CS & IT ENGINEERING

Algorithm

Miscellaneous Topics

Lecture No.- 09

By- Dr. Khaleel Khan sir

Recap of Previous Lecture

Topic Multistage Graph

Topic Travelling Salesperson Problem

Topics to be Covered

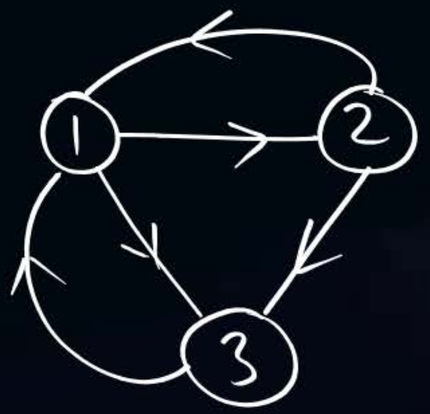
Topic All Pairs Shortest Paths

Topic 0/1 Knapsack

LCS

Topic : Dynamic Programming: (DP)

3) All-Pairs Shortest Paths <Floyd-Warshall's Algo>



Let $A^K(i,j)$ refer Cost of the Path from vertex 'i' (Src) to vertex 'j' (Dest), with 'K' being the highest Intermediate vertex along the path;

$A^K(i,j)$: (i) ... (K) .. (j)

C	1	2	3
1	0	4	11
2	6	0	2
3	3	∞	0

Src ... Dest

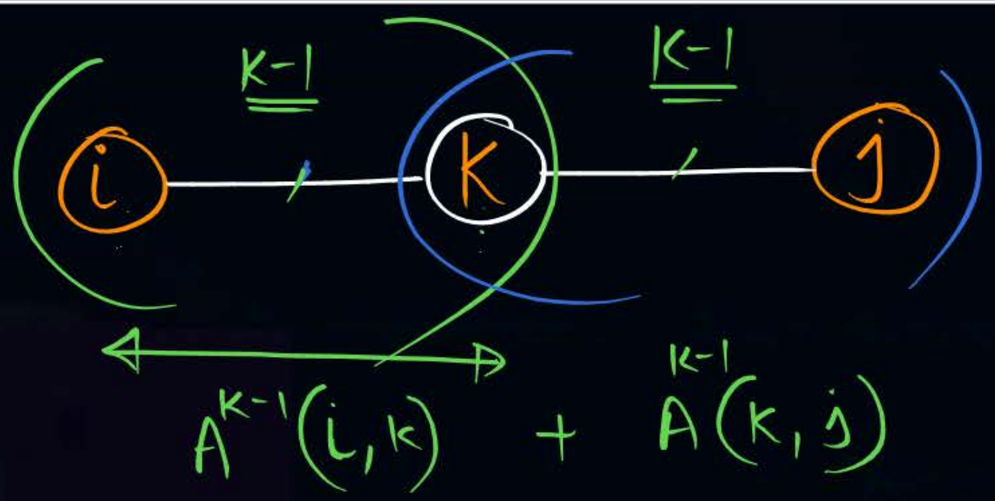
(3) - - - - (4)

3-6-5-1-4

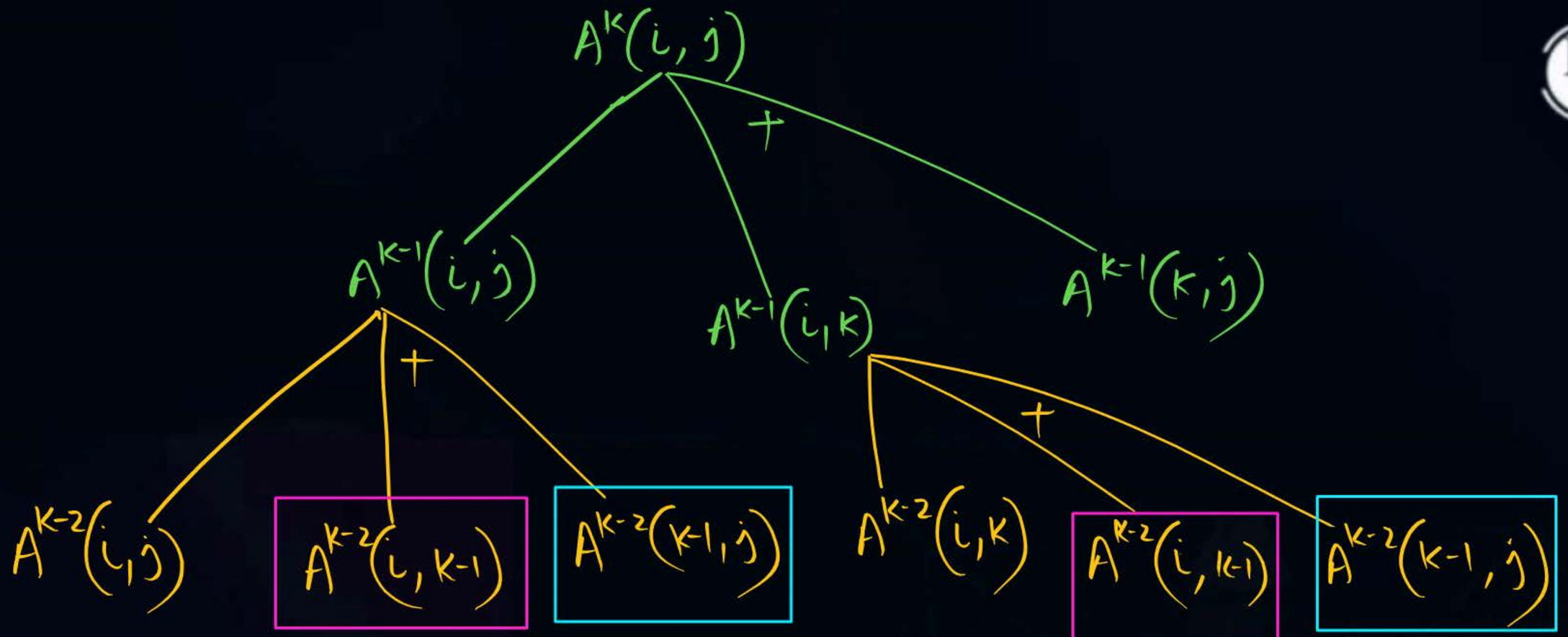
3-2-10-1-4

3-8-9-10-4

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

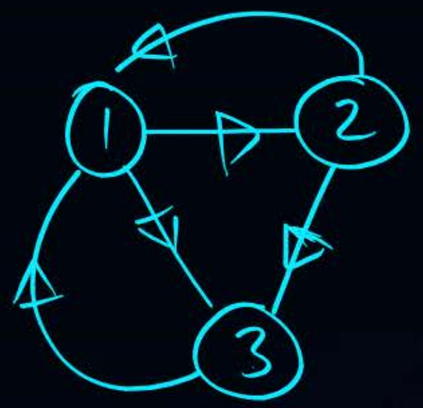


$A^0(i,j) = c(i,j)$


$A^k(i,j)$
+
$A^{k-1}(i,j)$
$A^{k-1}(i,k)$
$A^{k-1}(k,j)$
+
+
$A^{k-2}(i,j)$
$A^{k-2}(i,k-1)$
$A^{k-2}(k-1,j)$
$A^{k-2}(i,k)$
$A^{k-2}(i,k-1)$
$A^{k-2}(k-1,j)$

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

$A^3 \Longrightarrow A^2 \Longrightarrow A^1 \Longrightarrow A^0$



C	1	2	3
1	0	4	11
2	6	0	2
3	3	∞	0

A^0	1	2	3
1	0	4	11
2	6	0	2
3	3	∞	0

A^1	1	2	3
1	0	4	11
2	6	0	2
3	3	7	0

$A^1(1,2) = \min\{ A^0(1,2),\ A^0(1,1) + A^0(1,2) \}$ (4, 0, +4)

$A^1(2,3)$

$2 - 1 - 3$

$(6 + 11) = 17$

A^2	1	2	3
1	0	4	6
2	6	0	2
3	3	7	0

A^3	1	2	3
1	0	4	6
2	5	0	2
3	3	7	0

$A^1(3,2)$

$3 - 1 - 2$

```
Algorithm FLOYD-WARSHAL(G, n, e, C
    integer C[1..n, 1..n]
{
    integer A[1..n, 1..n]

1.  for i ← 1 to n
        for j ← 1 to n
            A[i,j] = C[i,j];

2.  for K ← 1 to n : Intermediate vertex
        for i ← 1 to n : Src
            for j ← 1 to n : dest
                A[i,j] = min{ A[i,j] , A[i,K] + A[K,j] }   (2m)

}
```

Time: $O(n^3)$

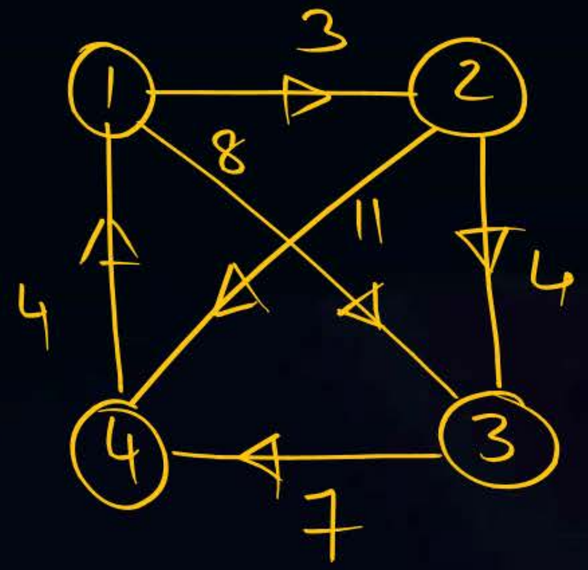
Space: $O(n^2)$

→ Transitive closure of a Matrix (representing a graph)

Flyod-warshall's Algo. can be used to obtain Transitive of a Matrix (repr. the graph)

↳ $O(n^3)$

$i \to j$
$j \to k$
$\Rightarrow i \to k$

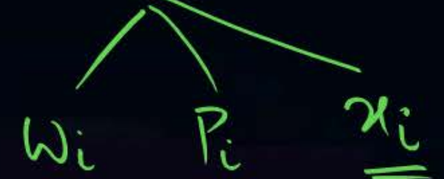

1
2
3
4
3
8
11
4
4
7

4) 0/1 KNAPSACK (Binary Knapsack)

→ KNAP Cap : M

→ no. of objects : n

(O_i)



$$\text{Max} \sum_{i=1}^{n} P_i x_i$$

$$\text{S.T.C} \sum_{i=1}^{n} w_i x_i \le M$$

$$x_i = 0/1$$

Sol^n Space : $O(2^n)$

$\langle x_1\ x_2\ x_3 \cdots x_n \rangle$

Let 0/1KNAP(n, M) repr. profit with 'n'-objects & KNAP of Capacity M;

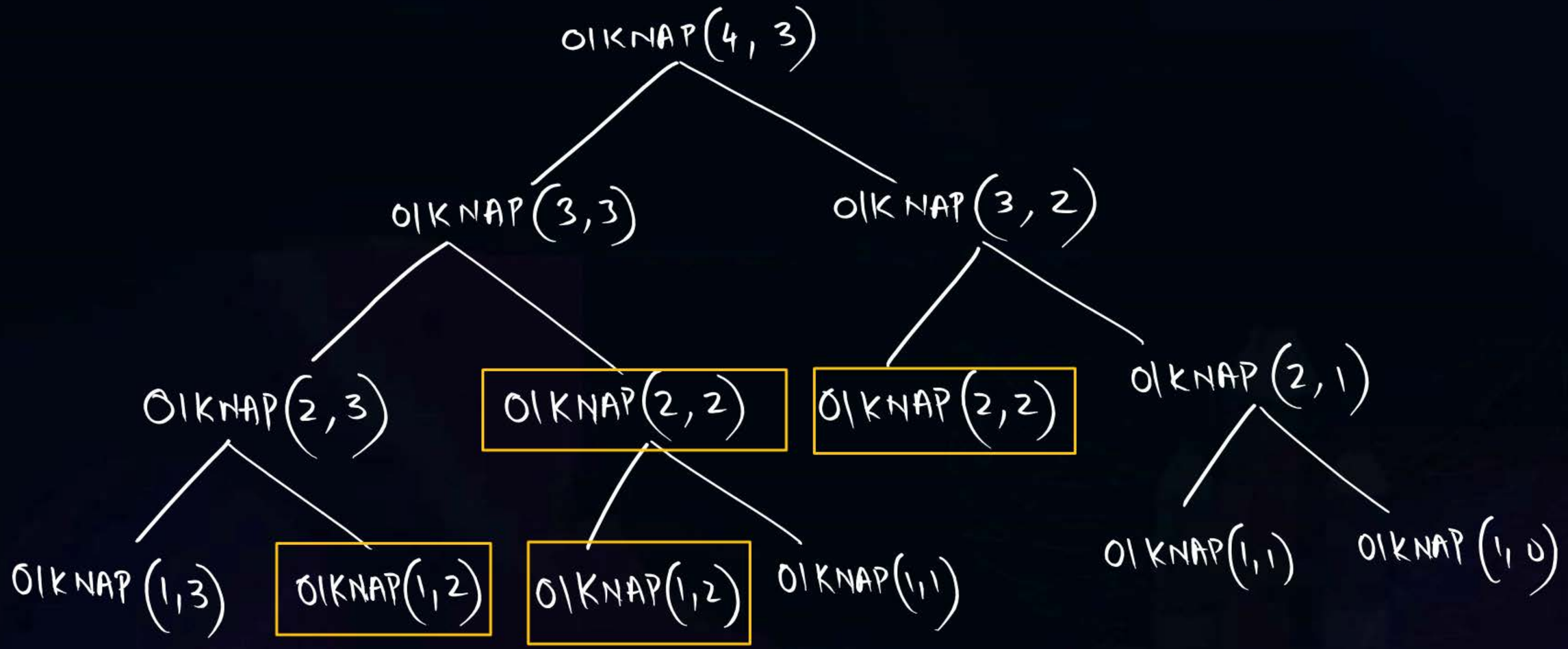
$$0/1KNAP(n,M) = 0/1KNAP(n-1,M), \quad w_n > M$$

$$= \text{Max}\begin{cases} x_n = 0 \\ 0/1KNAP(n-1,M), & w_n \le M \\ x_n = 1 \\ 0/1KNAP(n-1, M-w_n)+P_n \end{cases}$$

$$0/1KNAP(n,M) = 0, \quad n=0 \text{ or } M=0$$

$n = 4$; $M = 3$; $\langle w_1 \cdots w_4 \rangle = \langle 1, 1, 1, 1 \rangle$

$\langle P_1 \cdots P_4 \rangle = \langle 10, 20, 30, 40 \rangle$



Ex: $n=4$; $M=8$; $\langle w_1 \cdots w_4 \rangle = \langle 2,3,4,5 \rangle$; $\langle P_1 \cdots P_4 \rangle = \langle 1,2,5,6 \rangle$

Let $X[0..n, 0..M]$ be an array of which $X[n, M]$ = Profit

$X(4,8) =$

(Bottom-up-Tabulation)

P_i	w_i	i / X	0	1	2	3	4	5	6	7	8
		0	0	0	0	0	0	0	0	0	0
1	2	1	0	0	1	1	1	1	1	1	1
2	3	2	0	0	1	2	2	3	3	3	3
5	4	3	0	0	1	2	5	5	6	7	7
6	5	4	0	0	1	2	5	6	6	7	(8)

$\langle x_1, x_2, x_3, x_4 \rangle$

0 1 0 1

$x_4 = 1$

$8 - 6 = 2$

$x_3 = 0$

(2)

$x_2 = 1$

0 $x_1 = 0$

$$x[1,2] = \max\{x[0,2],\ x[0,1]+P_1\}$$

$$0\ ,\quad 0+1$$

$n=3;\quad M=6;$

$\langle P_1, P_2, P_3 \rangle = \langle 1, 2, 5 \rangle$

$\langle W_1, W_2, W_3 \rangle = \langle 2, 3, 4 \rangle$

Topic : Algorithms

Time : $O(n * M)$

Space : $O(n * M)$

```
Algo 01KNAP (M, n, ,W, P)
P [1.... n], W [1..... n] ; Input
{
      Integer   X[ 0..n, 0..M];
1.    for i ← 0 to n
          for j ← 0 to m
              If (i = 0  or  j = 0)
*                 x [i,j] = 0;
*     else
          if (W[i] ≤ j)
          {
              x[i, j] = max {x [i–1, j ], x[i–1, j-W [i]]+ P_i }
          }
          else
                x [i, j] = x [i–I , j];
          }
```

Boundary Condition

Slide 5

5) Longest Common Subsequence (LCS) < String Manipulation

1) String : a group of one/more characters,

Substring

A group of 1/more characters taken in Contiguous from the given String,

Subsequence

A group of one/more characters taken from the given String, that may not be Contiguous, but however their relative order is maintained.

Ex: < A B C > = 3 (1+2+3)

- <A>
- <B>
- <C>

} 3

- <A B>
- <B C>

} 2

- <A B C>

} 1

Q) Given a String of length n-characters, then the no of Substrings possible are $O(n^2)$;

1) ⟨A B C⟩

⟨A⟩ ; ⟨B⟩ ; ⟨B⟩

⟨A B⟩ ; ⟨B C⟩ ; ⟨A C⟩

⟨A B C⟩

⟨C A⟩ X

⟨C B⟩ X

⟨A B C D⟩

⟨1+2+3+4⟩

⟨ . . . n . . . ⟩

$1+2+3+\cdots+n$

$= \frac{n(n+1)}{2} + \emptyset$

$= O(n^2)$

String : <C A B D A B B C D>

: <B B D> ✓

<B A C> ✓

<C A D C> ✓

<A D A B D> ✓

<C D C A> ✗

Every Substring is also a Subsequence ;

Every Subsequence is NOT a Substring

Q2) Given a String of length n-characters, the no. of Subsequences possible are $O(2^n)$;

$\longrightarrow$ (Common Subsequence)

THANK - YOU