Slide 1

Administrivia

- Reminder: Homework 3 due today.
- Reminder: Quiz 3 Monday. Midterm Wednesday. Brief review Monday.

Slide 2

Evaluating Scheduling Algorithms

- How to decide which scheduling algorithm to use?
- One way — evaluate several choices, see which one best meets system goal(s). E.g., if the goal is minimum turnaround time, try to come up with an average turnaround time for each proposed choice.
- Several approaches possible . . . (This discussion is from another operating systems textbook, by Silberschatz and Galvin.)

Deterministic Modeling

- Idea — use a predetermined workload, compute values of interest (e.g., average turnaround time, as in Homework 3 problem).
- How well does it work?

Deterministic Modeling, Continued

- Simple, fast, gives exact numbers.
- Requires exact numbers as input, and only applies to them.

Slide 5

Queueing Models

- Idea — use “queueing theory” to model system as a network of “servers”, each with a queue of waiting processes. (E.g., CPU is a server, with input queue of ready processes.)
- Input to model — distribution of process arrival times, CPU and I/O bursts for processes, as mathematical formulas. (Base this on measuring, approximating, or estimating.) In queueing-theory terms, “arrival rates” and “service rates”.
- Queueing theory lets you then compute utilization, average queue length, average wait time, etc.
- How well does it work?

Slide 6

Queueing Models, Continued

- Seems more general than deterministic modeling.
- But can be tricky to set up model correctly, and need to approximate / make assumptions may be a problem.

Slide 7

Simulations

- Idea — program a model of the computer system, simulating everything, including hardware.
- Two ways to get input for simulation:
  - Generate processes, burst times, arrivals, departures, etc., using probability distributions and random-number generation.
  - Create “trace tape” from running system.
- How well does it work?

Slide 8

Simulations, Continued

- Potentially very accurate.
- Time-consuming to program and to run!

Slide 9

Implementation

- Idea — code it up and try it!
- How well does it work?

Slide 10

Implementation, Continued

- Seems like potentially the most accurate approach.
- Requires a lot of work, resources.
- Involves implicit assumption that users' behavior is fairly constant.

  (So it's good to build into the algorithm some parameters that can be changed at run time, by users and/or sysadmin. In textbook's phrase, "separate mechanism from policy". Notice, though, users are apt to figure out how to game any system.)

Slide 11

Recap — Scheduling Algorithms

- Main idea — decide which process to run next (when running process exits, becomes blocked, or is interrupted).
- Many possibilities, ranging from simple to complex. Real systems seem to use hybrid strategies.
- How to choose one?
  - Be clear on goals.
  - Maybe evaluate some possibilities to see which one(s) meet goals — analytic or experimental evaluation.
  - Build in some tuning knobs — “separate policy from mechanism”.

Slide 12

Memory Management — Preview

- One job of operating system is to “manage memory” — assign sections of main memory to processes, keep track of who has what, protect processes’ memory from other processes.
- As with CPU scheduling, we’ll look at several schemes, starting with the very simple. For each scheme, think about how well it solves the problem, how it compares to others.
- As with processes, there’s a tradeoff between simplicity and providing a nice abstraction to user programs.

Minute Essay

- This wraps up the discussion of processes and scheduling. Anything that wasn't clear, or that you'd like to hear more about?