

BSD 4.4 Scheduling Algorithm

- It is a *multilevel feedback queue* based algorithm
- What is multilevel feedback queue scheduling ?
 - Maintain multiple queues of ready processes, each with different priorities
 - Jobs can be moved between the queues

Specifying a Multilevel Feedback Queue Based Algorithm

- How many queues (priority levels) ?
- How do you move processes between queues ? (or change priority level of a process)
- Scheduling Algorithm to be used within a queue
- How to choose which queue to run process from

BSD 4.4 Basics

- Priority level can be between 0 and 127. Divide by 4 and have 32 queues
- Round robin scheduling within a queue, time quantum of 0.1 seconds
- Select a new process to run from the queue with the lowest priority value
- Adjust the priority every 4 clock ticks or 40 milliseconds

More on Priority Levels

- Priority levels 0 to 49 are reserved for kernel processes - we will only handle user processes - priority levels between 50 and 127
- Adjust priorities using the equation

$$p_usrpi = 50 + (p_estcpu/4) + 2 \times p_nice$$

- *p_estcpu* is the estimated and weighted cpu utilization
- *p_nice* is set by the user and is between -20 and 20
- If *p_usrpi* comes out lower than 50, adjust to 50
- If *p_usrpi* comes out higher than 127, adjust to 127

Accumulated CPU time

- The factor p_estcpu , used for adjusting priorities
- Every clock tick, add one to p_estcpu for the process that is running
 - can be implemented in OSP using the length of last CPU burst
- After every 1 second, *decay* the CPU usage
- The equation to be used is

$$p_estcpu = \frac{2 \times load}{2 \times load + 1} \times p_estcpu + p_nice$$

- *load* is the sampled average of the length of run queues over the previous 1 minute interval of the system operation

- If *load* is 1, only 13% of the time remains after 5 decay computations

Processes that are not running or in run queue

- If sleeping less than 1 second, use old values of p_estcpu
- Use time of last dispatch and length of last cpu burst to tell how long it was sleeping
- If sleeping greater than 1 second, use the equation

$$p_estcpu = \frac{2 \times load}{2 \times load + 1} \times p_estcpu$$

- p_sptime is the time it was asleep in seconds
- Use the load values when it awakes