**Operation Systems Lab 3 (210010032)**

**Part 1:**

Modified file schedule.c in usr/src/minix/servers/sched by inserting print statement inside schedule\_process function. As follows.

if (rmp->priority >= USER\_Q){                      //

        printf("Minix 210010032 : PID %d swapped in\n", \_ENDPOINT\_P(rmp->endpoint));

    }

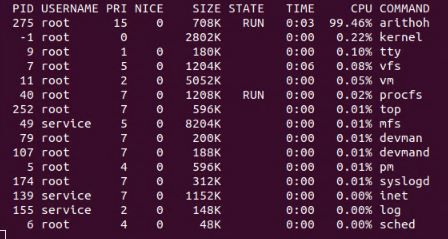
The condition rmp->priority >= USER\_Q is written to ensure that we print only user level processes when it is scheduled.

**Part 2:**

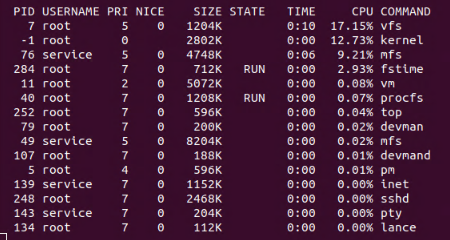
The task is to run several workmixes of different benchmarks like aritho, fstime, syscall, spawn and pipe.

fstime and pipe seem to be I/O bound. Aritho, syscall and spawn are CPU bound.

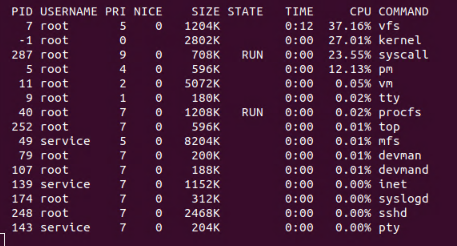
1. CPU usage of aritho



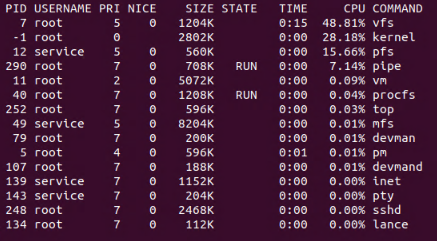
1. CPU usage of fstime



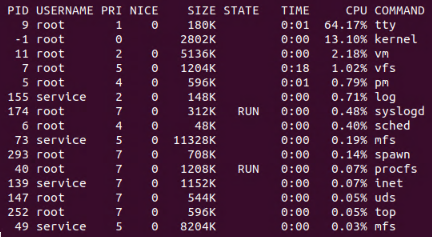
1. CPU usage of syscall



1. CPU usage of pipe



1. CPU usage of spawn



1. workload\_mix1.sh

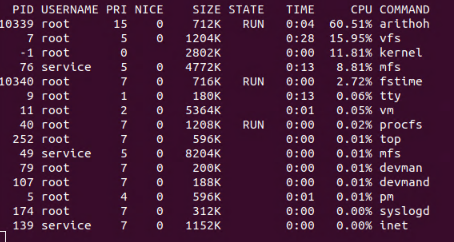
#!/bin/sh

./arithoh.sh &

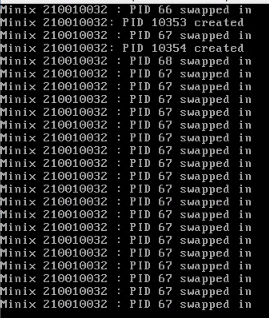
./fstime.sh &

wait

CPU usage:



Here process 68 is that of fstime, 67 if of aritho. 67 runs many times since it is CPU intensive while process with PID 68 (fstime) waits until it gets input, since it is I/O bound process.



1. workload\_mix2

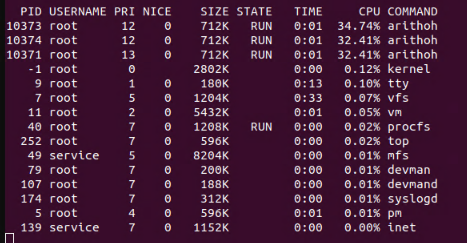
#!/bin/sh

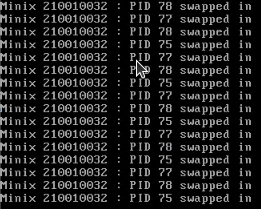
./arithoh.sh &

./arithoh.sh &

./arithoh.sh &

Wait

CPU usage:  




These 3 processes with PIDs 75,77 and 78 are all three different aritho processes. This is highly CPU intensive and interval between swaps is very less, since other process enters queue after current process spends considerable amount of time in queue.

1. workload\_mix3

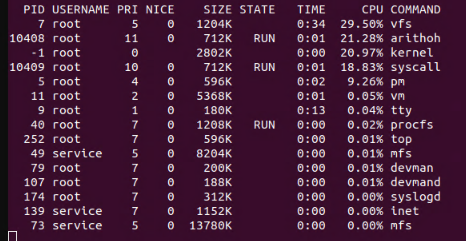
#!/bin/sh

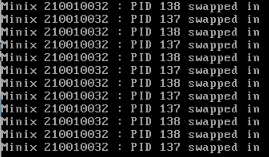
./arithoh.sh &

./syscall.sh &

wait

CPU usage:





Syscall is also CPU bound process like aritho. Hence both are in queue one after the other. But it is to be noted that syscall is less CPU intensive than aritho. Hence aritho is scheduled towards the end.