CS 481: Operating Systems
Lab 2 Part 1

Jacob Hurst

April 11, 2019

1 Hypothesis

Linux's Completely Fair Scheduler (CFS) will prioritize IO bound processes over CPU bound processes by often preempting CPU bound tasks to run IO bound tasks. As a result, the count of schedule-ins by the scheduler will be greater for IO bound processes, while the IO bound processes will have a lesser minimum, maximum, and average run-times.

2 Methods

Python program 'lab2.py' drives 3 smaller C programs, 'cpubound.c', 'iobound.c', and 'mixed-bound.c'.

The python driver works by first cleaning the directory and then compiling, running, and tracing a total of n runs of m seconds of the C programs, where n and m are provided via command-line. Each of the C programs run simultaneously and share a single core of the machines CPU. 'Perf' is used to trace and view the scheduling information of each of the programs.

The CPU bound C code works by calculating fibonacci sequence for m seconds and then exiting. The IO bound C code works by alternating between sleeping for 5ms and calculating fibonacci sequence for 5ms. The Mixed bound C code works similarly to the IO bound C code, however, it only sleeps for 1ms.

```
ssg@thelios:~/Desktop/cs481$ python3 lab2.py 1 15
Initializing experiment, this should take 0.25 minutes...
Current time:  22:21:06
Starting CPU bound process. PID=14345.
Stopping CPU bound process.
Starting IO bound process. PID=14343.
Stopping IO bound process.
Starting mixed bound process. PID=14346.
Stopping mixed bound process.
Treatment 1 of 1 complete.
Experiment concluded.
Current time:  22:21:21
Run 'sudo perf sched timehist -s --pid=14345,14343,14346' to view results.
ssg@thelios:~/Desktop/cs481$ [ perf record: Woken up 840 times to write data ]
[ perf record: Captured and wrote 3014.845 MB perf.data (27650223 samples) ]

ssg@thelios:~/Desktop/cs481$ sudo perf sched timehist -s --pid=14345,14343,14346
Samples do not have callchains.

Runtime summary
                        comm  parent   sched-in     run-time    min-run     avg-run     max-run
                                          (count)       (msec)     (msec)      (msec)      (msec)
---------------------------------------------------------------------------------------------------
<no still running tasks>

Terminated tasks:
                :14343[14343]      -1       1214     4960.393      0.050       4.085       5.025
                :14345[14345]      -1       1003     4958.403      0.025       4.943       7.983
                :14346[14346]      -1       1759     4966.678      0.000       2.823       5.018
```

Figure 1: A single sample run of 15s. Note: all processes start at the same time and run simultaneously (as verified by htop). However, due to how pythons subprocess module handles output, it appears as if they are occurring one after another - this is not the case.

[1]

2.1 Factors

1. Sleep duration: The portion of sample duration the C code spends doing any sleeping before switching to a computation, measured in ms.
2. Compute duration: The portion of sample duration the C code spends doing a computation before switching to sleep, measured in ms.
3. Sample Count: The number of runs completed for the 3 simultaneous processes, 30 for this experiment.
4. Sample Duration: The length of time each process is allotted to run, 30s for this experiment.

2.2 Response

Per Process, measured over the duration of the run.

1. Schedule-in count: How often the process was scheduled in by the scheduler to run next.
2. Average run-time: The average duration of time a process was able to run.
3. Minimum run-time: The smallest duration of time a process was able to run.
4. Maximum run-time: The largest duration of time a process was able to run.

[1] See code samples for implementation specifics.

2.3 Treatments

Each treatment ran each C program for 30s simultaneously and traced the scheduling information via perf. Sleep durations: In cpubound, this was the none of the duration. In iobound, this was 5ms. In mixedbound, this was 1ms. Compute durations: In cpubound, this was the full duration. In iobound and mixedbound, this was 5ms.

2.4 Repetitions

A total of 30 repetitions of the above treatment were ran in order to ensure reproducible results.

3 Results

Below are figures generated from averaging the 30 samples on each process. For more information, see 'results' text file.

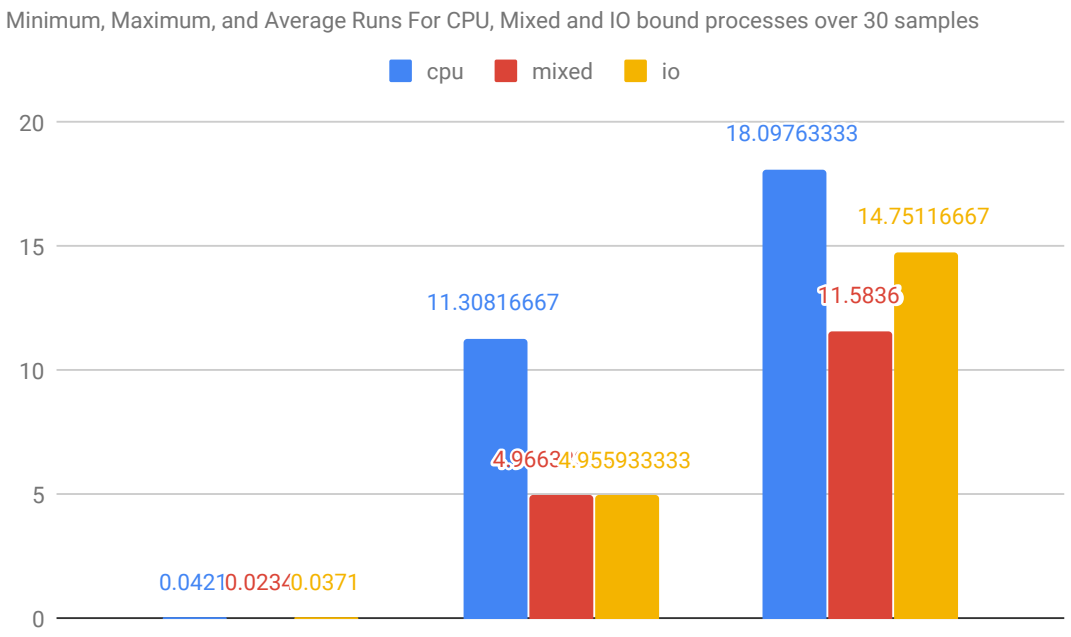

Figure 2: Average minimum, average, and maximum samples are lesser for IO and mixed than for CPU.

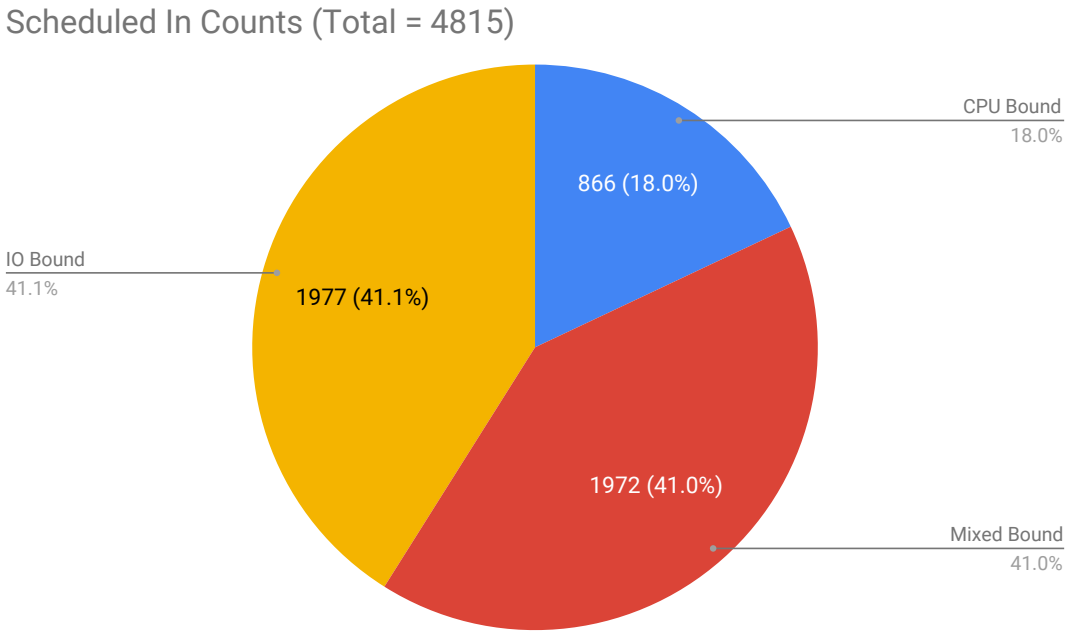

Figure 3: Schedule in counts for IO and mixed are greater than for CPU.

As the above figures confirm, CFS will prioritize IO bound processes over CPU bound processes by often preempting CPU bound tasks to run IO bound tasks.

This is seen by the count of schedule-ins by the scheduler for IO/Mixed bound processes versus that of CPU bound processes. The count of schedule-ins will be greater for IO/Mixed bound processes. The IO/Mixed bound processes will have shorter minimum, average, and maximum run-times as they will not ask for as much of the CPU for calculations.