Slide 1

Administrivia

- As mentioned in e-mail, I graded the first reading quiz and uploaded results in your individual folders, plus a sample solution in the shared folder. Generally I tried not to be picky about your answers not quite being what I intended!
- Reminder: Homework 1a, 1b due today.
- Next assignments coming soon, I intend before next Monday. I'll send mail. You'll have a full week on them.

Slide 2

Minute Essay From Last Lecture

- Most people thought turning things in on Google Drive was fine (but then several forgot to turn in Reading Quiz 2 — "hm!"?).
- One person asked how it was working for me. I do miss noticing work coming in, with an easy way for me to respond to it, but I don't miss trying to be sure I don't misfile something!

  For what it's worth, I'm using a tool that lets me do mass downloads, but it does rely on set folder names, which is why I'm setting them up for you. (And if you don't see one, I forgot, so please ask!).

Slide 3

CPU Scheduling — Review/Recap

- Several situations in which we need to, or might want to, switch between processes.
- Nuts and bolts of how — "mechanism" that doesn't change. What changes are the decisions — "policy".
- Which policy is best? Depends on several factors — workload, what we mean by "best" ("metric").
- Discussion in terms of "jobs" and starts with many unrealistic assumptions, later relaxed one by one.

Slide 4

Simple Scheduling Algorithms

- Previous class: FCFS, SJF, SJF with preemption, round robin.
- Two more the textbook doesn't discuss . . .

Slide 5

Priority Scheduling

- Basic ideas:
  - Keep a queue of ready processes, as before.
  - Assign a priority to each process.
  - When a process starts (or, if we also allow for blocking, when it becomes unblocked), add it to the end of the queue.
  - Switch when the running process exits (or blocks), or possibly when a process starts. (I.e., preemption may be allowed.)
  - Next process is the one with the highest priority.
- Points to consider:
  - What happens to low-priority processes? (So, maybe we should change priorities sometimes?)
  - How do we decide priorities? (external considerations versus internal characteristics)

Slide 6

Multiple-Queue Scheduling

- Basic idea — variant on priority scheduling:
  - Divide processes into “priority classes”.
  - When picking a new process, pick one from the highest-priority class with ready processes.
  - Within a class, use some other algorithm to decide (round-robin, e.g.).
  - Optionally, periodically lower processes’ priorities.
- If we let the algorithm manage priorities, then we have . . .

Slide 7

Multi-Level Feedback Queue Scheduling

- Same idea as MLQ, just with some details specified — how priorities are initially set and how they change.
- Goal is to balance good response time and good turnaround time, sometimes at odd.
- Key idea is one used in many context in computer science: It often works to assume that the near future will be much like the recent past.
- How this applies: A process that hasn't had long CPU bursts recently won't start having them in the near future.
- Textbook goes into some detail; I'll skip for now, except to note: "Gaming the system" (to get more than one's share of computing resources) is a real thing!

Slide 8

"Fair Share" Algorithms

- (Skim these too for now.)
- Lottery scheduling:

  Give each process one or more "lottery tickets" — more or fewer depending on its priority (so to speak); pick one at random to decide who's next.
- Guaranteed scheduling (as used in Linux Completely Fair Share?):

  "Guarantee" each process (of N) 1/N of the CPU cycles; (try to) schedule to make this true.

  Calculate, for each process, fraction of the time it has had the CPU in its lifetime, fraction it "should" have had; choose process for which actual time / entitled time is smallest.

Slide 9

Minute Essay

- I'm planning to record one more lecture on the topic of scheduling, with some of the skipped details. Anything more about scheduling, or about CPU virtualization in general, you want to hear about?