CS & IT ENGINEERING

Algorithms

Lecture No.- 01

1500 Series

By- Dr. Khaleel Khan sir

Recap of Previous Lecture

Topic

Topics to be Covered

Topic Analysis of Algorithms

Topic Divide and Conquer

1) Analysis
- Time Complexities
  - ASN's
  - Recursive & Non-Recursive Alg's
  - Loop Complexities

Math Background
- Logs
- Exponents
- Sum Series

2) Divide & Conquer
3) Greedy Method
4) Dynamic Programming
5) Graph Techniques
6) Heap Algorithms
7) Sorting Tech's

#Q. Given three algorithms for solving a graph problem having n vertices & e edges. Algorithm 'A' takes n^2 ms, algorithm B takes 100n ms; while C takes $100\left(2^{n/10}-1\right) ms$.

Which algorithm is fastest for all very large value of n?

A. A

B. B

C. C

D. All are equally fast.

$A \rightarrow n^2$ ms

$B \rightarrow 100 \times n$ ms

} Poly

$C \rightarrow 100\left(2^{n/10}-1\right)$ ms } Expo.

Poly < Expo

$B < A < C$

Fastest

Slowest

#Q. Let B(n) and A(n) respectively denote the best case and average case running time of an algorithm of input size 'n' which of the following is always false?

A. $B(n) = \Omega(A(n))$ May be True

B. $B(n) = \theta(A(n))$ May be TRUE

C. $B(n) = w(A(n))$

(small omega)

D. $B(n) = O(A(n))$ May be TRUE

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

$\boxed{B(n) > A(n)}$ Always false

#Q. $f(n) = \sum_{i=1}^{n} i^{\frac{1}{2}}$ is___.

$$f(n) = \sum_{i=1}^{n} i^{1/2} \sim \int_1^n i^{1/2} = \left[\frac{i^{1/2+1}}{\frac{1}{2}+1}\right]_1^n$$

$$\int x^n = \frac{x^{n+1}}{n+1}$$

$$= \left[\frac{n^{3/2}}{} \times \frac{2}{3}\right]$$

$$= \Theta\left(n^{1.5}\right)$$

$$= \Theta\left(n\sqrt{n}\right)$$

A $\theta(n)$

B θ (nlogn)

C $\theta\left(n\sqrt{n}\right)$

D $\theta\left(\sqrt{n}\right)$

$$f(n) = \sum_{i=1}^{n} i^{-1/2} = O(\quad)$$

$$= \int_1^n 1/i^{1/2} = \log n^{1.5} \Rightarrow 1.5 \times \log n$$

$$= O(\log n)$$

#Q. Let k be an integer greater than 1. Which of the following represents the order of growth of the expression $\sum_{i=1}^{n} k^i$ as a function of n ?

$$S_n = \frac{a(r^n - 1)}{r-1}$$

$$= \frac{\cancel{k}(k^n - 1)}{\cancel{k-1}} = \Theta(k^n)$$

A $\Theta(k^n)$

B $\Theta(k^{n \log n})$

C $\Theta(k^n \text{ lon } n)$

D $\Theta(k^{2kn})$

E $\Theta(n^{k+1})$

#Q. Consider the following functions:

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

$g(n) = n \log^{10} n$

$h(n) = n^3$

Inc. order : $n(\log n)^{10} < n^2 \cdot \log n < n^3$

$g < f < h$

which of the following is incorrect?

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

B. $g(n) = O(h(n))$: T

C. $f(n) = O(h(n))$: T

D. None of the above

#Q. Consider the functions below:

$$\log_2(n!), \frac{n}{2^n}, \frac{n}{\log_2 n}, (\log_2 n)!, \log_2 n$$

$(n\log n)$ (1) (2) (3) (4) (5)

P P P E P

Arrange them in the decreasing order of rates of growth.

$x! = \Theta(x^x)$

$(\log n)! = (\log n)^{\log n}$

a) 1, 4, 5, 3, 2

b) 1, 4, 3, 5, 2

c) 2, 3, 5, 4, 1

d) 2, 3, 5, 1, 4

e) 4, 1, 5, 3, 2

$(\log n)! > \log(n!) >$

$(4) > (1) > (3) > (5) > (2)$

$\frac{n}{2^n}$ (2) $\frac{n}{\log n}$ (3) $\log n$ (5)

$\frac{n}{\log n} > \log n > \frac{n}{2^n}$

#Q. Arrange the following functions in the increasing order of their rates of growth:

$(\text{sqrt}(2))^n$, $2^{\text{sqrt}(n)}$, $n^2\log n$, $n(\log n)^2$, $(n\log n)^2$, $n^{\log n}$, $n^{\text{sqrt}(n)}$, n^n, $(\log n)^n$.

E E P P P E E E E

(1) (2) (3) (4) (5) (6) (7) (8) (9)

$n(\log n)^2 < n^2\log n < n^2\log^2 n$

(4 3 5)

$\langle 2 \quad 1 \rangle$

$(\sqrt{2})^n > 2^{\sqrt{n}}$

6	7	8	9
$n^{\log n}$	$n^{\sqrt{n}}$	n^n	$(\log n)^n$

6 7 9 8

(4 3 5 6 2 1 7 9 8)

$n^{\sqrt{n}} < (\log n)^n$

$\sqrt{n}\cdot\log n$ $\quad$ $\sqrt{n}\cdot\sqrt{n}\log\log n$

a) 435679 18

b) 436912578

c) 65432178

d) 45312968

#Q. Consider the equality

MSQ

$$\left[\sum_{i=0}^{i=n+1} 2^i\right]\left[\sum_{i=0}^{i=n} = \frac{1}{2^i}\right] = P$$

$$\frac{1\left(2^{n+2}-1\right)}{2-1} \qquad \frac{1\left(1-\frac{1}{2^{n+1}}\right)}{1-\frac{1}{2}}$$

$\langle A+B+C \rangle$

Which of the following notation asymptotically is / are remains correct, with respect to P?

$$\left(2^{n+2}-1\right)\left(1-\frac{1}{2^{n+1}}\right)$$

$$= 2^{n+2} - 2 - 1 + \frac{1}{2^{n+1}}$$

$$= \left[2^{n+2} - 3 + \frac{1}{2^{n+1}}\right]$$

$$\left(2^2\right)^n = 4^n$$

A. $\theta(2^{n+2})$ ✓

B. $\Omega(2^n)$ ✓

C. $O(2^{2n})$ ✓

D. $\theta(2^{2n})$ ✗

#Q. Consider the following C function:

```
main()
{
    for (i = 1; i ≤ n; ++i)
    {
        j = 1;
        while ( j ≤ n)
            j = 2*j;
        for (k = 1; k ≤ n; ++k)
            c = c + 1;
    }
}
```

: n Total

: n

Logn : n Logn

n : n^2

$n + n.\log n + n^2 = O(n^2)$

The time complexity of above algorithm will be?

A. O(n log n)

B. $O(n^2 \log n)$

C. $O(n^2)$

D. None of the above

The return (final) value of K is $\Theta(___)$;

#Q. Consider the following function:

```
int unknown (int n)
{
    int i, j, k = 0;
    for (i = n/2; i < n; i++)
    {
    for (j = 2; j ≤ n; j = j * 2)
        k = k+ n/2;
    }
    return (k);
}
```

$K = \left(\frac{n}{2} * \log n\right) * \frac{n}{2}$

$\Theta(n^2 \log n)$

What will be the time complexity of above C code?

A. $\theta(n^2 \log n)$

B. $\theta(n \log n)$

C. $\theta(n^2)$

D. None of the above

#Q. Consider the following code segment

```
for (int x = 0; x < n; x ++)
for (int y = 1; y < =n*n*n; y = 2*y)
    print("spread positivity");
```

$n * 3 * \log n$

$= O(n \log n)$

$\log z = \log n^3 = 3 * \log n$

What is the running time complexity of the above code?

A. $\theta(n^2)$

B. $\theta(n \log n)$

C. $\theta(n(\log n)^3)$

D. $\theta(n \log \log n)$

#Q. If the given program takes 10 milliseconds to run n = 50, how long, in milliseconds, will it take to run for n = 500? ___.

```
addn = 0;
for (a = 0; a < n/2; a++)
for (b = 0; b < n/4; b++)
        addn ++;
```

$O(n^2)$

(1000 ms)

$(50)^2$ units $\Longrightarrow$ 10 ms

1 unit $\Longrightarrow$?

$$\boxed{1 \text{ unit} = \frac{10}{50 \times 50} = 1/250 \text{ ms}}$$

$(500)^2$ units $\Longrightarrow$?

$$\frac{500 \times 500}{250} = 1000 \text{ ms}$$

#Q. Determine the Time Complexity of the following iterative function:

```
int f (int A[SIZE] [SIZE], int n)
{
    int i, j, sum = 0;
    for (i = 1; i ≤ n; ++i)
    {
        if (i % 2 ==0)
            for (j =1; j ≤ i; j = j + 1) sum = sum + A[i] [j];
        else
            for (j = n ; j > i; j = j – 1) sum = sum – A[i] [j];
    }
}
```

$i=1 \quad i=2 \quad i=3 \quad i=4 \quad i=5 \quad i=6$

Total Step Count $= (n-1) + 2 + (n-3) + 4 + (n-5) + 6 \cdots + 1 + n$

$= 2+4+6+\cdots+n + n*\frac{n}{2} - (1+3+5+\cdots+\frac{n-1}{})$

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

#Q. What does the following function compute ?

```
int F (int n, int *Fprev)
{
        int Fn_1, Fn_2;
        if (n == 0) {
                *Fprev = 1;
                return (0);
        }
        if (n==1)  {
                *Fprev = 0;
                return (1);
        }
        Fn_1 = F(n -1, & Fn_2);
        *Fprev = Fn_1;
        return (Fn_1+Fn_2);
}
```

Space: $O(n)$

Recursion Stack

Return n^{th} Fibonacii Number

Time: $O(n)$ ✓

0 1 1 2 3 5 . . .

n: 0 1 2 3 4

Time Complexity Recurrence of $F(n)$

$$T(n) = c, \quad n \le 1$$
$$= a + T(n-1) + b$$
$$= T(n-1) + d, \quad n > 1$$

2 mins Summary

Topic One — Analysis of Algorithms

Topic Two — Divide and Conquer

Topic Three

Topic Four

Topic Five

THANK - YOU

CS & IT ENGINEERING

Algorithms

Lecture No.- 02

1500 Series

By- Dr. Khaleel Khan sir

Recap of Previous Lecture

Topic ALGORITHMS 01

Topics to be Covered

Topic Analysis of Algorithms

Topic Divide and Conquer

#Q. Let f(n) and g(n) be asymptotically positive functions then consider the following statements

$\langle a+b \rangle$

(a) $f(n) = O(g(n))$ implies $2^{f(n)} = O(2^{g(n)})$ ×

(b) $f(n) = O((f(n))^2)$ ×

A. a is false ✓

B. b is false ✓

C. a, b are true

D. a is true

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

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

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

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

2^{2n} is it $O(2^n)$ ×

3) $f(n) = n$

n is it $O(n^2)$ ✓

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

2^n is it $O((2^n)^2)$ ✓

5) $f(n) = \frac{1}{n}$

$1/n$ is it $O(1/n^2)$ ×

#Q.

```
for (int i = 2; i ≤ n; i *=i)
{
    for (int j = 2; j ≤ i; j*=2)
    {
        print( " * " );
    }
}
```

$\log i$

The Time Complexity of the code snippet is _____.

$i = 2 \quad i = 2*2 \quad i = 4\times4 \quad i = 16\times16 \quad \ldots$

$$\text{Time} = \left[\log_2 2^{2^0} \quad \log_2 2^{2^1} \quad \log_2 2^{2^2} \quad \log_2 2^{2^3} \quad \ldots \quad \log_2 2^{2^k}\right]$$

$$= \log_2 2^{2^0+2^1+2^2+\ldots+2^k} = \log_2 2^{(2^{k+1}-1)} = (2^{k+1}-1) = (2^{\log\log n+1}-1) = O(\log n)$$

$2^{2^k} = n \Rightarrow k = \log\log n$

a) $\sqrt{n}\cdot\log n$

b) $\log n$

c) $n\log n$

d) n^2

e) $(\log n)^2$

f) $n\cdot\log\log n$

g) $\log n * \log\log n$

#Q. The following function recursively determines whether a given string is a Palindrome. Determine its Time Complexity.

```
int isPalindrome (char A[ ], int n)
{
    if (n ≤ 1) return 1;
    if (A[0] != A[n–1]) return 0;
    return isPalindrome (&A[1], n–2);
}
```

Let $T(n)$ Time Complexity of isPalindrome(n)

$T(n) = c, \quad n = 1$

$= T(n-2) + a,$

$T(n) = T(n-2) + a$ — (1)

$= T(n-4) + 2a$ — (2)

$= T(n-6) + 3a$ — (3)

$= T(n-2\cdot 3)$

$= T(n-2\cdot k) + k\cdot a$

$= T(1) + a\frac{(n-1)}{2}$

$n - 2k = 1$

$\frac{n-1}{2} = k$

$= O(n)$ ✓

#Q. The running time of the following procedure

```
procedure foo(n)
{
    if (n < =2) return (1)
    else
        return (foo(√n) + n)
}
```

is best described as

Value₁ Value₂

- A. 0 (loglogn)
- B. 0(1)
- C. 0(log n)
- D. $0(2^n)$

Let T(n) repr. T.C of foo(n)

$T(n) = c, \; n \le 2$

$= a + T(\sqrt{n}) + b, \; n > 2$

$= T(\sqrt{n}) + d, \; n > 2$

```
main()
{ int n, m;
  n=10; m=20;
  return (n+m);
}
```

O(1)

O(log log n)

```
Procedure Foo(n)
{
    if (n ≤ 2) return(1);
    else
        return(Foo(√n) + B(n));
}
```

$O(B(n))$

$$T(n) = c, \quad n \le 2$$

$$= a + T(\sqrt{n}) + O(B(n)) + a$$

#Q. Let T(n) be a recurrence relation such that $T(n) = 4T(\sqrt{n}) + c$ $n > 2$ otherwise $= 0$

(Time Complexity)

What is the maximum depth of recursion required?

A. $\lceil \log_4 n \rceil$

B. $\lceil \log_2 \log_2 n \rceil$

C. $\lceil \log_4 \log_4 n \rceil$

D. $\lceil \log_4 n \rceil$

$T(n) = 4 \cdot T(n^{1/2}) + c$

$T(n^{1/2}) = 4 \cdot T(n^{1/2^2}) + c$

$\vdots$

$= 4 \cdot T(n^{1/2^k})$

$n^{1/2^k} = 2$

$k = \lceil \log_2 \log_2 n \rceil$

$T(2^k) = 4 \cdot T(2^{k/2}) + c$

$\boxed{S(k) = 4 \cdot S(k/2) + c}$

c is $O(k^{2-\epsilon})$

$\therefore T(k) = O(k^2)$

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

1) $T(n) = T(n/2) + c \rightarrow$ Binary Search,

2) $T(n) = 2 \cdot T(n/2) + \underline{\underline{f(n)}} \rightarrow$ M.S / Q.S

$\Big\}$ $\log n$

$$T(n) = T(n/2) + c$$

$$T(n/2) = T(n/4) + c$$

$$= T(n/4) + 2c$$

$$\vdots$$

$$= T(n/2^k) + k \cdot c$$

k = depth of Stack

$$\frac{n}{2^k} = 1$$

$$\Rightarrow n = 2^k$$

$$\boxed{k = \log n}$$

#Q. Consider the following recursive program

```
void abc(int n)
{
  int x = 0;
  if (n>1)
  {
      abc(n – 1);
      abc(n – 1);
  }
  for (a = 0; a <n; a++)
  printf("$");
}
```

$T(n) = 2 \cdot T(n-1) + n$

What is the recurrence relation for the given program ?

A. $T(n) = 2T(n-1)+1$

B. $T(n) = T(n-1) + n$

C. $T(n) = 2T(n-1) + n$

D. None of these

#Q. Match the Pairs

A	$O(\log n)$	(P)	Merge sort
B	$O(n)$	(Q)	Binary Search
C	$O(n\log n)$	(R)	Insertion sort
D	$O(n^2)$	(S)	Partitioning / Pivoting

#Q. An array 'A' contains n integers ~~in increasing order~~. The Algorithm to find two indices 'i' & 'j' such that A[i] + A[j] = 'M' a given integer, if they exists , will take:

(Most efficient Algorithm)

A. Constant time

B. Logarithmic time

C. Linear Polynomial time

D. Exponential time

1) Brute_Force $= O(n^2)$

2) Using Bin-Srch $= O(n \cdot \log n)$

3) Two-Pointer [Partition-Proc] $O(n)$

#Q. Which of the following has the lowest worst-case complexity

[Priority-Q] Heap : $\log n$

A. Shortest job first

B. [Dijkstra's Banker's Algorithm] $O(n^2)$

C. Round Robin : $O(1)$

→ circular L.L

D. [Shortest seek time first] Heap

Evaluates

$\frac{n}{2^k} = 1 \Rightarrow k = \log_2 n$

#Q. The solution to the recurrence equation T(n) = 3T (n/2) +1, T(1) =1 is-

A. $2^{\log 3} . \log n$

B. $3^{\log_2 n + 1} - 2$

C. $3^{\log_2 n}$

D. None of the above

$T(n) = 3 \cdot T(n/2) + 1$ — (1)

$T(n/2) = 3 \cdot T(n/4) + 1$ — (2)

$T(n) = 3(3 \cdot T(n/4) + 1) + 1$

$= 9 \cdot T(n/4) + 3 + 1$ — (3)

$= 3^2 \cdot T(n/2^2) + \sum_{i=0}^{1} 3^i$

$= 3^3 \cdot T(n/2^3) + \sum_{i=0}^{2} 3^i$

$= 3^k \cdot T(n/2^k) + \boxed{\sum_{i=0}^{k-1} 3^i}$

$= 3^k \cdot 1 + \frac{3^k}{2} - \frac{1}{2}$

$= \frac{3}{2} \cdot 3^k - \frac{1}{2}$

$= \frac{3^{k+1}}{2} - \frac{1}{2}$

$= \frac{1}{2}\left[3^{\log_2 n + 1} - 1\right]$

$O(n^{\log_2 3})$

$S_n = \frac{a(r^n - 1)}{r-1} = \frac{1(3^k - 1)}{3-1}$

$= \left(\frac{3^k - 1}{2}\right)$

#Q. Let T(n) be defined by T(1) = 7 and T(n + 1) = 3n + T(n) for all integers n ≥ 1. Which of the following represents the order of growth of T(n) as a function of n?

$T(n+1) = 3n + T(n)$

$\boxed{T(n) = T(n-1) + 3(n-1)}$

$T(1) = 7$

$\boxed{T(n) = T(n-1) + n}$

$\rightarrow O(n^2)$

A $\Theta(n)$

B $\Theta(n \log n)$

C $\Theta(n^2)$

D $\Theta(n^2 \log n)$

E $\Theta(2^n)$

#Q. Let $T(n)$ be defined by $T(0) = T(1) = 4$ and $T(n) = T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + T\left(\left\lfloor \frac{n}{4} \right\rfloor\right) + cn$ for all integers $n \geq 2$, where c is a positive constant. What is the asymptotic growth of $T(n)$?

Cost

A $\Theta(\log n)$

B $\Theta(n)$

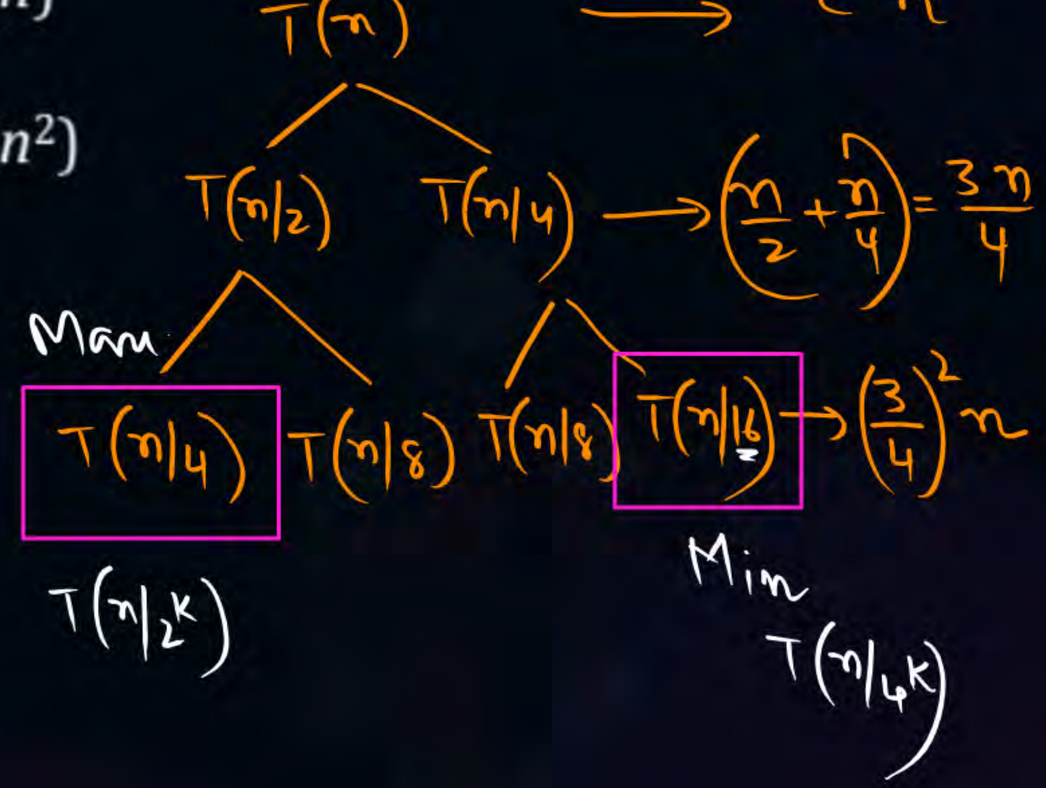
C $\Theta(n \log n)$

D $\Theta(n^2)$

E $\Theta\left(n^{\log_3 4}\right)$

$T(n) = T(n/2) + T(n/4) + n$

Recursion Tree



#Q. Consider a recursive algorithm for sorting an array of $n \geq 2$ integers that works as follows.

(a) If there are only 2 elements to be sorted, compare them and swap them if they are out of order.

(b) Otherwise, do the following steps in order.

(1) Recursively sort the first $n-1$ elements of the array.

(2) In the resulting array, recursively sort the last $n-1$ elements.

(3) In the resulting array, recursively sort the first 2 elements of the array.

What is the asymptotic running time complexity of this algorithm measured in terms of the number of comparisons made?

A $\Theta(n \log n)$

B $\Theta(n^2)$

C $\Theta(n^3)$

D $\Theta(2^n)$

E $\Theta(3^n)$

$T(n) = c, \quad n = 2$

$= T(n-1) + T(n-1) + a, \quad n > 2$

$= 2T(n-1) + a \quad - (1)$

$O(2^n)$

#Q. Mergesort works by splitting a list of n numbers in half, sorting each half recursively, and merging the two halves, Which of the following data structures will allow mergesort to work in O(n log n) time?

I. A singly linked list

II. A doubly linked list

III. An array

A. None

B. III only

C. I and II only

D. II and III only

E. I, II and III

#Q. The time complexity to multiply two long integers of size n- digit represented in an array of size n, using divide & conquer strategy with 3-way split is-

A. $O(n)$

B. $O(n\log n)$

C. $O(n^{1.32})$

D. $O(n^2)$

2 mins Summary

Topic One

Topic Two

Topic Three

Topic Four

Topic Five

THANK - YOU

CS & IT ENGINEERING

Algorithms

Lecture No.- 03

1500 Series

By- Dr. Khaleel Khan sir

Recap of Previous Lecture

Topic ALGORITHMS 01

Topics to be Covered

Topic D and C

Topic Greedy Method

#Q. The time complexity to multiply two long integers of size n- digit represented in an array of size n, using divide & conquer strategy with 3-way split is-

A. $O(n)$

B. $O(n\log n)$

C. $O(n^{1.32})$

D. $O(n^2)$

LIM <Long Integer Multiplication>

1) Non-DC (School Method) : $O(n^2)$

2) (D and C)

2-way Split | 3-way Split | K-way Split

$O(n^2)$

$T(n) = 4T(n/2) + n$

$T(n) = 9 \cdot T(n/3) + n$

$T(n) = K^2 \cdot T(n/K) + n$

$O(n^2)$

LIM

3) A.K optimiz. (DC)

2-way

$T(n) = 3 \cdot T(n/2) + n$

3-way

$T(n) = 8T(n/3) + n$.

K-way Split

$T(n) = (K^2 - 1) T(n/K) + n$

4) Toom & Cook :

2-way Split

$T(n) = 3T(n/2) + n$

3-way-Split

$T(n) = 5T(n/3) + n$.

K-way Split

$T(n) = (2K-1) \cdot T(n/3) + n$

#Q. The recurrence equation arises in solving which of the following problems:

MSQ

$T(n) = T(n/2) + 1,$

$T(1) = 1$

A. Towers of Hanoi : $T(n) = 2T(n-1) + c$

B. Merge sort : $T(n) = 2T(n/2) + n$

C. Binary search : $T(n) = T(n/2) + c$

D. Power of n (x^n) : $T(n) = T(n/2) + c$

→ D and C

#Q. Consider the following pseudocode of the recursive function goAgain.

```
//assume N is a positive integer
integer goAgain (integer a, integer N)
begin
  if(N ==1) then
        return a;
  else
        return (a + goAgain (a, N/2));
end
```

$T(n) = c, \quad N = 1$

$= a + T(n/2) + b, \quad N > 1$

$T(n) = T(N/2) + d$

$O(\log N)$

What is the order of growth, as a function of N, of the number of times the function goAgain is recursively executed?

A. $\Theta(\log N)$

B. $\Theta(N)$

C. $\Theta(N \log N)$

D. $\Theta(N^2)$

E. $\Theta(N^3)$

#Q. Which of the following algorithms has running time $\Theta(n^2)$ in the worst case but $\Theta(n \log n)$ on average?

A. Bubblesort : $n^2 + n^2$

B. Mergesort : $n \log n + n \log n$

C. Heapsort : $n \log n + n \log n$

D. Quicksort : $n^2 + n \log n$

#Q. Suppose that the splits at every level of quick sort are in the proportion $1-\alpha$ to α, where $0 < \alpha \le \frac{1}{2}$ is a constant. Minimum depth of a leaf in the recursion tree is approximately:

A. $-\frac{\log(n)}{\log(\alpha)}$

B. $\log(n\alpha)$

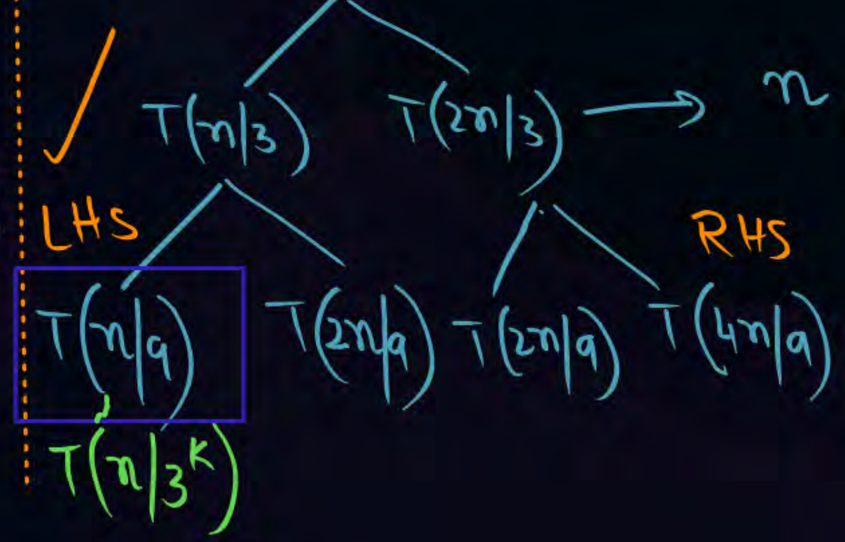
C. $n \log n$

D. $-\frac{n\log(n)}{\log(\alpha)}$

Let $\alpha = 1/3$

$n/3$ $\quad$ $2n/3$

$T(n) = T(n/3) + T(2n/3) + n$



$\frac{n}{3^k} = 1 \Rightarrow k = \log_3 n$

For general value of "α"

$T((\alpha)^k \cdot n) = 1$

$\Rightarrow (\alpha)^k \cdot n = 1$

$(\alpha)^k = 1/n$

$= n^{-1}$

$k \cdot \log \alpha = -\log n$

$\therefore$

$k = -\frac{\log n}{\log \alpha}$

#Q. Suppose that the splits at every level of quick sort are in the proportion $1-\alpha$ to α, where $0 < \alpha \leq \frac{1}{2}$ is a constant. Maximum depth of a leaf in the recursion tree is approximately:

$$T((1-\alpha)^k \cdot n) =$$

$$(1-\alpha)^k \cdot n = 1$$

$$k = -\frac{\log n}{\log(1-\alpha)}$$

A. $\log n$

B. $-\log(n\alpha)$

C. $-\dfrac{\log(n)}{\log(1-\alpha)}$

D. $n \log n$

#Q. An algorithm's real-time readiness (RTR) ratio is defined as the ratio of its average-case running time to its worst-case running time. Which of the following algorithm has an RTR ratio closest to 0?

$$RTR = \frac{A \cdot C}{W \cdot C}$$

A. Bubble sort $\frac{n^2}{n^2} = 1$

B. Heap sort $\frac{n \cdot \log n}{n \cdot \log n} = 1$

C. Insertion sort : $\frac{n^2}{n^2} = 1$

D. Merge sort $\frac{n \cdot \log n}{n \cdot \log n} = 1$

E. Quick sort : $\frac{n \log n}{n^2} \sim 0$

#Q. Consider the following statements:

S_1: Quick sort has divide procedure but not combine.

S_2 : Number of comparisons in partition algorithm are same for best case and worst case

S_3 : Merge sort always has more comparisons than swaps.

Which of the following is/are correct?

A. S_1 and S_2 only

B. S_2 and S_3 only

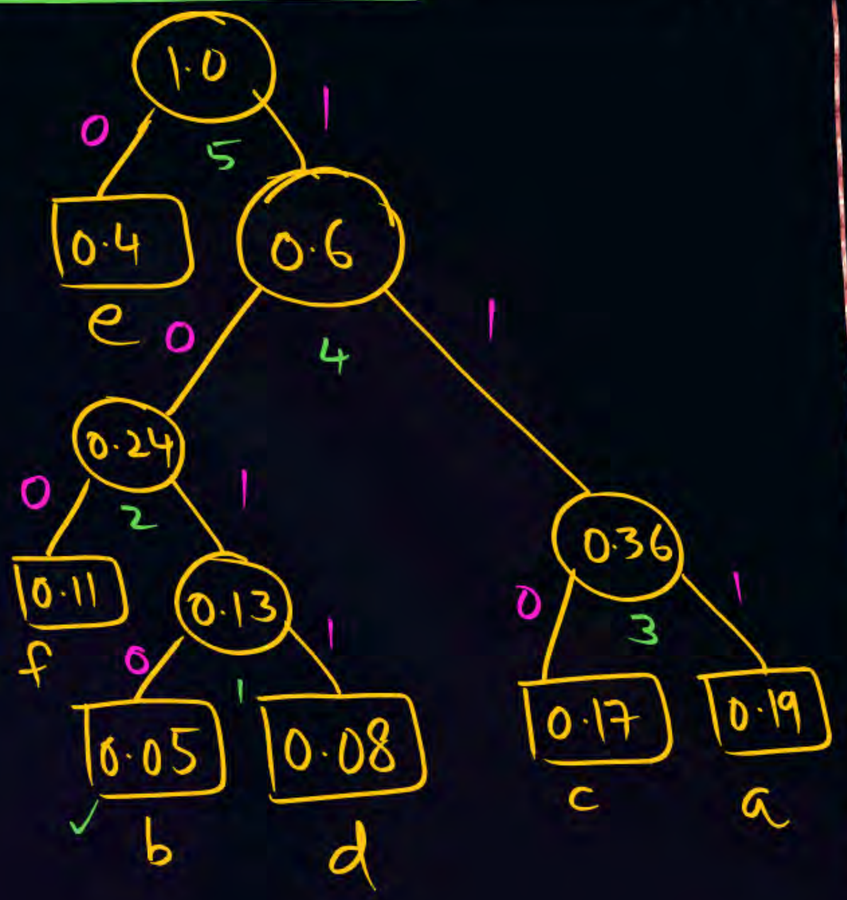
C. All are correct

D. None of the above

#Q. Consider a language L, defined as L=Σ(a – f) with the relative frequencies respectively given as (0.19, 0.05, 0.17, 0.08, 0.4, 0.11), then average length of prefix binary code has 2.33 bits.

1

Huffman Coding:



a → 111 (3)
b → 1010 (4)
c → 110 (3)
d → 1011 (4)
e → 0 (1)
f → 100 (3)

$$\text{Avg bits} = \sum_{i=1}^{n} d_i \cdot q_i$$

$$= (3*0.19 + 4*0.05 + 3*0.17 + 4*0.08 + 1*0.4 + 3*0.11)$$

$$=$$

#Q. Two classical algorithms for finding a minimum spanning tree in a graph are Kruskal's algorithm and Prim's algorithm. Which of the following are the design paradigms used by these algorithms?

	Kruskal's algorithm	Prim's Algorithm
A	The greedy method	The greedy method
B	The greedy method	Dynamic programming
C	Dynamic programming	The greedy method
D	Dynamic programming	Divide and conquer
E	Divide and conquer	Dynamic programming

#Q. Consider the following possible data structure for a set of n distinct integers.

H/W

I. A min-heap

II. An array of length n sorted in increasing order

III. A balanced binary search tree

For which of these data structure is the number of steps needed to find and remove the 7th largest element in $O(\log n)$ in the worst case?

A. I only

B. II only

C. I and II

D. I and III

E. II and III

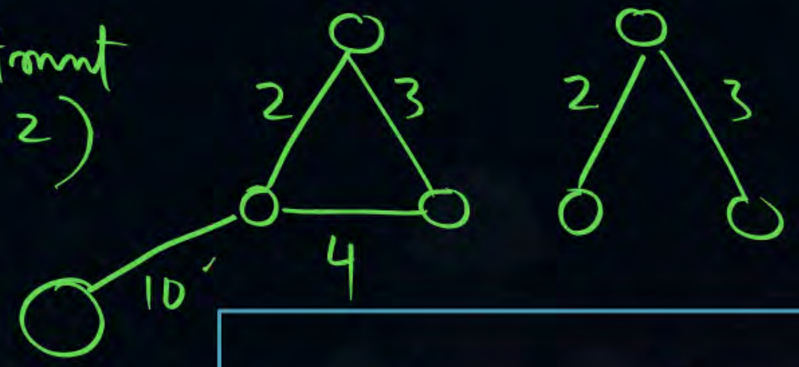
[NAT]

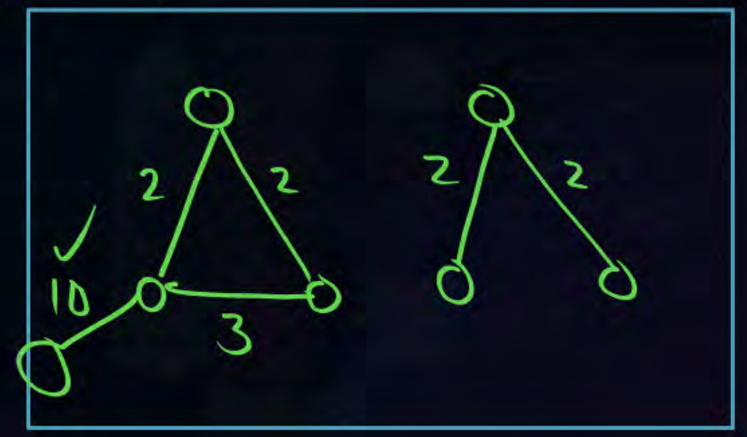
#Q. Consider the following statement with respect to a connected undirected graph G:

I : If G has a unique minimal spanning tree then all its edges are unique.

II : If G has all unique edges, then it always has a unique minimum spanning tree.

The number of correct statement is/are 1 (Stmnt 2).





#Q. Which of the following properties must be true of a Minimum Spanning Tree (MST) of a connected graph G with at least 3 edges?

I. The MST must contain the shortest edge of G.
II. The MST must contain the second-shortest edge of G.
III. The MST can never contain the longest edge of G.

A. None
B. I only
C. I and II only
D. I and III only
E. I, II and III

#Q. A hiker faces the 0/1 knapsack problem. There are 7 items to be packed into the knapsack, each with value v_i and weight w_i as shown in the following table.

i	1	2	3	4	5	6	7
v_i	3	6	8	1	2	5	7
w_i	7	3	5	1	4	2	6

Profit v_i — weight w_i

0.42, 2, 1.6, 1, 0.5, 2.5, 1.16

2, 3, 5, 6, 1, 4

$\Rightarrow 29$

0/1 KNAP GM H.A

H/W optimal Algo.

{0/1 KNAP D.P}

M

The knapsack, which is initially empty, can hold a maximum weight of 24, so some item(s) must be left behind, and fractions of items cannot be packed. The optimality criterion is to maximize the total value of the items that are placed in the knapsack. The hiker fills the knapsack one item at a time, using a heuristic algorithm that is greedy on value density, where the value density of an item is its value/weight ratio. When this heuristic algorithm is used, what is the total value of the items that are packed, and is this total optimal?

P_i/w_i

	Total Value	Optimal / Not Optimal
A	29	Optimal
B	29	Not optimal
C	30	Optimal
D	30	Not optimal
E	32	Optimal

[NAT]

<J.S.D>

#Q. Consider the following tasks with their profits and deadlines. What is the number of task than cannot be scheduled? (If the answer is 2 that means T_2 cannot be scheduled)

T_5

Tasks	Profit	Deadline
T_1	20	4
T_2	12	4
T_3	17	3
T_4	15	3
T_5	4	4
T_6	3	5

T_2	T_4	T_3	T_1	T_6	

0 1 2 3 4 5

#Q. The Character 'p' to 'u' have the set of frequencies based on the given table:

H/W

Character	Frequency
p	10
q	20
r	24
s	11
t	4
u	18

87

Frequencies :

Probab : <root : 1.0>

$\frac{Freq_i}{\sum_{i=1}^{n} Freq_i}$

$P = 10/87$

$q = 20/87$

Which of the following is/are correct based on Huffman code?

A. For character sequence "t p t u q" corresponding hexadecimal code is "E F E 1".

B. For character sequence "t p p q u" corresponding hexadecimal code is "E F F 1".

C. For bit stream "10101100100" sequence of the character is "r r s q u".

D. For bit stream "10101100001" sequence of character is "r r s u q".

2 mins Summary

Topic One

Topic Two

Topic Three

Topic Four

Topic Five

D and C

Greedy Method

THANK - YOU

CS & IT ENGINEERING

Algorithms

1500 series

Lecture No.- 04

By- Dr. Khaleel Khan sir

Recap of Previous Lecture

Topic ALGORITHMS 03

Topics to be Covered

Topic D and C

Topic Greedy Method

#Q. Which of the following represents the order of the edges while construction of MST using Prim's algorithm.

A. (C, A) (C , F) (A, G) (G, F) (F, E) (E, D) (D, C) (G, H) (I, J)

B. (F, E) (C, A) (C, F) (C, D) (A, G) (I, J) (F, I) (A, B) (D, E)

C. (D, E) (E, F) (F, C) (C, A) (A, G) (F, I) (I, J) (G, H) (A, B)

D. None of these.

#Q. Consider the following statements.

S1 : In a connected graph G with distinct weighted edges, different spanning trees of G will have different total weights. F

S2 : In a connected graph G with weighted edges, if edge e has a weight smaller than any other edge in G, then e must be included in every minimal spanning tree of G. F

Least Cost Edge

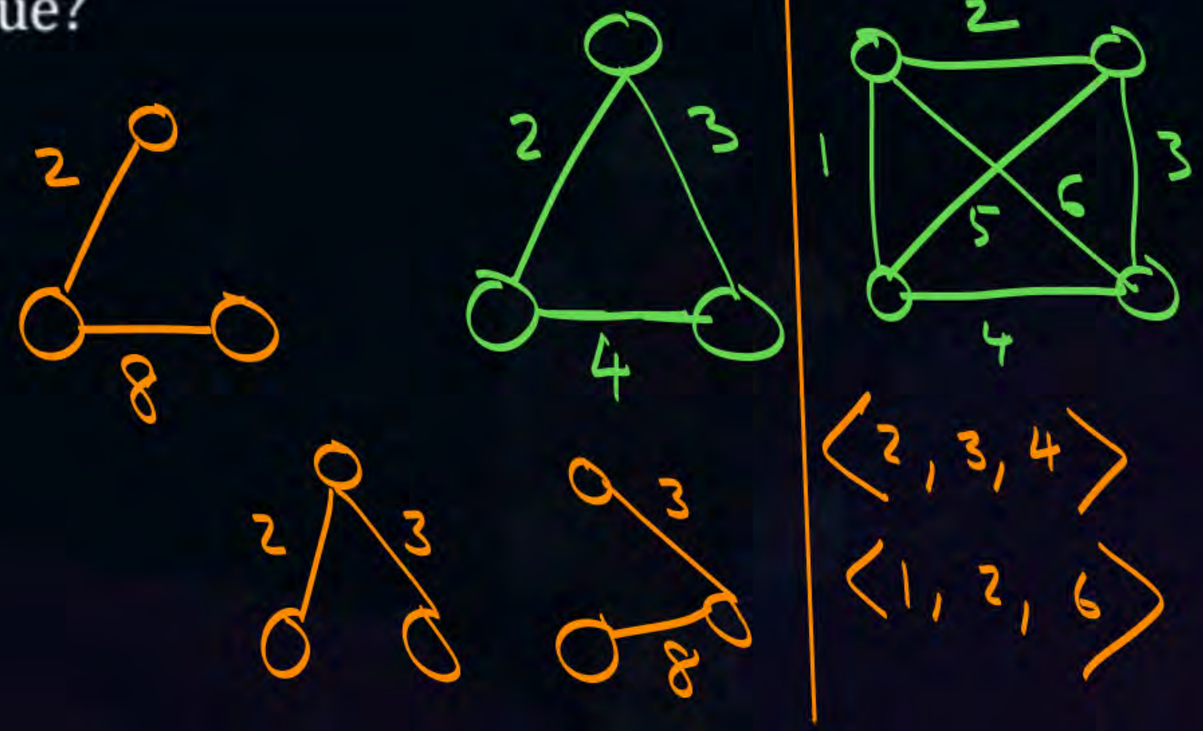
Which of the above statement is/are true?

A. Only S_1

B. Only S_2

C. Both S_1 and S_2

D. Neither S_1 nor S_2



#Q. Which of the following algorithm design techniques can be used for solving the problem of 0/1 Knapsack? <MSQ>

<A+C>

A. Backtracking

B. Greedy method

C. Dynamic Programming

D. Divide & conquer

#Q. Which of the following problems can be solved by Dynamic Programming

Backtracking

A. N-Queens

B. 0/1 Knapsack

C. Sum of Subsets

D. Optimal Binary Search Tree

#Q. What is the number of element comparisons needed to sort n elements using Radix sort (Non-Comparison based)

A. 0

B. n

C. nlogn

D. n^2

#Q. Let the numbers 794, 332, 561, 342, 200, 607 and 893 be sorted using RADIX SORT. What will be the sixth number in the sequence of numbers after sorting the second digit?

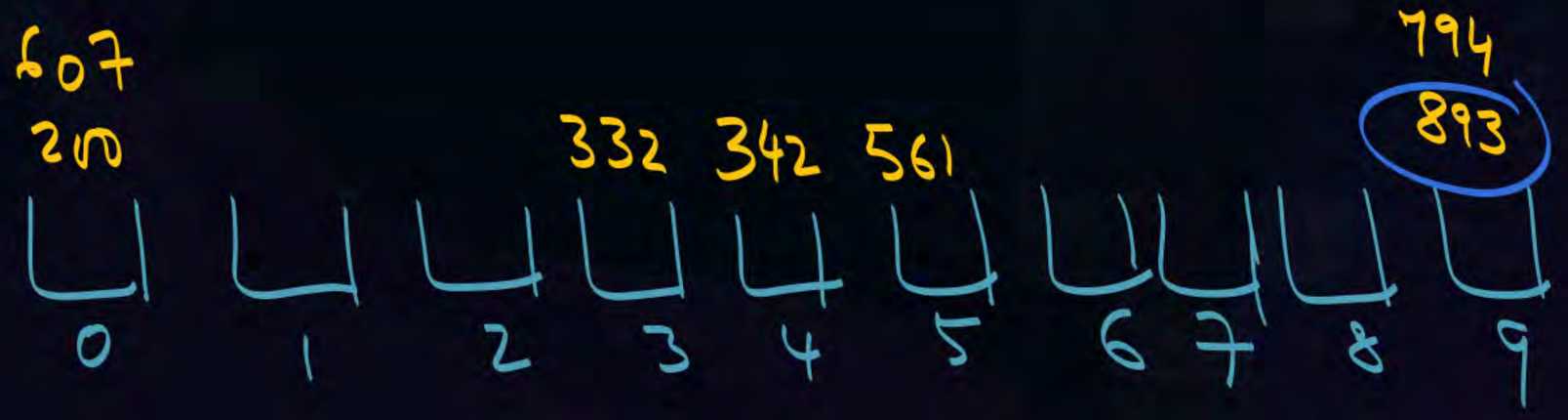
Pass 1 : 200 ; 561 ; 332 ; 342 ; 893 ; 794 ; 607

A. 561

B. 794

C. 893

D. 332



#Q. Which algorithm can be used to sort n elements in the range $[1.....n^k]$ $k > 0$, in linear time

A. Radix sort

B. Selection sort

C. Address calculation sort

D. Merge sort

#Q. Consider the following array with 9 elements

a[9] = {25, 20, 15, 35, 30, 16, 11, 7, 105}

Count the total number of swaps done, if insertion sort is applied on the above array __21__.

15	20	25	35	

: 1+2+

#Q. Which of the following algorithms may use Heap data structures in its implementation

<MSQ>

<A+B+C>

A. Prim's algorithm ✓

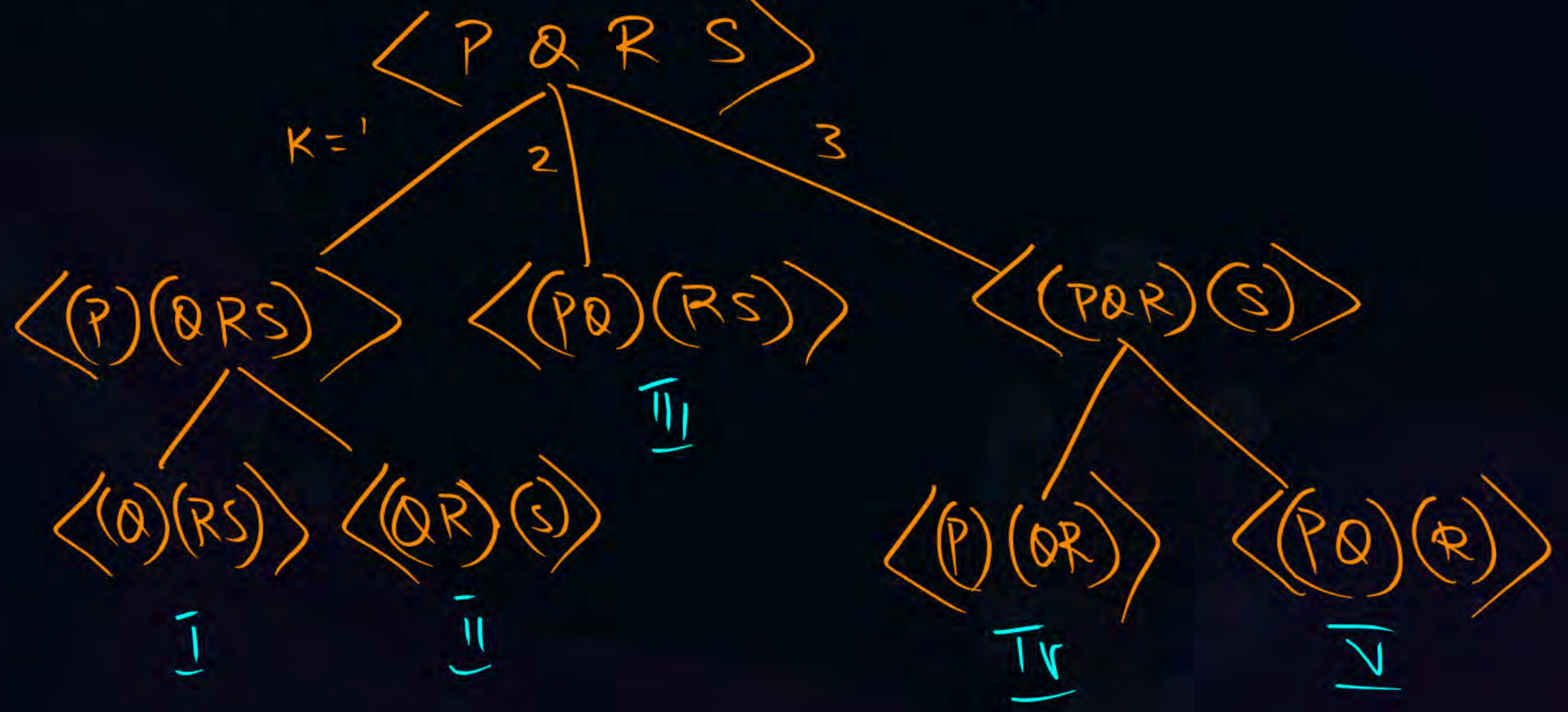
B. Kruskal's algorithm ✓

C. Shortest Job First ✓

D. Round Robin ✗

#Q. Let P, Q, R and S be four matrices of dimensions

20 × 15, 15 × 25, 25 × 20 and 20 × 5 respectively. The total optimal number of multiplication using the best parenthesization is 5875 (?).



Next 2 Questions refer to the following information.

The All Pairs Shortest Path Problem can be specified as follows.

Input:

Directed graph G(V, E), where V = {1, 2,..., n}

Cost C(i, j) $\in R^+ \cup \{\infty\}$ for all i, j $\in$ V, where C(i, j) = ∞ if and only if (i, j) $\notin$ E

Definition:

D(i, j) is the length of the shortest path from i to j for all i, j $\in$ V.

If there is no path from i to j, then D(i, j) = ∞.

If i = j, then D(i, j) = 0.

$$A^k(i,j) = \min\{A^{k-1}(i,j),\ A^{k-1}(i,k) + A^{k-1}(k,j)\}$$

Problem:

Find D(i, j) for all i, j $\in$ V.

The Floyd-Warshall algorithm gives a dynamic programming solution to the problem by defining an array A(k, i, j) for $0 \le k \le n$ and i, j $\in$ V by the following condition.

A(k, i, j) is the length of a shortest path from i to j such that all intermediate nodes on the path are in {1, 2,...,k} (where no intermediate nodes are allowed if k = 0).

Then D(i, j) = A(n, i, j).

The algorithm computes A(k, i, j) using a recurrence on k. where the initial step is given as follows.

A(0, i, j) = C(i, j) for all i, j $\in$ V such that I $\neq$ j

A(0, i, i) = 0 for all i $\in$ V

#Q. Which of the following is the general step in the recurrence, where $1 \le k \le n$?

A $A(k,i,j) = \min_{l<k}\{A(l,i,k\} + A(l,k,j)\}$

B $A(k,i,j) = \min_{l<k}\{A(k-1,i,l) + A(k-1,l,j)\}$

C $A(k,i,j) = \min\{A(k-1,i,j), A(k-1,i,k) + A(k-1,k,j)\}$

D $A(k,i,j) = \min\{C(i,j), A(k,i,k) + A(k,k,j)\}$

E $A(k,i,j) = \min\{C(i,j), A(k-1,i,k) + A(k-1,k,j)\}$

#Q. Which is the running time of the Floyd-Warshall's Algorithm?

A $\Theta(n)$

B $\Theta(n^2)$

C $\Theta(n^3)$

D $\Theta(n^3 \log n)$

E $\Theta(n^4)$

#Q. The Transitive Closure of a binary relation on a set of 'n' elements can be found by which of the following algorithm-

A. Kadane's Algorithm

B. Belady's Algorithm

C. Dijkstra's Algorithm

D. Floyd-Warshall's Algorithm

#Q. Considering the brute force approach in which we explore all possible ways of multiplying a given set of 'n' matrices, what is the time complexity of the matrix chain multiplication implementation?

Exponential (Catalan No)

$\frac{1}{(n+1)} {}^{2n}C_n$

$\Omega(2^n)$

A. Linear

B. Cubic polynomial

C. Quadratic polynomial

D. Exponential

#Q. Which of the following is incorrect?

A. Floyd-Warshall is based on Dynamic Programming.

B. Kruskal is based on Greedy approach.

C. Sum of Subsets problem is solved by Dynamic Programming

D. None of these.

#Q. Consider the following graph:

⟨MSQ⟩

⟨C+D⟩

Which of the following is/are correct BFS traversing order?

A. G E H F D A B C

B. H E A F G D B C

C. F G H E D A B C

D. C A B H D F E G

#Q. Consider the following directed graph.

Which of the following is a Topological sort of the nodes of the graph?

A. 5, 7, 10, 13, 14, 17, 20, 30

B. 10, 5, 13, 14, 7, 30, 17, 20

C. 10, 5, 13, 17, 20, 14, 7, 30

D. 10, 5, 20, 13, 17, 30, 14, 7

E. 10, 20, 5, 17, 13, 14, 7, 30

#Q. Which of the following data structure is used in topological sort on directed graph?

D.F.S
<Stack>

A. Heap

B. Linked list

C. Array

D. Stack

#Q. What is the time complexity to find the maximum element in a min heap of n-elements?

Max-Heap (using Heapify)

A. $\theta(\log n)$

B. $\theta(1)$

C. $\theta(n)$

D. $\theta(n \log n)$

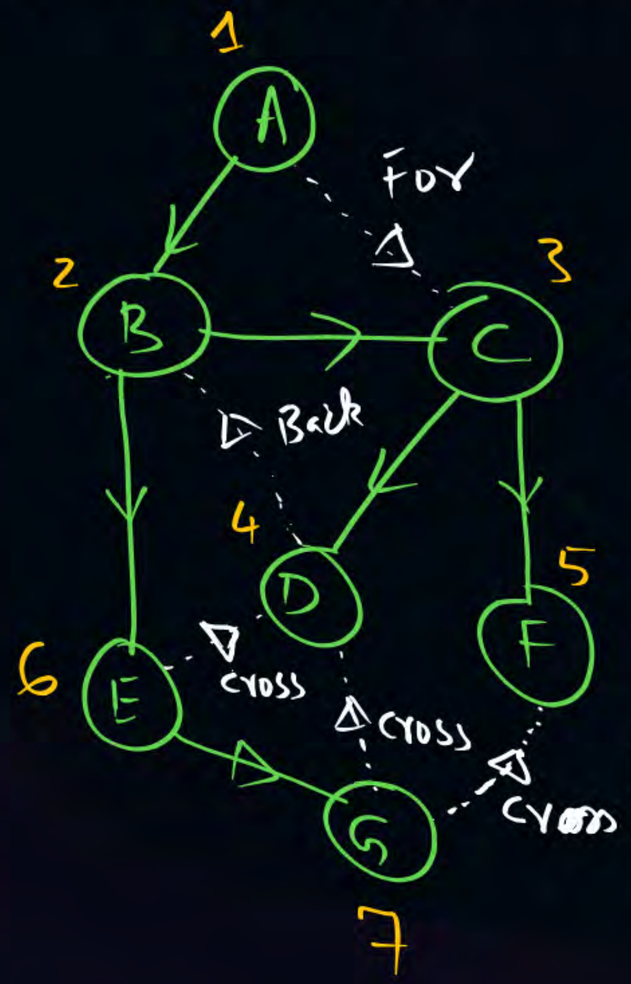
#Q. Consider the given graph. If DFS algorithm runs from vertex A and adjacency are chosen based on lexicographic order of vertex labels. Then the number of possible tree edges a, back edges b, cross edges f and forward edges g are:

A. a = 6, b = 2, g = 3, f = 4

B. a = 6, b = 1, f = 1, g = 4

C. a = 6, b = 1, f = 3, g = 1

D. a = 3, f = 2, g = 1, b = 2



#Q. Number of Topological Sort Sequences possible from the given Graph is ___8___.

$2 \times 2 \times 2 = 8$

#Q. Which of the following option(s) is/are correct? <MSQ>

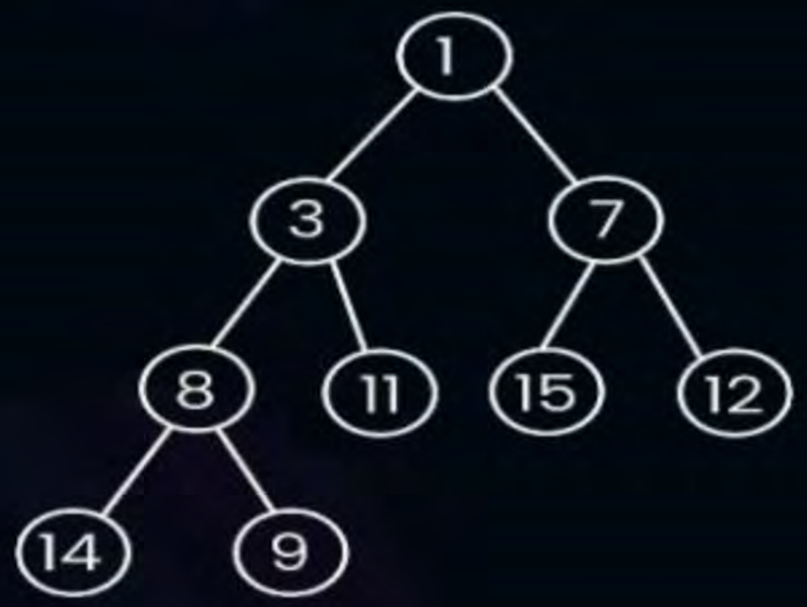
A. Time complexity of Breadth first search and Depth first search are same as O(V+E) in case of adjacency list.

B. Time complexity of Breadth first traversal and Depth first traversal are same as O(V+E) in case of adjacency list.

C. Space complexity of Breadth first search and Depth first search are same as O(V).

D. Time complexity of ~~Breadth~~ Depth first search and Breadth First traversal are same as $O(v^2)$ in case of adjacency matrix.

#Q. Consider the given min heap:



What will be the result from doing 2 delete min. operations on the above given min. heap?

A

B

C

D

#Q. Consider a max heap-

85, 83, 80, 75, 76, 48, 15, 32, 68, 18

A list of operations are performed-

Del(max), insert(87) delete(18), insert(82), inset(81), delete(max)

What is the content of the max heap after these operations?

A. 83, 75, 80, 68, 32, 76, 81, 48, 15, 82.

B. 83, 82, 80, 75, 81, 48, 15, 68, 32, 76.

C. 83, 82, 80, 81, 76, 48, 15, 32, 68, 75

D. 83, 82, 80 75, 81, 48, 15, 32, 68, 76

#Q. Given a directed weighted graph having n-vertices and n edges (edge weight may be negative) . The time complexity of the most efficient algorithm to determine whether there are paths of large weights is-

A. O(nlogn)

B. $O(n\sqrt{n})$

C. $O(2^n)$

D. $O(n^3)$

#Q. Which of the following is not implied by P = NP? Cook's Theorem

A. 3 SAT can be solved in polynomial time

B. Factoring can be solved in polynomial time

C. Halting problem can be solved in polynomial time

D. TSP can be solved in polynomial time

#Q. Consider the following possible data structure for a set of n distinct integers.

I. A min-heap

II. An array of length n sorted in increasing order

III. A balanced binary search tree

For which of these data structure is the number of steps needed to find and remove the 7th largest element in $O(\log n)$ in the worst case?

A. I only

B. II only

C. I and II

D. I and III

E. II and III

#Q. The Character 'p' to 'u' have the set of frequencies based on the given table:

Character	Frequency
p	10
q	20
r	24
s	11
t	4
u	18

Which of the following is/are correct based on Huffman code?

A. For character sequence "t p t u q" corresponding hexadecimal code is "E F E 1".

B. For character sequence "t p p q u" corresponding hexadecimal code is "E F F 1".

C. For bit stream "10101100100" sequence of the character is "r r s q u".

D. For bit stream "10101100001" sequence of character is "r r s u q".

2 mins Summary

Topic One D and C

Topic Two Greedy Method

Topic Three

Topic Four

Topic Five

THANK - YOU