changes the priority of a given process. It takes the pid of the process, whose priority is required to be changed and the new priority of said process as inputs and it implements this change. It uses the set_priority system call to implement this

Scheduling Algorithms

The Following Scheduling Algorithms have been implemented (RR is the original scheduling algorithm used, the other 3 have been added):

-**Round-Robin (RR)** - preemptive scheduling algorithm that executes each process for a given time period. After the completion of that time period, it is preempted and other processes are allowed to execute for the same time period

-**First Come First Serve (FCFS)** - non-preemptive scheduling algorithm that selects the process with the lowest creation time and runs it to completion

-**Priority Based Scheduling (PBS)** - selects the process with the highest priority. Implements Round-Robin in case multiple processes have the same priority. By default, the priority of each process has been set to 60

-**Multi-Level Feedback Queue (MLFQ)** - allows processes to move between various priority queues (in this case 5), based on how much CPU time they take to execute and CPU bursts

Reason for Scheduling Getting Exploited

If a process calls a large number of CPU cycles and then calls a few I/O calls before exceeding its time slice, then it will stay in a queue in a high priority and still be able to continue to call a large number of CPU cycles

Performance Comparisons Between Scheduling Algorithms

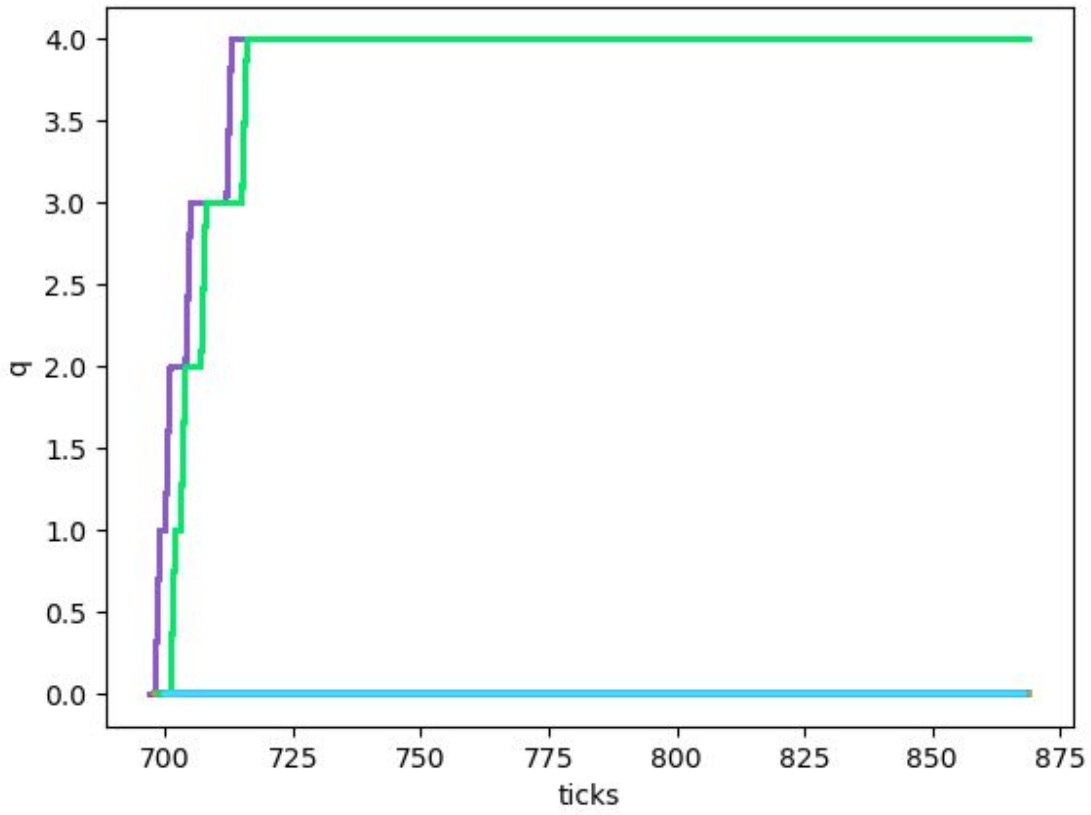
On running 'benchmark', we get the total number of ticks taken to run each scheduling algorithm for 2 separate runnings as,

RR - 8941, 10326
FCFS - 12755, 11175
PBS - 5763, 4724
MLFQ - 7481, 8304

Bonus

The Bonus has been implemented as well
Here there are 2 different graphs, both with 4 processes

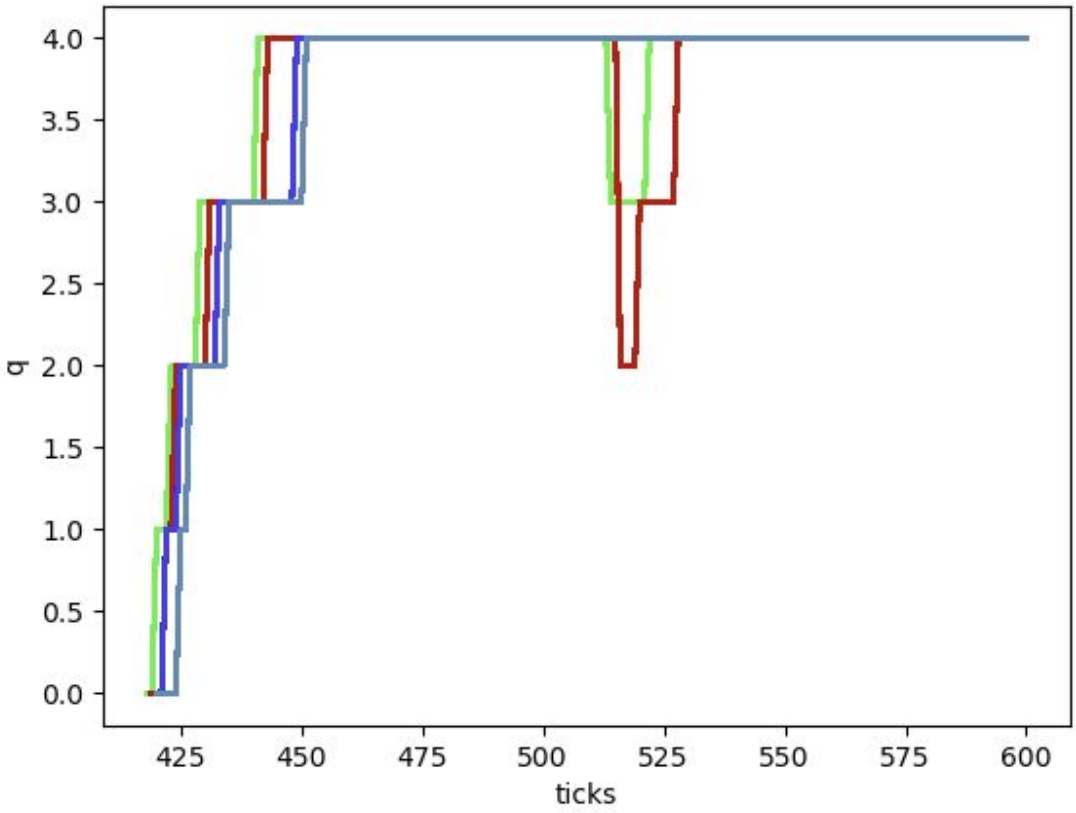
In the first graph, 2 of the processes are CPU bound processes and 2 are I/O bound processes. The 2 processes that are CPU bound quickly climb to the 5th queue and are executed there while the I/O bound processes stay at the first queue as they are sleeping for a large portion of time.



Graph1

We can show that the 2 processes that climb are CPU bound while the 2 that remain with the 1st queue are I/O bound with the second graph. In the second graph, all 4 processes

are CPU bound and we can see that all the processes climb quickly to the 5th queue and are executed there. Hence the 2 processes that climbed in the first graph are the CPU bound processes while the other 2 processes that stayed at the first queue are I/O bound



Graph 2