Report - Assignment 4 Enhancing XV6

Team - Prayush Rathore(2020114009) and Yash Agarwal(2020114005)

Specification 1: syscall tracing

- Added $U/_strace to UPROGS in Makefile
- Added `sys_trace()` in kernel/sysproc.c that reads the syscall argument and calls the `trace()` defined in proc.c with the argument
- `trace()` sets the `mask` field of the process struct
- Modified `fork()` to copy `mask` from parent to child
- Modified `syscall()` to print syscall info if the syscall is traced
- Created user program `strace` in user/strace.c that calls the trace system call to set the mask of the process and run the function passed.

##

Specification 2: scheduling

- Modified Makefile to take argument which then defines a macro with the compiler to identify the scheduling algorithm

###

FCFS Policy

- Edited `struct proc` to store the time it was created
- Edited `allocproc()` to initialise the new variable created above
- Edited `scheduler()` to run the process with the lowest time created
- Edited `kerneltrap()` in kernel/trap.c to disable premption with timer interrupts

LBS Policy

- Edited `struct proc` to store the number of tickets for each process
- Edited `allocproc()` to initialise the new variable created above
- Edited `scheduler()` to run the process which wins the lottery
- Created `rand_max()` to generate a random number less than a given number
- Edited `kerneltrap()` in kernel/trap.c to disable premption with timer interrupts

Variables in struct proc have been added to record all of the information needed by the scheduler to determine tickets. In scheduler() , I added a new scheduling algorithm for LBS that schedules according to lottery prize. By incrementing temp_count , I was able to find a process to schedule.

PBS Policy

- Edited `struct proc` to store the priority, time dispatched, runtime during allocated time, and time when it ready to run
- Edited `allocproc()` to initialise the new variables created above
- Edited `scheduler()` to run the process with the highest priority
- Edited `clockintr()` to track runtime and wait time
- Added a new sycall `set_priority` to change the priority of a process

Variables in struct proc have been added to record all of the information needed by the scheduler to determine priority. In scheduler() , I added a new scheduling algorithm for PBS that schedules according to dynamic priority. By incrementing update_time , I was able to find the run and sleep times. set_priority is a new syscall that modifies the process's static priority.

###

MLFQ Policy

- Edited `struct proc` to store the priority, allocated time, times dispatched, time added to queue, and time spent in each queue
- Edited `allocproc()` to initialise the new variables created above
- Created 5 queues of different priority
- Edited `scheduler()` to run the process with the highest priority
- Edited `clockintr()` to track runtime, add processes to queue and handle aging
- Edited `kerneltrap()` and `usertrap()` to yield when process has exhausted its time slice

To store the processes, I created five queues. When a process is started, it is pushed to the end of the highest priority queue, which is queue 0. New variables were added to struct proc and initialized in allocproc . The process is preempted and inserted at the end of the next lower level queue if it uses the entire time slice assigned to its current priority queue. This is done in the kernel as well as the user trap. To avoid starvation, aging was implemented.

Performance Comparison

Note: All are tested on 1 CPU

RR

Average rtime 32, wtime 229

FCFS

Average rtime 68, wtime 301

LBS

Average rtime 54, wtime 289

PBS

Average rtime 34, wtime 197