Slide 1

Administrivia

- Reminder: Homework 6 due today.
- Reminder: Quiz 6 Tuesday.
- Homework 7 on the Web; due a week from today. (Again, start early for some clues about what might be on the next quiz.)

Slide 2

Order of Magnitude of Functions

- By now you've probably heard "this is an $O(N)$ algorithm", etc., many times. Here we'll define it formally.
- First: When we talked about analysis of algorithms (chapter 2), we came up with estimates of "total work" of the algorithm as a function of size of input ("problem size"). Useful and interesting, but a bit fine-grained — what we usually care about is behavior as problem size gets very big.
- So — idea is to come up with an "order of magnitude" for functions, analogous to "order of magnitude" for numbers. If the functions for two algorithms have the same order of magnitude, the functions are in some sense about equally fast/slow.
- Example: If you have two algorithms for processing an image with N pixels, one that takes time proportional to $1000N$ and one that takes time proportional to time N^2, which do you pick? (Does the size of N matter?)

Slide 3

Order of Magnitude of Functions, Continued

- How to determine an order of magnitude for functions?

  If we look at graphs of functions, we might notice that we can classify them into groups based on their "shape".

  For nondecreasing functions, we also notice that some shapes "grow" faster than others.

  (Compare x^2, $10x^2$, x^3, etc.)

- Idea is that we want functions that have the same shape to have the same order of magnitude.

Slide 4

Order of Magnitude of Functions, Continued

- Formal definition:

  Write $f = \Theta(g)$ to mean that f and g have the same order of magnitude.

  Define to be true iff there are positive constants n_0, c_1, c_2 such that for all $x \geq n_0$

  $$c_1 g(x) \leq f(x) \leq c_2 g(x)$$

  In other words, these functions are roughly proportional to each other.

- Can guess values c_1, c_2 and more or less show that they work by plotting resulting curves — but to really show that the definition holds, must do algebra to show. (Example: f(x)=3x+2, g(x)=x-10.)

Slide 5

"Big-O Notation"

- The $O(f(N))$ you see in computer science is similar, but it's a "less than or equal" rather than a "strictly equal" — i.e., $f(N) = O(g(N))$ means f's order of magnitude is no bigger than g's (and might be less).

  Formally, true iff there are positive constants n_0 and c such that for all $x \geq n_0$

  $$f(x) \leq cg(x)$$

- (If you wonder why you haven't seen this done before — it's the formal definition, but quite tedious to apply, so people have come up with (and proved) general rules for polynomials, other common functions.)
- Interesting(?) to observe that Θ is an equivalence relation, and O is a partial ordering.

Slide 6

Order of Magnitude of Functions, Continued

- So we have a way to compare orders of magnitude of functions, with an "equals" (Θ) and a "less-than-or-equal-to" (O).
- In general, function's order of magnitude determined by fastest-growing term. Some categories of interest:
  - x^2 grows faster than x, x^3 faster than x^2, etc. x^2 and cx^2 "the same".
  - $\log_b x$ grows more slowly than x.
  - b^x grows faster than all polynomials.
  - x^x grows faster than all b^x.

Minute Essay

- Which of the following functions are $O(N^2)$?
  $g(N) = 100N^2 + N - 1000$
  $h(N) = N^3$
- Which of the following functions are $O(2^N)$?
  $f(N) = 2^N - 5$
  $h(N) = N!$

Slide 7

Minute Essay Answer

- $O(N^2)$?
  $g(N) = 100N^2 + N - 1000$ — yes
  $h(N) = N^3$ — no
- $O(2^N)$?
  $f(N) = 2^N - 5$ — yes
  $h(N) = N!$ — no

Slide 8