XV-SIX

Added system call `waitx(int waiting_time,int running time)` , user program `ps` and three scheduler policies `FCFS` `PBS` `MLFQ` to existing xv6 os.

Running XV-SIX

`make clean`

`make qemu-nox [SCHEDULER=RR/FCFS/PBS/MLFQ]`

NOTE: default scheduler is RR

`waitx()` syscall

```
int waitx(int* wtime, int* rtime)
```

The two arguments are pointers to integers to which waitx syscall assigns the total number of clock ticks during which process was waiting and the total number of clock ticks when the process was running. The return values for waitx is the same as that of the wait system-call.

A test program utilizes the waitx system call by creating a `time` like command for the same.

The test command can be run by the following command

`time [command]`

`ps` user program

The ps user program returns some basic information about all the active processes.The different columns of the ps user program are explained below

● Priority- Current priority of the process (defined as per the need of schedulers below)

● State- Current state of the process

● r-time- Total time for which the process ran on CPU till now (use a suitable unit of time)

● w-time- Time for which the process has been waiting (reset this to 0 whenever the process gets to run on CPU or if a change in the queue takes place (in the case of MLFQ scheduler))

● n_run- Number of times the process was picked by the scheduler

● cur_q- Current queue (check task 2 part C)

● q{i}- Number of ticks the process has received at each of the 5 queues

Note:

● i={0,1,2,3,4} - Since 5 priority queues are implemented, with the highest priority being number as 0 and the bottom queue with the lowest priority as 4

● cur_q and q{i} displayed only in case of MLFQ scheduler

Schedulers

RR

Round-robin (RR) is one of the algorithms employed by process and network schedulers in computing.[1][2] As the term is generally used, time slices (also known as time quanta)[3] are assigned to each process in equal portions and in circular order, handling all processes without priority (also known as cyclic executive). Round-robin scheduling is simple, easy to implement, and starvation-free

Round Robin is the default scheduler in xv6 os. run `make qemu-nox` or `make qemu-nox SCHEDULER=RR` to run processes on RR scheduling

For RR: Avg waiting time : Avg total time :

FCFS

First Come First Serve (FCFS) is an operating system scheduling algorithm that automatically executes queued requests and processes in order of their arrival. It is the easiest and simplest CPU scheduling algorithm. In this type of algorithm, processes which requests the CPU first get the CPU allocation first. This is managed with a FIFO queue

● The major change for FCFS Scheduler implementation is to remove yielding in trap.c. i.e. we do not allow the CPU to pre-empt any process that is running.

● To run FCFS Scheduler run the command `make qemu-nox SCHEDULER=FCFS`

● FCFS is implemented by traversing the ptable and finding the process with least creation time and running it without pre-emption.

PBS

Priority scheduling is a non-preemptive algorithm and one of the most common scheduling algorithms in batch systems. Each process is assigned first arrival time (less arrival time process first) if two processes have same arrival time, then compare to priorities (highest process first). Also, if two processes have same priority then we choose process by RR scheduling here. This process is repeated while all process get executed.

MLFQ

In a multilevel queue-scheduling algorithm, processes are permanently assigned to a queue on entry to the system. Processes do not move between queues. This setup has the advantage of low scheduling overhead

● There 5 priority queues numbered 0-4 , with 0 being the highest priority queue and 4 being the least.

● Any process at the time of creation is sent into the first priority queue and is demoted downwards as the time slice of the process in respective queues is completed.

● The time-slice for priority 0 is 1 timer tick. The times-slice for priority 1 is 2 timer ticks; for priority 2, it is 4 timer ticks; for priority 3, it is 8 timer ticks; for priority 4, it is 16 timer ticks.

● Aging is also implemented.

Testing and comparing performances of different Schedulers

The `time` command can be used to run a benchmark file on different schedulers and the performance can be compared based on respective wait time and total time. eg:

```
time benchmark
```

These following results are recorded on testing the time command with a sample benchmark file :

RR : wtime:1993 rtime:17

FCFS : wtime:1987 rtime:23

PBS : wtime:1995 rtime:17

MLFQ : wtime:1990 rtime:20

License