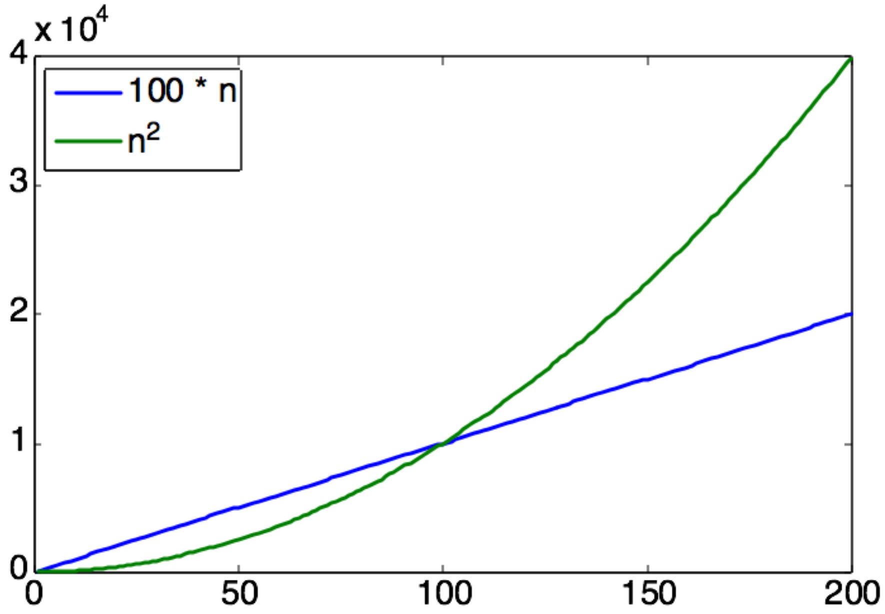
Asymptotic Analysis

Asymptotic Analysis

- Run time: # simple steps that are executed
- Depends on the size of the input (n)
  - Size of input, n, is generally defined as the number of input elements
  - Larger array takes more time to sort
  - T(n): run time for input with size n
- Look at ***growth*** of $T(n)$ as $n \to \infty$.
  - High order term dominates



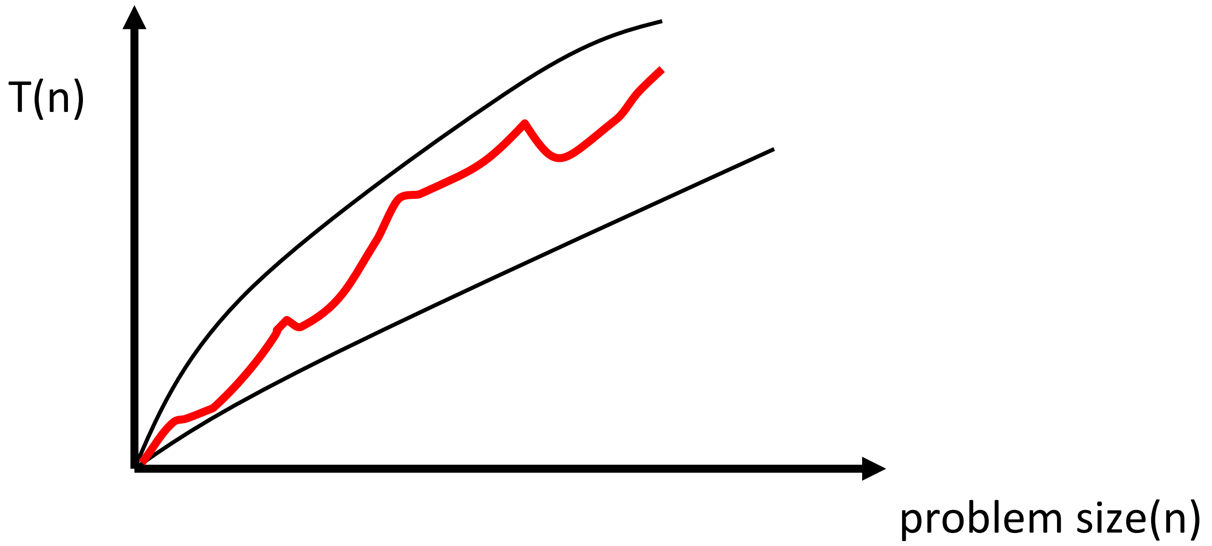
Comparison of Time Complexity Functions

	$\log_2 n$	n	$n\log_2 n$	n^2	n^3	2^n	$n!$
$n=10$	3.3	10	33	10^2	10^3	10^3	10^6
$n=10^2$	6.6	10^2	660	10^4	10^6	10^{30}	10^{158}
$n=10^3$	10	10^3	10^4	10^6	10^9		
$n=10^4$	13	10^4	10^5	10^8	10^{12}		
$n=10^5$	17	10^5	10^6	10^{10}	10^{15}		
$n=10^6$	20	10^6	10^7	10^{12}	10^{18}		

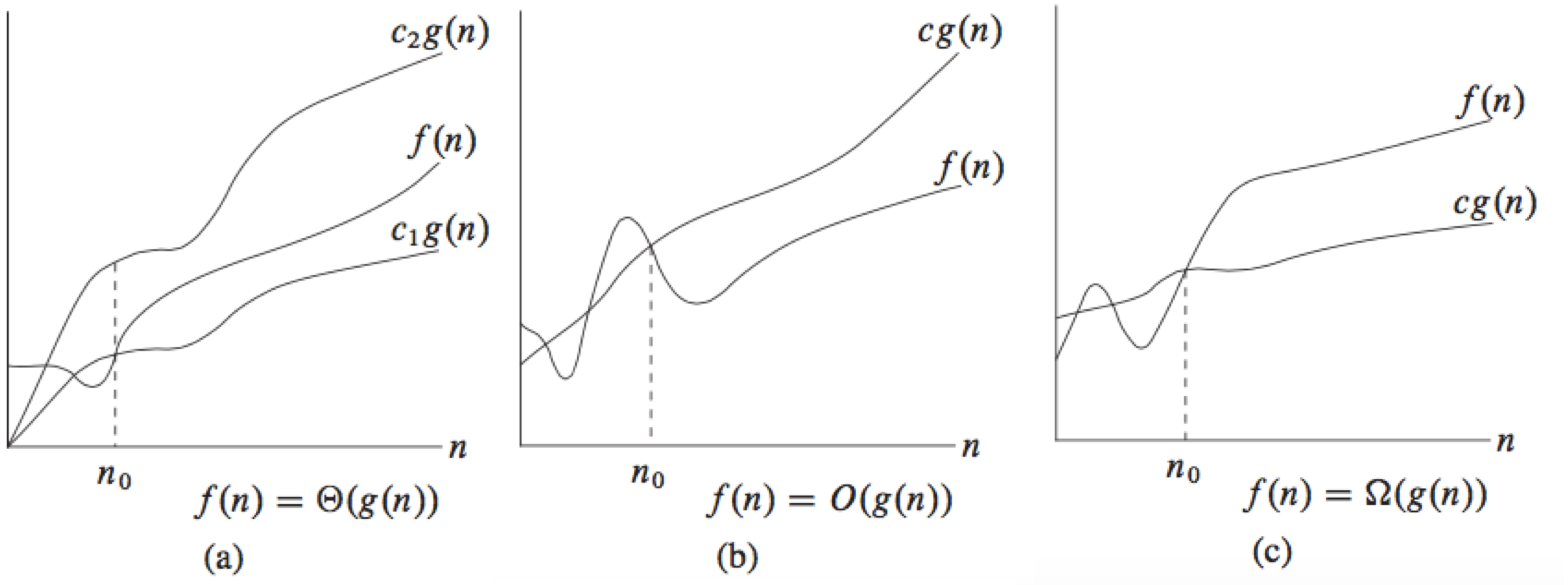
For a super computer that does 1 trillion operations per second, it will be longer than 1 billion years

Exact analysis is hard!

- easier to talk about upper and lower bounds of the function.



O, Ω, and Θ



The definitions imply a constant n_0 *beyond which* they are satisfied. (We do not care about small values of n.)

Asymptotic notations

- O: Big-Oh
- Ω: Big-Omega
- Θ: Theta
- o: Small-oh
- ω: Small-omeg

- $f(n) = O(g(n)) \rightarrow f(n) \leq c*g(n)$
- $f(n) = \Omega(g(n)) \rightarrow f(n) \geq c*g(n)$
- $f(n) = \Theta(g(n)) \rightarrow f(n) = c*g(n)$
- $f(n) = o(g(n)) \rightarrow f(n) < c*g(n)$
- $f(n) = \omega(g(n)) \rightarrow f(n) > c*g(n)$

Example: Repeated Elements

```
def findRepeated(L):
"""
determines whether all elements in a given
list L are distinct
"""
  n=len(L)
  for i in range(n):
    for j in range(i+1, n):
      if L[i]==L[j]:
        return True
  return False
```

Run Time Complexity

- Best case
  - A[1] = A[2]
  - $T(n) = \Theta(1)$
- Worst-case
  - No repeated elements
  - $T(n) = (n-1) + (n-2) + \ldots + 1 = n(n-1)/2 = \Theta(n^2)$
- Average case?
  - What do you mean by “average”?
  - Need more assumptions about data distribution.
    - How many possible repeats are in the data?
  - Average-case analysis often involves probability.

Run Time Complexity:

```
import collections
m=2; n = 2;
print ("#1")
for i in range (n):
  print(i)


print ("#2")
i=0
while (i*i < n):
  print ( i*i)
  i += 1

print ("#3")
for i in range (m):
  for j in range (n):
    print ( i*j )
```

```
m=0
print ("#4")
for i in range (n):
  for j in range (m):
    print (j)
  m += i

print ("#5: OrderedDict")
v = collections.OrderedDict()
for i in range(n):
  v[i] = i

print ("#6: list")
v = []
for i in range (n):
  v.append(i)
for i in range (n):
  v.pop(0)
```

Run Time Complexity:

1) Find the element x in an *unsorted* array of size N ($N \leq 1{,}000{,}000$).
2) In a grid of size N x M ($1<=N, M<=1{,}000$), find the shortest path between 2 points marked S and E.
3) Given a number A and a number B ($1 \leq A, B \leq 10{,}000{,}000$), find A to the power of B. As this number can be quite large, find it modulo 1,000,007.
4) Given a number P ($1 \leq P \leq 10{,}000{,}000$), determine if P is prime.
5) Given an array of size N ($1<=N<=100{,}000$, N odd), find the *median* of the array.
6) Given an array of size N ($1 \leq N \leq 2{,}000$), count the number of *inversions* in the array, where an inversion is a pair of indexes (i, j) such that $i < j$ and $ar[i] > ar[j]$.

Bonus Slides

Q#3. Which of the following is false?

a) $2^{2n} \in O(2^n)$
b) $\log(n!) \in O(n \log n)$
c) $2^{n+1} \in \Theta(2^n)$
d) $2^{2n} \in \omega(2^n)$
e) $9n^3 + 12n \in o(2^n)$

Q#4 Which of the following is true?

a) $10n^2 + 50 \in O(n \log n)$
b) $3n^2 + 12n + 2 \in \Omega(n^3)$
c) If $f(n) \in \omega(g(n))$, then $g(n) \in o(f(n))$.

Q#5. Which of the following is true?

a) If $f(n) = O(g(n))$, then $f(n) = o(g(n))$.
b) If $f(n) = \Theta(g(n))$, then $f(n) = \omega(g(n))$.
c) If $f(n) = O(g(n))$, then $g(n) = \omega(f(n))$.
d) If $f(n) = \Theta(g(n))$, then $g(n) = \Omega(f(n))$.

Q#1. Prove $n \in O(n^2)$ by choosing some $c>0$ and n_0 such that $n \le c \cdot n^2$ for $n \ge n_0$. Which of the following is NOT the correct choice of c and n_0?

a) $c=1$, $n_0=2$
b) $c= 1/2$, $n_0= 1$
c) $c= 10$, $n_0= 2$
d) $c= 1/20$, $n_0= 20$
e) $c=1$, $n_0=1$

Q#2. Prove $1000*n \in O(n^2)$ by choosing some $c>0$ and n_0 such that $1000*n \le c * n^2$ for $n \ge n_0$. Which of the following is NOT the correct choice of c and n_0?

a) $c=1$, $n_0=1000$
b) $c=1000$, $n_0=1$
c) $c=1000$, $n_0=1000$
d) $c=1$, $n0=1$

Q#1. Prove n $\in O(n^2)$ by choosing some c>0 and n_0 such that $n \le c \cdot n^2$ for $n \ge n_0$. Which of the following is NOT the correct choice of c and n_0?

a) c=1, n_0=2
b) c= 1/2, n_0= 1
c) c= 10, n_0= 2
d) c= 1/20, n_0= 20
e) c=1, n_0=1

Q#2. Prove 1000*n $\in O(n^2)$ by choosing some c>0 and n_0 such that $1000 * n \le c * n^2$ for $n \ge n_0$. Which of the following is NOT the correct choice of c and n_0?

a) c=1, n_0=1000
b) c=1000, n_0=1
c) c=1000, n_0=1000
d) c=1, n0=1

Student ID:________________________

Q#3. Which of the following is false?

a) $2^{2n} \in O(2^n)$
b) $\log(n!) \in O(n \log n)$
c) $2^{n+1} \in \Theta(2^n)$
d) $2^{2n} \in \omega(2^n)$
e) $9n^3 + 12n \in o(2^n)$

Q#4 Which of the following is true?

a) $10n^2 + 50 \in O(n \log n)$
b) $3n^2 + 12n + 2 \in \Omega(n^3)$
c) If $f(n) \in \omega(g(n))$, then $g(n) \in o(f(n))$.

Q#5. Which of the following is true?

a) If $f(n) = O(g(n))$, then $f(n) = o(g(n))$.
b) If $f(n) = \Theta(g(n))$, then $f(n) = \omega(g(n))$.
c) If $f(n) = O(g(n))$, then $g(n) = \omega(f(n))$.
d) If $f(n) = \Theta(g(n))$, then $g(n) = \Omega(f(n))$.

Student ID:_________________________

Q#3. Which of the following is false?

a) $2^{2n} \in O(2^n)$
b) $\log(n!) \in O(n \log n)$
c) $2^{n+1} \in \Theta(2^n)$
d) $2^{2n} \in \omega(2^n)$
e) $9n^3 + 12n \in o(2^n)$

Q#4 Which of the following is true?

a) $10n^2 + 50 \in O(n \log n)$
b) $3n^2 + 12n + 2 \in \Omega(n^3)$
c) If $f(n) \in \omega(g(n))$, then $g(n) \in o(f(n))$.

Q#5. Which of the following is true?

a) If $f(n) = O(g(n))$, then $f(n) = o(g(n))$.
b) If $f(n) = \Theta(g(n))$, then $f(n) = \omega(g(n))$.
c) If $f(n) = O(g(n))$, then $g(n) = \omega(f(n))$.
d) If $f(n) = \Theta(g(n))$, then $g(n) = \Omega(f(n))$.