Data Structures and Algorithms

Spring 2009-2010

Outline

When the Inputs Get Bigger...

- Chief concern in algorithm analysis is determining the underlying running time and space requirements in the *long term*, when the input size grows larger
- Running-time is highly dependent on machine architecture, so analysis cannot be machine-specific
- To do this we introduce the *order* notation that describes the time and storage requirements only in terms of the algorithm and its parameters

Order of Functions

- Which function is bigger, $f(n) = 20 \cdot n$ or $g(n) = 0.002 \cdot n^2$?
- Although $f(n)$ is larger initially, the n^2 of $g(n)$ dominates *eventually*
- To help us quantify functions and to make the notion of comparison more precise we introduce four definitions

$O(n)$ – Big-Oh

Definition

$T(n) = O(f(n))$ if there are *constants* c and n_0 so that
$T(n) \leq cf(n)$ when $n \geq n_0$

$T(n)$ is "less than or equal to" $f(n)$

That is, once $n \geq n_0$, $T(n)$ is always less than or equal to some constant, c, times $f(n)$.

$\Omega(n)$ – Big-Omega

Definition

$T(n) = \Omega(f(n))$ if there are *constants* c and n_0 so that $T(n) \geq cf(n)$ when $n \geq n_0$

$T(n)$ is "greater than or equal to" $g(n)$

Therefore, if $f(n) = O(g(n))$ then $g(n) = \Omega(f(n))$

$\Theta(n)$ – Big-Theta

Definition

$T(n) = \Theta(h(n))$ if and only if $T(n) = O(h(n))$ and $T(n) = \Omega(h(n))$

$T(n)$ is "behaves like" $h(n)$;
Alternative view of $\Theta(h(n))$:

$$\lim_{n \to \infty} \frac{T(n)}{h(n)} \to C$$

$o(n)$ – Little-oh

Definition

$T(n) = o(p(n))$ if $T(n) = O(p(n))$ and $T(n) \neq \Theta(p(n))$

$T(n)$ is "is smaller than" $p(n)$;

Alternative view of $o(p(n))$:

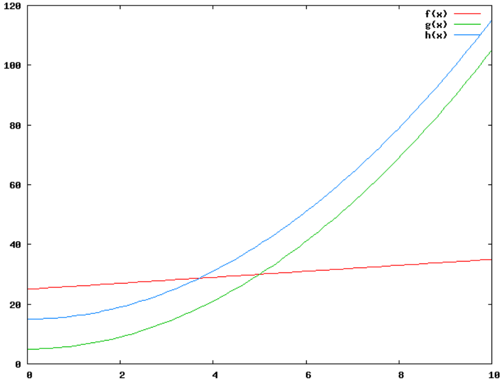
$$\lim_{n \to \infty} \frac{T(n)}{p(n)} \to 0$$

Function Comparisons

Let

- $f(n) = n + 25$
- $g(n) = n^2 + 5$, and
- $h(n) = n^2 + 15$

Then, we can say:

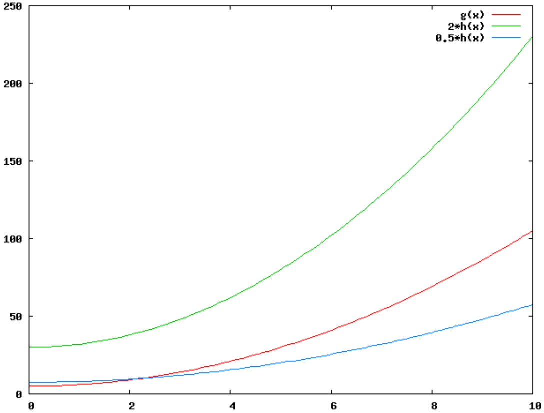


- $f(n) = O(g(n))$ and $f(n) = O(h(n))$
- $g(n) = \Omega(f(n))$ and $h(n) = \Omega(f(n))$
- $g(n) = \Theta(h(n))$ and therefore, $h(n) = \Theta(g(n))$
- $f(n) = o(g(n))$

More on $\Theta(n)$

With

- $g(n) = n^2 + 5$
- $h(n) = n^2 + 15$
- $2 \cdot h(n)$
- $\frac{1}{2} \cdot h(n)$
- so we can say that $g(n) = \Theta(h(n))$



For $n > 0$, $g(n) \leq 2 \cdot h(n)$, and so $g(n) = O(h(n))$ with $n_0 = 0$ and $c = 2$;
similarly, for $n \geq 3$, $g(n) \geq 0.5 \cdot h(n)$, and so $g(n) = \Omega(h(n))$ with $n_0 = 3$ and $c = 0.5$;

Even more on $\Theta(n)$

Likewise, from below we can argue that $g(n) = \Theta(n^2)$

- $g(n) = n^2 + 5$
- $h(n) = n^2$
- $2 \cdot h(n)$
- $\frac{1}{2} \cdot h(n)$
- so we can say that $g(n) = \Theta(n^2)$