Scheduling

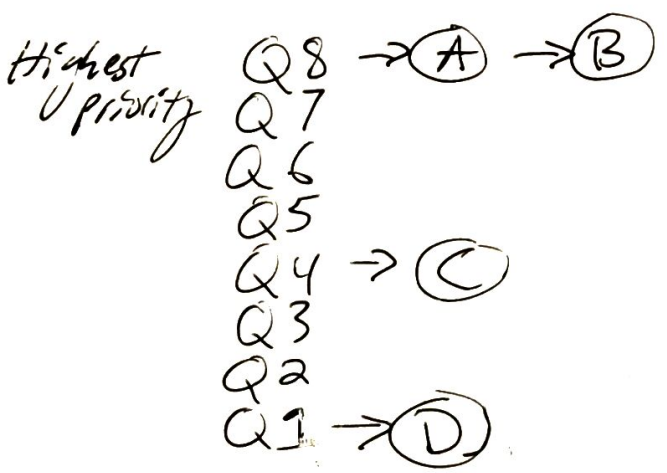
Multi-Level Feedback Queue

- optimize turn around
- Responive

has multiple queue's that represent different priorities then works off Rand Robin

Rule 1: Priority(A) > Priority(B)? A Runs
Rule 2: If Priority(A) = Priority(B), Round Robin

MLFQ tries to learn about processes as they Run, and bases priority off that



Changing Priority

Rule 3: When a Job enters the system it is placed at the highest priority
Rule 4a: If Job uses entire time slice while running it's priority is reduced
Rule 4b: If a Job gives up CPU before time runs out it keeps priority

Problems with MLFQ

Starvation: If there are too many high priority tasks, low priority jobs get starved

Gaming the Scheduler: a smart programmer can make their program take 99% and monopolize the system

Programs change Behavior: with current system it couldn't move back up the queue

Priority Boost

Rule 5: After some time "S" all jobs are moved to the topmost queue

Better Accounting

Revised Rule 4: Once a Job uses up its time allotment at a given level (Regardless of how many times it has given up the CPU) its priority is reduced

-Defaults*

≈60 queues with slowly increasing time slices.

Priority boost every ≈ 1 second

Proportional - Share Scheduling

- Lottery: Uses Randomness
- Stride: Deterministic
- CFS (Completely Fair Scheduler): weighted Round Robin with dynamic time slices
  - Uses Red/Black Trees
  - Scales well