This Lecture

- Big O
- Omega
- Theta
- Properties of Order
- Examples of Growth Rate

Practicing Algorithms?

- Practice minimum 1 hour a week
- Solve/Learn two extra problems a week

Time Complexity Analysis

- ***A time complexity analysis*** of an algorithm is the determination of how many times the **basic operation** is done for each value of the **input size**.

Input Size

- Input size examples:
    - Sequential Search (input size: n)
    - Add Array Members (input size: n)
    - Exchange Sort (input size: n)
    - Binary Search (input size: n)
    - Matrix multiplication (input size: n)
    - Graph Algorithms: (input size depends on two numbers; number of vertices and number of edges)

Time Complexity Analysis – Basic Operations

- Pick some instruction or group of **instructions that the total work done by the algorithm is roughly proportional to** the number of this instruction or group of instructions.
- Examples of basic instructions/operations:
  - Each pass in a while loop
  - Comparison instruction
- In general, do not include the control structure (like increment and compare the index) into the basic operations
- There is no hard and fast rule for choosing the basic operation. It is largely a **matter of judgment and experience**.

Time Complexity Analysis – Basic Operations…

- The core of the algorithm analysis: **to find out how the number of the basic operations depends on the size of the input.**

- The basic operation is usually selected to be the operation that:
  - is needed to solve the problem
  - is the most time consuming
  - is the most frequently used

Execution times for algorithms with the given time complexities

n	$f(n) = \lg n$	$f(n) = n$	$f(n) = n \lg n$	$f(n) = n^2$	$f(n) = n^3$	$f(n) = 2^n$
10	0.003 μs*	0.01 μs	0.033 μs	0.10 μs	1.0 μs	1 μs
20	0.004 μs	0.02 μs	0.086 μs	0.40 μs	8.0 μs	1 ms†
30	0.005 μs	0.03 μs	0.147 μs	0.90 μs	27.0 μs	1 s
40	0.005 μs	0.04 μs	0.213 μs	1.60 μs	64.0 μs	18.3 min
50	0.006 μs	0.05 μs	0.282 μs	2.50 μs	125.0 μs	13 days
10^2	0.007 μs	0.10 μs	0.664 μs	10.00 μs	1.0 ms	4×10^{13} years
10^3	0.010 μs	1.00 μs	9.966 μs	1.00 ms	1.0 s	
10^4	0.013 μs	10.00 μs	130.000 μs	100.00 ms	16.7 min	
10^5	0.017 μs	0.10 ms	1.670 ms	10.00 s	11.6 days	
10^6	0.020 μs	1.00 ms	19.930 ms	16.70 min	31.7 years	
10^7	0.023 μs	0.01 s	2.660 s	1.16 days	31,709 years	
10^8	0.027 μs	0.10 s	2.660 s	115.70 days	3.17×10^7 years	
10^9	0.030 μs	1.00 s	29.900 s	31.70 years		

*1 μs $= 10^{-6}$ second.

†1 ms $= 10^{-3}$ second.

Linear-Quadratic Algorithms

- **linear-time algorithms:** Algorithms with time complexities such as n and 100n are called linear-time algorithms because their time complexities are linear in the input size n.
- **quadratic-time algorithms:** Algorithms with time complexities such as n^2 and $0.01n^2$ are called quadratic-time algorithms because their time complexities are quadratic in the input size n.

- Any linear-time algorithm is eventually be more efficient than any quadratic-time algorithm.

- We are **interested in eventual behavior**.

Low Order Terms

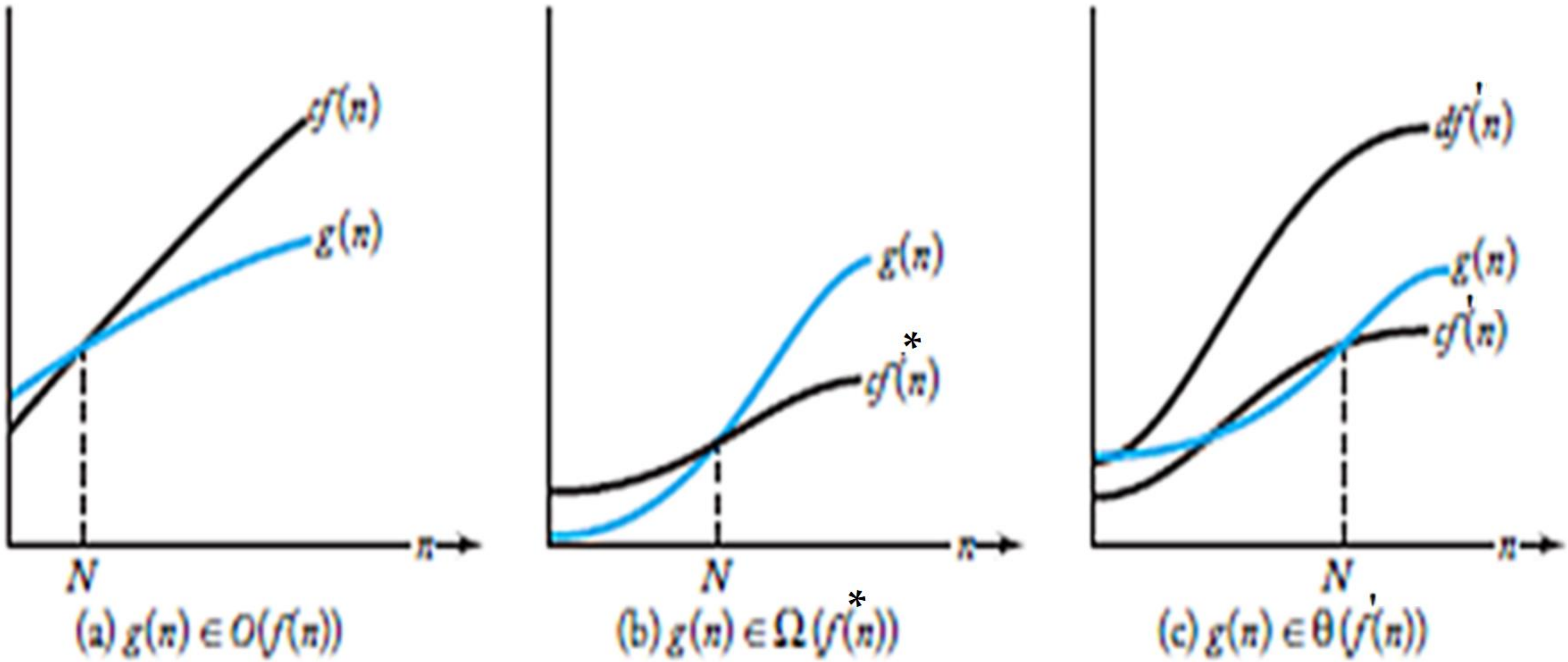
- The values of the other terms eventually become insignificant compared with the value of the quadratic term.
- We should always be able to **throw away low-order terms** when classifying complexity functions.

n	$0.1n^2$	$0.1n^2 + n + 100$
10	10	120
20	40	160
50	250	400
100	1,000	1,200
1,000	100,000	101,100

Other '–time' Algorithms

- Cubic function (3)
- Quartic function (4)
- Quintic function (5)
- Sextic equation (6)
- Septic equation (7)
- Octic equation (8)

Big O, Omega, Theta



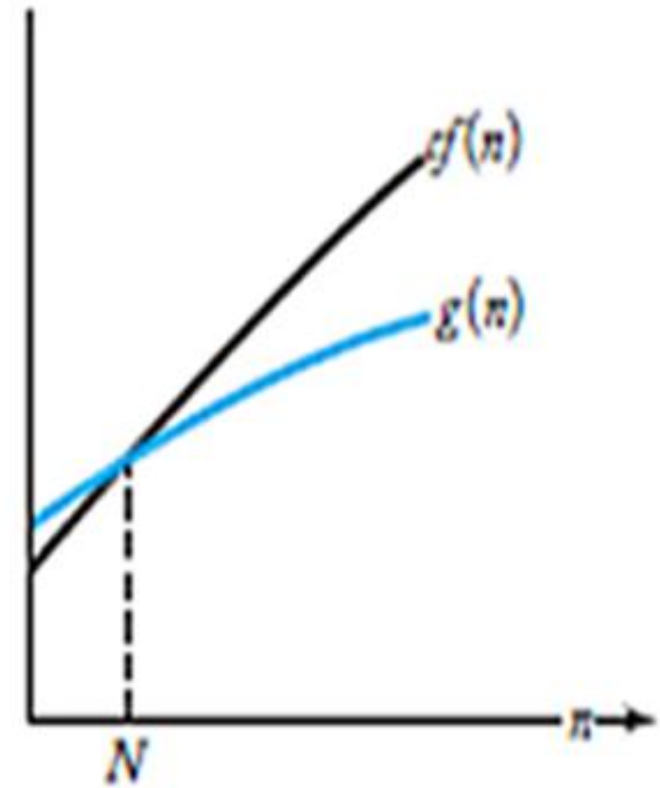
We analyze our algorithm and find out that g(n) is the time-complexity for it. (Time complexity: How many times the basic operation is done for each value of the input size (n))

It is important to note that BigO, Omega and Theta define **sets**

We analyze and then put the algorithm complexities into these sets.

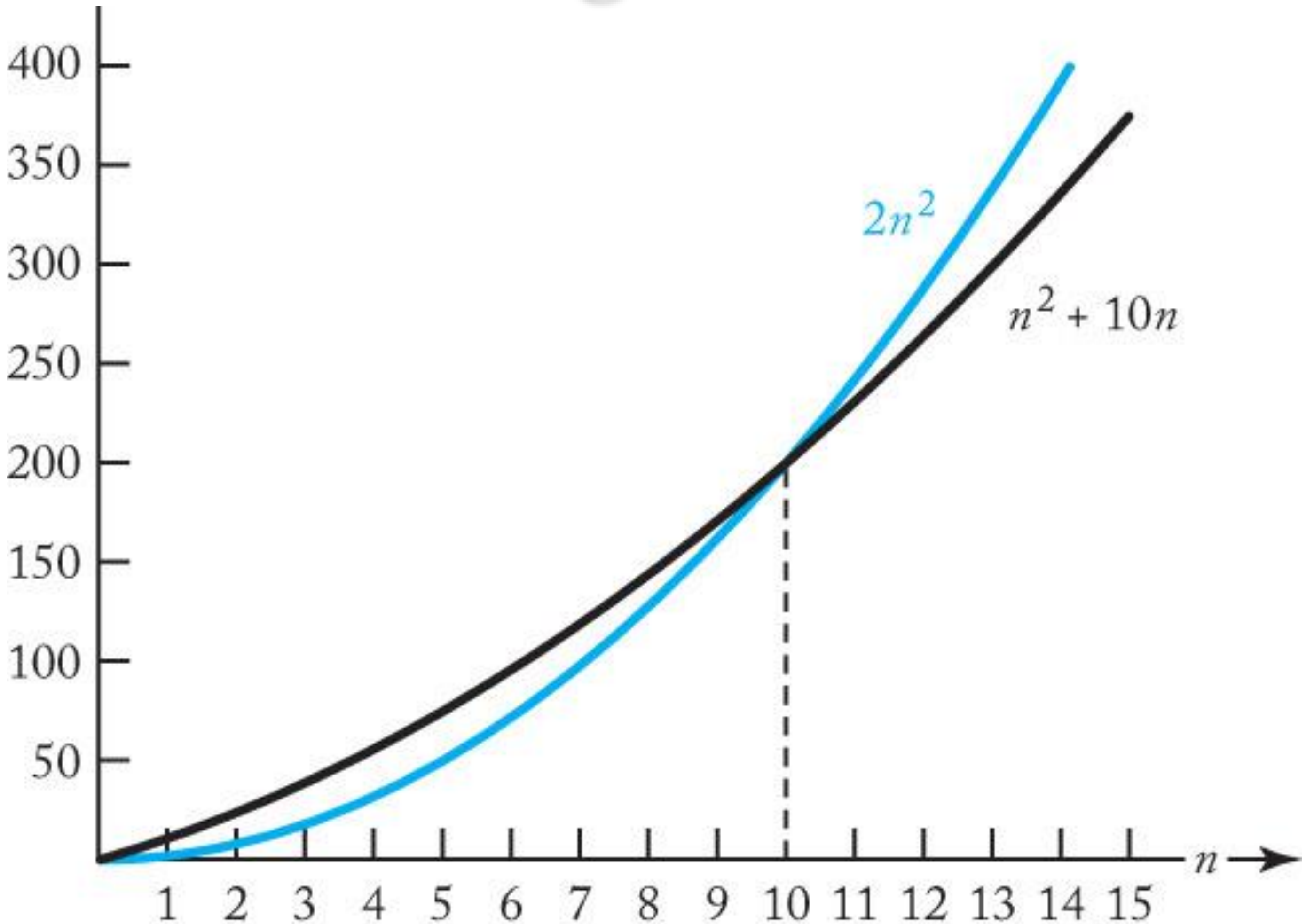
Big O

- For a given complexity function f (n), O(f(n)) is **the set of complexity functions g(n)** for which there exists **some positive real constant** c and some nonnegative integer N such that for all $n \geq N$,
- $g(n) \leq c \times f(n)$
- $g(n) \in O(f(n))$
- “big O” puts an asymptotic upper bound on a function.



(a) $g(n) \in O(f(n))$

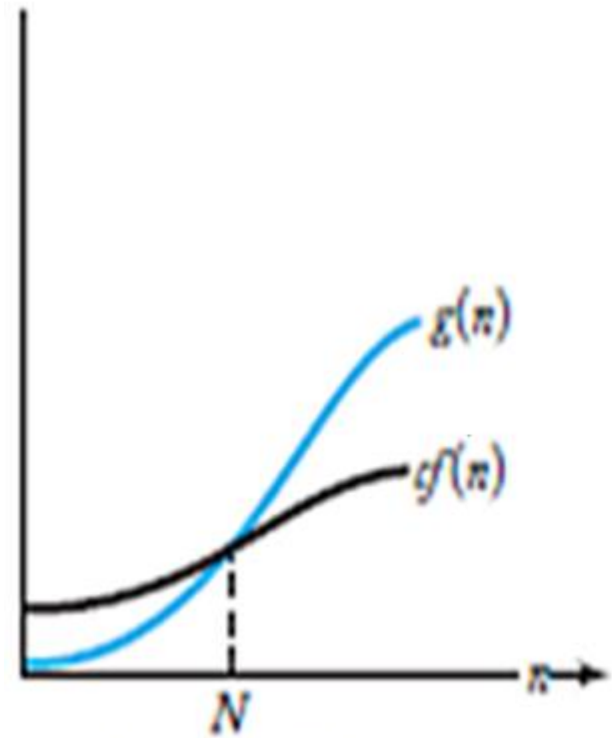
Finding c and N



c = 2 and N = 10 in the definition of “big O”

Omega

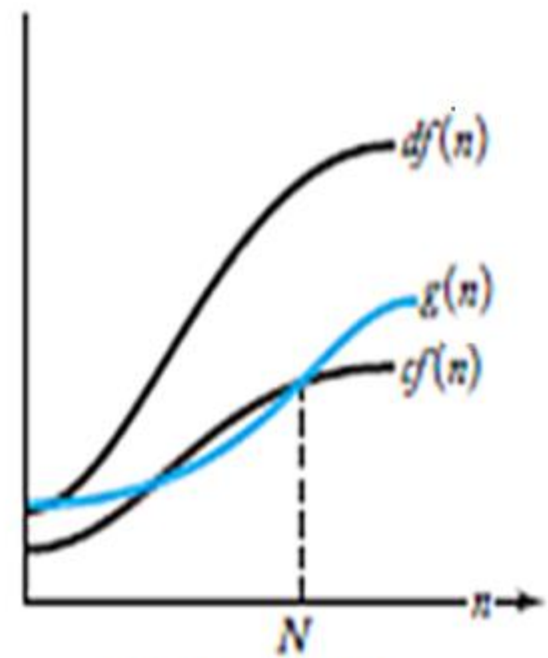
- For a given complexity function f(n), Ω(f (n)) is **the set of complexity functions g (n)** for which there exists some positive real constant c and some nonnegative integer N such that, for all $n \geq N$,
- $g(n) \geq c \times f(n)$
- $g(n) \in \Omega(f(n))$
- "Omega" puts an asymptotic lower bound on a function.



(b) $g(n) \in \Omega(f(n))$

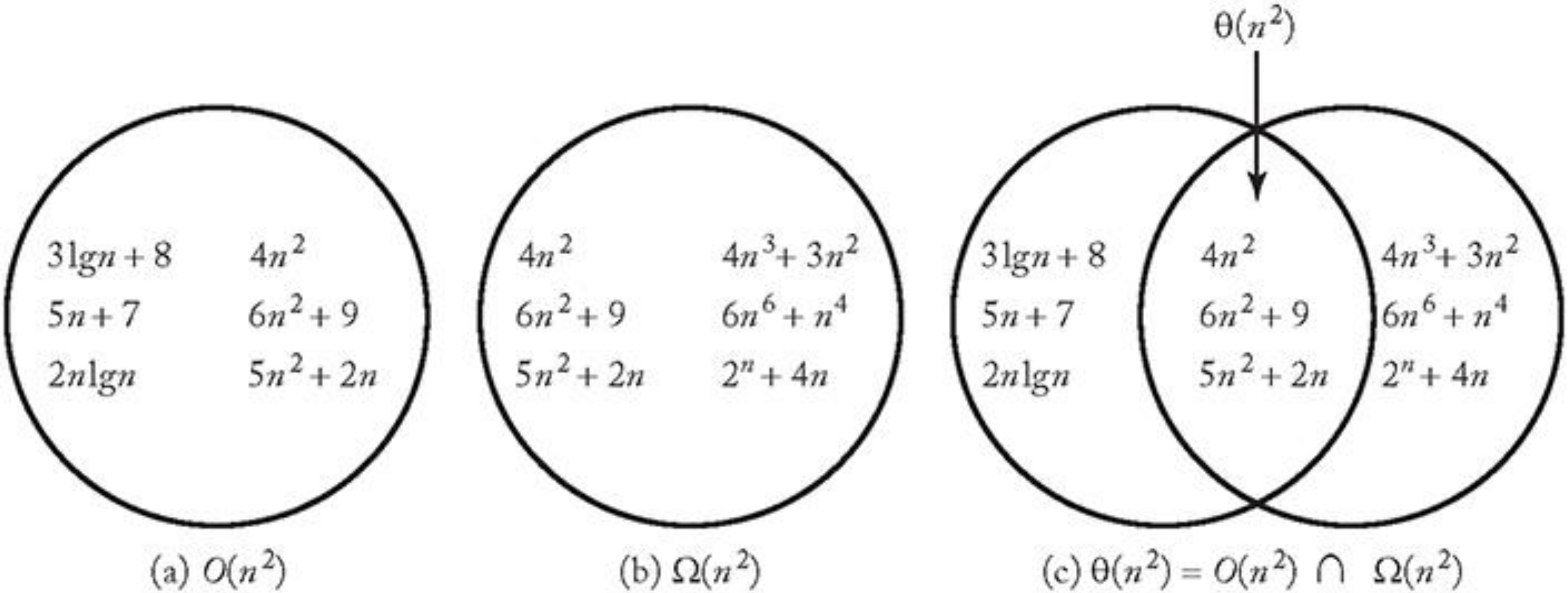
Theta

- For a given complexity function $f(n)$, $\Theta(f(n)) = O(f(n)) \cap \Omega(f(n))$
- This means that $\Theta(f(n))$ is **the set of complexity functions** $g(n)$ for which there exists some positive real constants c and d and some nonnegative integer N such that, for all $n \geq N$,
- $c \times f(n) \leq g(n) \leq d \times f(n)$.
- $g(n) \in \Theta(f(n))$



(c) $g(n) \in \Theta(f(n))$

Examples of O(),Ω (), Θ ()



The sets O (n^2) ,Ω (n^2), Θ (n^2). Some exemplary members are shown.

Order – Classes of Functions – Growth Rate

- Θ(f) - At the same rate of f
- O(f) - At most as fast as f
- Ω(f) - At least as fast as f

- n grows more slowly than n^3 => n ε $O(n^3)$
- n^3 grows faster than n => n^3 ε $\Omega(n)$
- n and 2n grow at the same rate => 2n ε $\Theta(n)$

(By definition)

- Θ(f), O(f), Ω(f) gives us asymptotic behavior of a function because we are concerned only with **eventual behavior**. The term **asymptotic means approaching a value or curve arbitrarily closely** (i.e., as some sort of limit is taken).

Properties of Order ..1

1. $g(n) \in O(f(n))$ if and only if $f(n) \in \Omega(g(n))$.

2. $g(n) \in \Theta(f(n))$ if and only if $f(n) \in \Theta(g(n))$.

3. If $b > 1$ and $a > 1$, then $\log_a n \in \Theta(\log_b n)$.
This implies that all logarithmic complexity functions are in the same complexity category. We will represent this category by $\Theta(\lg n)$.

4. If $b > a > 0$, then $a^n \in o(b^n)$
This implies that all exponential complexity functions are not in the same complexity category.

Properties of Order ..2

5. For all $a > 0$ $a^n \in o(n!)$

This implies that n! is worse than any exponential complexity function.

6. Consider the following ordering of complexity categories:

$$\Theta(\lg n) \quad \Theta(n) \quad \Theta(n \lg n) \quad \Theta(n^2) \quad \Theta(n^j) \quad \Theta(n^k) \quad \Theta(a^n) \quad \Theta(b^n) \quad \Theta(n!)$$

where $k > j > 2$ and $b > a > 1$. If a complexity function g(n) is in a category that is to the left of the category containing f(n), then $g(n) \in o(f(n))$

7. If $c \geq 0$, $d > 0$, $g(n) \in O(f(n))$, and $h(n) \in \Theta(f(n))$, then

$$c \times g(n) + d \times h(n) \in \Theta(f(n))$$

Order of Complexity Categories

$\Theta(\lg n)$ $\Theta(n)$ $\Theta(n \lg n)$ $\Theta(n^2)$ $\Theta(n^j)$ $\Theta(n^k)$ $\Theta(a^n)$ $\Theta(b^n)$ $\Theta(n!)$

where k > j > 2 and b > a > 1.

If a complexity function g(n) is in a category that is to the left of the category containing f(n),

then g(n) ∈ o(f(n))

Growth Rate Examples…

Rank the following functions by complexity category from most efficient (1) to least efficient (8)

Try it , you have two minutes

$n\log n$

$n^{5/2}$

$5n^{2} + 7n$

6^{n}

4^{n}

$n^{10} + 10^{n}$

$8n + 12$

$n!$

$\Theta(\lg n)$ $\Theta(n)$ $\Theta(n \lg n)$ $\Theta(n^2)$ $\Theta(n^j)$ $\Theta(n^k)$ $\Theta(a^n)$ $\Theta(b^n)$ $\Theta(n!)$

Growth Rate Examples…

Rank the following functions by complexity category from most efficient (1) to least efficient (8)

$n\log n$

$n^{5/2}$

$5n^2 + 7n$

6^n

4^n

$n^{10} + 10^n$

$8n + 12$

$n!$

1- $O(8n + 12)$

2- $O(n\log n)$

3- $O(5n^2 + 7n)$

4- $O(n^{5/2})$

5- $O(4^n)$

6- $O(6^n)$

7- $O(n^{10} + 10^n)$

8- $O(n!)$

$\Theta(\lg n) \quad \Theta(n) \quad \Theta(n \lg n) \quad \Theta(n^2) \quad \Theta(n^j) \quad \Theta(n^k) \quad \Theta(a^n) \quad \Theta(b^n) \quad \Theta(n!)$

Growth Rate Examples

Rank the following functions by complexity category from most efficient (1) to least efficient (8)

Try it , you have two minutes

nlogn

$5n^{2\,+}\,7n$

6^n

10^5logn

$n^{10} + 10^n$

$10^{10}\,n + 7$

$n! + 6^n$

$\Theta(\lg n) \quad \Theta(n) \quad \Theta(n \lg n) \quad \Theta(n^2) \quad \Theta(n^j) \quad \Theta(n^k) \quad \Theta(a^n) \quad \Theta(b^n) \quad \Theta(n!)$,

Growth Rate Examples

Rank the following functions by complexity category from most efficient (1) to least efficient (7)

$n\log n$
$5n^2 + 7n$
6^n
$10^5\log n$
$n^{10} + 10^n$
$10^{10} n + 7$
$n! + 6^n$

3- $n\log n$
4- $5n^2 + 7n$
5- 6^n
1- $10^5\log n$
6- $n^{10} + 10^n$
2- $10^{10} n + 7$
7- $n! + 6^n$

$\Theta(\lg n)$ $\Theta(n)$ $\Theta(n \lg n)$ $\Theta(n^2)$ $\Theta(n^j)$ $\Theta(n^k)$ $\Theta(a^n)$ $\Theta(b^n)$ $\Theta(n!)$,

Find the line on the graph that corresponds to the functions listed below:

Try it , you have two minutes

2^n

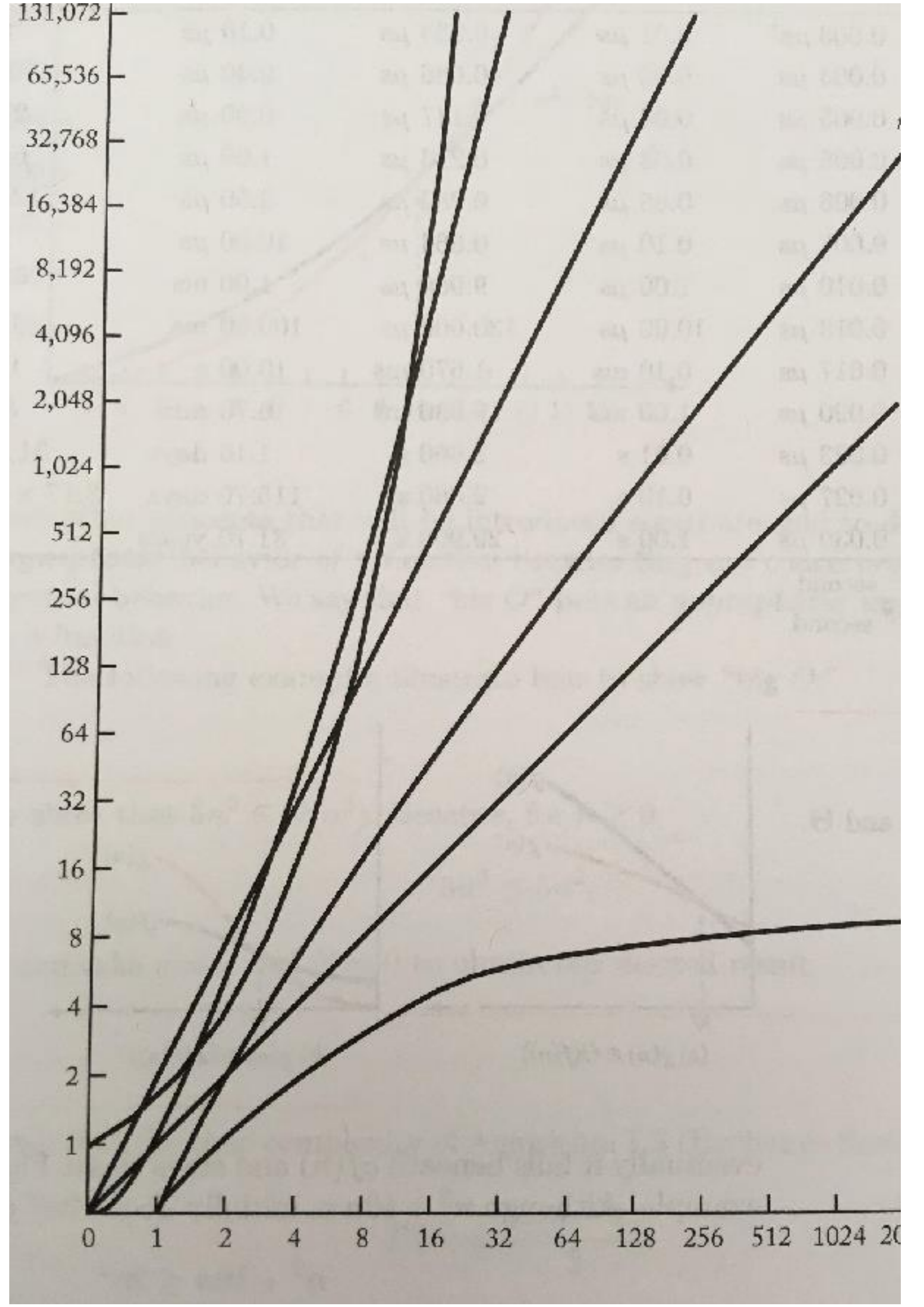
n^3

n^2

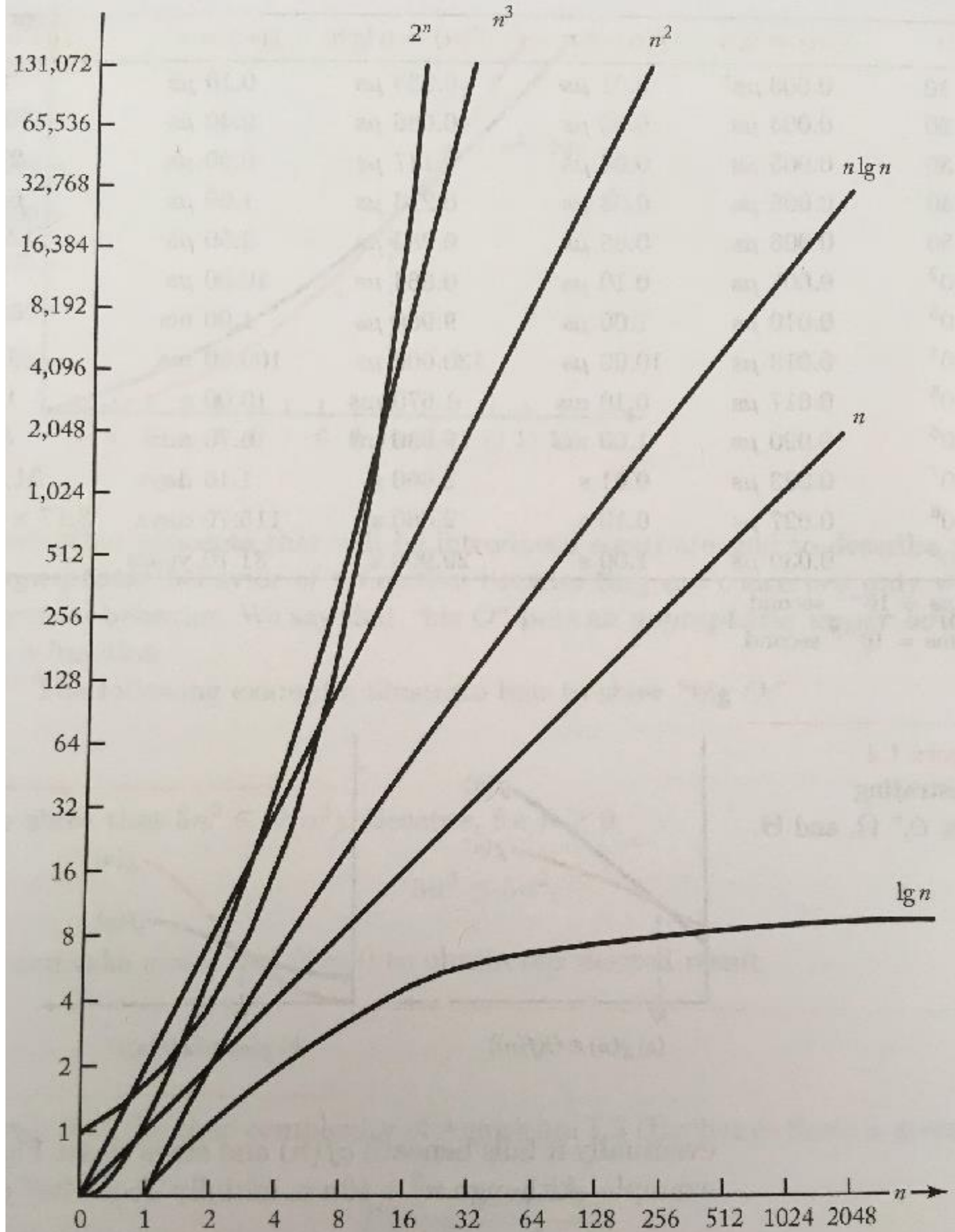
nlogn

n

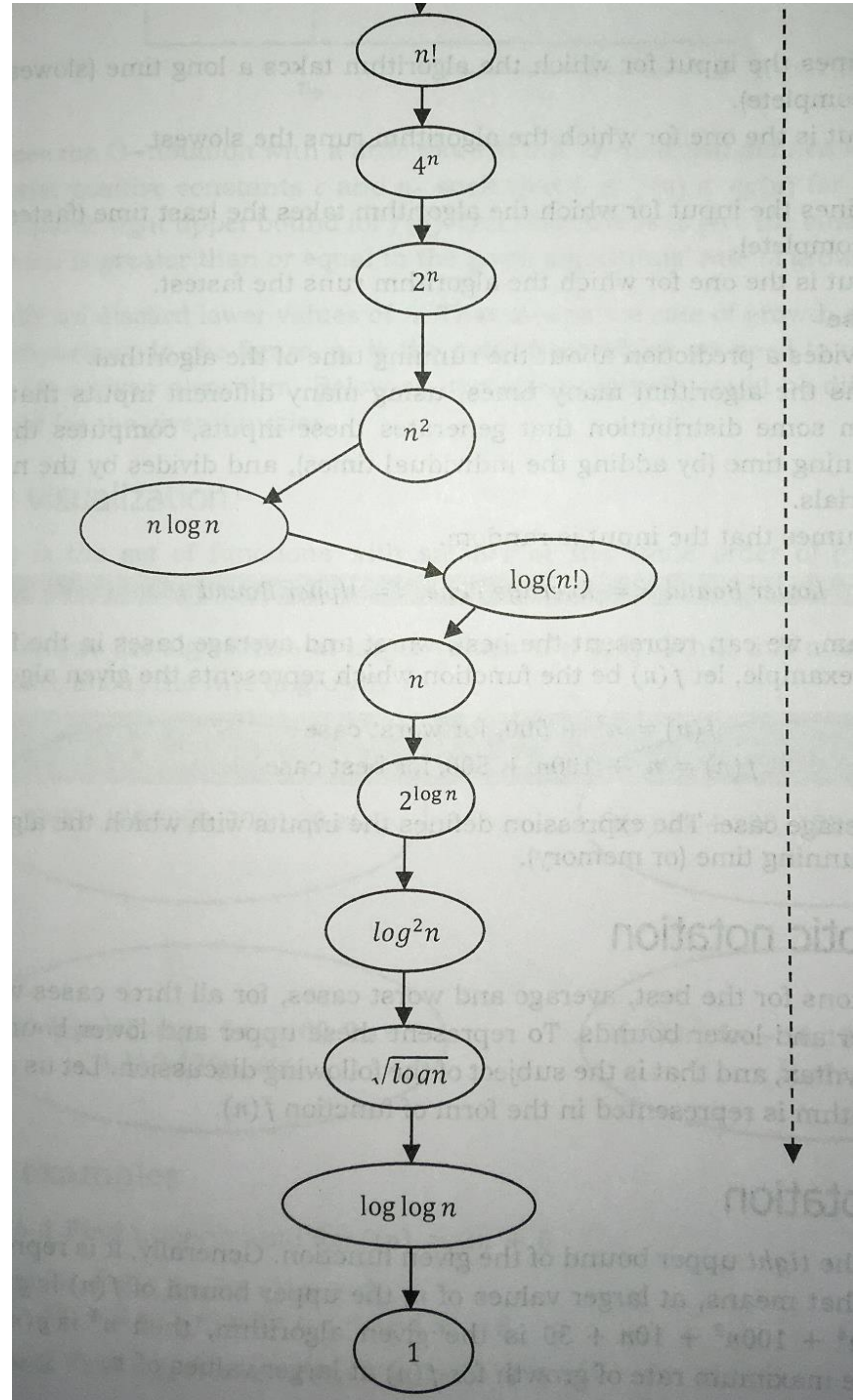
logn



Growth Rates of Common Complexity Functions



Growth Rates of Common Complexity Functions



Quiz

For the functions, n^k and c^n, what is the asymptotic relationship between these functions?

Assume that $k \geq 1$ and $c > 1$ are constants.

Choose all answers that apply:

- (A) n^k is $O(c^n)$
- (B) n^k is $\Omega(c^n)$
- (C) n^k is $\Theta(c^n)$

Quiz answer

For the functions, n^k and c^n, what is the asymptotic relationship between these functions?

Assume that $k \geq 1$ and $c > 1$ are constants.

Choose all answers that apply:

CORRECT (SELECTED)

n^k is $O(c^n)$

INCORRECT

n^k is $\Omega(c^n)$

INCORRECT

n^k is $\Theta(c^n)$

Quiz

For the functions, $\log_2 n$ and $\log_8 n$, what is the asymptotic relationship between these functions?

Choose all answers that apply:

(A) $\log_2 n$ is $O(\log_8 n)$

(B) $\log_2 n$ is $\Omega(\log_8 n)$

(C) $\log_2 n$ is $\Theta(\log_8 n)$

Quiz answer

For the functions, $\log_2 n$ and $\log_8 n$, what is the asymptotic relationship between these functions?

Choose all answers that apply:

CORRECT (SELECTED)

$\log_2 n$ is $O(\log_8 n)$

CORRECT (SELECTED)

$\log_2 n$ is $\Omega(\log_8 n)$

CORRECT (SELECTED)

$\log_2 n$ is $\Theta(\log_8 n)$

Quiz

What is the asymptotic relationship between the functions $n^3 \log_2 n$ and $3n \log_8 n$?

Choose all answers that apply:

(A) $n^3 \log_2 n$ is $O(3n \log_8 n)$

(B) $n^3 \log_2 n$ is $\Omega(3n \log_8 n)$

(C) $n^3 \log_2 n$ is $\Theta(3n \log_8 n)$

Quiz Answer

What is the asymptotic relationship between the functions $n^3 \log_2 n$ and $3n \log_8 n$?

Choose all answers that apply:

INCORRECT

$n^3 \log_2 n$ is $O(3n \log_8 n)$

CORRECT (SELECTED)

$n^3 \log_2 n$ is $\Omega(3n \log_8 n)$

INCORRECT

$n^3 \log_2 n$ is $\Theta(3n \log_8 n)$

Quiz

For the functions, 8^n and 4^n, what is the asymptotic relationship between these functions?

Choose all answers that apply:

(A) 8^n is $O(4^n)$

(B) 8^n is $\Omega(4^n)$

(C) 8^n is $\Theta(4^n)$

Clue $\log_2 a^b = b \log_2 a$

Quiz-Answer

For the functions, 8^n and 4^n, what is the asymptotic relationship between these functions?

Choose all answers that apply:

INCORRECT

8^n is $O(4^n)$

CORRECT (SELECTED)

8^n is $\Omega(4^n)$

INCORRECT

8^n is $\Theta(4^n)$

Quiz

For the functions, $\log_2 n^{\log_2 17}$ vs. $\log_2 17^{\log_2 n}$, what is the asymptotic relationship between these functions?

Choose all answers that apply:

(A) $\log_2 n^{\log_2 17}$ is $O(\log_2 17^{\log_2 n})$

(B) $\log_2 n^{\log_2 17}$ is $\Omega(\log_2 17^{\log_2 n})$

(C) $\log_2 n^{\log_2 17}$ is $\Theta(\log_2 17^{\log_2 n})$

Quiz answer

For the functions, $\log_2 n^{\log_2 17}$ vs. $\log_2 17^{\log_2 n}$, what is the asymptotic relationship between these functions?

Choose all answers that apply:

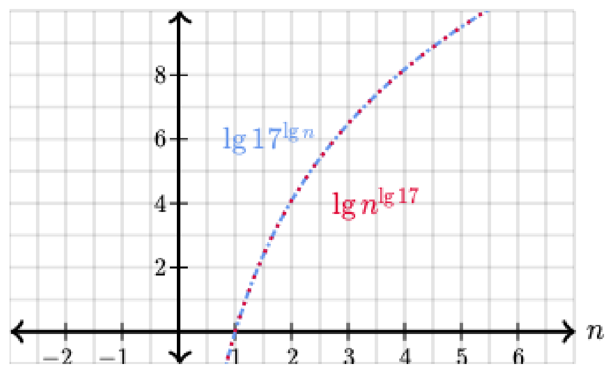
CORRECT (SELECTED)
$\log_2 n^{\log_2 17}$ is $O(\log_2 17^{\log_2 n})$

CORRECT (SELECTED)
$\log_2 n^{\log_2 17}$ is $\Omega(\log_2 17^{\log_2 n})$

CORRECT (SELECTED)
$\log_2 n^{\log_2 17}$ is $\Theta(\log_2 17^{\log_2 n})$

To answer this, we need to think about the function, how it grows, and what functions bind its growth.

Both $\log_2 n^{\log_2 17}$ vs. $\log_2 17^{\log_2 n}$ are functions with logarithmic growth, and the same base. They differ in what they take the logarithm of: $n^{\log_2 17}$ versus $17^{\log_2 n}$. Here's a graph of the two functions:



Notice something? It's the same graph! They're actually exactly equivalent functions, because of a particular property of logarithms:

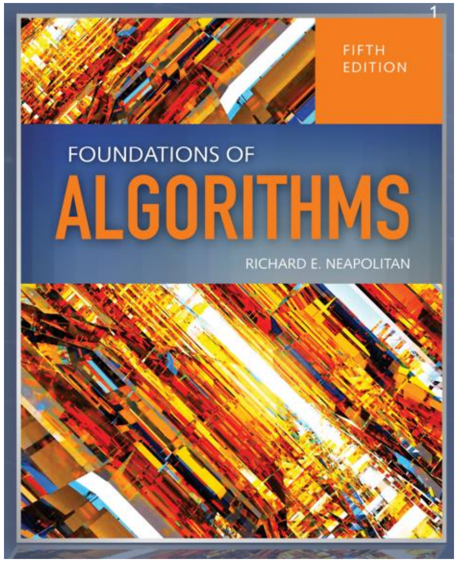
$$\log_2 a^b = b \log_2 a$$

Let's re-write both of the original functions using that property:

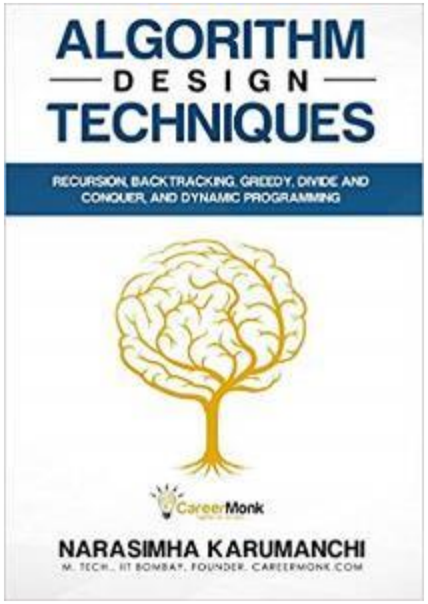
original	becomes
$\log_2 \left(n^{(\log_2 17)}\right)$	$\log_2 (17) \cdot \log_2 n$
$\log_2 \left(17^{\log_2 n}\right)$	$\log_2 (n) \cdot \log_2 (17)$

References

- Foundations of Algorithms
by Richard Neapolitan



- Algorithm Design Techniques: Recursion, Backtracking, Greedy, Divide and Conquer, and Dynamic Programming
by Narasimha Karumanchi



Questions?

Some more definitions Pure, Complete - Quadratic Algorithms

- **pure quadratic functions:**
  - $5n^2$; $5n^2 + 100$
  - because they contain no linear term,
- **complete quadratic:**
  - $0.1n^2 + n + 100$
  - because it contains a linear term