Lesson-1-Class Notes

How to analyze an algorithm?

Sum of first n numbers 1 + 2 + 3 +… + n

How can you write an algorithm?

Algorithm 1: sum = 0 for i = 1 to N sum = sum + i end for The loop runs N times, where N is the input size. Time Complexity O(n)	**Algorithm 2:** sum = 0 sum = (N * (N + 1)) / 2 The calculation can be done in constant time regardless of the input size Time Complexity: O(1)

The best one is Algorithm 2. You are going to learn how to analyze which algorithm is best.

Slide-19

Algorithm ***arrayMax***(***A***, ***n***)
currentMax ← ***A***[0]
 	
for ***i*** ← 1 **to** n-1 **do**
 	
if ***A***[*i*] > ***currentMax*** **then**
 	
currentMax ← ***A***[***i***]
 	
/* Not necessary to show in Pseudocode, but for analysis included here */
{ increment counter ***i*** }
 	
return ***currentMax***
	# operations (concern with Worst case)
	
	→ Accessing 0^{th} element and assignment (2)
	
	→ Assigning index to i and n^{th} time condition becomes false. So (n+1)
	→ accessing A[i] and > comparison (2)
	loop n-1 times = 2(n-1)
	→ accessing A[i] and assignment (2)
	loop n-1 times = 2(n-1)
	
	→ increment and assignment(2) i= i+1, loop n-1 times = 2(n-1))
	→ (1)
	Total = 2+ n+ 1+ 2(n-1) + 2(n-1) + 2(n-1) + 1
	= 4 – 6 + 7n
	= 7n-2

Arr = [10 15 20 17 -7] z = 100

Problem: For the Analysis to understand best case, Worst case

- **The Problem**: Given an array arr of ints and an int z, determine whether z belongs to arr.

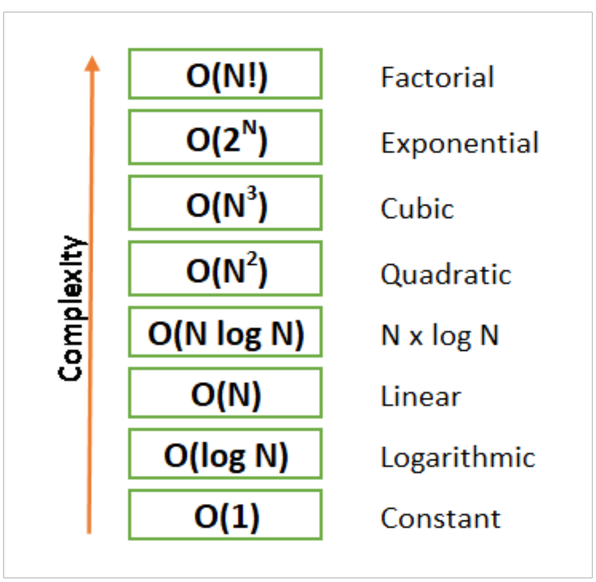
Algorithm Search(arr,z)
  Input An array arr of ints of length n and an int z
  Output True if z is found in arr, false otherwise
    for i = 0 to n – 1 **do**
        if arr[i] = z then
        return true
return false

Best Case: Found z in the first step. O(1), to find z=10 in the given list.
Worst Case: Element is not in the list. O(n), to find z=100 which is not in given list

In general, **Asymptotic analysis** is a method used in computer science and mathematics to describe the behavior of functions as they approach a particular value or infinity. It's particularly important in the analysis of algorithms.

In the context of algorithm analysis, asymptotic analysis is used to describe the running time of an algorithm in terms of the input size, typically as the input size approaches infinity.

Complexity Classes



Asymptotic analysis graph representation

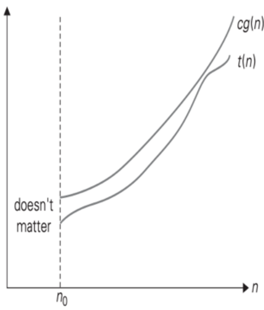


FIGURE 2.1 Big-oh notation: $t(n) \in O(g(n))$.

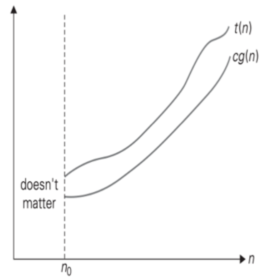


FIGURE 2.2 Big-omega notation: $t(n) \in \Omega(g(n))$.

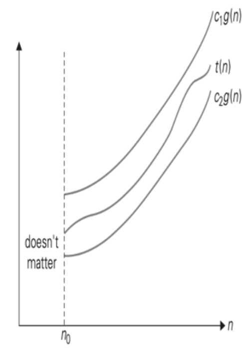


FIGURE 2.3 Big-theta notation: $t(n) \in \Theta(g(n))$.

Lesson-1- Review Questions [for Reading]

1. Able to know how to write a Pseudo code for the given problem.
2. Need to analyze runtime of the given algorithm/partial loop coding.
3. What is Big-oh?
4. Standard complexity classes and you are able arrange the given complexity from lower to higher or vice-versa.
5. Knowledge on various Asymptotic Notation.

<u>Optional Part -1</u>

Detecting Growth Rates Using Limits

A much more convenient method for detecting the growth rate is based on computing the limit of the ratio of two functions in question. It's a powerful calculus technique.

We can tell linear functions $f(n) = an + b$ always grow more slowly than the quadratic $g(n) = n^2$ because the quotient $f(n)/g(n)$ tends to 0 as n becomes large:

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{an+b}{n^2} = \lim_{n\to\infty} \frac{\frac{a}{n} + \frac{b}{n^2}}{1} = 0.$$

To simplify the expression, we divide each term in the numerator and denominator by the highest power of n, which is n^2

Conclusion: In this case, the limit evaluates to zero. Therefore, we can conclude that lim n→∞ f(n)/g(n) is zero. It indicates that the function in the numerator grows at a significantly slower rate than the function in the denominator as the input size increases.

Optional Part -2

Detecting Growth Rates Using Limits

Example: Show that 5n + 3 grows more slowly than n^2

Solution:
$$\lim_{n\to\infty} \frac{5n+3}{n^2} = \lim_{n\to\infty} \frac{\frac{5}{n}+\frac{3}{n^2}}{1} = 0$$

To simplify the expression, we divide each term in the numerator and denominator by the highest power of n, which is n^2

Conclusion: In this case, the limit evaluates to zero. Therefore, we can conclude that lim n→∞ f(n)/g(n) is zero. It indicates that the function in the numerator grows at a significantly slower rate than the function in the denominator as the input size increases.

Optional Part -3

(continued)

On the other hand, all quadratic functions always grow at the same rate. We can see this using limits: If $f(n) = an^2 + bn + c$ and $g(n) = dn^2 + en + r$, where $a \neq 0$ and $d \neq 0$, then the quotient $f(n)/g(n)$ tends to a nonzero number:

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{an^2+bn+c}{dn^2+en+r} = \lim_{n\to\infty} \frac{a+\frac{b}{n}+\frac{c}{n^2}}{d+\frac{e}{n}+\frac{r}{n^2}} = \frac{a}{d} \neq 0.$$

To simplify the expression, we divide each term in the numerator and denominator by the highest power of n, which is n^2

Example: Show that $3n^2 + 7$ grows at the same rate as $5n^2 - n$.

Solution:
$$\lim_{n\to\infty} \frac{3n^2+7}{5n^2-n} = \lim_{n\to\infty} \frac{3+\frac{7}{n^2}}{5-\frac{1}{n}} = \frac{3}{5} \neq 0$$

Conclusion: The limit evaluates to a finite number. This result tells us that both functions have the same dominant term, which is n^2, and the coefficient in front of n^2 doesn't affect their growth rate as n approaches infinity. It means that both functions exhibit the same quadratic growth rate.