CS & IT ENGINEERING

Algorithms

Analysis of Algorithm

Lecture No.- 07

By- Dr. Khaleel Khan Sir

Recap of Previous Lecture

- **Topic** Small Notations
- **Topic** Properties of Asymptotic Notations
- **Topic** Problem Solving
- **Topic**
- **Topic**

Topics to be Covered

Topics

Problem Solving with ASN

Framework for Analysing Recursive Algo

Framework for Analysing Non- Recursive Algo

$f(n), g(n)$: are fn's

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

a) Is $f(n) = O(f(n)^2)$? FALSE

b) $2^{f(n)} = O(2^{g(n)})$ FALSE

Ex. $f(n) = n$; $g(n) = n^2$

$2^n = O(2^{n^2})$ ✓

$f(n) = O(g(n))$ $\boxed{f(n) = 2n \; ; \; g(n) = n}$

$2^{2n} = O(2^n)$

$4^n \neq O(2^n)$

Ex

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

2) $f(n) = 1/n$ $(f(n))^2 = 1/n^2$

$\frac{1}{n} > 1/n^2$

$f(n) \neq O(f(n)^2)$

Topic : Asymptotic Comparisons

Q) Which one of the following statements is TRUE for all positive functions f(n)?

(GATE-22)

(a) $f(n^2) = \Theta(f(n)^2)$, when f(n) is a polynomial

(b) $f(n^2) = o(f(n)^2)$ ×

× (c) $f(n^2) = O(f(n)^2)$, when f(n) is an exponential function

(d) $f(n^2) = \Omega(f(n)^2)$

$f(n) = n^3$

$\Rightarrow f(n^2) = (n^2)^3 = n^6$

$f(n)^2 = (n^3)^2 = n^6$

$f(n) = n$

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

$(f(n))^2 = n^2$ ×

$f(n) = 2^n$

$f(n^2) = 2^{n^2}$

$(f(n))^2 = (2^n)^2 = 2^{2n}$

$= (2^2)^n = 4^n$

$\log 2^{n^2} \quad \log 2^{2n}$

$n^2 > 2n$

1) $f(n^2) = \Omega\left(f(n)^2\right)$ ✗

$\Rightarrow$ 1) $f(n) = n^2$

$f(n^2) = n^4$ ✓

$(f(n))^2 = n^4$

✗ $\log\log n$

2) $f(n) = \log n$

$f(n^2) = \log n^2$

$= 2 \cdot \log n$

$(f(n))^2 = (\log n)^2 = (\log n)(\log n)$

$2 \cdot \log n$

$(\log n + \log n) < (\log n) \cdot (\log n)$

Topic: Adding Functions

The sum of two functions is governed by the dominant one, namely:

$$O\left(f(n)\right) + O\left(g(n)\right) \rightarrow O\left(\max\left(f(n), g\ (n)\right)\right)$$

$$\Omega f(n)) + \Omega(g(n)) \rightarrow \Omega(\max(f(n), g\ (n)))$$

$$\ominus\left(f(n)\right) + \ominus\left(g(n)\right) \rightarrow \ominus\left(\max\left(f(n), g(n)\right)\right)$$

Topic: Adding Functions

$$O(f(n)) * O(g(n)) \rightarrow O(f(n) * g(n))$$

$$\Omega(f(n)) * \Omega(g(n)) \rightarrow \Omega(\mathrm{f(n)} * \mathrm{g(n)})$$

$$\ominus(f(n)) * \ominus(g(n)) \rightarrow \ominus(f(n) * g(n))$$

Topic : Asymptotic Comparisons

01. $f(n) = n$, $g(n) = \log n$ — $g(n)$ is $O(f(n))$ ✓

02. $f(n) = n^2 \log n$, $g(n) = n.\log^{10} n$ ✓

03. $f(n) = n^3$, $0 < n \leq 10{,}000$ ✓ *

$= n$, $n > 10{,}000$

$g(n) = n$, $0 < n \leq 100$

$= n^3$, $n > 100$

$\boxed{g(n) \text{ is } O(f(n))}$

$f(n) = n^2 \cdot \log n$; $g(n) = n \cdot \log^{10} n$

$= \cancel{(n \cdot \log n)} \cdot n$ $\quad$ $\cancel{(n \cdot \log n)} \cdot \log^9 n$

$= n$ $\quad$ $= (\log n)^9$

$= \log n$ $\quad > \quad$ $\log (\log n)^9 = 9 \cdot \log \log n$

03. * Two Packages are available for processing a Data Base having 10^x records. Package A takes a times of $10.n.\log_{10} n$ while package B takes a time of $0.0001n^2$ for processing 'n' records. Determine the smallest integer x for which Package 'A' outperforms Package 'B'.

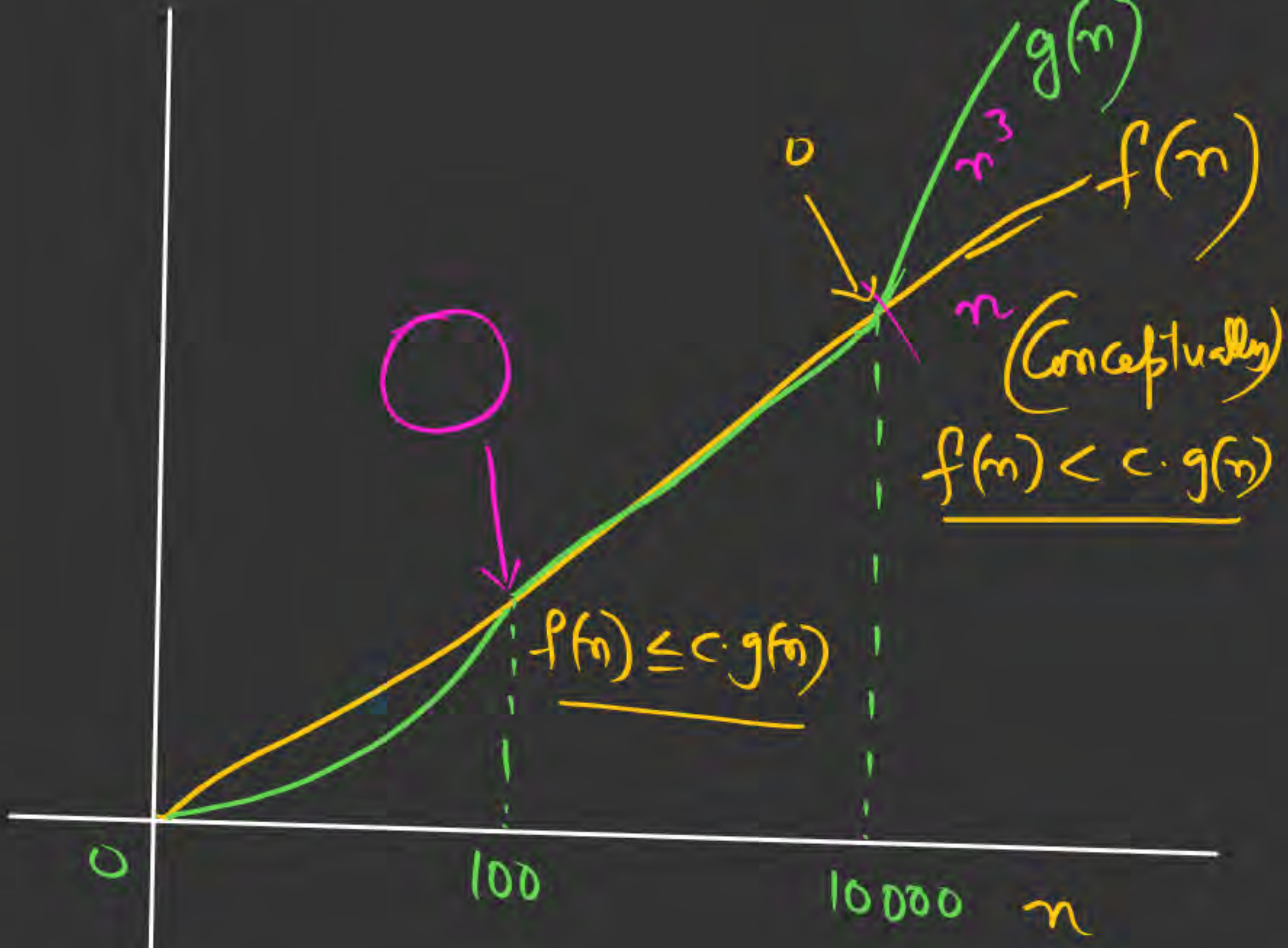
$$f(n) = n^3, \quad 0 < n \le 10{,}000$$
$$= n, \quad n > 10{,}000$$

$$g(n) = n \checkmark, \quad 0 < n \le 100$$
$$= n^3, \quad n > 100$$

$$f(n) = n^3 \checkmark, \quad n > 100, \le 10{,}000$$
$$= \underline{n}, \quad n > 10{,}000$$

$$\checkmark\, g(n) = n^3, \quad n > 100, \le 10{,}000$$
$$= \underline{n^3}, \quad n > 10{,}000$$

$(f < g)$



$f(n)$ is it $O(g(n))$, $n > 100$

$f(n)$ is it $o(g(n))$ ✓, $n > 10{,}000$

Topic: Asymptotic Notations

$$n \log n < n^2$$
$$A < B$$

$A = 10 \cdot n \cdot \log n$

$B = 0.0001\, n^2$

$n = 10^x$ records

DB

if $x = 2$

$n = 10^2$

$A : 10 \cdot 10^2 \cdot \log_{10} 10^2$

$: 2000$ units

$B : 10^{-4} \times (10^2)^2$

$: 10^{-4} \times 10^4 = 1$ unit

if $x = 3$

$n = 10^3$

$A : 10 \cdot 10^3 \cdot \log_{10} 10^3$

$: 30{,}000$

$B : 10^{-4} \times (10^3)^2$

$: 100$

$\underline{\underline{A}} \rightarrow 10 \cdot n \cdot \log_{10} n \quad ; \quad \underline{\underline{B}} \rightarrow 0.0001 n^2$

$$10 n \cdot \log_{10} n < 0.0001 n^2$$

$n = 10^x$

$$10 \cdot \cancel{10^x} \cdot \log_{10} 10^x < 10^{-4} \times 10^{\cancel{2}x}$$

$$\boxed{x < \frac{10^x}{10^5}}$$

$\therefore x = 6$

$x = 6$

$$\boxed{6 < \frac{10^6}{\cancel{10^5}}}$$

Topic : Asymptotic Comparisons

$\log 2^n \quad \log n^{\log n}$

$n \cdot \log_2 2 > \log n \cdot \log n$

0.4 Arrange the functions in increasing order of rates of growth.

0.1 $n^2; n.\log n; n\sqrt{n}; e^n; n;\ 2^n; (1/n)$

02. 2^n; $n^{3/2}$; nlogn; $n^{\log n}$

03. $n^{(1/3)}$; e^n; $n^{7/4}$; $n\log^9 n$; 1.001^n

1: $\underset{P}{n^2}\quad \underset{P}{n\log n}\quad \underset{P}{n\sqrt{n}}\quad \underset{E}{e^n}\quad \underset{P}{n}\quad \underset{E}{2^n}\quad \underset{P}{1/n}$

$\frac{1}{n} < n < n\log n < n\sqrt{n} < n^2 < 2^n < e^n$

2) $\underline{2^n}$; $\underline{n^{3/2}}$; $\underline{n.\log n}$; $\underline{n^{\log n}}$

$n^{1.5}$

$n\sqrt{n}$

$$\boxed{n\log n < n^{3/2} < n^{\log n} < 2^n}$$

$\underset{P}{n^{1/3}}$; $\underset{E}{e^n}$; $\underset{P}{n^{7/4}}$; $\underset{P}{n.\log^9 n}$, $\underset{E}{1.001^n}$

$\left(n^{1/3} < n.\log^9 n < n^{7/4}\right) < (1.001)^n < e^n$

$n^{7/4}$ $\quad n \cdot \log^9 n$

$n^{1.75}$

$(\cancel{n} \cdot n^{0.5} \cdot n^{0.25})$ $\quad \cancel{n} \cdot \log^9 n$

$(\sqrt{n}) \cdot n^{0.25} > (\log n)^9$

Topic: Asymptotic Notations

PYQ

Q) Consider the following functions from positive integers to real numbers:

$$10,\ \sqrt{n}, n, \log_2 n,\ \frac{100}{n}.$$

$10;\ \sqrt{n};\ n;\ \log_2 n;\ \frac{100}{n}$

The CORRECT arrangement of the above functions in increasing order of asymptotic complexity is:

$\frac{100}{n} < 10 < \log n < \sqrt{n} < n$

(a) $\log_2 n, \frac{100}{n}, 10, \sqrt{n}, n$ (b) $\frac{100}{n}, 10, \log_2 n, \sqrt{n}, n$

(c) $10, \frac{100}{n}, \sqrt{n}, \log_2 n, n$ (d) $\frac{100}{n}, \log_2 n, 10, \sqrt{n}, n$

Topic : Asymptotic Comparisons

MSQ

06. Which of the following is TRUE?

PYQ

1. f(n) is O(g(n))
2. g(n) is NOT O(f(n))
3. g(n) is O(h(n))
4. h(n) is O(g(n))

$f^{n} < g^{n^2}$

$g = h^{n^2}$

$$\boxed{f^{n} < \left(g^{n^2} = h^{n^2}\right)}$$

(a) f(n) is O(h(n)) ✓

(b) f(n) + h(n) is O(g(n)+h(n)) ✓

$n + n^2 \quad O(n^2 + n^2)$ ✓

(c) h(n) ≠ O(f(n)) ✓

(d) f(n) . g(n) ≠ O(g(n)). h(n)) ✗

$n \cdot n^2 \quad n^2 \cdot n^2$

$n^3 < n^4$

Topic : Asymptotic Comparisons

#Q. $f(n)=2^n$; $g(n) = n^n$

$\log 2^n \qquad \log n^n$

$n \cdot \log_2 2 \qquad n \cdot \log_2 n$

$n < n \log_2 n$

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

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

C. $f(n) = \theta(g(n))$

D. None of these

Topic : Asymptotic Comparisons

#Q. $f(n) = n.2^n$; $g(n) = 4^n$

$f < g$

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

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

C. $f(n) = \theta(g(n))$

D. None of these

$\log_2(n \cdot 2^n) < \log_2 4^n$

$\log_2 n + \log_2 2^n \qquad n \cdot \log_2 4$

$n + \log n < n + n$

Topic : Asymptotic Comparisons

H/W

#Q. $f(n) = n^2 . \log n$; $g(n) = n^{100}$

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

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

C. $f(n) = \theta(g(n))$

D. None of these

Topic : Asymptotic Comparisons

H/W

#Q. $f(n) = n^{\log n}$ $g(n) = 2^n$

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

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

C. $f(n) = \theta(g(n))$

D. None of these

Topic : Asymptotic Comparisons

H/W

#Q. $f(n) = \log_2^n ; g(n) = \log_{10}^n$

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

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

C. $f(n) = \theta(g(n))$

D. None of these

Topic : Asymptotic Comparisons

H/W

#Q. $f(n) = 2^n; g(n) = n^{\sqrt{n}}$

A. f(n) = 0 (g(n))

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

C. $f(n) = \theta(g(n))$

D. None of these

Topic : Asymptotic Comparisons

#Q. $f(n) = n^{\log_2^n}$; $g(n) = n^{\log_{10}^n}$ H/W

$n^{\log_2 n}$ $n^{\log_{10} n}$

A. f(n) = O (g(n))

B. f(n) = Ω(g(n))

C. f(n) = θ (g(n))

D. None of these

Topic : Arrange in increasing order:

H/W

#Q. $\log n;\ \log_n^{10};\ \log\ \log n; (\log\ \log n)^{10}$

$(\log n)^{10}$

Topic : Arrange in increasing order:

H/W

#Q. 2^{2^n} ; $n!$; 4^n ; 2^n

Topic : Arrange in increasing order:

H/W

#Q. $2^{\log n}; (\log n)^2; \sqrt{\log n}\ ;\ \log\ \log n$

Q) Let $W(n)$ and $A(n)$ repr respectively, the worst case & Average case running time of an Algorithm with input size of n; which is always TRUE?

S.T a) $A(n) = o(W(n))$

b) S.T $A(n) = \Omega(W(n))$

c) S.T $A(n) = \Theta(W(n))$

d) Always $A(n) = O(W(n))$

(will always be false)

e) $A(n) = \omega(W(n))$

Always false

$$\boxed{B(n) \le \underline{A(n)} \le \underline{W(n)}}$$

$\log n \quad n \quad n^3$

$\log n \quad n^2 \quad n^2$

Take-Home Lesson:

The Big Oh notation and worst-case analysis are tools that greatly simplify out ability to compare the efficiency of algorithms.

$3n^2 - 100n + 6 = O(n^2)$ because I choose $c = 3$ and $3n^2 > 3n^2 - 100n+6$;

$3n^2 - 100n + 6 = O(n^3)$, because I choose $c = 1$ and $n^3 > 3n^2 - 100n + 6$ when $n > 3$;

$3n^2 - 100n + 6 \neq O(n)$, because for any c I choose $c \times n < 3n^2$ when $n > c$;

Take-Home Lesson:

$3n^2 - 100n + 6 = \Omega(n^2)$, because I choose $c = 2$ and $2n^2 < 3n^2 - 100n + 6$ when $n > 100$;

$3n^2 = 100n + 6 \neq \Omega(n^3)$, because I choose $c = 3$ and $3n^2 - 100n + 6 < n^3$ when $n > 3$;

$3n^2 - 100n + 6 = \Omega(n)$, because for any c I choose $cn < 3n^2 - 100n + 6$ when $n > 100c$;

$3n^2 - 100n + 6 = \Theta(n^2)$, because both O and Ω apply;

$3n^2 - 100n + 6 \neq \Theta(n^3)$, because only O applies;

$3n^2 - 100n + 6 \neq \Theta(n)$, because only Ω applies.

Topic : Analysing Non-Recursive Algo

General Problems
(Time Complexity)

01. An element in an Array is called Leader if it is greater than all elements to the right of it. The Time Complexity of the most efficient Algorithm to print all Leaders of the given Array of size 'n' is $O(n)$.

	1	2	3	4	5	6	7	8	9
A	30	19	20	9	15	12	7	8	5
	L		L		L	L		L	

```
Algo BF_LEADER(A, n)
{
    for i ← 1 to (n-1)
  * {
        for j ← (i+1) to n
        {
            if (A[i] < A[j])
                Break;
        }
        if (j = n+1)
            print(A[i]);
    } *
}
```

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

(i) Best-Case : (Inc. order)

A	5	10	15	25	40

Time : $(n-1)$: $O(n)$

(ii) Worst Case : (Decr. order)

A	100	90	70	50	10	5

(ii) Linear Search from R-L :

	1	2	3	4	5	6	7 (n)
A	30	6	18	15	9	10	8
	L		L	L		L	

```
Algo EFF_LEADER(A, n)
{
    1. L ← A[n];                          1
    2. for i ← (n-1) downto 1
       {
           if (A[i] > L)
           {
               print(A[i]);               O(1)
               L ← A[i];
           }
       }
}
```

Time : $1 + (n-1)\cdot O(1)$

: $n-1+1$

: $O(n)$

$\Omega(n)$

$\Theta(n)$ ✓

THANK - YOU