

Operating Systems Project

Author: Papadopoulos Charalampos

Topic: Modifying the scheduler of the XV6 operating system

Description

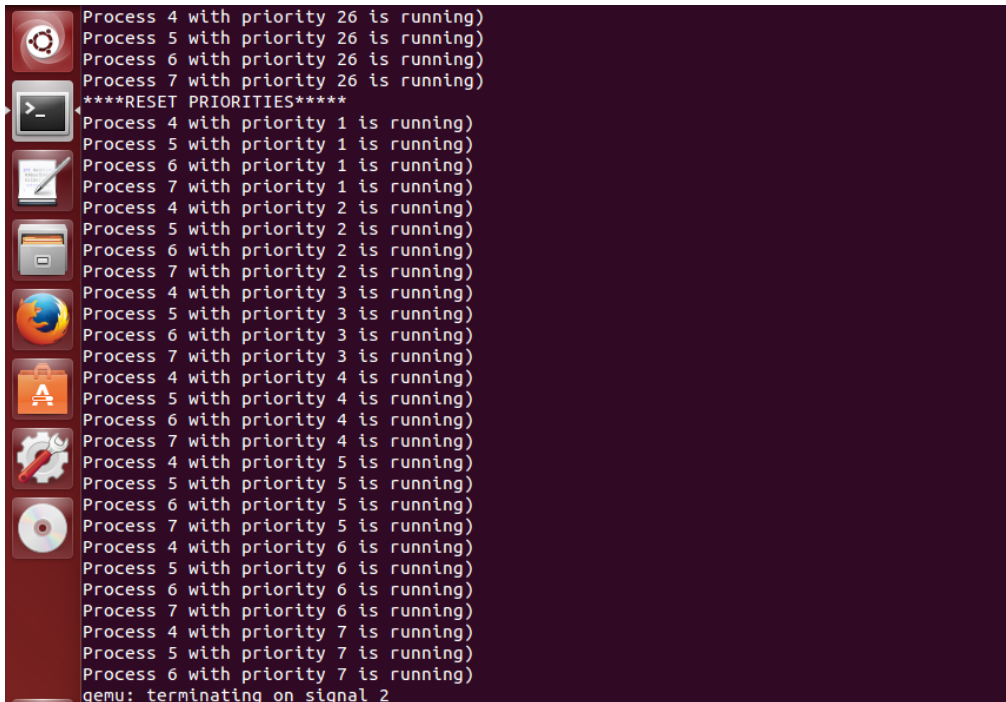
The xv6 OS scheduler, normally runs through the table that contains all the processes and runs the first process that it finds that is runnable.

I changed that policy, implementing a priority based scheduler. I added a field `proc->priority` to each process. The priority of each process is initialized to one . It is increased by one each time that the scheduler function runs, by running my function `"increasepriority()"` inside `proc.c` . Each time that the `scheduler()` function is entered, we run through the table with the available processes, find the smallest priority. Then, the first process with the smallest priority runs. If the priority reaches the maximum level of 200 then it is reset automatically. Also, every 500 clock ticks, all the priorities are reset to 1.

This policy implements the MLFQ scheduling policy, with 200 possible queues. This functionality is implemented inside the `proc.c` file, in the scheduler function.

In order to run my code from my tar file, you need to connect to the server, untar it, do `ssh -p 27 mu` and then do `make -CPUS=1 QEMU`. Immediately when it starts running, you must enter at the command prompt the command `test`. This command spawns 4 children, in order to demonstrate the correct functionality.

You can see a demo of this functionality below:



```
Process 4 with priority 26 is running)
Process 5 with priority 26 is running)
Process 6 with priority 26 is running)
Process 7 with priority 26 is running)
****RESET PRIORITIES****
Process 4 with priority 1 is running)
Process 5 with priority 1 is running)
Process 6 with priority 1 is running)
Process 7 with priority 1 is running)
Process 4 with priority 2 is running)
Process 5 with priority 2 is running)
Process 6 with priority 2 is running)
Process 7 with priority 2 is running)
Process 4 with priority 3 is running)
Process 5 with priority 3 is running)
Process 6 with priority 3 is running)
Process 7 with priority 3 is running)
Process 4 with priority 4 is running)
Process 5 with priority 4 is running)
Process 6 with priority 4 is running)
Process 7 with priority 4 is running)
Process 4 with priority 5 is running)
Process 5 with priority 5 is running)
Process 6 with priority 5 is running)
Process 7 with priority 5 is running)
Process 4 with priority 6 is running)
Process 5 with priority 6 is running)
Process 6 with priority 6 is running)
Process 7 with priority 6 is running)
Process 4 with priority 7 is running)
Process 5 with priority 7 is running)
Process 6 with priority 7 is running)
qemu: terminating on signal 2
```

Additional Points

For additional points, I implemented the nice system call, which I call inside the test.c file. There, at the 3rd child – which corresponds to pid=6, I periodically do nice(100), so I set its priority is set to 100. For some reason, this only runs only before the start of the massive printing of the priority messages. I include a demo below. You can see here that process 6 is not scheduled before the RESET priorities, so the nice system call works correctly. For some reason, when the massive printing starts, the nice system call is not working correctly.

```
neu@ubuntu: ~
****RESET PRIORITIES****
NICE SYSTEM CALL!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
****RESET PRIORITIES****
NICE SYSTEM CALL!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Process 4 with priority 31 is running)
Process 5 with priority 31 is running)
Process 7 with priority 31 is running)
Process 4 with priority 32 is running)
Process 5 with priority 32 is running)
Process 7 with priority 32 is running)
Process 4 with priority 33 is running)
Process 5 with priority 33 is running)
Process 7 with priority 33 is running)
Process 4 with priority 34 is running)
Process 5 with priority 34 is running)
Process 7 with priority 34 is running)
****RESET PRIORITIES****
Process 4 with priority 1 is running)
Process 5 with priority 1 is running)
Process 6 with priority 1 is running)
Process 7 with priority 1 is running)
Process 4 with priority 2 is running)
Process 5 with priority 2 is running)
Process 6 with priority 2 is running)
Process 7 with priority 2 is running)
Process 4 with priority 3 is running)
Process 5 with priority 3 is running)
Process 6 with priority 3 is running)
```