CSE 101 10-29-20

Algorithm Efficiency & Runtime

how do we measure runtime?
not in seconds! we count # of executions of some basic operation.
(also called Barometer operations)

Ex A(n)

```
for i = 1 to n
    for j = 1 to n
        Print('blah') ← basic op.
```

(# Print stmts) = n^2 ← runtime in # of basic ops. on input of size n.

Ex. $R(n)$

```
for i = 1 to n
    for j = 1 to i
        print('blah')  ← basic op.
```

$$(\#\text{print stmts}) = 1 + 2 + 3 + \cdots + n$$

$$= \frac{n(n+1)}{2} = \frac{1}{2}n^2 + \frac{1}{2}n$$

note:

(1) the lower order term $\frac{1}{2}n$ in $R(n)$ is negligible for large values of n.

$$\lim_{n\to\infty}\left(\frac{\frac{1}{2}n^2 + \frac{1}{2}n}{n^2}\right) = \lim_{n\to\infty}\left(\frac{1}{2} + \frac{1}{2n}\right) = \frac{1}{2}$$

$\frac{1}{2n} \to 0$

Thus we might as well consider $\frac{1}{2}n^2 + \frac{1}{2}n$ to be $\frac{1}{2}n^2$, i.e. ignore lower order term.

(2) now compare A: n^2 to B: $\frac{1}{2}n^2$

we seek a measure of runtime that is independent of computing device. since A & B can be equalized by running A on a machine that's 2x faster, we don't distinguish between A and B.

Measure of runtime is called:

Asymptotic runtime.

Procedure

- define basic operation(s)
- determine # of basic OPs executed on input of size n, getting a function $f(n)$.
- determine asymptotic growth rate of $f(n)$

Informal:

(1) drop lower order terms

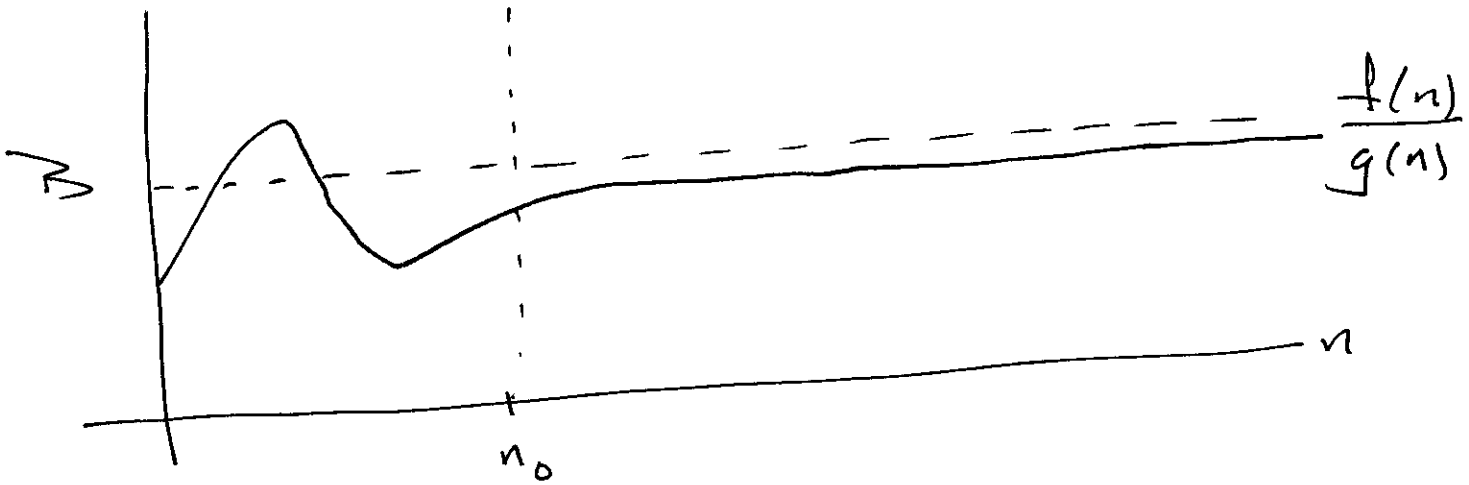
(2) replace coeff. of highest order term by 1.

Asymptotic growth of functions

Defn $f(n) = O(g(n))$ iff

$$0 < \frac{f(n)}{g(n)} \leq R \quad \leftarrow \text{some upper bound}$$

for all sufficiently large n. [i.e. There exists $R > 0$ and there exists $n_0 > 0$ such that for all $n \geq n_0$: $0 < \frac{f(n)}{g(n)} \leq R$.]



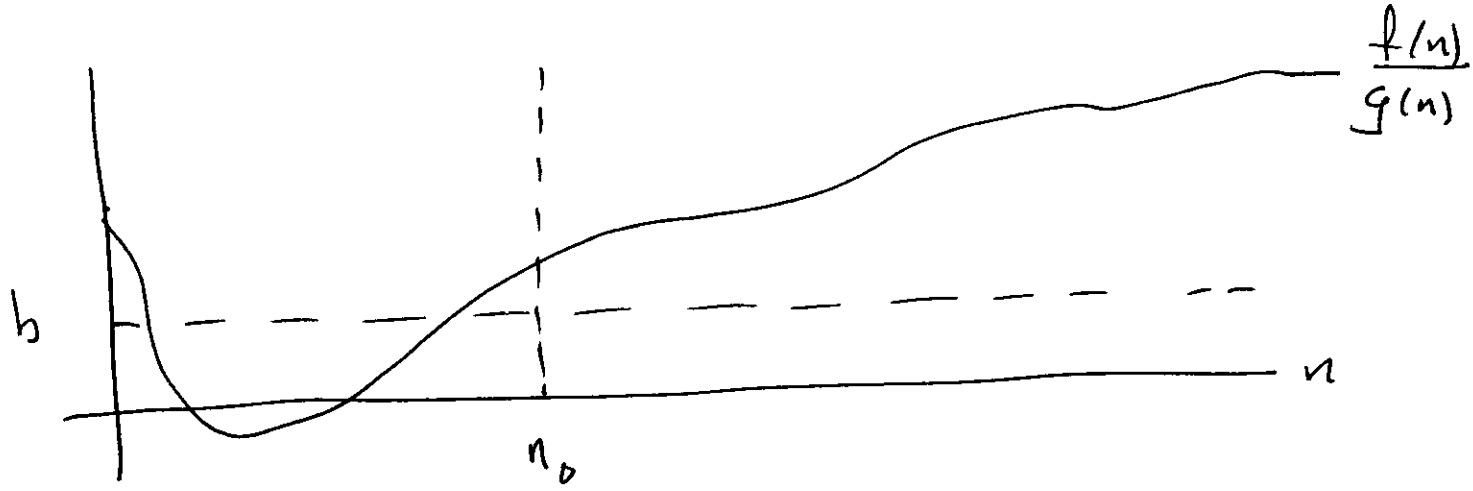
Say: $g(n)$ is asymp. u.b. for $f(n)$

Def $f(n) = \Omega(g(n))$ iff

$$0 < b \leq \frac{f(n)}{g(n)}$$

↑ some lower bound.

for all sufficiently large n.

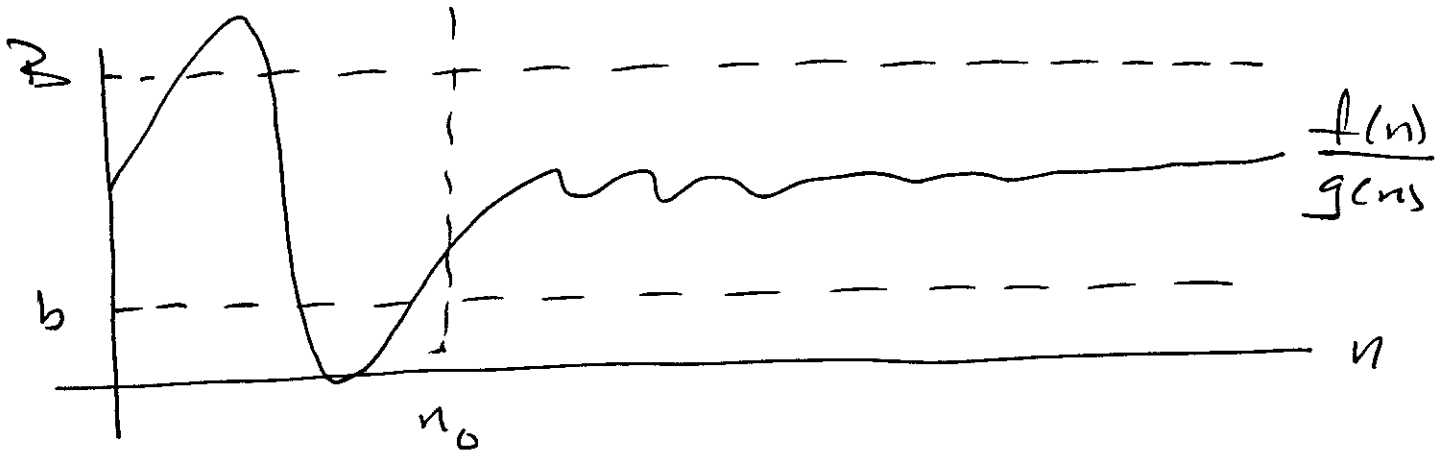


we say: $g(n)$ is an asym. l.b. for $f(n)$

Defn $f(n) = \Theta(g(n))$ iff

$$0 < b \leq \frac{f(n)}{g(n)} \leq R$$

for all suff. large n.



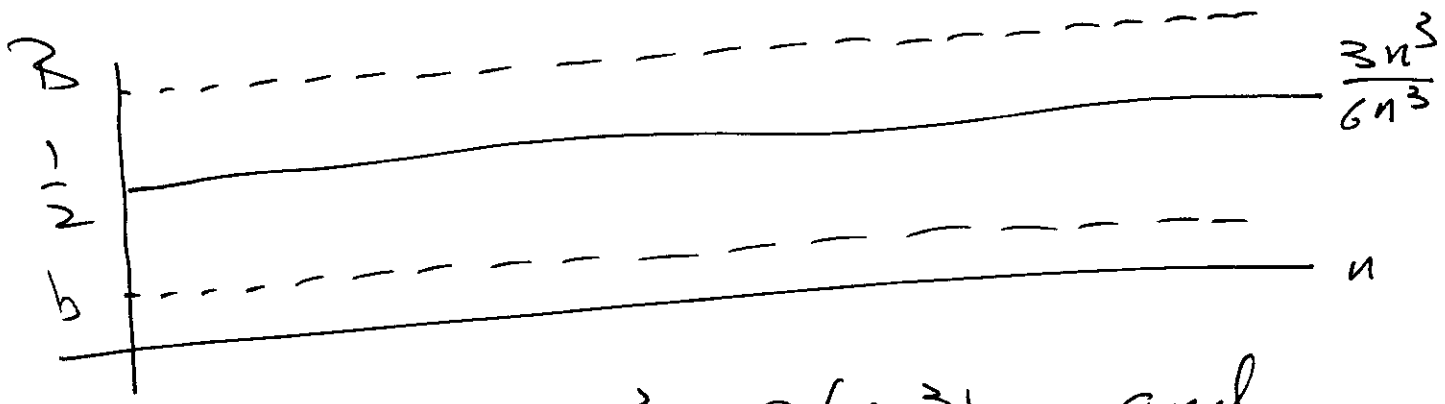
we say: $g(n)$ is a tight asymp. bound for $f(n)$

Ex. $n^3 = O(n^4)$. Why?

$$\frac{n^3}{n^4} = \frac{1}{n} \longrightarrow 0 \quad \text{as } n \to \infty$$

Ex. $3n^3 = \Theta(6n^3)$

$$\frac{3n^3}{6n^3} = \frac{1}{2} \longrightarrow \frac{1}{2} \quad \text{as } n \to \infty$$

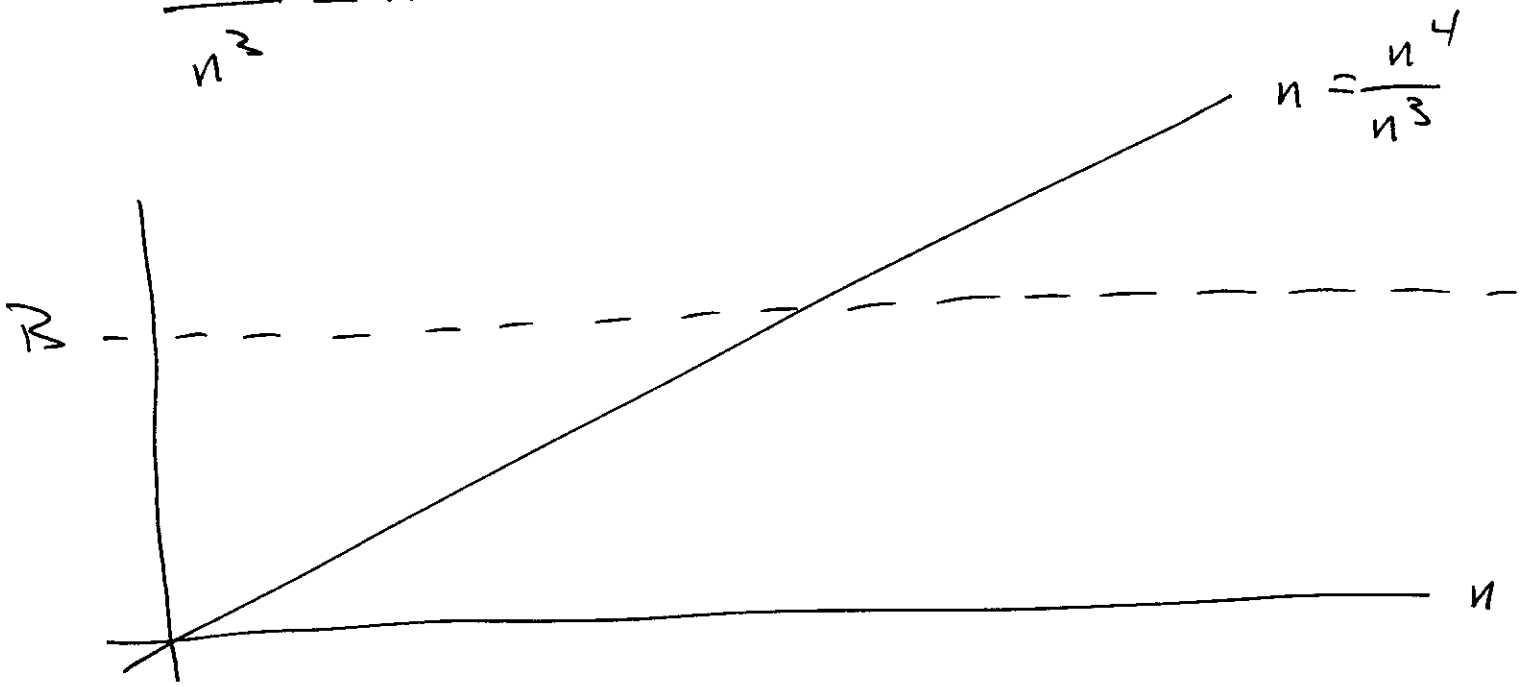


also have: $3n^3 = O(6n^3)$, and

$3n^3 = \Omega(6n^3)$

Ex. $n^4 \neq O(n^3)$

$$\frac{n^4}{n^3} = n \longrightarrow \infty \quad \text{as} \quad n \to \infty$$

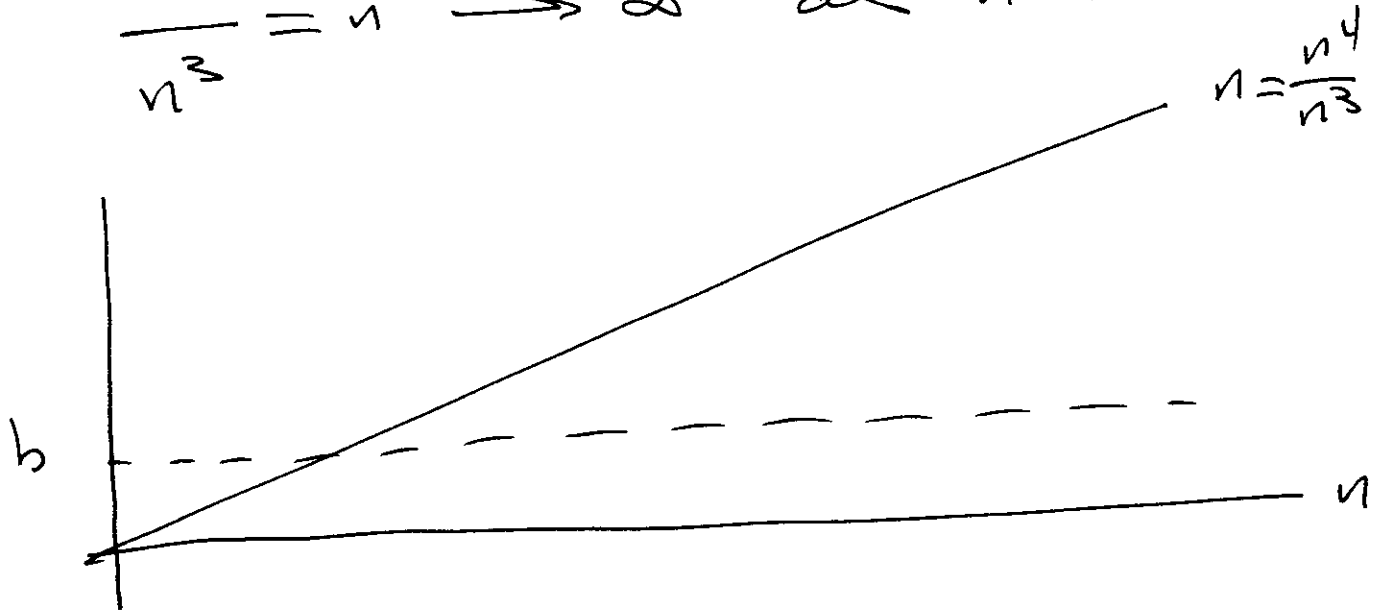


stmt. $0 < \frac{n^4}{n^3} \leq R$ n false

for <u>every</u> $R > 0$, and suff. large n.

Ex. $n^4 = \Omega(n^3)$

$\frac{n^4}{n^3} = n \to \infty$ as $n \to \infty$



Thm

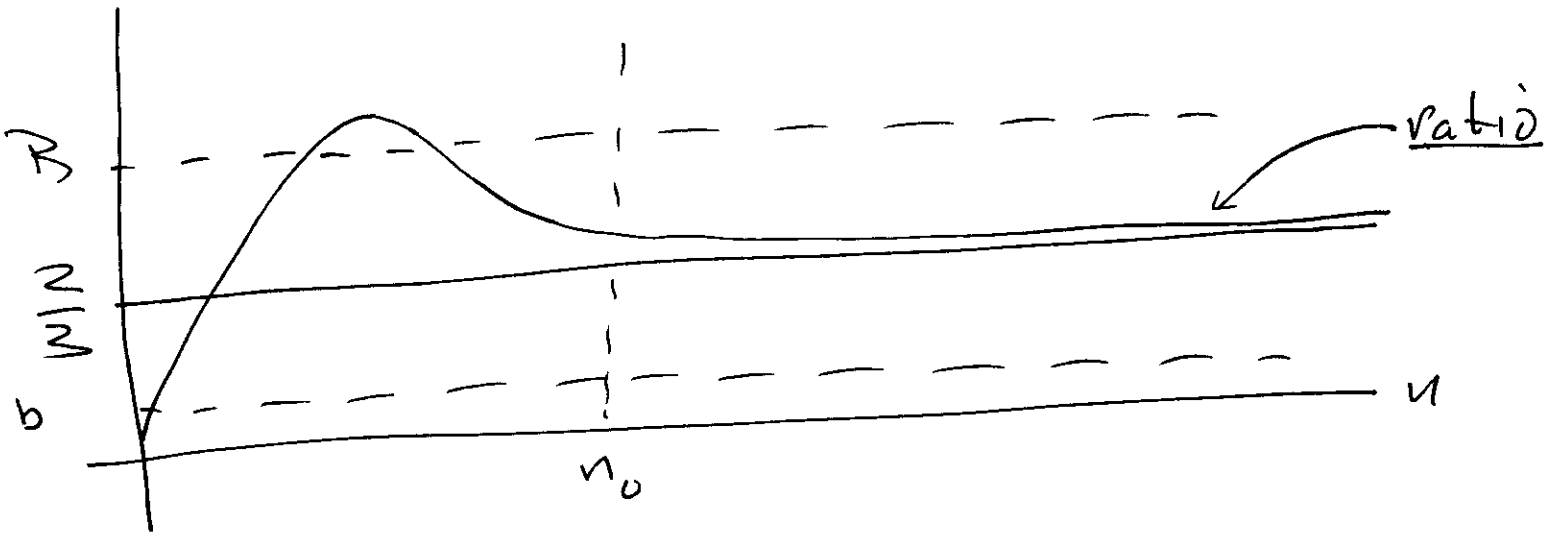
- $f(n) = O(g(n))$ iff $g(n) = \Omega(f(n))$
- $f(n) = \Theta(g(n))$ iff both

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

$f(n) = \Omega(g(n))$

Ex. $2n^2+5n-1 = \Theta(3n^2-6n+4)$

$$\frac{2n^2+5n-1}{3n^2-6n+4} = \frac{2+\frac{5}{n}-\frac{1}{n^2}}{3-\frac{6}{n}+\frac{4}{n^2}} \longrightarrow \frac{2}{3} \quad \text{as } n \to \infty$$



next time:

- analogy
- $o(g(n))$
- $\omega(g(n))$