Lecture 5: finish sched / synchronization

- SRTF / Adaptive scheduling
- Real time scheduling

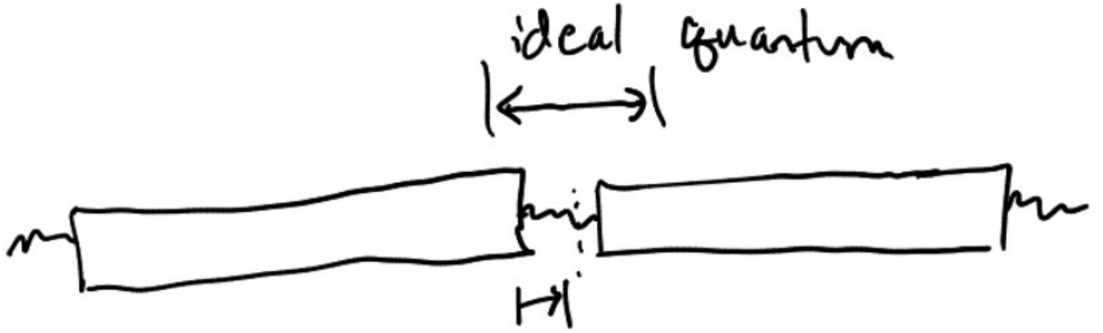
- Mutual exclusion

Round-robin

- like FCFS, periodically preempt processes
- how long should quantum be?
  - short quantum: lots of context switches
  - long quantum: · potentially long delays
    · low responsiveness
    · high avg. waiting time ← time spent in ready state

(infinite quantum: FCFS)

Two kinds of processes:



- I/O bound processes
  - text editor
  - web server
  - need responsiveness: short quantum

higher priority (handle first)

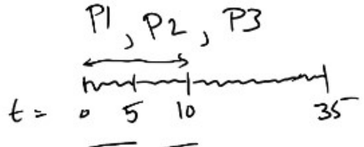
- CPU-bound processes

  - data processing.
  - need efficient use of CPU: long quantum.

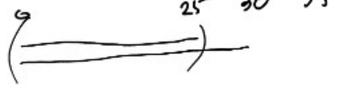
Optimal algorithm: Shortest remaining time first (SRTF)

- Run whichever process will finish first (do I/O)

P1: will run for 5
P2: " 5
P3: " 25



turnaround time
P1: 5 P3: 35
P2: 10 total: 50



turnaround:
P1: 30 P3: 25 total: 90
P2: 35

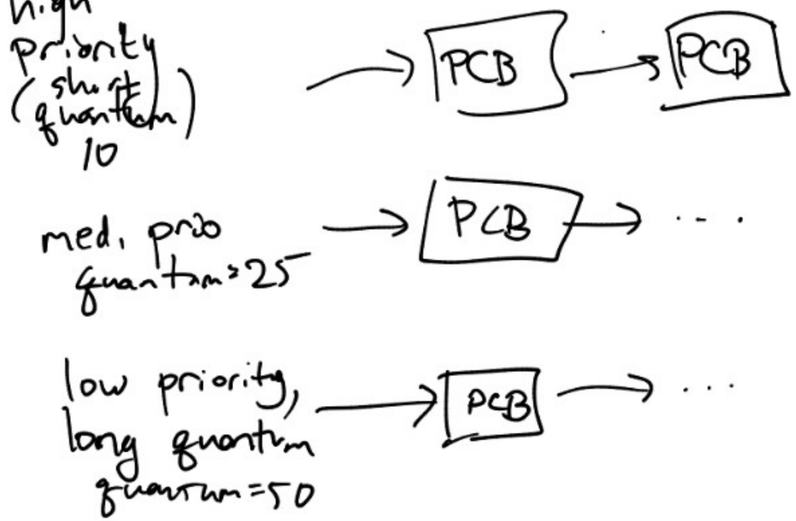
Pro:
- optimal (waiting time)
- if we assume user-facing procs are I/O bound: responsive.
- minimal ctx switches

Con:
- impossible to know how long procs will take.
- unfair if lots of short jobs continually arrive (long jobs would starve)

Adaptive multi-level scheduler

. idea: observe procs, I/O bound procs get short quanta, CPU bound procs get long quanta.

← I/O bound procs.



- new procs (ones that returned from I/O) go in high priority queue
- if proc uses whole quantum, move it to lower priority queue, moved down.

divide time between queues:

- proc. high queue for 50 units
- med queue for 50 units
- low queue for 50

Multiple processors

- want to schedule threads that communicate quickly (e.g. locking & unlocking) together (called gang scheduling)
- processor affinity: all things being equal, might be better to run same thread on same processor.
- fairness: want to ensure that all processes get time.

Real-time scheduling

- processes specify deadlines.
  - e.x. audio processing (live)
  - e.x: car, fighter jet, ...
- scheduler makes a schedule
  (can be optimal if runtimes are given)
- Soft / Hard RT sched:
  Soft: "best-effort" OS scheduler tries to meet constraints, but might not be possible.

  hard: access control: scheduler can refuse requests for resources (CPU before deadline)
  - guarantee that demand is met.

Milk problem / Mutual exclusion:

See typed lecture notes