Discrete Mathematics
CS 2610

February 26, 2009 -- part 1

Big-O Notation

- Big-O notation is used to express the time complexity of an algorithm
  - We can assume that any operation requires the same amount of time.
  - The time complexity of an algorithm can be described independently of the software and hardware used to implement the algorithm.

Big-O Notation

Def.: Let f, g be functions with domain $\mathbf{R}_{\geq 0}$ or $\mathbf{N}$ and codomain $\mathbf{R}$.

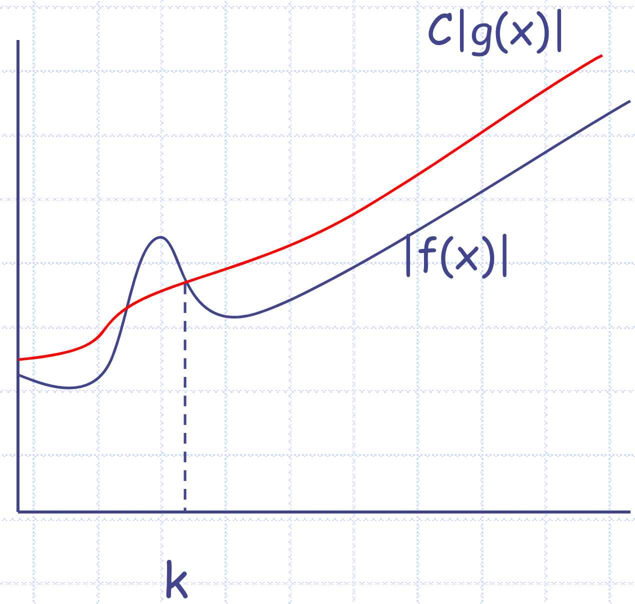
f(x) is O(g(x)) if there are constants ***C*** and ***k*** st

$$\forall x > k, |f(x)| \leq C \cdot |g(x)|$$

$f(x)$ is asymptotically dominated by $g(x)$

C|g(x)| is an upper bound of f(x).

C and k are called witnesses to the relationship between f & g.



Big-O Notation

- To prove that a function $f(x)$ is $O(g(x))$
  - **Find values for k and C, not necessarily the smallest one, larger values also work!!**
  - **It is sufficient to find a certain k and C that works**
  - **In many cases, for all $x \geq 0$,**

    if $f(x) \geq 0$ then $|f(x)| = f(x)$

Example: $f(x) = x^2 + 2x + 1$ is $O(x^2)$ for $C = 4$ and $k = 1$

Big-O Notation

Show that $f(x) = x^2 + 2x + 1$ is $O(x^2)$.

When $x > 1$ we know that $x \leq x^2$ and $1 \leq x^2$

then $0 \leq x^2 + 2x + 1 \leq x^2 + 2x^2 + x^2 = 4x^2$

so, let $C = 4$ and $k = 1$ as witnesses, i.e.,

$f(x) = x^2 + 2x + 1 < 4x^2$ when $x > 1$

Could try $x > 2$. Then we have $2x \leq x^2$ & $1 \leq x^2$

then $0 \leq x^2 + 2x + 1 \leq x^2 + x^2 + x^2 = 3x^2$

so, $C = 3$ and $k = 2$ are also witnesses to $f(x)$ being $O(x^2)$. Note that $f(x)$ is also $O(x^3)$, etc.

Big-O Notation

Show that $f(x) = 7x^2$ is $O(x^3)$.

When $x > 7$ we know that $7x^2 < x^3$ (multiply $x > 7$ by x^2)
so, let $C = 1$ and $k = 7$ as witnesses.

Could try $x > 1$. Then we have $7x^2 < 7x^3$
so, $C = 7$ and $k = 1$ are also witnesses to $f(x)$
being $O(x^3)$. Note that $f(x)$ is also $O(x^4)$, etc.

Big-O Notation

Show that $f(n) = n^2$ is not $O(n)$.

Show that no pair of C and k exists such that

$$n^2 \le Cn \text{ whenever } n > k.$$

When $n > 0$, divide both sides of $n^2 \le Cn$ by n to get $n \le C$. No matter what C and k are, $n \le C$ will not hold for all n with $n > k$.

Big-O Notation

- Observe that $g(x) = x^2$ is $O(x^2 + 2x + 1)$

Def: Two functions $f(x)$ and $g(x)$ have the same order iff $g(x)$ is $O(f(x))$ and $f(x)$ is $O(g(x))$

Big-O Notation

- Also, the function $f(x) = 3x^2 + 2x + 3$ is $O(x^3)$
- What about $O(x^4)$?
- In fact, the function $Cg(x)$ is an upper bound for $f(x)$, but not necessarily the tightest bound.
  - When Big-O notation is used, $g(x)$ is chosen to be as small as possible.

Big-Oh - Theorem

Theorem: If $f(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0$ where $a_i \in R$, $i=0,\ldots n$; then $f(x)$ is $O(x^n)$. Leading term dominates!

Proof: if $x > 1$ we have

$$|f(x)| = |a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0|$$
$$\leq |a_n| x^n + |a_{n-1}| x^{n-1} + \ldots + |a_1| x + |a_0|$$
$$= x^n(|a_n| + |a_{n-1}|/x + \ldots + |a_1|/x^{n-1} + |a_0|/x^n)$$
$$\leq x^n(|a_n| + |a_{n-1}| + \ldots + |a_1| + |a_0|)$$

So, $|f(x)| \leq Cx^n$ where $C = |a_n| + |a_{n-1}| + \ldots + |a_1| + |a_0|$ whenever $x > 1$ (what's k? k = 1, why?)

What's this: $|a + b| \leq |a| + |b|$

Big-O

Example: Prove that $f(n) = n!$ is $O(n^n)$

Proof (easy): $n! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 \cdots n$

$\leq n \cdot n \cdot n \cdot n \cdot n \cdots n$

$= n^n$

where our witnesses are $C = 1$ and $k = 1$

Example: Prove that $\log(n!)$ is $O(n\log n)$

Using the above, take the log of both sides:

$\log(n!) \leq \log(n^n)$ which is equal to $n \log(n)$

Big-O

- Lemma: A constant function is O(1).

Proof: Left to the viewer ☺

- The most common functions used to estimate the time complexity of an algorithm. (in increasing O() order):

$1, \ (\log n), \ n, \ (n \log n), \ n^2, \ n^3, \ \ldots \ 2^n, \ n!$

Big-O Properties

- **Transitivity:** if f is $O(g)$ and g is $O(h)$ then f is $O(h)$
- **Sum Rule**:
  - *If* f_1 is $O(g_1)$ and f_2 is $O(g_2)$ then f_1+f_2 is $O(\max(|g_1|,|g_2|))$
  - *If* f_1 is $O(g)$ and f_2 is $O(g)$ then f_1+f_2 is $O(g)$
- **Product Rule**
  - *If* f_1 is $O(g_1)$ and f_2 is $O(g_2)$ then $f_1 f_2$ is $O(g_1 g_2)$
- For all $c > 0$, $O(cf)$, $O(f + c)$, $O(f - c)$ are $O(f)$

Big-O – Properties Example

Example: Give a big-O estimate for $3n \log(n!) + (n^2+3)\log n$, $n>0$

1) For $3n \log(n!)$ we know $\log(n!)$ is $O(n\log n)$ and $3n$ is $O(n)$ so we know $3n \log(n!)$ is $O(n^2\log n)$

2) For $(n^2+3)\log n$ we have $(n^2+3) < 2n^2$ when $n > 2$ so it's $O(n^2)$; and $(n^2+3)\log n$ is $O(n^2\log n)$

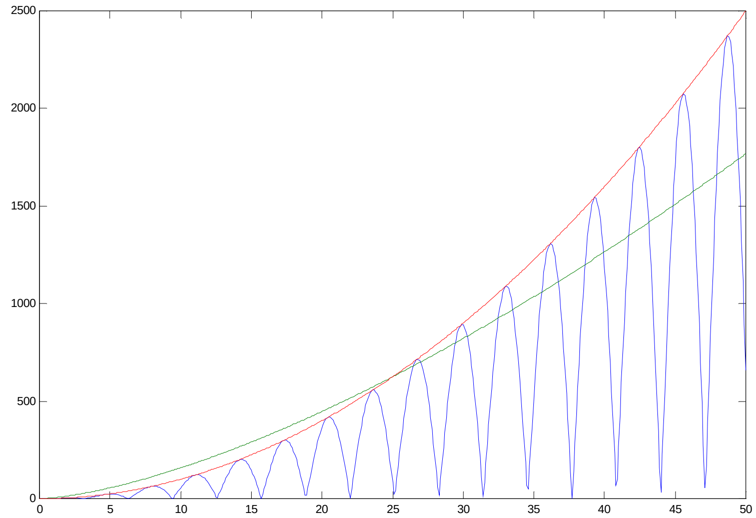
3) Finally we have an estimate for $3n \log(n!) + (n^2+3)\log n$ that is: $O(n^2\log n)$

Big-O Notation

Def.:Functions f and g are incomparable, if f(x) is not O(g) and g is not O(f).

f: $R^+ \rightarrow R$, $f(x) = 5\,x^{1.5}$

g: $R^+ \rightarrow R$, $g(x) = |x^2 \sin x|$



-- $5\,x^{1.5}$

-- $|x^2 \sin x|$

-- x^2

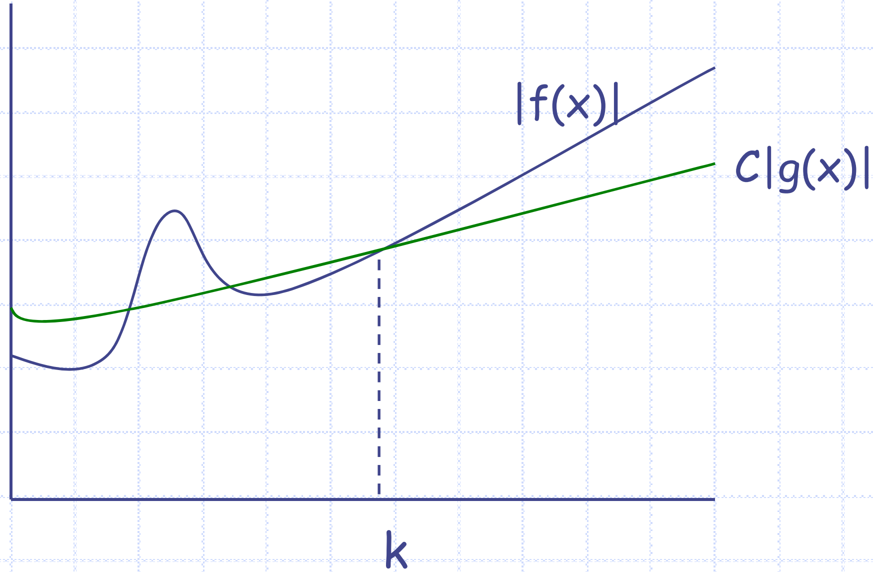
Big-Omega Notation

Def.: Let *f, g* be functions with domain $R_{\geq 0}$ or N and codomain R.

f(x) is Ω(g(x)) if there are **positive constants** ***C*** **and** ***k*** such that

$$\forall x > k,\ C \cdot |g(x)| \leq |f(x)|$$

- C · |g(x)| is **a lower bound** for |f(x)|



Big-Omega Property

Theorem: **$f(x)$ is $\Omega(g(x))$ iff $g(x)$ is $O(f(x))$.**

Is this trivial or what?

Big-Omega Property

Example: prove that $f(x) = 3x^2 + 2x + 3$ is $\Omega(g(x))$
where $g(x) = x^2$

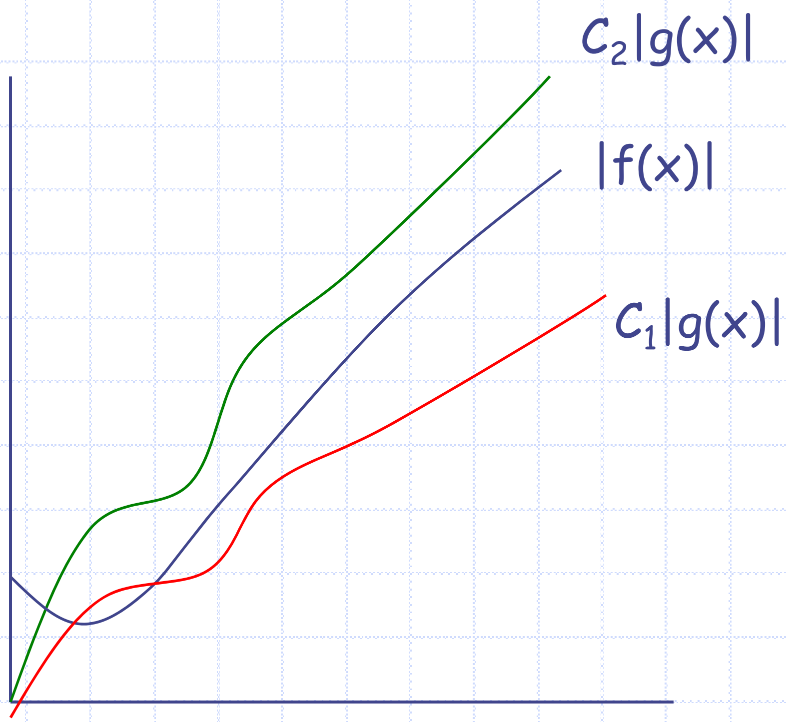
Proof: first note that $3x^2 + 2x + 3 \geq 3x^2$ for all $x \geq 0$.

That's the same as saying that
$g(x) = x^2$ is $O(3x^2 + 2x + 3)$

Big-Theta Notation

Def.:Let f, g be functions with domain $\mathbf{R}_{\geq 0}$ or $\mathbf{N}$ and codomain $\mathbf{R}$.

$$f(x) \text{ is } \Theta(g(x)) \text{ if } f(x) \text{ is } O(g(x)) \text{ and } f(x) \text{ is } \Omega(g(x)).$$



Big-Theta Notation

When $f(x)$ is $\Theta(g(x))$, we know that $g(x)$ is $\Theta(f(x))$.

Also, $f(x)$ is $\Theta(g(x))$ iff

$f(x)$ is $O(g(x))$ and

$g(x)$ is $O(f(x))$.

Typical g functions: x^n, c^x, $\log x$, etc.

Big-Theta Notation

- To prove that f(x) is order g(x)
  - Method 1
    - Prove that f is O(g(x))
    - Prove that f is Ω(g(x))
  - Method 2
    - Prove that f is O(g(x))
    - Prove that g is O(f(x))

Big-Theta Example

show that $3x^2 + 8x \log x$ is $\Theta(x^2)$ (or order x^2)

$0 \le 8x \log x \le 8x^2$ so $3x^2 + 8x \log x \le 11x^2$ for $x > 1$.

So, $3x^2 + 8x \log x$ is $O(x^2)$ (can I get a witness?)

Is x^2 $O(3x^2 + 8x \log x)$? You betcha! Why?

Therefore, $3x^2 + 8x \log x$ is $\Theta(x^2)$

Big Summary

Upper Bound – Use Big-Oh

Lower Bound – Use Big-Omega

Upper and Lower (or Order of Growth) –

Use Big-Theta

Time to Shift Gears Again

Number Theory

Number Theory

Livin' Large