CS 304: Operating Systems Lab

170010025, S V Praveen 170010039, Anudeep Tubati

We study how the standard Minix3 scheduler schedules our workloads and presents our observations and inferences. We further implement Pseudo-FIFO policy for the MINIX3 scheduler and make observations and inferences,

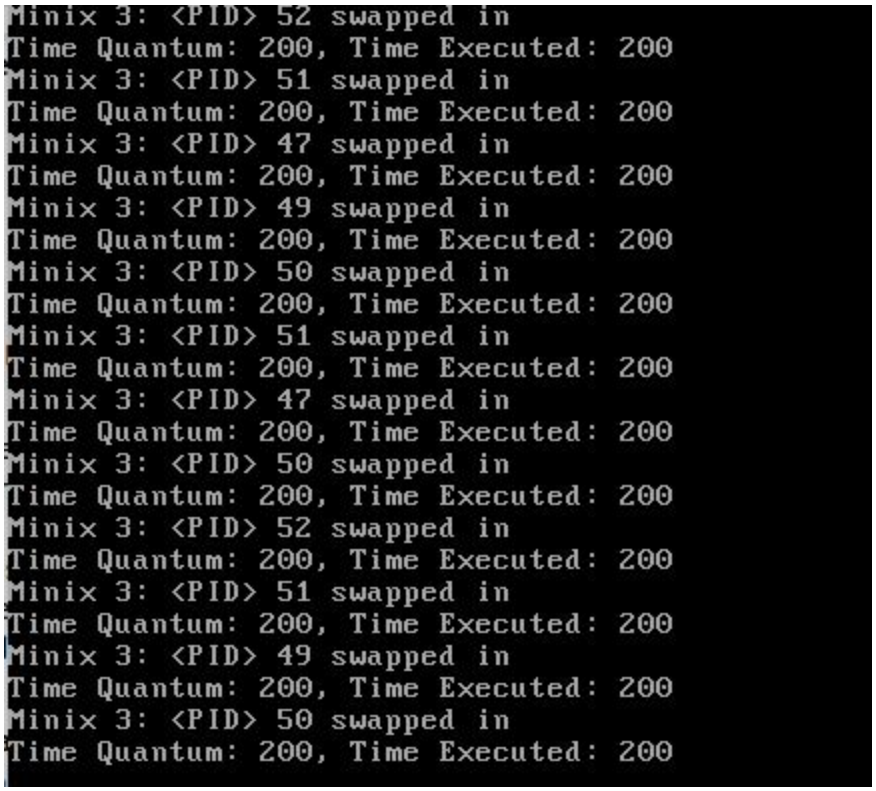
Part 1 (Standard Scheduler)

★ Workload Mix 1

5 Parallel executions of ./arithoh.sh

```
./arithoh.sh & ./arithoh.sh & ./arithoh.sh & ./arithoh.sh & ./arithoh.sh
```

- **Observations:** We see that the 5 processes execute parallelly in round-robin fashion. The time quanta which is 200 by default is fully used every time the process is scheduled.
- **Inference:** CPU fairly schedules all the processes in a round-robin fashion. CPU intensive processes do not need to wait for I/O and they execute fully in the time quanta they are allocated.

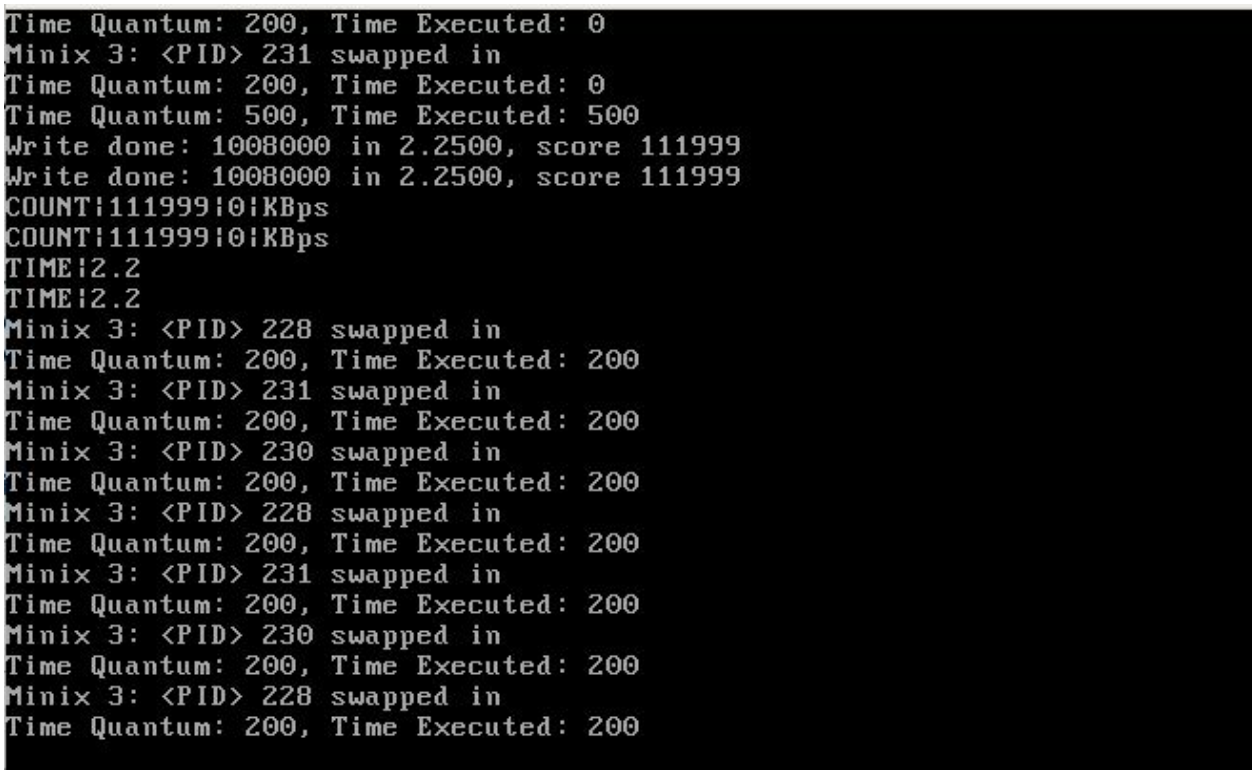


★ Workload Mix 2

3 Parallel executions of ./arithoh.sh and 2 Parallel executions of ./fstime.sh

```
./arithoh.sh & ./fstime.sh & ./arithoh.sh & ./fstime.sh & ./arithoh.sh
```

- **Observations & Inference:** We see that the 3 CPU intensive processes and 2 I/O bound processes execute parallelly in round-robin fashion. However, the other 2 I/O bound processes wait for their I/O work and then are scheduled to work on the CPU. As seen in the screenshot, CPU bound tasks utilize all of their quanta whereas I/O bound tasks don't do the same.



★ Workload Mix 3

4 Parallel executions of ./pipe.sh and 1 Parallel executions of ./fstime.sh

```
./pipe.sh & ./pipe.sh & ./fstime.sh & ./pipe.sh & ./pipe.sh
```

- **Observations & Inference:** As pipe.sh involves IPC, it invokes system calls and hence gets a higher quantum (500 ms for system calls). All the 5 processes execute in a round-robin fashion and fstime.sh uses CPU rarely as it is I/O bound.

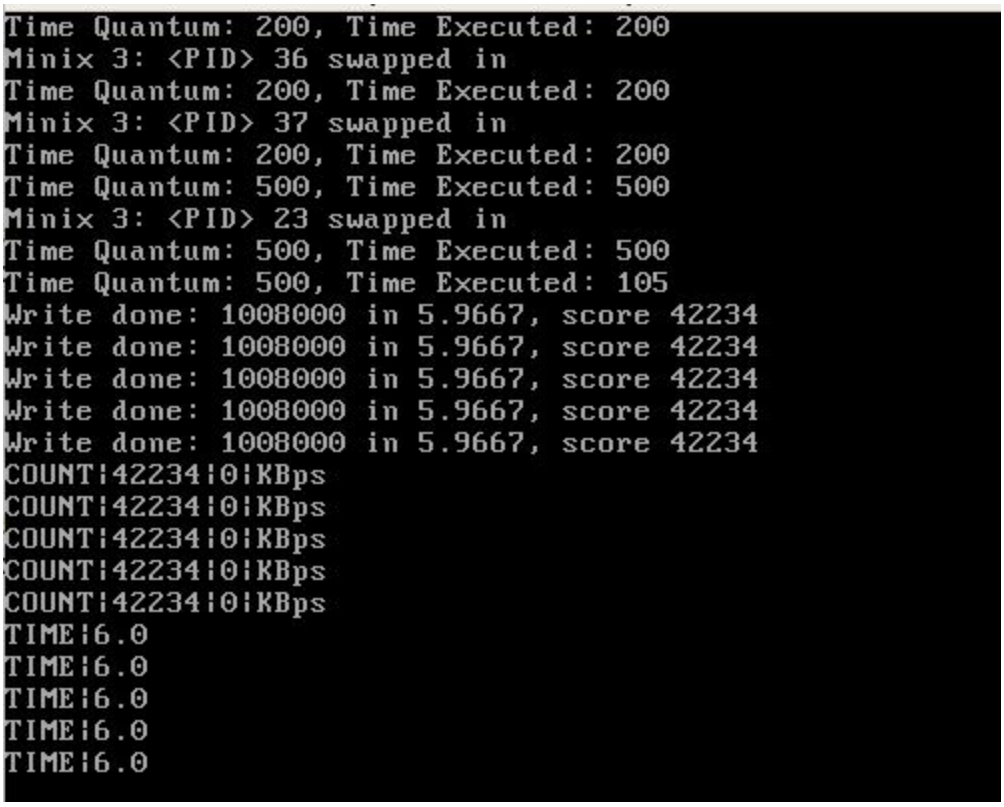
★ Workload Mix 4

5 Parallel executions of ./fstime.sh

```
./fstime.sh & ./fstime.sh & ./fstime.sh & ./fstime.sh & ./fstime.sh
```

- **Observations & Inference:** All the 5 processes execute in a round-robin fashion. Also, as modifying files is a system call, it has a higher quantum. The

tasks don’t utilise their quanta to the fullest as they aren’t CPU intensive.



Part 2 (Modified pseudo-FIFO Scheduler)

★ Workload Mix 1

5 Parallel executions of ./arithoh.sh

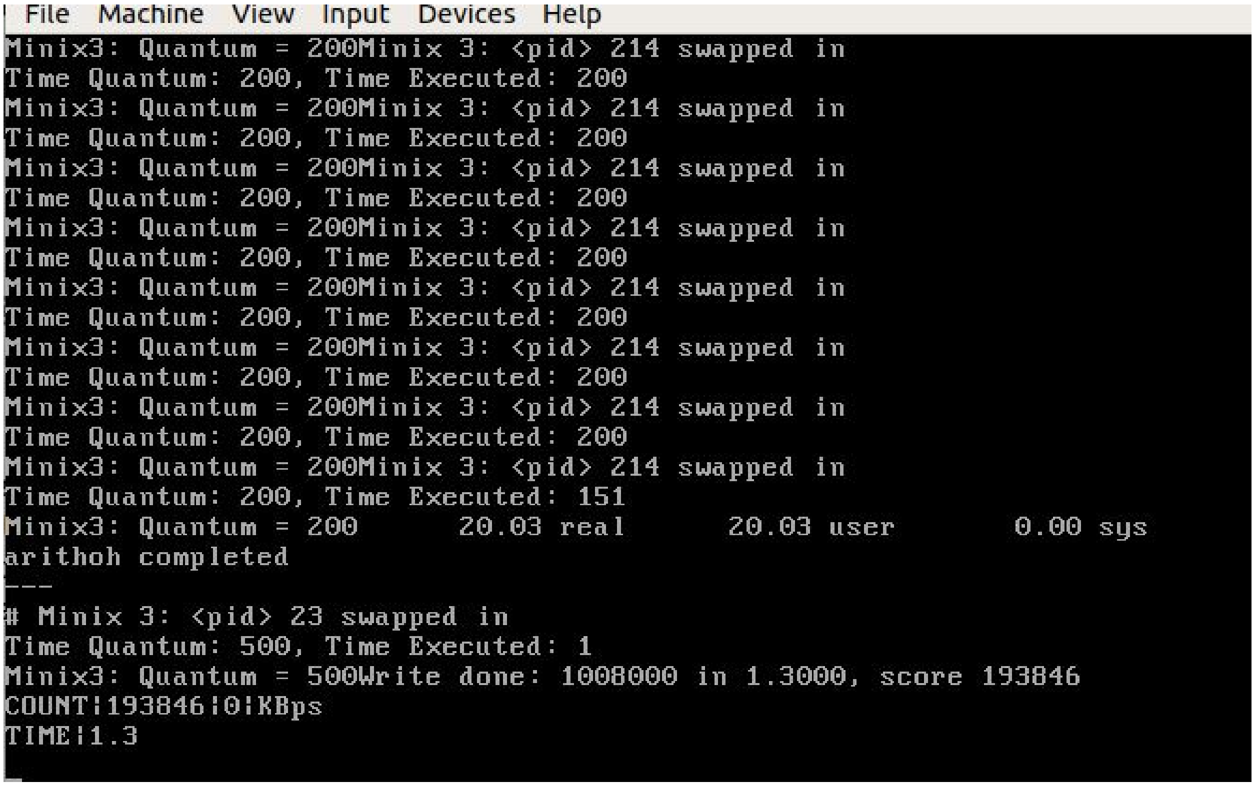
```
./arithoh.sh & ./arithoh.sh & ./arithoh.sh & ./arithoh.sh & ./arithoh.sh
```

- **Observations:** We see that the 5 processes execute one after the other in first come first serve fashion. The time quanta which is 200 by default is fully used every time the process is scheduled. Occasionally, we see a process allocated 500 ms quanta given zero execution time. This is most probably the execution of a system call.
- **Inference:** CPU fairly schedules all the processes on a first come first serve basis. CPU intensive processes do not need to wait for I/O and they execute fully in the time quanta they are allocated.

★ Workload Mix 2

3 Parallel executions of ./arithoh.sh and 2 Parallel executions of ./fstime.sh

```
./arithoh.sh & ./fstime.sh & ./arithoh.sh & ./fstime.sh & ./arithoh.sh
```



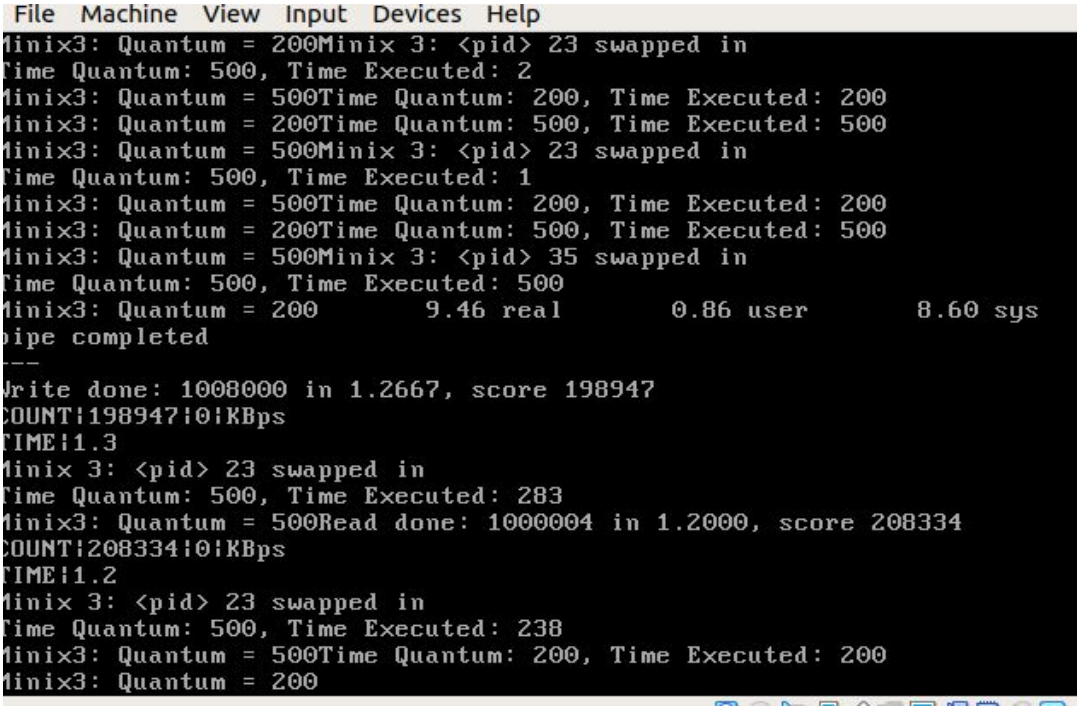

```
Minix3: Quantum = 200Minix 3: <pid> 214 swapped in
Time Quantum: 200, Time Executed: 200
Minix3: Quantum = 200Minix 3: <pid> 214 swapped in
Time Quantum: 200, Time Executed: 200
Minix3: Quantum = 200Minix 3: <pid> 214 swapped in
Time Quantum: 200, Time Executed: 200
Minix3: Quantum = 200Minix 3: <pid> 214 swapped in
Time Quantum: 200, Time Executed: 200
Minix3: Quantum = 200Minix 3: <pid> 214 swapped in
Time Quantum: 200, Time Executed: 200
Minix3: Quantum = 200Minix 3: <pid> 214 swapped in
Time Quantum: 200, Time Executed: 200
Minix3: Quantum = 200Minix 3: <pid> 214 swapped in
Time Quantum: 200, Time Executed: 200
Minix3: Quantum = 200Minix 3: <pid> 214 swapped in
Time Quantum: 200, Time Executed: 151
Minix3: Quantum = 200      20.03 real      20.03 user       0.00 sys
arithoh completed
---
# Minix 3: <pid> 23 swapped in
Time Quantum: 500, Time Executed: 1
Minix3: Quantum = 500Write done: 1008000 in 1.3000, score 193846
COUNT|193846|0|KBps
TIME|1.3
```

- **Observations & Inference:** We see that the 3 CPUs have a tendency to be completed before the 2 I/O bound processes which are not perfectly in FCFS order. However, this is due to the fact that the 2 I/O bound processes are sent to the waiting queue after requesting for I/O and are then placed back in the ready queue and scheduled to work on the CPU. As seen in the screenshot, CPU bound tasks utilize all of their quanta whereas I/O bound tasks don't always do the same.

★ Workload Mix 3

4 Parallel executions of ./pipe.sh and 1 Parallel executions of ./fstime.sh

```
./pipe.sh & ./pipe.sh & ./fstime.sh & ./pipe.sh & ./pipe.s
```

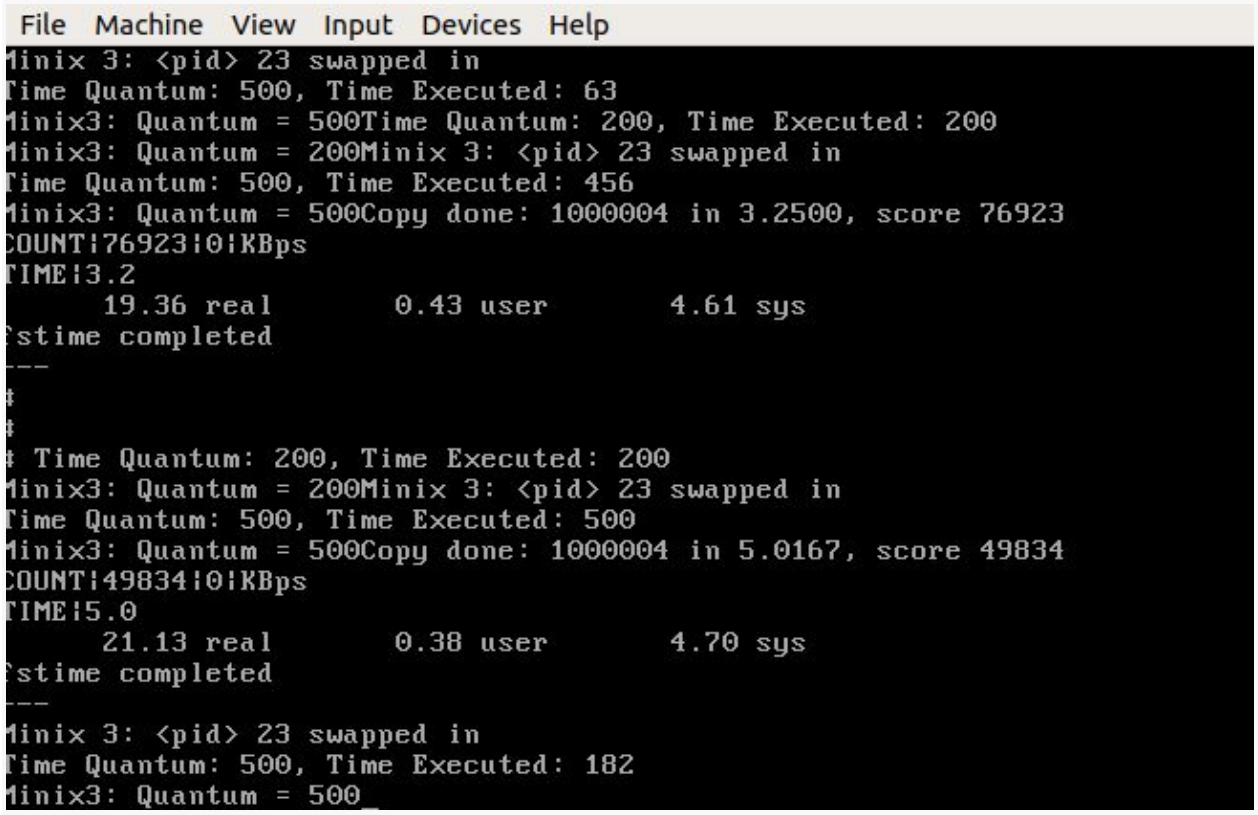


- **Observations & Inference:** As pipe.sh involves IPC, it invokes system calls and hence gets a higher quantum (500 ms for system calls). The processes run in nearly the FCFS manner. The 2 I/O bound processes are sent to the waiting queue after requesting for input and finally get executed after being scheduled back into the ready queue.

★ Workload Mix 4

5 Parallel executions of ./fstime.sh

```
./fstime.sh & ./fstime.sh & ./fstime.sh & ./fstime.sh & ./fstime.sh
```



- **Observations & Inference:** The five processes execute in FCFS manner. Here the quanta assigned to each process is not exactly 500 and is little less than that. This is because it doesn't require the full quanta when the process is nearing completion.