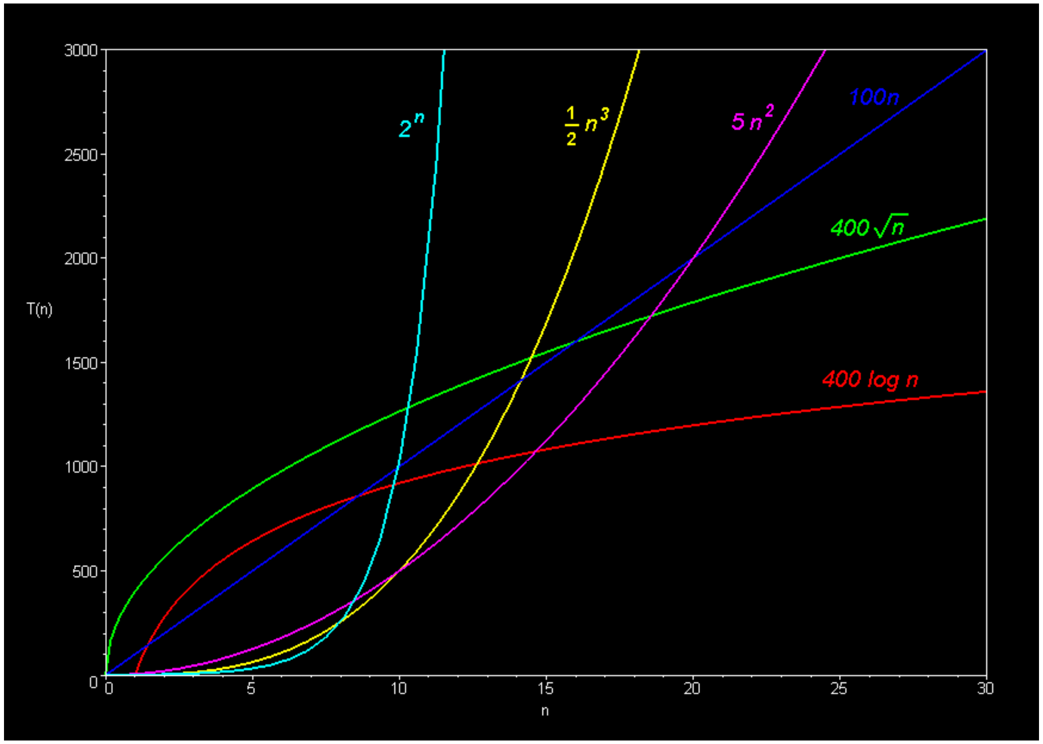
Data Structures and Algorithm Analysis (CSC317)



Week 2: Growth of Functions

Picture from
http://science.slc.edu/~jmarshall/courses/2002/spring/cs50/BigO/

Growth of functions

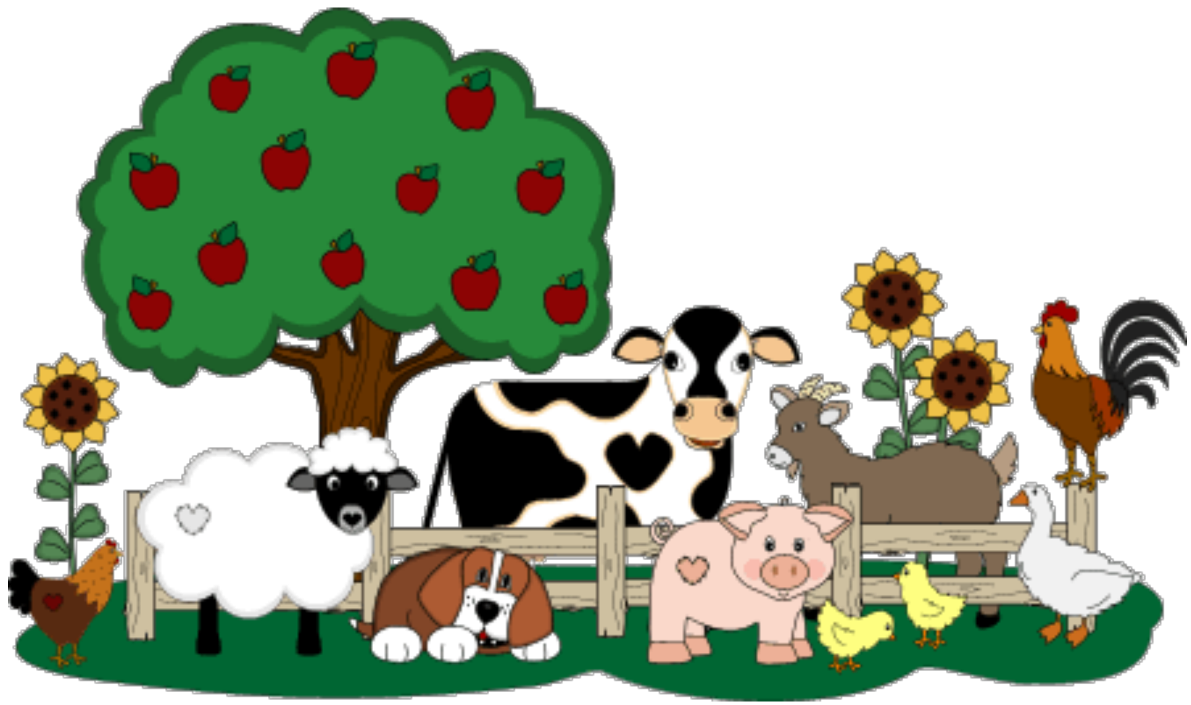
We've already been talking about "Grows as" for the sort examples, but what does this really mean?

We already know that:

- We ignore constants and low order terms; why?

- Asymptotic analysis: we focus on large input size; growth of function for large input; why do we care?

Complexity petting zoo



This is a petting zoo, because there are many more complexity classes, and we are only exploring the surface…

Complexity petting zoo (see notes of prof Burt Rosenberg: http://blog.cs.miami.edu/burt/2014/09/01/a-complexity-petting-zoo/)

Complexity classes

Constant time $T(1)$

Example? First number in an array
Also second number…

Complexity classes

$T(\log n)$

Example?

Binary search: Sorted array A; find value v between range low and high

Middle value

A = [1 3 4 10 15 23 35 40 45]
Find v=4

Solution: Search in middle of array:
value found, or recursion left side, or recursion right half

Growth of functions

$$T(n)$$

Example?
Largest number in sequence
Sum of fixed sequence
Whenever you step through entire sequence or array
Even if you have to do this 20 times

Complexity classes

$T(n \log n)$

Example?
We've seen; merge sort…

Growth of functions

$$T(n^2)$$

Example?
We've seen; insertion sort…

Complexity classes

$$T(n^3)$$

Example?
Naïve matrix multiplication (for an n by n matrix) is classical example; we shall see more later…

Complexity classes

All of these are polynomial time (class P)

$$T(n); T(n \log n); T(n^2); T(n^3)$$

$T(n^k)$ K nonnegative

Complexity classes

More than polynomial time? Exponential

$$T(2^n)$$

Complexity classes

What about this problem: subset sum problem?
How long to find a solution??

Input: set of integers size n
Output: is there a subset that sums to 0?

A={1; 4; -3; 2; 9; 7}
Is there a subset that sums to 0?

Complexity classes

What about this problem: subset sum problem?
How long to find a solution??

Input: set of integers size n
Output: is there a subset that sums to 0?

A={1; 4; -3; 2; 9; 7}
Is there a subset that sums to 0?

Might take exponential time if we have to go through every possible subset (brute force)

Complexity classes

What about this problem: subset sum problem?

Input: set of integers size n
Output: is there a subset that sums to 0?

A={1; 4; -3; 2; 9; 7}
Is there a subset that sums to 0?

What about if I hand you a subset:
{1; -3; 2}
How long to verify if this sums to 0?

Complexity classes

What about this problem: subset sum problem?

Input: set of integers size n
Output: is there a subset that sums to 0?

A={1; 4; -3; 2; 9; 7}
Is there a subset that sums to 0?

What about if I hand you a subset:
{1; -3; 2}
How long to verify if this sums to 0? Polynomial, linear, time.

Complexity classes

Algorithms that are *verifiable* in polynomial time (good) are called NP class

But might take exponential number to go through every possible input (possibly bad)

Example: Subset sum problem

A={1; 4; -3; 2; 9; 7}
Is there a subset that sums to 0?

{1; -3; 2} is verifiable to sum to 0 quickly

Complexity classes

Class NP = Nondeterministic Polynomial

Algorithms that are verifiable in polynomial time (good)

But might take exponential number to go through every possible input! (possibly bad)

Nondeterministic = random = if I was magically handed solution. Originally from nondeterministic Turing machine

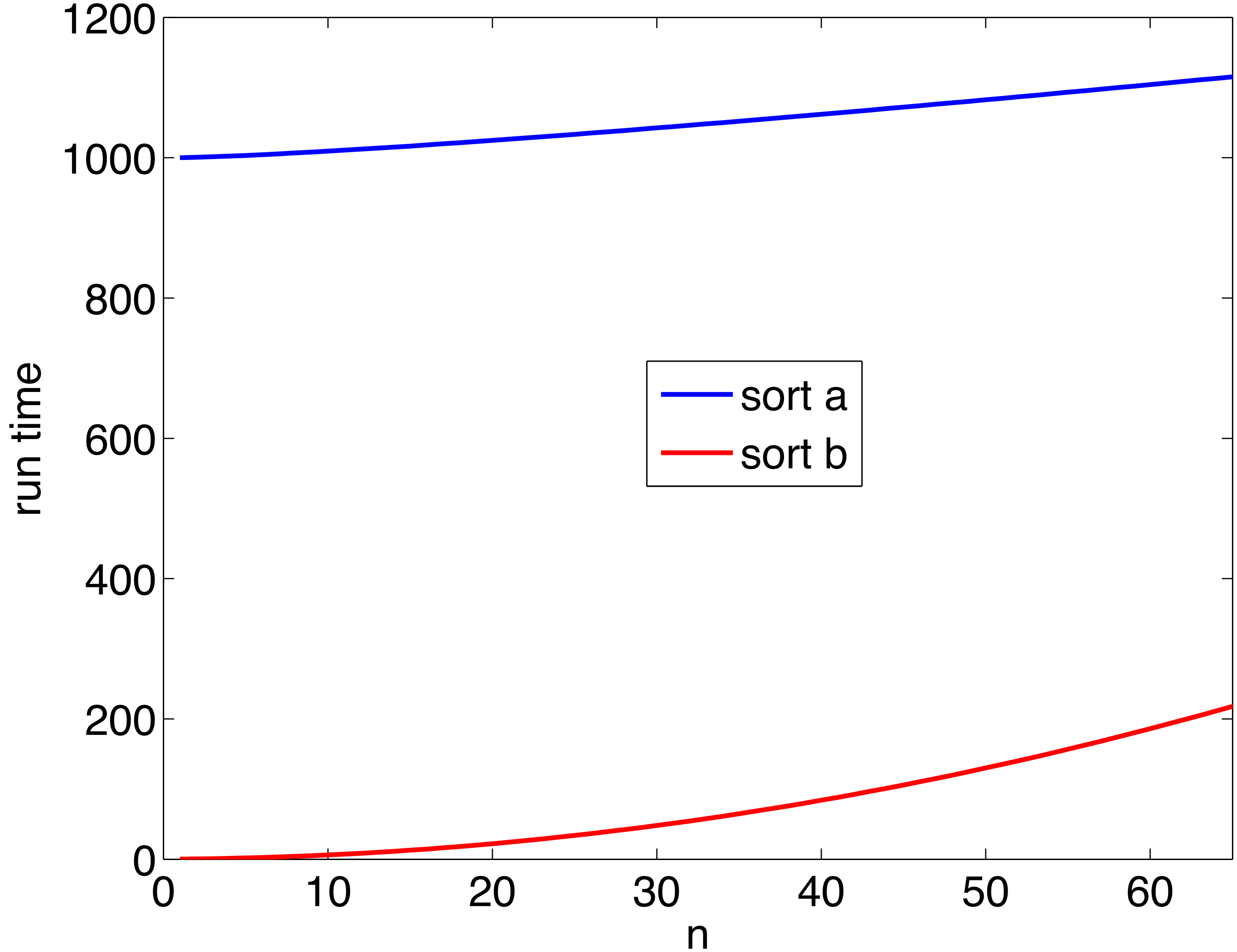
Complexity classes

P = NP ???

Can problem that is quickly verifiable (ie, polynomial time) be quickly solved (ie, polynomial time)?

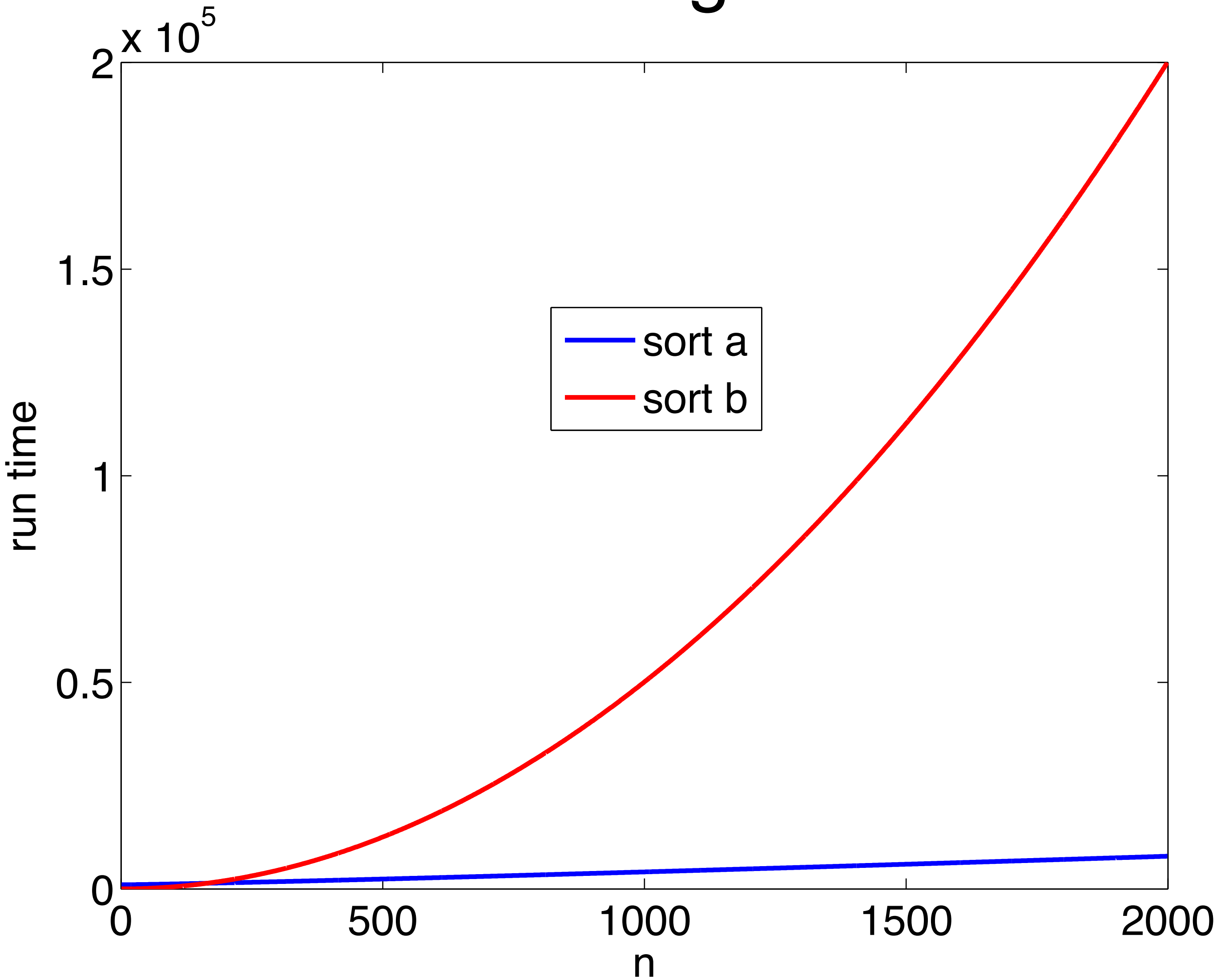
Unknown; Millenium prize problem

Growth of functions & Big Oh



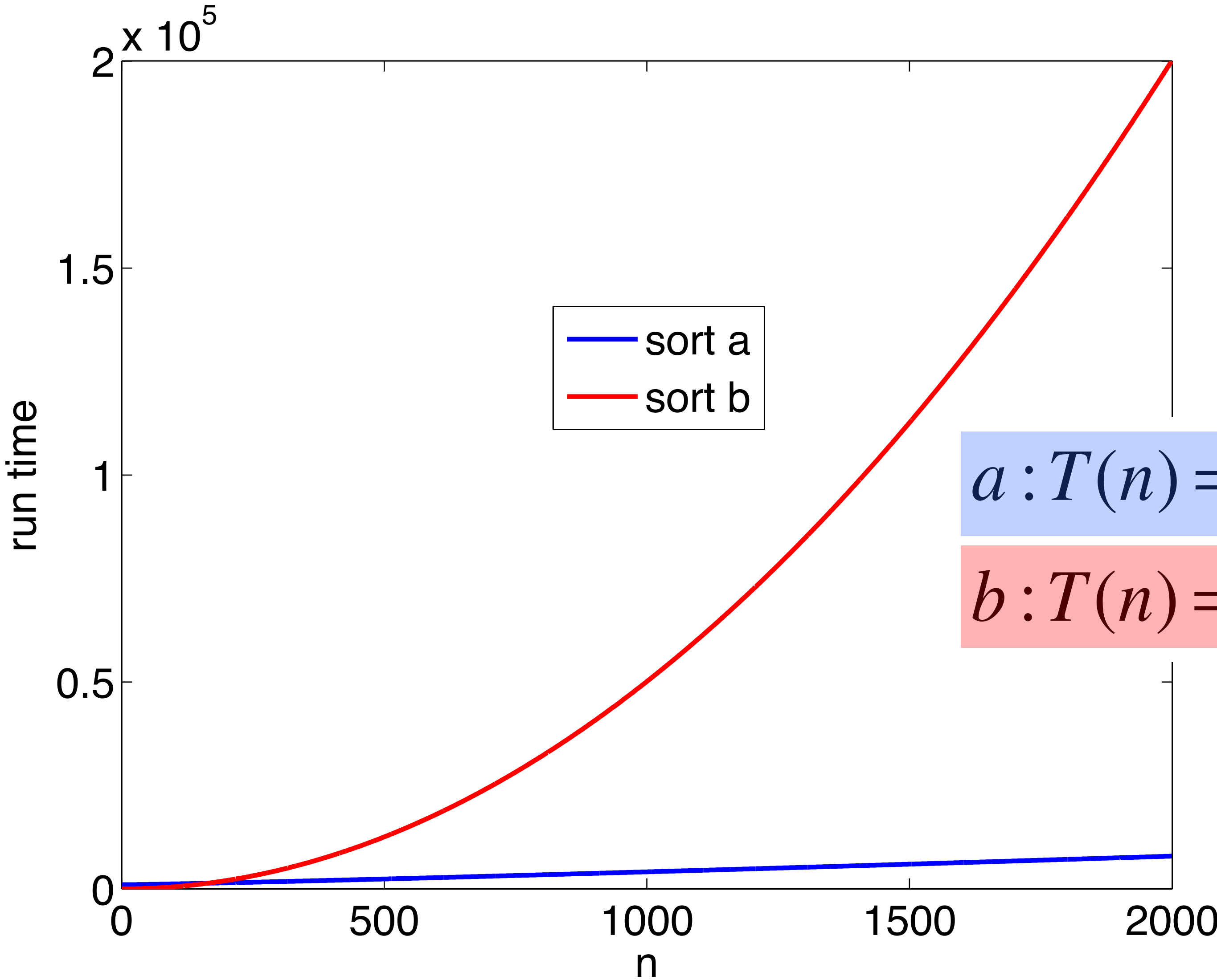
Which sort function is faster?

Growth of functions & Big Oh



Which sort function is faster?

Growth of functions & Big Oh



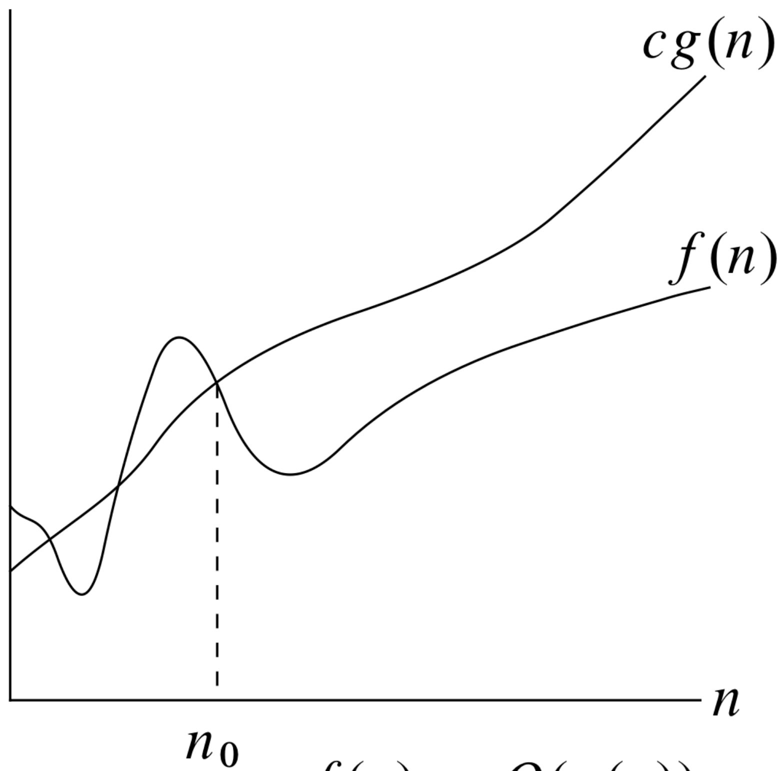
$$a : T(n) = n \log n + 1000$$

$$b : T(n) = 0.2n^2$$

Low asymptotic run time = faster

Big Oh notation

- Asymptotic upper bound; bounded from above by g(n) for large enough n (why do we care?)
- **Definition:** O(g(n))={ f(n): there exist positive constants c and n_0 such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0$}



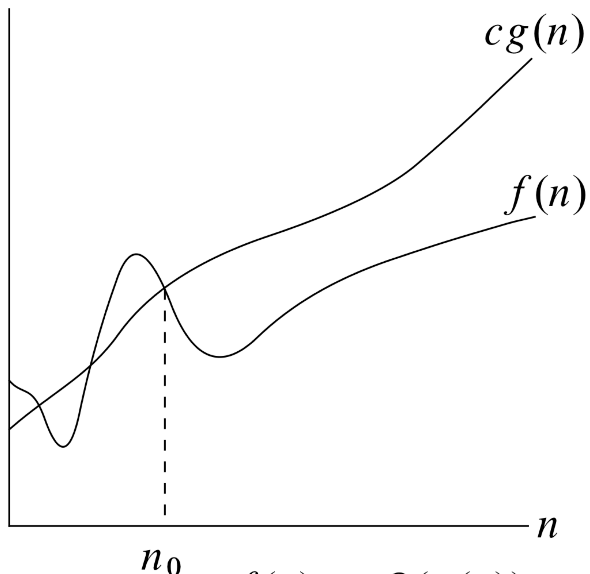
$f(n) = O(g(n))$

Big Oh notation

- Asymptotic upper bound; bounded from above by g(n) for large enough n
- **Definition:** O(g(n))={ f(n): there exist positive constants c and n_0 such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0$}

There exist -> need to find c and n_0

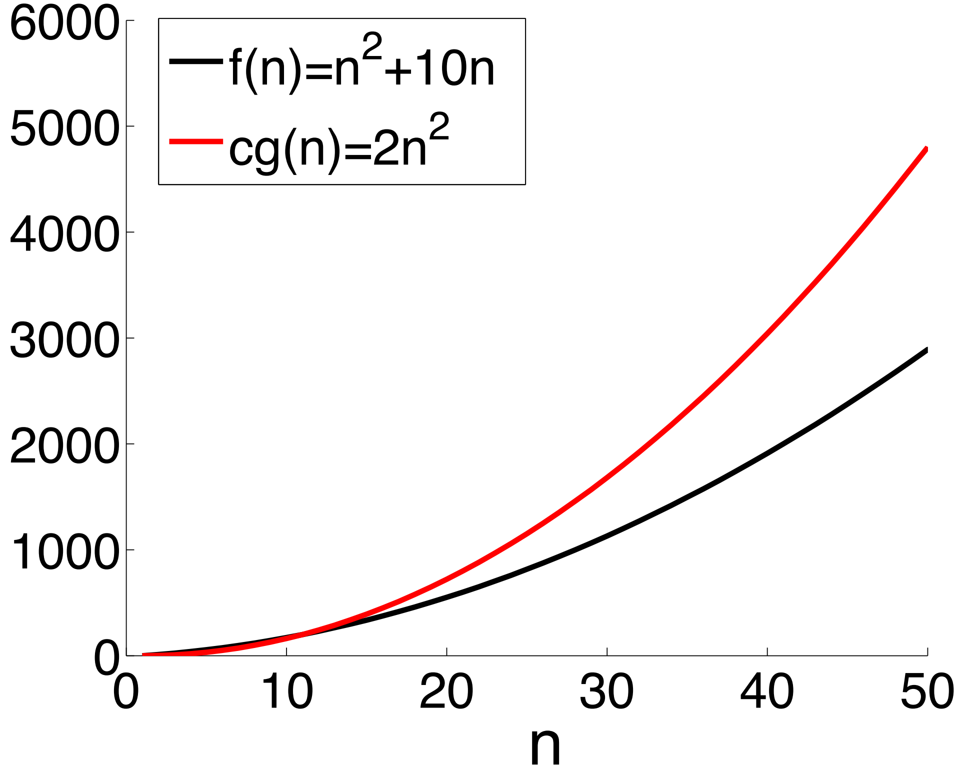
Enough to show one such pair that exists!



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

Big Oh notation

- **Definition:** O(g(n))={ f(n): there exist positive constants c and n_0 such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0$}
- Example: $f(n) = n^2 + 10n$ is $O(n^2)$



Big Oh notation

- **Definition:** O(g(n))={ f(n): there exist positive constants c and n_0 such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0$}
- Example of functions f(n) in $O(n^2)$

$$f(n) = n^2;$$

$$f(n) = n^2 + n$$

$$f(n) = n^2 + 1000n$$

- All bound above asymptotically by n^2
- Intuitively, constants and lower order don’t matter…

Big Oh notation

- **Definition:** O(g(n))={ f(n): there exist positive constants c and n_0 such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0$}
- Example of functions f(n) in $O(n^2)$

$$f(n) = n^2;$$

$$f(n) = n^2 + n$$

$$f(n) = n^2 + 1000n$$

- What about?

$$f(n) = n;$$

Big Oh notation

- **Definition:** O(g(n))={ f(n): there exist positive constants c and n_0 such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0$}

- What about?

$$f(n) = n;$$

Yes. $g(n) = n^2$ is **not a tight upper bound but it's an upper bound.**

$$n \le 1n^2$$

For all $n \ge 1$

Big Oh notation

- **Definition:** O(g(n))={ f(n): there exist positive constants c and n_0 such that $0 \leq f(n) \leq cg(n)$ for all $n \geq n_0$}

- What about?

$$f(n) = n;$$

Yes. $g(n) = n^2$ is **not a tight upper bound but it's an upper bound.**

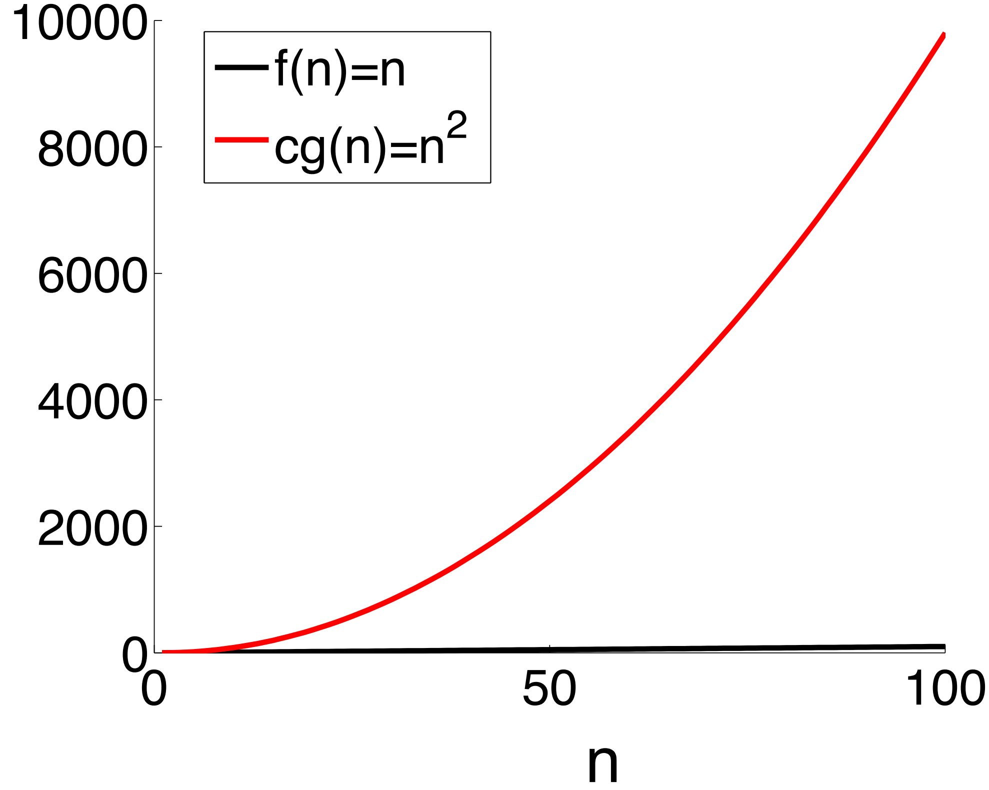
$$n \leq 1n^2$$

For all $n \geq 1$

There thus exists $c = 1; n_0 = 1$ such that $0 \leq f(n) \leq g(n)$

Big Oh notation

- **Definition:** O(g(n))={ f(n): there exist positive constants c and n_0 such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0$}
- Example: $f(n) = n$ is $O(n^2)$



Big Oh notation

- **Example:** $f(n) = a_k n^k + .. + a_1 n^1 + a_0$

  Then

  $f(n) = O(n^k)$

- Intuition: we can ignore lower order terms and constants

Big Oh notation

- **Example:** $f(n) = a_k n^k + .. + a_1 n^1 + a_0$

  Then

  $f(n) = O(n^k)$

- Proof : we want to find $n_0; c$ such that $f(n) \leq cn^k$

Big Oh notation

- **Example:** $f(n) = a_k n^k + .. + a_1 n^1 + a_0;$

  $a_k > 0$

  Then: $f(n) = O(n^k)$

- Proof : we want to find $n_0; c$ such that $f(n) \le cn^k$

$$f(n) = a_k n^k + .. + a_1 n^1 + a_0$$

$$\le |a_k| n^k + .. + |a_1| n^1 + |a_0|$$

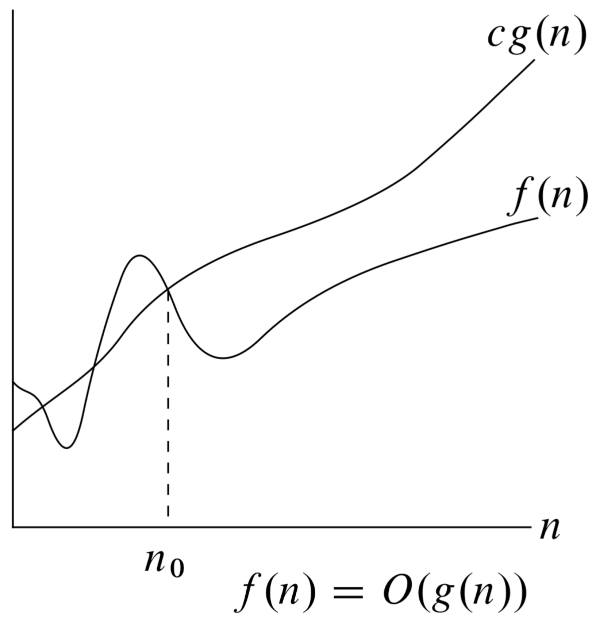
$$\le |a_k| n^k + .. + |a_1| n^k + |a_0| n^k = (|a_k| + .. |a_1| + |a_0|) n^k$$

What are? $n_0; c$

- Note also $f(n) \ge 0$

Big Oh: Most commonly used!

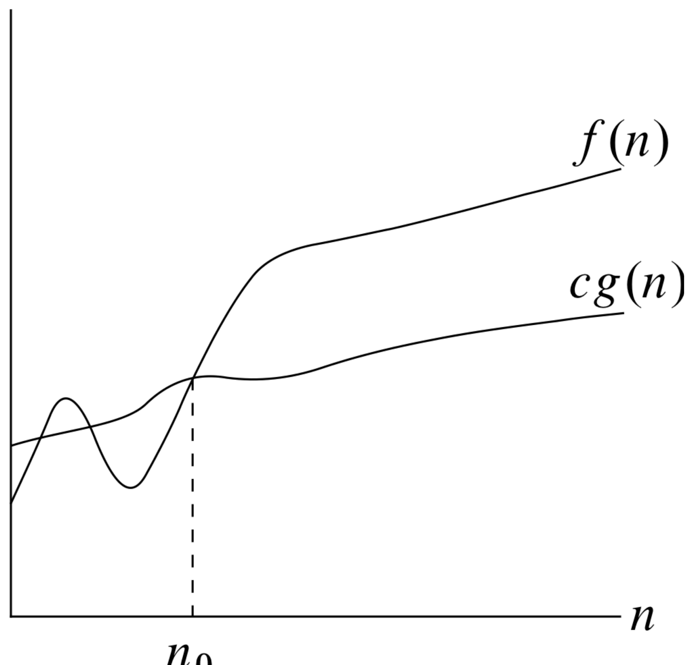
- Asymptotic upper bound; bounded from above by g(n) for large enough n
- **Definition:** O(g(n))={ f(n): there exist positive constants c and n_0 such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0$}



But there are other bounds

Big Omega

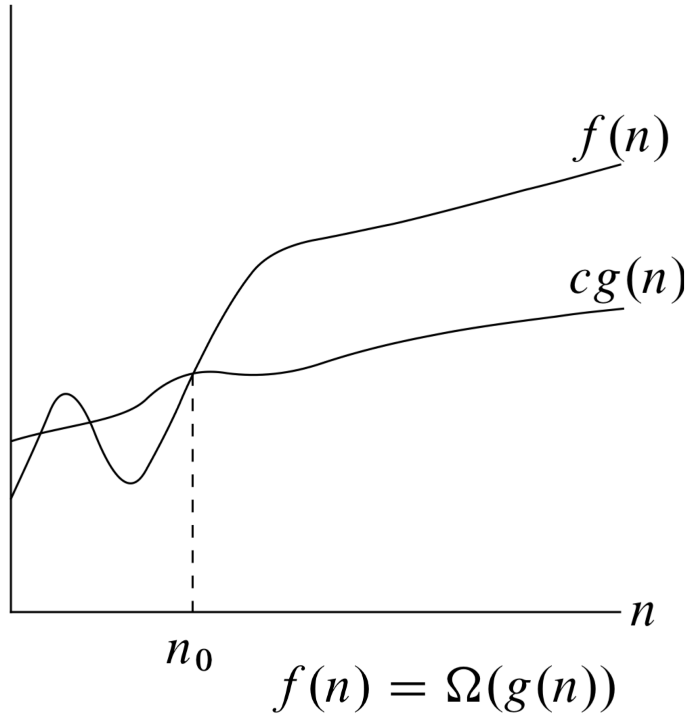
- Asymptotic lower bound; bounded from below by g(n) for large enough n
- **Definition:** $\Omega(g(n))$ = { f(n): there exist positive constants c and n_0 such that $0 \le cg(n) \le f(n)$ for all $n \ge n_0$ }



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

Big Omega

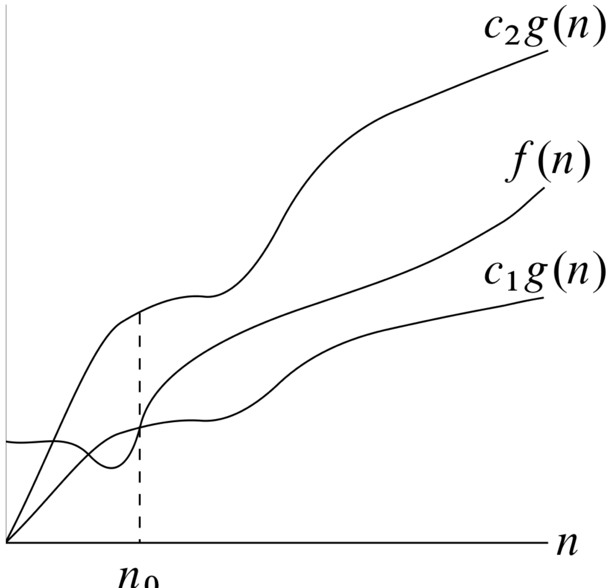
- Asymptotic lower bound; bounded from below by g(n) for large enough n
- **Definition:** $\Omega(g(n))$={ f(n): there exist positive constants c and n_0 such that $0 \le cg(n) \le f(n)$ for all $n \ge n_0$}



Why is this less often used?

Big Theta

- Asymptotic tight bound; bounded from below and above by g(n) for large enough n
- **Definition:** $\Theta(g(n))$ = { f(n): there exist positive constants $c_1; c_2$ and n_0 such that $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$ for all $n \geq n_0$ }



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

Stronger statement (note literature sometimes sloppy and says Oh when actually Theta)

Examples Oh, Omega, Theta

- Example of functions f(n) in $O(n^2)$

$$f(n) = n^2; f(n) = n^2 + n; f(n) = n$$

- Example of functions f(n) in $\Omega(n^2)$

$$f(n) = n^2; f(n) = n^2 + n; f(n) = n^5$$

- Example of functions f(n) in $\Theta(n^2)$

$$f(n) = n^2; f(n) = n^2 - n$$

Summary Oh, Omega, Theta

- Oh
  $O(n)$ asymptotic upper, like $\leq$ — $f(n) \leq cg(n)$
- Omega
  $\Omega(n)$ asymptotic lower, like $\geq$ — $f(n) \geq cg(n)$
- Theta
  $\Theta(n)$ asymptotic tight, like $=$ — $c_1 g(n) \leq f(n) \leq c_2 g(n)$

More on Oh, Omega, Theta

Theorem: $f(n) = \theta(n)$

if and only if (iff)

More on Oh, Omega, Theta

Theorem: $f(n) = \theta(n)$

if and only if (iff)

$f(n) = O(n)$ and $f(n) = \Omega(n)$