Administrivia

- Reminder: Homework 4 due Monday.

Slide 1

Solving Recurrence Relations — Review/Recap

- Goal is to take a recursive definition of a sequence and come up with a“closed-form” (non-recursive) definition of the same sequence.
- One method is what textbook calls “expand, guess, verify” — example last time.

Slide 2

Slide 3

Solving Recurrence Relations, Continued

- Is there another way? In general, probably not, but there are some formulas for some frequently-occurring special cases.
- One is "first-order linear" recurrence relations. If

  $$S(n) = cS(n-1) + g(n)$$

  then we can show (see textbook for derivation) that

  $$S(n) = c^{n-1}S(1) + \sum_{i=2}^{n}(c^{n-i}g(i))$$

- Apply this to the two problems we did earlier — we should get the same results.

Slide 4

Analysis of Algorithms, Overview

- Often there's more than one way to solve a given problem, i.e., more than one algorithm. Which one is "best"? Depends on what "best" means. If we mean "fastest":
- A useful measure of approximate execution time is worst-case (or sometimes average-case) execution time expressed as a function of "problem size" (e.g., for operations on array, size of array) — "time complexity" of algorithm. (Another measure is "space complexity".)
- Customary to skip over housekeeping operations and count only "important stuff" — arithmetic operations, comparisons, etc.

  Also customary to "round off" the estimate to an "order of magnitude" — for a problem of size N, we say an algorithm is $O(f(N))$ if execution time is $f(N)$.

Slide 5

Analysis of Algorithms, Examples

- Example — computing a sum of N numbers. How many additions?
- Example — sequential search of array of size N. How many comparisons (worst case)?
- Example — binary search of sorted array of size N. How many comparisons (worst case)?

Slide 6

Minute Essay

- None — quiz.