Slide 1

Administrivia

- Reminder: Reading Quiz 2 due today.
- First homework posted, in two parts (1a is written problems, 1b is a programmming problem).
- I'm planning to record a second lecture for this week, by Friday, and ask you to watch over the weekend. Hope that's okay!
- Also, about assignments: My plan is to have one more reading quiz and one more homework about virtualizing the CPU and then move on.
- (And yes, I *am* hoping to get some feedback to you soon about what you're turning in!)

Slide 2

Minute Essay From Last Lecture

- Opinions differ as to whether it would help to relax the collaboration rule for reading quizzes. I think I will.
- Most people seemed to find them worthwhile, anyway, though some found the questions vague. I can believe it! coming up with good questions — for these, homeworks, etc. — is harder than it might seem?

Slide 3

CPU Scheduling

- (The textbook organizes this discusssion a bit differently; I'll approach it more as the textbook I used to do did. Same material!)
- We've talked about how processes can be in different states (ready, running, blocked, others) and that some transitions (e.g., "running" to "blocked") don't really involve any decision-making, but others ("ready" to "running") do.
- Who/what makes these decisions? "scheduler".

Slide 4

Scheduling, Continued

- When to make scheduling decisions?
  - When a new process is created.
  - When a running process exits.
  - When a process becomes blocked (I/O, etc.).
  - After an interrupt (external source, timer, etc.).

  One possible decision — "go back to interrupted process" (e.g., after I/O interrupt). But there are other choices.
- How to choose? various "scheduling algorithms".

Slide 5

Scheduler Goals

- Importance of scheduler can vary; extremes are
  - Single-user system — often only one runnable process, complicated decision-making may not be necessary (though still might sometimes be a good idea).
  - Mainframe system — many runnable processes, queue of "batch" jobs waiting, "who's next?" an important question.
  - Servers / workstations somewhere in the middle.
- First step is to be clear on goals — want to make "good decisions", but what does that mean? (Textbook frames this as "what should we use for a metric"?)

Slide 6

Scheduler Goals, Continued

- Typical goals for any system:
  - Fairness — similar processes get similar service.
  - Policy enforcement — "important" processes get better service.
  - Balance — all parts of system (CPU, I/O devices) kept busy (assuming there is work for them).
- Other goals depend on system type.

Slide 7

Terminology

- Discussion often in term of "jobs" — holdover from mainframe days, means "schedulable piece of work".
- Processes usually alternate between "CPU bursts" and I/O, can be categorized as "compute-bound" ("CPU-bound") or "I/O-bound".
- Scheduling can be "preemptive" or "non-preemptive".

Slide 8

Scheduler Goals By System Type

- For batch (non-interactive) systems, possible goals (might conflict):
  - Maximize throughput — jobs per hour.
  - Minimize (average?) turnaround time (time from when user submits work to time they get results back).
  - Maximize CPU utilization.

  Preemptive scheduling may not be needed.
- For interactive systems, possible goals:
  - Minimize response time.
  - Make response time proportional to user's perception of task difficulty.

  Preemptive scheduling probably needed.

Slide 9

Scheduler Goals By System Type, Continued

- For real-time systems, possible goals:
  - Meet time constraints/deadlines.
  - Behave predictably.

Slide 10

Scheduling Algorithms

- Many, many scheduling algorithms, ranging from simple to not-so-simple.
- Point of reviewing lots of them? notice how many ways there are to solve the same problem ("who should be next?"), strengths/weaknesses of each.

Slide 11

First Come, First Served (FCFS)

- Basic ideas:
  - Keep a (FIFO) queue of ready processes.
  - When a process starts or becomes unblocked, add it to the end of the queue.
  - Switch when the running process exits or blocks. (I.e., no preemption.)
  - Next process is the one at the head of the queue.
- Points to consider:
  - How difficult is this to understand, implement?
  - What happens if a process is CPU-bound?
  - Would this work for an interactive system?

Slide 12

Shortest Job First (SJF)

- Basic ideas:
  - Assume we know ahead of time how long each "job" will take, and each job consists of a single CPU burst (so, no blocking).
  - Keep a queue of these jobs.
  - When a process (job) starts, add it to the queue.
  - Switch when the running process exits (i.e., no preemption).
  - Next process is the one with the shortest running time.
- Points to consider:
  - How difficult is this to understand, implement?
  - What if we don't know running time in advance?
  - What if all jobs are not known at the start?
  - Would this work for an interactive system?

Slide 13

SJF, Continued

- Key advantage — if all jobs are in the queue at start, this gives the best average turnaround time — provably.
- Key disadvantage — what happens if we're running a medium-length job and a shorter one arrives?

Slide 14

SJF Plus Preemption

- A possible fix — allow preempting running job.
- (Various names — PSJF (preeptive SJF), STCF (shortest time to completion first), SRTN (shortest remaining time next).)
- Basic idea:
  - Keep a queue of ready processes as before.
  - Switch when the running process exits *or* a new process starts. (I.e., preemption allowed — requires recomputing time left for preempted process.)
  - Next process is the one with the shortest time left.

Slide 15

Round-Robin Scheduling

- Basic ideas:
  - Keep a queue of ready processes, as before.
  - Define a "time slice" — maximum time a process can run continuously before it must yield to another. (Should be a multiple of timer-interrupt period.)
  - When a process starts or becomes unblocked, add it to the end of the queue.
  - Switch when the running process uses up its time slice, or it exits or blocks. (I.e., preemption allowed!).
  - Next process is the one at the head of the queue.

Slide 16

Round-Robin Scheduling, Continued

- Points to consider:
  - How difficult is this to understand, implement?
  - Would this work for an interactive system?
  - How do you choose the time slice? (What if it's really long? really short?)

Slide 17

Minute Essay

- How are you doing with how I ask you to turn work in (e-mail for minute essays, Google Drive for homeworks)?

  (For what it's worth, I'm aiming here for a compromise between what might work best for y'all — TLEARN? Google Classroom? — and what works well for me.)