Data Structures and Algorithms

Spring 2009-2010

Outline

Outline

Growth Rates of Some Commonly Occurring Functions

Function	**Name**	**Example**
c	Constant	Adding two numbers
$\log n$	Logarithmic	Searching a *balanced* binary tree
$\log^2 n$	Log-squared	
n	Linear	Searching a list
$n \log n$	"n log n"	*mergesort*, *quicksort*
n^2	Quadratic	Insertion/Selection/Bubble Sort
n^3	Cubic	Matrix Multiplication (the slow way!)
2^n, a^n, $n!$	Exponential	No. of ways to permute n objects ($n!$)
		No. of unique patterns of n bits (2^n)

Outline

Some Rules

Some rules for combining "Big-Ohs"

Rule 1 If $T_1(n) = O(f(n))$ and $T_2(n) = O(g(n))$, then
(a) $T_1(n) + T_2(n) = max(O(f(n)) + O(g(n)))$;
(b) $T_1(n) * T_2(n) = O(f(n) * g(n))$

Rule 2 If $T(x)$ is a polynomial of degree n, then $T(x) = \Theta(x^n)$

Rule 3 $log^k n = O(n)$ for any constant k. Logarithms grow very slowly.

Outline

Assumptions of Model

When analysing the running time of an algorithm we assume the following *computer model*:

- Instructions are executed sequentially
- Any of the fundamental mathematical operations take a *constant* single unit of time
- We do not care about what the units of time are
- No tricky operations permitted without paying for them
- Our computer has an infinite amount of memory – no paging worries

Weaknesses of Model

Weaknesses of this model:

- Not all operations take same time
- e.g., addition vs. multiplication vs. disk reading

What to measure

- Almost always running time...
- When we quantify running time it will be as a function of the input size, the number of items the algorithm works on, expressed in “Big-Oh” notation
- Two types of running time of interest: *worst* and *average*
- Worst-case running time is the time the algorithm would take on the worst possibly arranged input
- Average-case running time is the time the algorithm can be expected to take on an “average” input
- Average-case is by far the most useful but by far the most difficult to derive
- We satisfy ourselves with finding a bound on the very worst our algorithm will perform
- We ignore the time taken to read the input, which must be $O(n)$