Report

Operating Systems and Networks

Pavan Karke

2023

Question statement

The report must contain explanation about the implementation of the various implemented Scheduling algorithms.

Include the performance comparison between the default and 2 implemented policies in the README by showing the average waiting and running times for processes. Set the processes to run on only 1 CPU for this purpose.

Report

MLFQ scheduling analysis

Create timeline graphs for processes that are being managed by MLFQ Scheduler. 'Variate the length of time that each process consumes the CPU before willingly quitting using the benchmark from Specification 2. The graph should be a timeline/scatter plot between queue_id(y-axis) and time elapsed(x-axis) from start with color-coded processes.

1)Round Robin:

it is default in xv6

intr_on(); is used to avoid deadlock by ensuring that devices can interrupt.

we first check ,if process is RUNNABLE

All processes that are ready to execute are placed in a queue, proc[NPROC]

critical section is locked in case of multiple CPU's like: acquire(&p->lock); to release: release(&p->lock);

each process can execute maximum of 1 quantum time of CPU

If the process completes its execution within the time quantum, it is removed from the queue.

However, if the process does not finish within the time quantum, it is

Round Robin may result in significant context-switching overhead if the time quantum is too short

commonly used in multitasking operating systems

Round Robin ensures fairness among processes by distributing CPU time equally, preventing any single process from monopolizing resources.

Its implementation is in kernel/proc.c

Average rtime 13, wtime 152 of RR

2)FCFS(First-Come, First-Served):

The function runs the infinite for loop for(;;)

Then, we loop over all processes to find a process with the least creation time (to avoid Convoy Effect)
In the first iteration of for(p = proc ...), the process seen is set as the process that has come first (first_process)
Then, the rest of the process creation times are checked
Any time we find a process with a lower creation time, first_come_process gets updated, and at the end of for(p = proc ...) we would have found the process with the least creation time
Then, this process' state is changed to RUNNING, and it is assigned the CPU
Its implementation is in kernel/proc.c

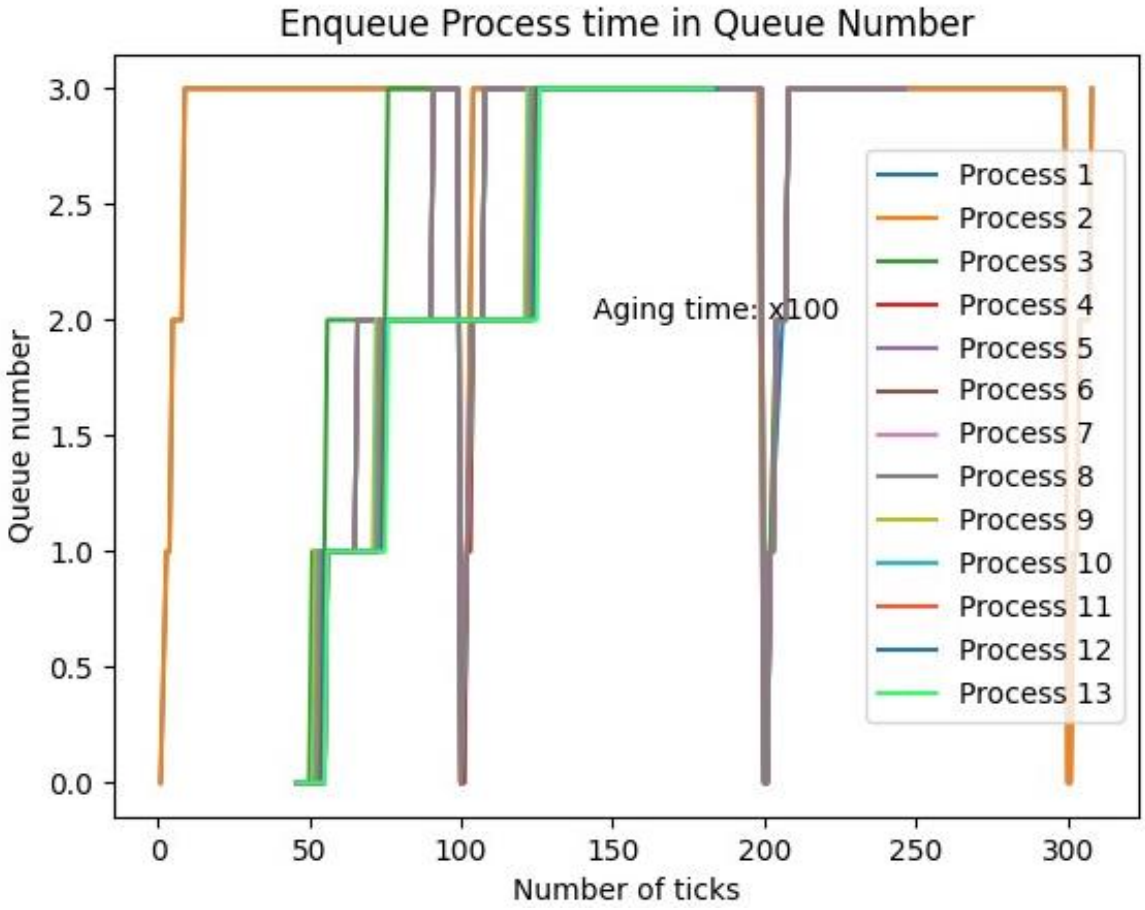
FCFS doesn't take into account the priority of processes, which can be a limitation in systems where certain tasks need to be executed with higher priority.
Average rtime 13 wtime 126

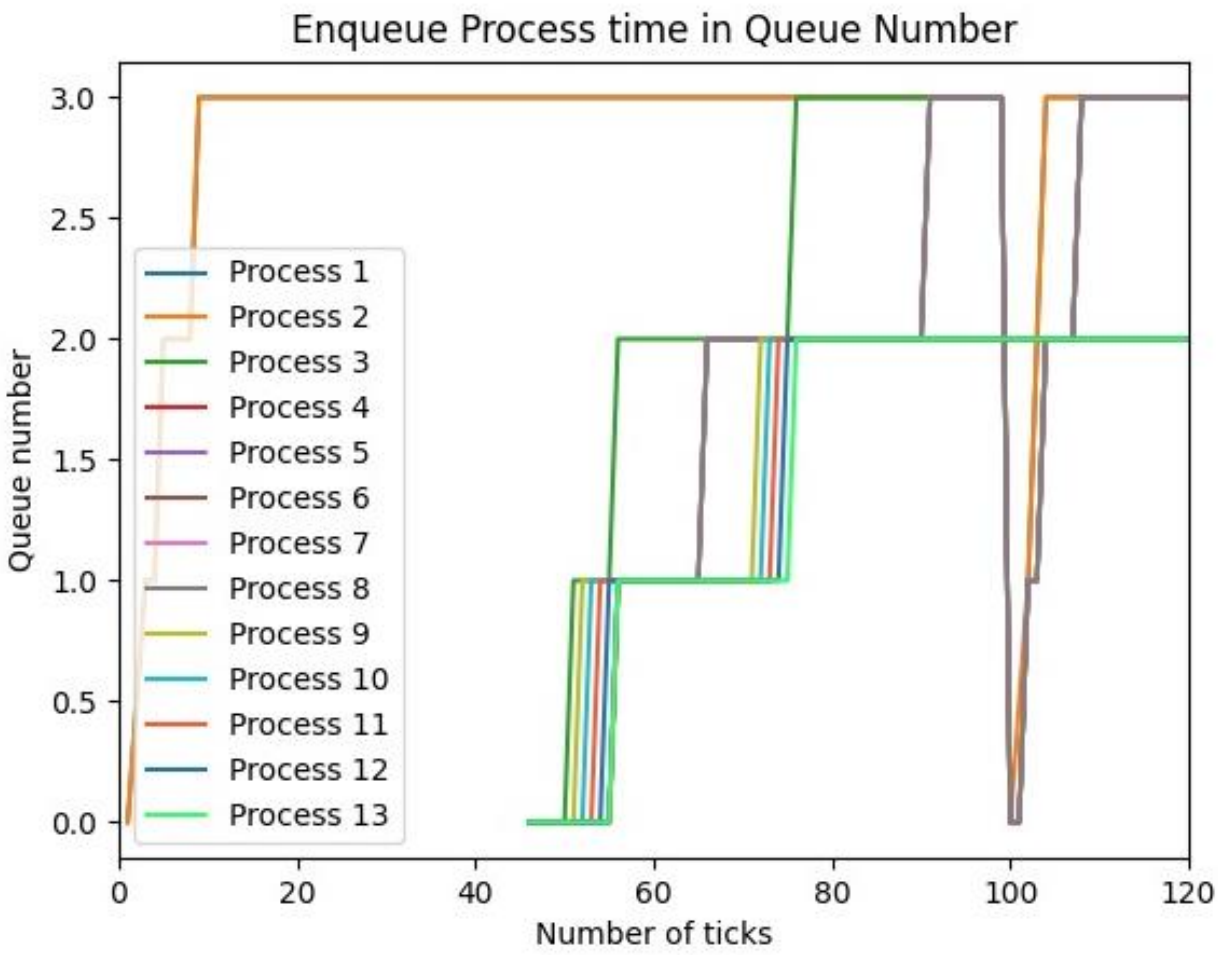
3)MLFQ(Multi-Level Feedback Queue):
MLFQ uses multiple queues,

I have made a queue[4][64] ,queue 0 is highest priority ... queue 3 of lowest
time slice at each priority is denoted by ,maxtick[4]={1,3,9,15}
On the initiation of a process, enqueue to the end of the highest priority queue ,i.e. 0th index queue
process compute for long are demoted(applies queue 0,1,2) ,and process that wait for long are promoted

When the process completes, removed using exit ,wait x function in proc.c
round-robin scheduling is implemented at each queue
aging is done with aging variables,sets after 100 ticks to prevent starvation
always run the processes that are in the highest priority queue that is not empty using RR
Its implementation is in kernel/proc.c
Average rtime 13 wtime 153

MLFQ Scheduling Analysis Graph





4

Zoom in MLFQ for showing RR