CS 330

LECTURES 2 & 3

Analysis of Algorithms

- Asymptotic notation
- Basic data structures

Dora Erdos
Adam Smith

Useful Functions and Asymptotics

Permutations and combinations

- Factorial: "n factorial"

$$n! = n \cdot (n-1) \cdot \ldots \cdot 2 \cdot 1$$

= number of permutations of $\{1, \ldots, n\}$

- Combinations: "n choose k"

$$\binom{n}{k} = \frac{n \times (n-1) \times \cdots \times (n-k+1)}{k \times (k-1) \times \ldots \times 2 \times 1} = \frac{n!}{k!(n-k)!}$$

= number of ways of choosing an unordered subset of k items in $\{1, \ldots, n\}$ without repetition

Review Question

- In how many ways can we select two disjoint subsets of $\{1, \ldots, n\}$, of size k and m, respectively?

- Answer:

$$\binom{n}{k}\binom{n-k}{m} = \binom{n}{m}\binom{n-m}{k} = \binom{n}{m+k}\binom{m+k}{k}$$

Asymptotic notation

O-notation (upper bounds):

$f(n) = O(g(n))$ means

there exist constants $c > 0$, $n_0 > 0$ such that $0 \leq f(n) \leq cg(n)$ for all $n \geq n_0$.

EXAMPLE: $2n^2 = O(n^3)$ $\quad (c = 1, n_0 = 2)$

functions, not values

Asymptotic Notation

- **One-sided equality:** $T(n) = O(f(n))$.
  - Not transitive:
    - $f(n) = 5n^3$; $g(n) = 3n^2$
    - $f(n) = O(n^3) = g(n)$
    - but $f(n) \neq g(n)$.
  - Alternative notation: $T(n) \in O(f(n))$.

Set Definition

$O(g(n)) = \{ f(n) :$ there exist constants $c > 0$, $n_0 > 0$ such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0$ $\}$

EXAMPLE: $2n^2 \in O(n^3)$

(*Logicians:* $\lambda n.2n^2 \in O(\lambda n.n^3)$, but it's convenient to be sloppy, as long as we understand what's *really* going on.)

Examples

- $10^6\, n^3 + 2n^2 - n + 10 = O(n^3)$

- $n^{1/2} + \log n = O(n^{1/2})$

- $n\,(\log n + \sqrt{n}) = O(n^{3/2})$

- $n = O(n^2)$

Ω-notation (lower bounds)

O-notation is an *upper-bound* notation. It makes no sense to say $f(n)$ is at least $O(n^2)$.

$\Omega(g(n)) = \{ f(n)$: there exist constants $c > 0, n_0 > 0$ such that $0 \le cg(n) \le f(n)$ for all $n \ge n_0 \}$

EXAMPLE: $\sqrt{n} = \Omega(\lg n)$ $(c = 1, n_0 = 16)$

Ω-notation (lower bounds)

- **Be careful:** "Any comparison-based sorting algorithm requires at least O(n log n) comparisons."
  - Meaningless!
  - Use Ω for lower bounds.

Θ-notation (tight bounds)

$$\Theta(g(n)) = O(g(n)) \;\cap\; \Omega(g(n))$$

EXAMPLE: $\frac{1}{2}n^2 - 2n = \Theta(n^2)$

Polynomials are simple:

$$a_d n^d + a_{d-1}n^{d-1} + \cdots + a_1 n + a_0 = \Theta(n^d)$$

o-notation and ω-notation

O-notation and Ω-notation are like $\leq$ and $\geq$.
o-notation and ω-notation are like $<$ and $>$.

$o(g(n)) = \{ f(n) :$ for **every** constant $c > 0$,
there is a constant $n_0 > 0$
such that $0 \leq f(n) < cg(n)$
for all $n \geq n_0 \}$

EXAMPLE: $2n^2 = o(n^3)$ $(n_0 = 2/c)$

Overview of Asymptotic Notation

Notation	… means …	Think…	E.g.	Lim $f(n)/g(n)$
$f(n)=O(n)$	$\exists\, c > 0, n_0 > 0$ $\forall\, n > n_0$: $0 \le f(n) < cg(n)$	Upper bound	$100n^2$ $= O(n^3)$	If it exists, it is $< \infty$
$f(n)=\Omega(g(n))$	$\exists c>0,\ n_0>0,\ \forall n > n_0$: $0 \le cg(n) < f(n)$	Lower bound	2^n $= \Omega(n^{100})$	If it exists, it is > 0
$f(n)=\Theta(g(n))$	both of the above: $f=\Omega(g)$ **and** $f = O(g)$	Tight bound	$\log(n!)$ $= \Theta(n \log n)$	If it exists, it is > 0 and $<\infty$
$f(n)=o(g(n))$	$\forall c>0,\ \exists n_0>0,\ \forall n > n_0$: $0 \le f(n) < cg(n)$	Strict upper bound	$n^2 = o(2^n)$	Limit exists, $=0$
$f(n)=\omega(g(n))$	$\forall c>0,\ \exists n_0>0,\ \forall n > n_0$: $0 \le cg(n) < f(n)$	Strict lower bound	n^2 $= \omega(\log n)$	Limit exists, $=\infty$

Common Functions: Asymptotic Bounds

- **Polynomials.** $a_0 + a_1 n + \cdots + a_d n^d$ is $\Theta(n^d)$ if $a_d > 0$.
- **Polynomial time.** Running time is $O(n^d)$ for some constant d independent of the input size n.
- **Logarithms.** $\log_a n = \Theta(\log_b n)$ for all constants a,b > 0.

can avoid specifying the base

log grows slower than every polynomial

For every $x > 0$, $\log n = o(n^x)$.

Every polynomial grows slower than every exponential

- **Exponentials.** For all $r > 1$ and all d > 0, $n^d = o(r^n)$.
- **Factorial.** By Sterling's formula,

$$n! = (\sqrt{2\pi n})\left(\frac{n}{e}\right)^n (1 + o(1)) = 2^{\Theta(n \log n)}$$

grows faster than every exponential