GATE

CRASH COURSE

DS & AI

Algorithms

Graph: Shortest Path
(Part 02) (Lecture 9)

By - Aditya sir

Topics to be Covered

1

2 Shortest Path Algos

3

4

About Aditya Jain sir

1. Appeared for GATE during BTech and secured AIR 60 in GATE in very first attempt - City topper
2. Represented college as the first Google DSC Ambassador.
3. The only student from the batch to secure an internship at Amazon. (9+ CGPA)
4. Had offer from IIT Bombay and IISc Bangalore to join the Masters program
5. Joined IIT Bombay for my 2 year Masters program, specialization in Data Science
6. Published multiple research papers in well known conferences along with the team
7. Received the prestigious excellence in Research award from IIT Bombay for my Masters thesis
8. Completed my Masters with an overall GPA of 9.36/10
9. Joined Dream11 as a Data Scientist
10. Have mentored working professions in field of Data Science and Analytics
11. Have been mentoring GATE aspirants to secure a great rank in limited time
12. Have got around 27.5K followers on Linkedin where I share my insights and guide students and professionals.

@ADITYASIR_PW

Telegram Link for Aditya Jain sir:

https://t.me/AdityaSir_PW

Topic : (Lecture Schedule)

3. Shortest Path Algos

1. Dijkstra ✓
2. Bellman Ford ✓ DP
3. Floyd Warshall
4. Multi-stage Graph
5. Travelling Salesman Problem

Questions

Topic : Algorithms

Algorithm Bellman-Ford (G, w, s)

```
Initialize-Single-Source(G,s)
for i ← 1 to |V[G]| – 1
    do for each edge (u, v) ∈ E[G]
        do Relax(u, v, w)
for each edge (u, v) ∈ E[G]
    do if d[u] > d[v] + w(u, v)
        then return FALSE
return TRUE
```

$\rightarrow O(n)$ $|V(G)| = n$

$\hookrightarrow O(e)$ $|E(G)| = e$

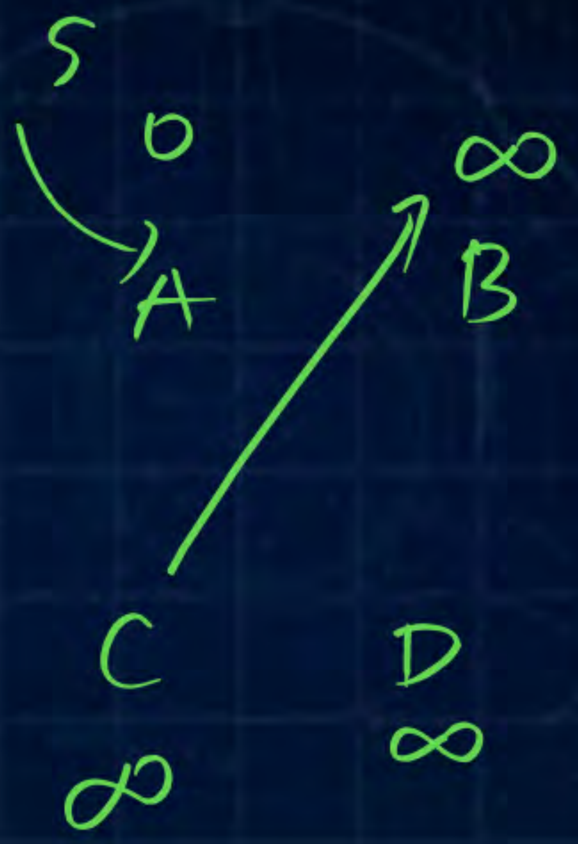
FALSE → There is a -ve wt cycle reachable from Source, hence BF does not give optimal ans

↳ it is solved by BF

Topic : Algorithms

Initialize – Single- Source (G, s)

1. For each vertex $u \in V[G]$
2. do $d[u] \leftarrow \infty$
3. $\pi[u] \leftarrow \text{NIL}$
4. $d[s] \leftarrow 0$



Relax (u, v, w)

1. if $d[v] > d[u] + w(u, v)$
2. ~~then $d[v] \leftarrow d[u] + w(u, v)$~~ → then $d[v] = d[u] + w[u, v]$
3. $\pi[v] \leftarrow u$

Time Complexity of Bellman Ford on $G(V, E)$

$|V| = n$

$|E| = e$

$$\boxed{TC = O(n * e)}$$

PYQ: Given a Complete graph with n vertices, the TC of Bellman Ford algo is? (in terms of n)

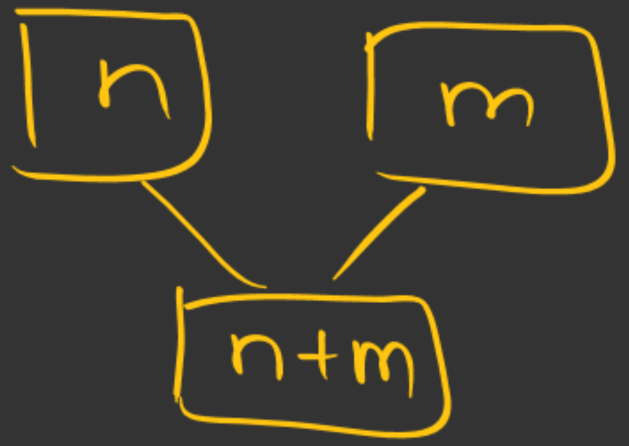
→ TC of BF = $O(n * e)$

Complete graph ⇒ $e = O(n^2)$

$$TC = O(n \times O(n^2)) = O(n^3)$$

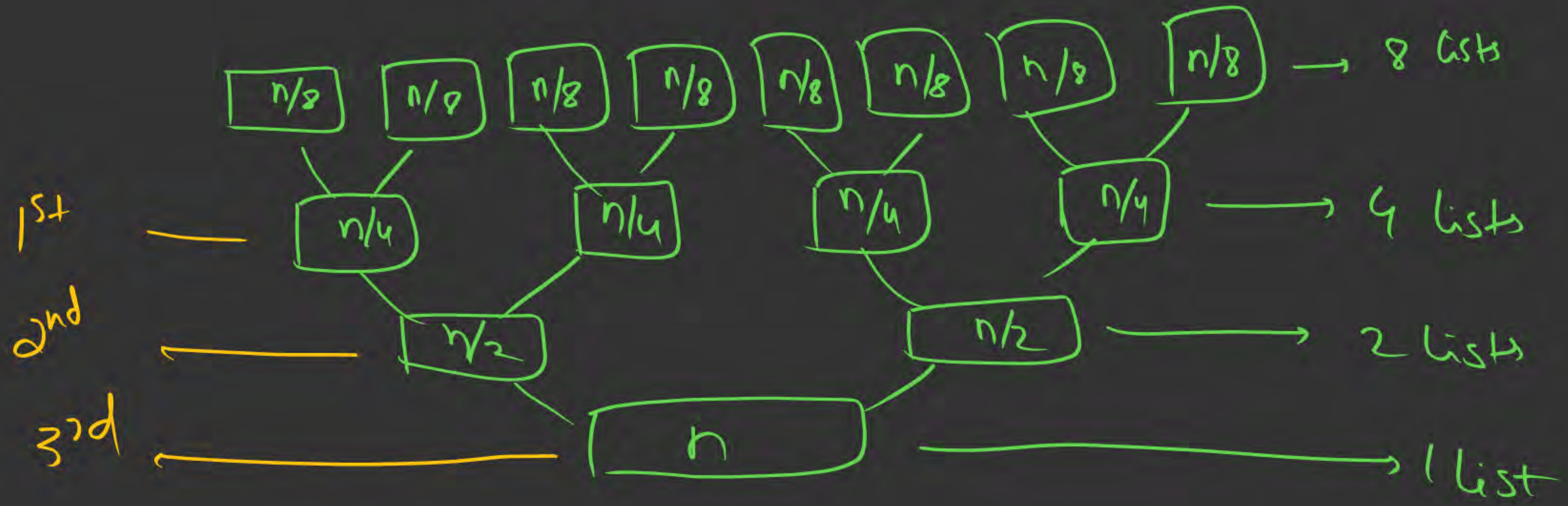
Question

#Q. Suppose, there are 8 sorted list of n/8 elements each if we merge them into a single sorted list of n elements, n is 1000 elements then , what is the* difference between key comparisons in worst case and best case ?



Best Case = min(m, n)

Worst Case = m+n-1



n/8
n/8
n/8
n/8
n/8
n/8
n/8
n/8
8 lists
n/4
n/4
n/4
n/4
4 lists
n/2
n/2
2 lists
n
1 list
1st
2nd
3rd

no. of Comp

1st level : Best Case :: $\min(n/8, n/8) = n/8$

(4 times) worst Case : $\left(n/4 - 1\right)$

2nd level: Best Case $= n/4$

(2 times) worst Case $= \left(n/2 - 1\right)$

3rd level : Best Case $= n/2$

(1 time) Worst Case $= (n-1)$

Overall Total Comp in Best Case

$$= 4\left(\frac{n}{8}\right) + 2\left(\frac{n}{4}\right) + 1 * \left(\frac{n}{2}\right)$$

$$= \frac{n}{2} + \frac{n}{2} + \frac{n}{2} = \boxed{\frac{3n}{2}}$$

Overall Total Comp in Worst Case

$$= 4\left(\frac{n}{4} - 1\right) + 2\left(\frac{n}{2} - 1\right) + 1 \times (n-1)$$

$$= (n-4) + (n-2) + (n-1) = (3n-7)$$

Reqd ans = WC − BC

$$= (3n-7) - \left(\frac{3n}{2}\right)$$

$$= \left(\frac{3n}{2} - 7\right)$$

For $n = 1000$

$$ans = \frac{3 \times 10\cancel{00}}{\cancel{2}} - 7$$

$$= 1500 - 7$$

$$= \boxed{1493}$$

Question

#Q. Assume that there are 8 sorted lists of n/8 elements each, if these lists are merged into a single sorted list of 'n' elements then how many key comparisons are required in the worst case using an efficient algorithm?

A) $3n - 7$

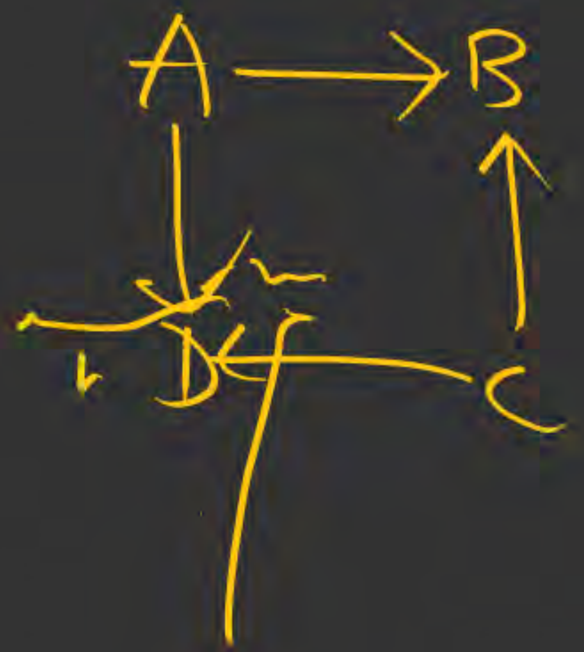
B) $\frac{7}{4}n - 3$

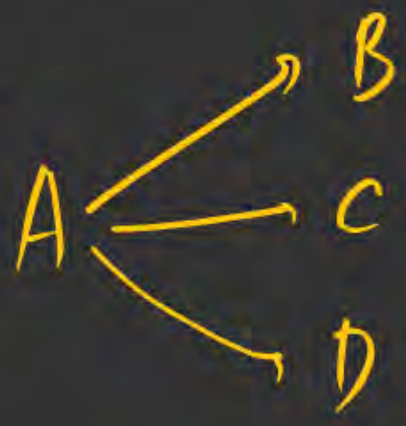
C) $7n - 3$

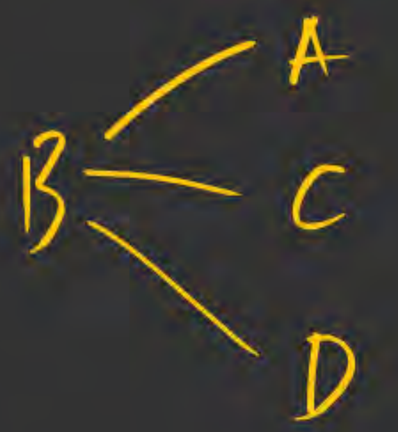
D) $\frac{6}{4}n - 3$

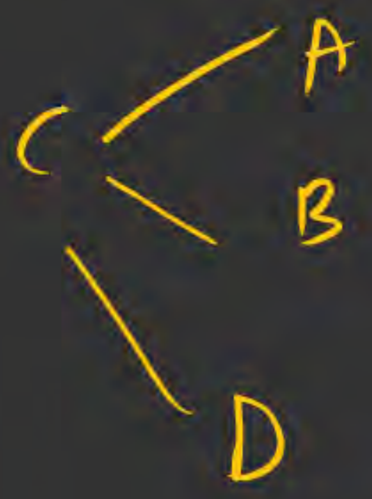
3) All Pairs Shortest Paths (APSP) $\longrightarrow$ DP

Floyd Warshall









Dijkstra → <u>SSSP</u>

Src = A → B

Src = A → C

Src = A → D

Src = B

Src = C

Src = D

Floyd Warshall $\longrightarrow$ <u>DP Recurrence</u> (<u>Top-Down</u>)

Let $A^{k}(i,j)$ $\qquad i \xrightarrow[k]{} j$

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

$$1 \leq k \leq n$$

<u>Base Condition</u>

$$\underline{A^{0}(i,j) = C[i,j]}$$

cost adj matrix

Bottom-up

```
Algo FloydWarshall(G, c, n, e)
{
    A[1...n, 1...n]

    for(i=1; i<=n; i++)
        for(j=1; j<=n; j++)
            A[i,j] = c[i,j]

    for(k=1; k<=n; k++)
        for(i=1; i<=n; i++)
            for(j=1; j<=n; j++)
                A[i,j] = min(A[i,j], A[i,k] + A[k,j])
}
```

$\rightarrow O(n^2)$

$n \times n$

$\rightarrow O(n^3)$

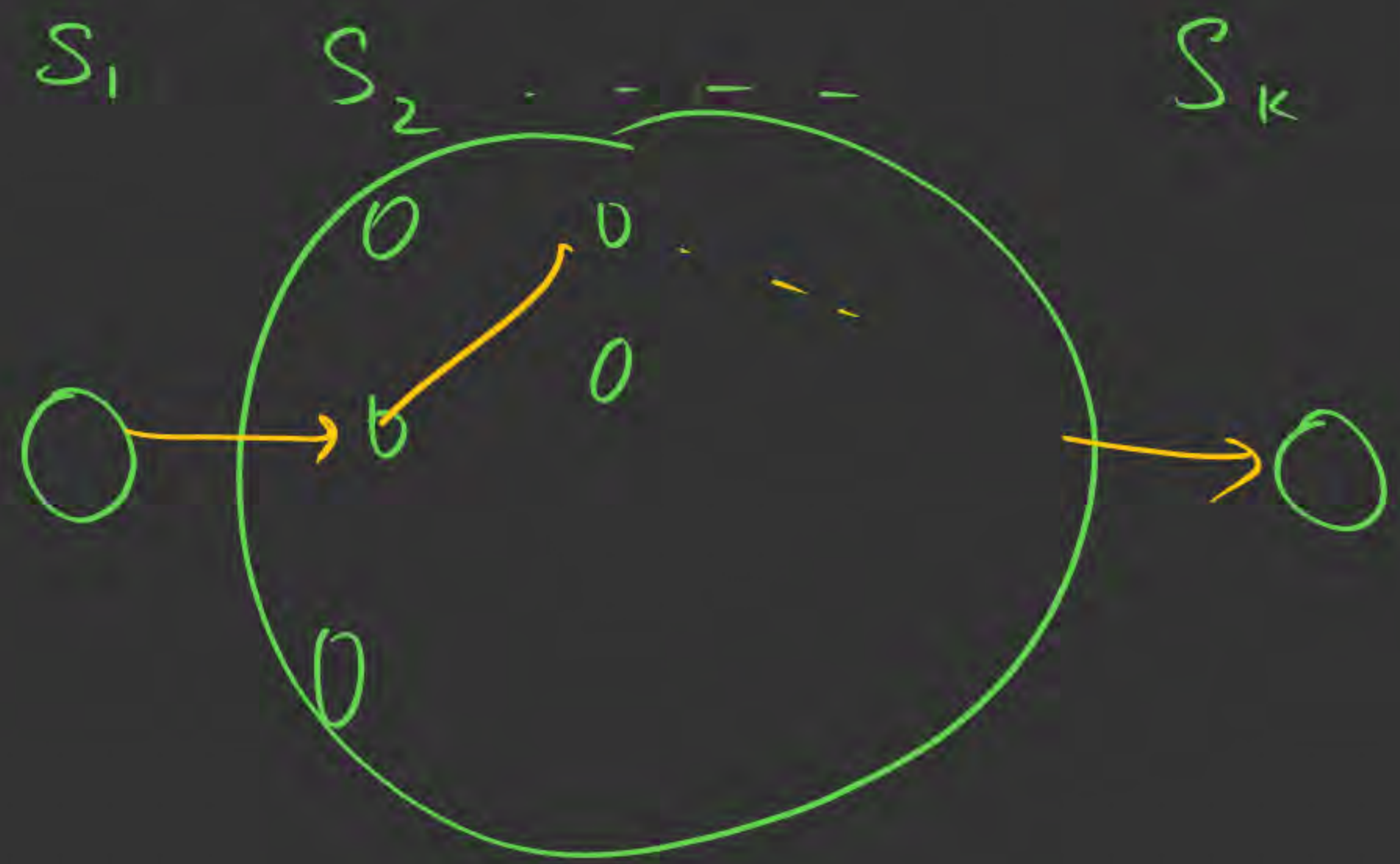
overall TC of Floyd Warshall

$|V| = n$

$$= O(n^2 + n^3)$$

$$= O(n^3)$$

4) Multi-Stage Graph :



$(V_i \longrightarrow V_{i+1})$

Cost adj matrix

$C[i, j]$

DP Appr to multi-stage graph

cost(i, j)

= represents the min Cost from a vertex 'j' that is present in Stage 'i' to reach the destination.

$C[i, j] \longrightarrow$ Cost adj matrix $\quad i \longrightarrow j$

$\text{Cost}[i, j] \longrightarrow$ optimal Soln

Stage i

$j \rightsquigarrow$ (Destination)

vertices $1 \longrightarrow n$

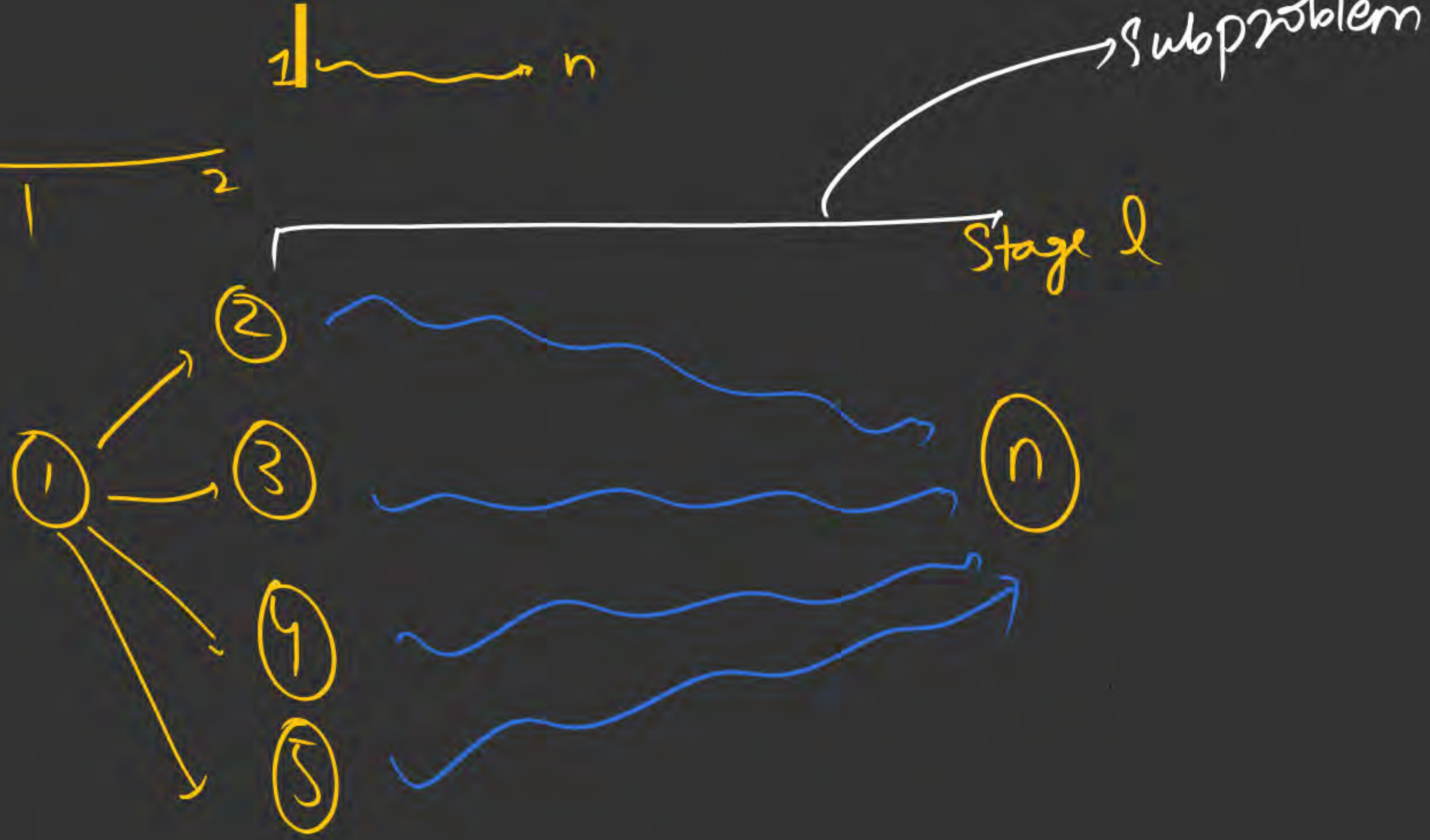
Stage 1 stage 'l'

$s \rightsquigarrow d$

Stage 1 2

cost(1,1)

$1 \rightsquigarrow n$



Stage 1 Stage 2

$$1 \longrightarrow k \rightsquigarrow n$$

$$\text{cost}(1,1) = \min_{\substack{(1,k) \in E \\ \text{and } k \in \text{Stage 2}}} \left\{ c[1,k] + \text{cost}(2,k) \right\}$$

In general,

$i \longrightarrow i+1$

$j \Rightarrow k \longrightarrow n$

$$\text{cost}(i, j) = \min_{\substack{(j,k) \in E \\ \text{and } k \in \text{Stage}(i+1)}} \{ c[j, k] + \text{cost}(i+1, k) \}$$

Base Condition

$$\text{Cost}(l-1, j) = c[j, n]$$

$n \to$ destination

2nd last stage $(l-1)$

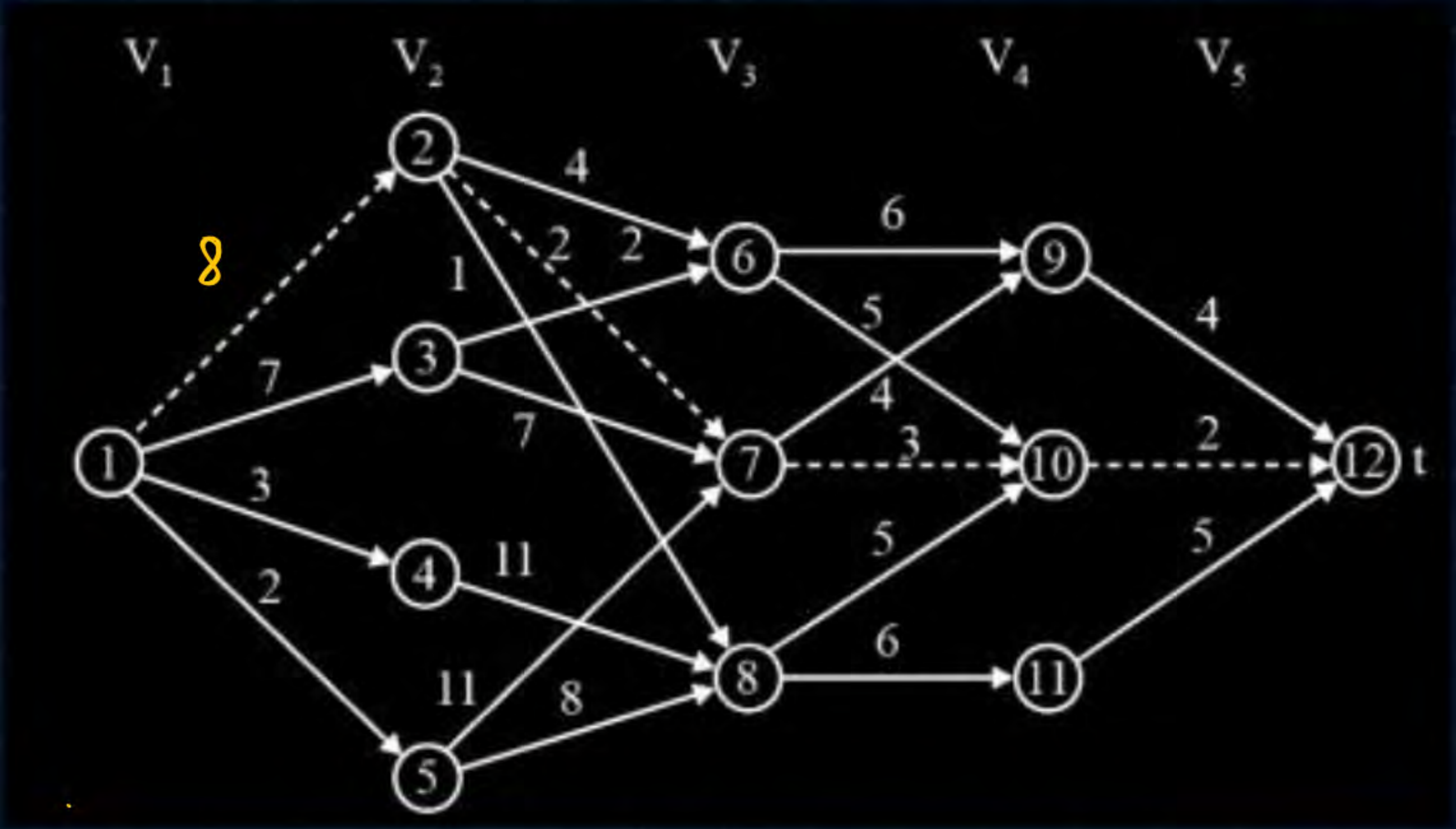
$l-1 \longrightarrow l$

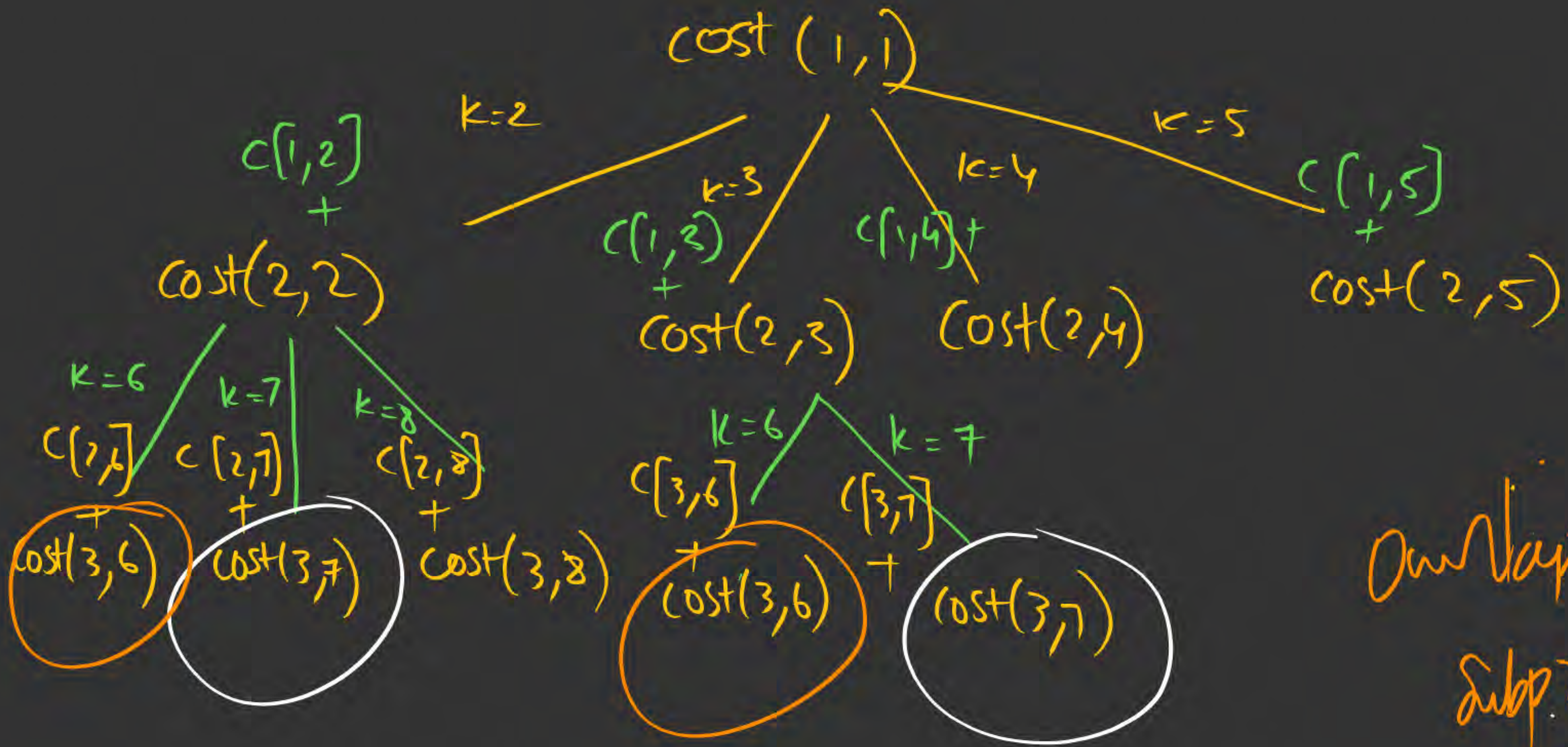
$j \longrightarrow n$

Topic : Dynamic Programming (DP)

cost(1,1)

8





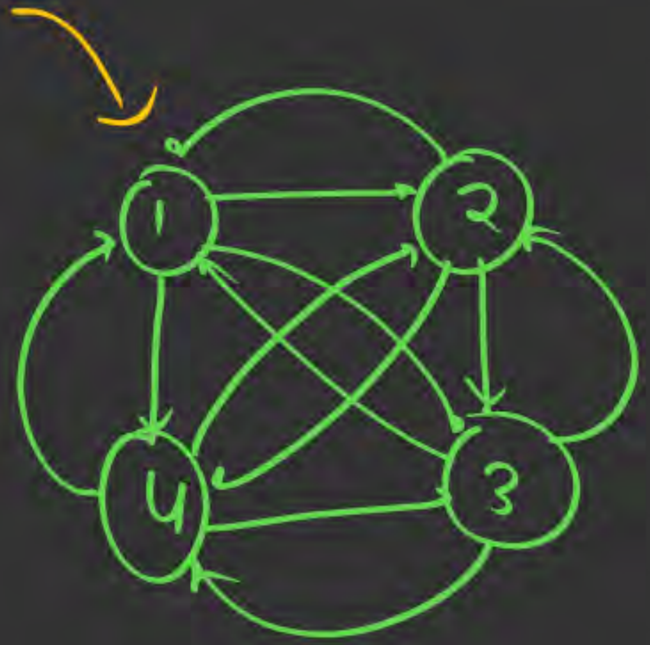
Overlapping subproblems

5) Travelling Salesman Problem (TSP):

shortest tour cost

S → all vertices exactly once → S

eg1:



Cost adj matrix

c	1	2	3	4
1	0	11	16	21
2	6	0	10	11
3	7	14	0	13
4	9	9	10	0

Greedy → Source = 1

1 → 2 → 3 → 4 → 1 → Tour

11 + 10 + 13 + 9 = 43

1 → 2 → 4 → 3 → 1

11 + 11 + 10 + 7

= 39

(actual minimum)

Let $g(i, S)$

$=$ Cost from 'i' to back to source by visiting all the vertices in set 'S' exactly once

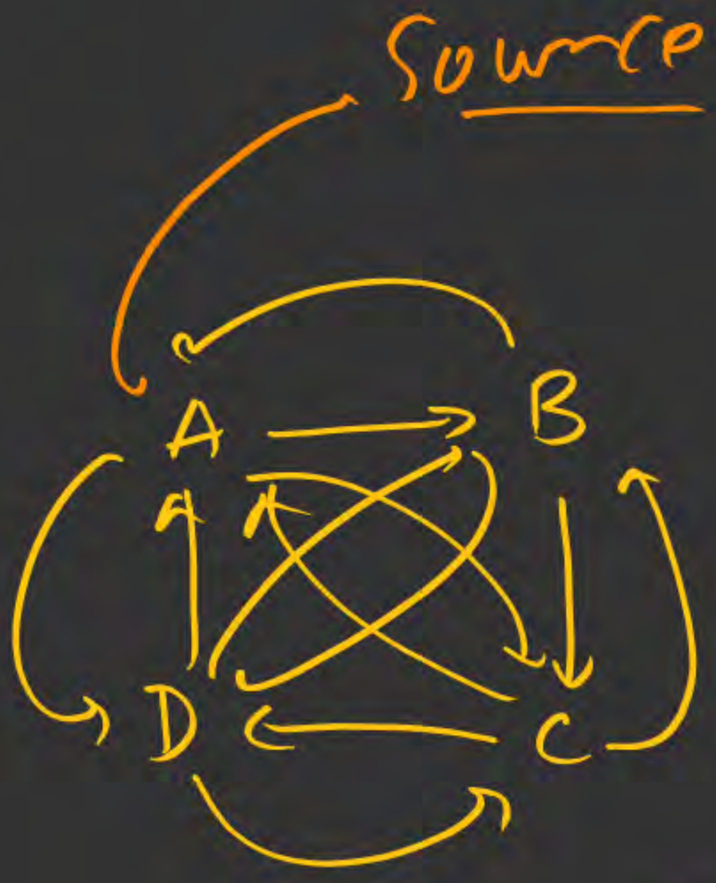
$$g(i,S) = \left\{ c[i,k] + g(k, S - \{k\}) \right\}$$

$k \in S$

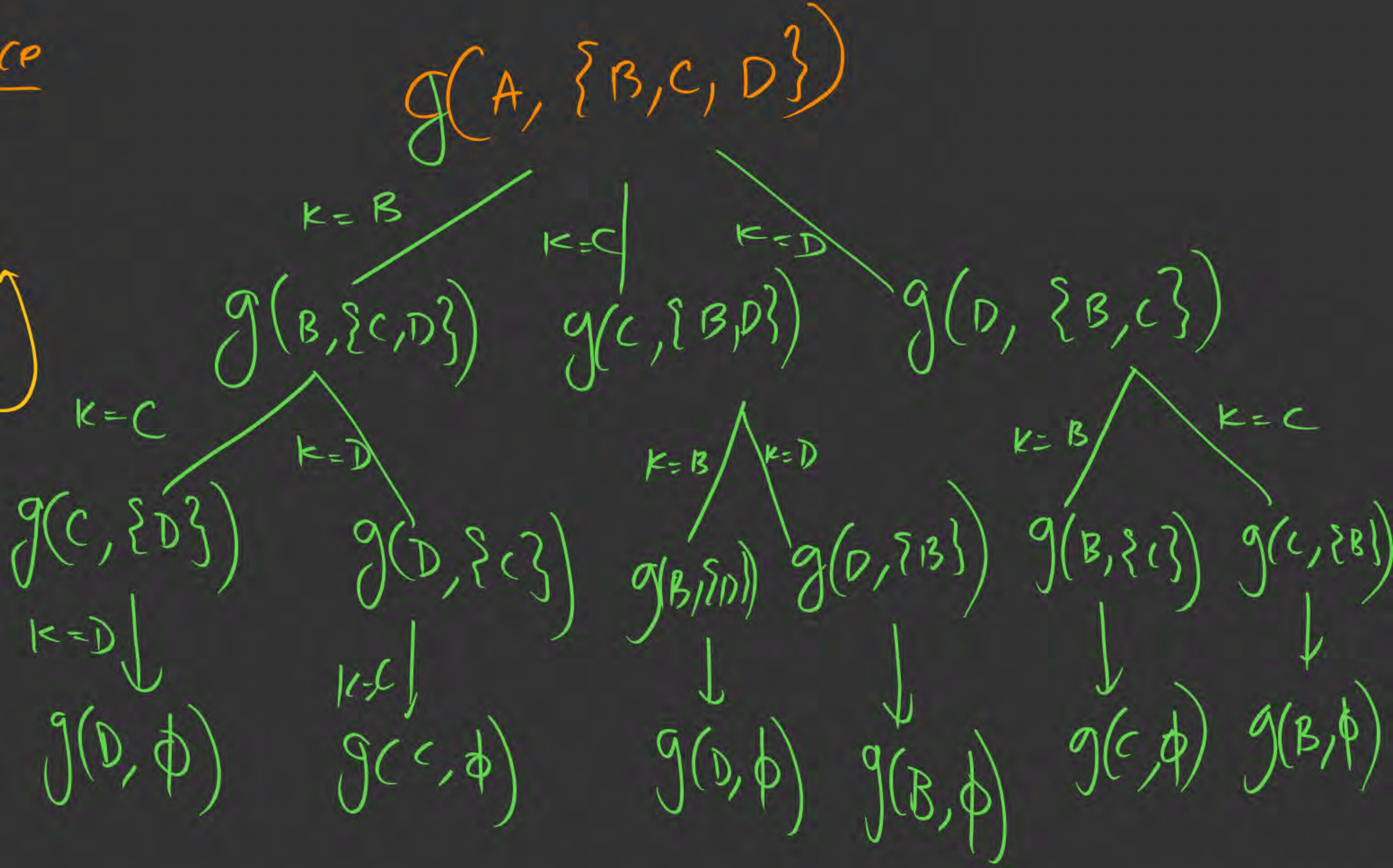
and $(i,k) \in E$

Base Condition

$$g(i, \phi) = c[i, \text{source}]$$



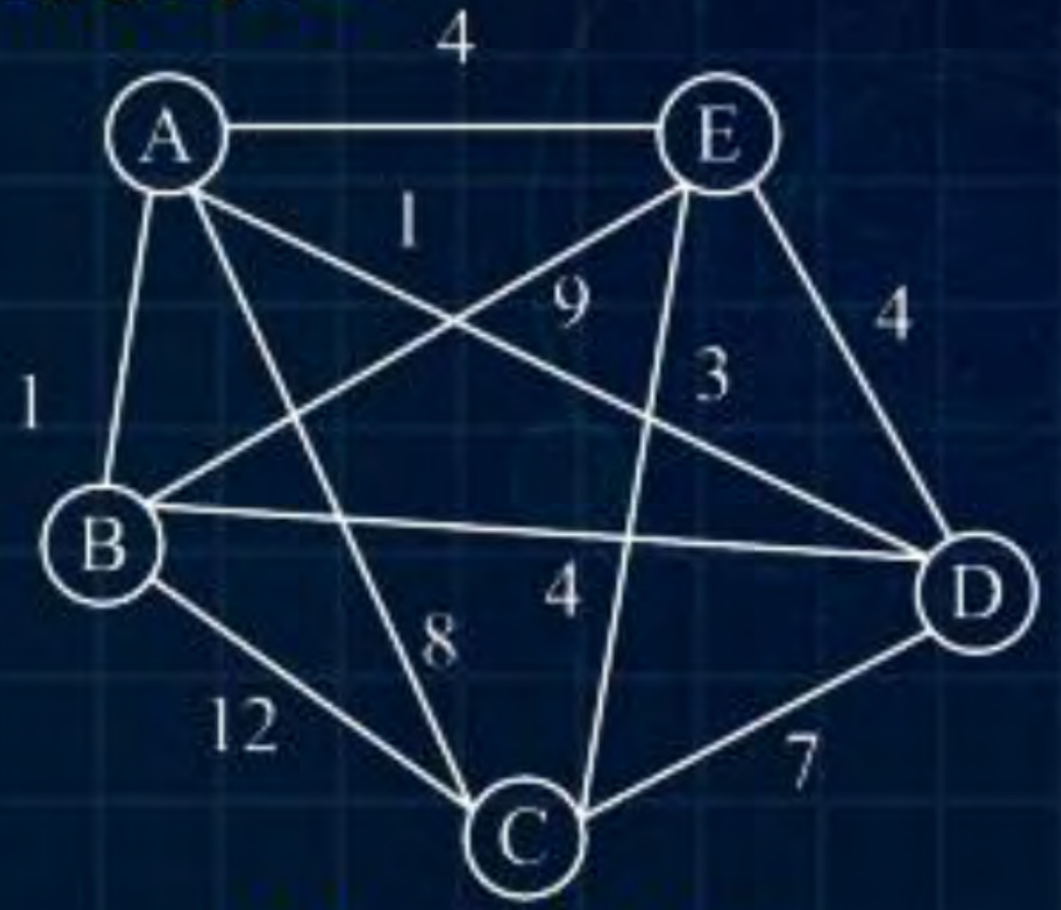
Source
A
B
C
D



g(A, {B, C, D})
K = B
K = C
K = D
g(B, {C, D})
g(C, {B, D})
g(D, {B, C})
K = C
K = D
K = B
K = D
K = B
K = C
g(C, {D})
g(D, {C})
g(B, {D})
g(D, {B})
g(B, {C})
g(C, {B})
K = D
K = C
g(D, ϕ)
g(C, ϕ)
g(D, ϕ)
g(B, ϕ)
g(C, ϕ)
g(B, ϕ)

Question

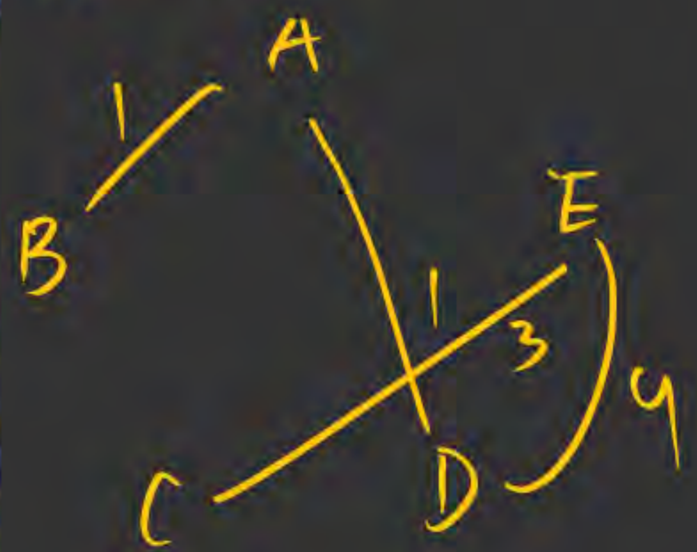
#Q. Consider the following graph G:



What is the minimum possible weight of a spanning tree such that vertex A is a leaf node ?

Soln:

mcST

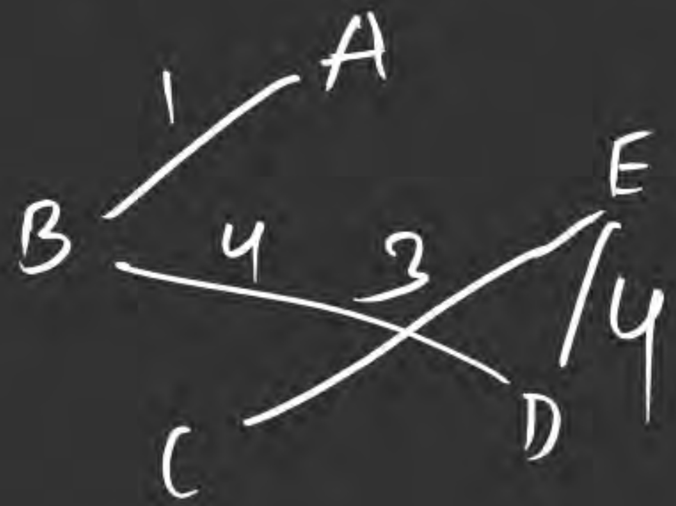


$n = 5$
$e = 4$

cost $= 1 + 1 + 3 + 4$
$= 9$

A → leaf node
(1 edge connected to A)

Real mcST

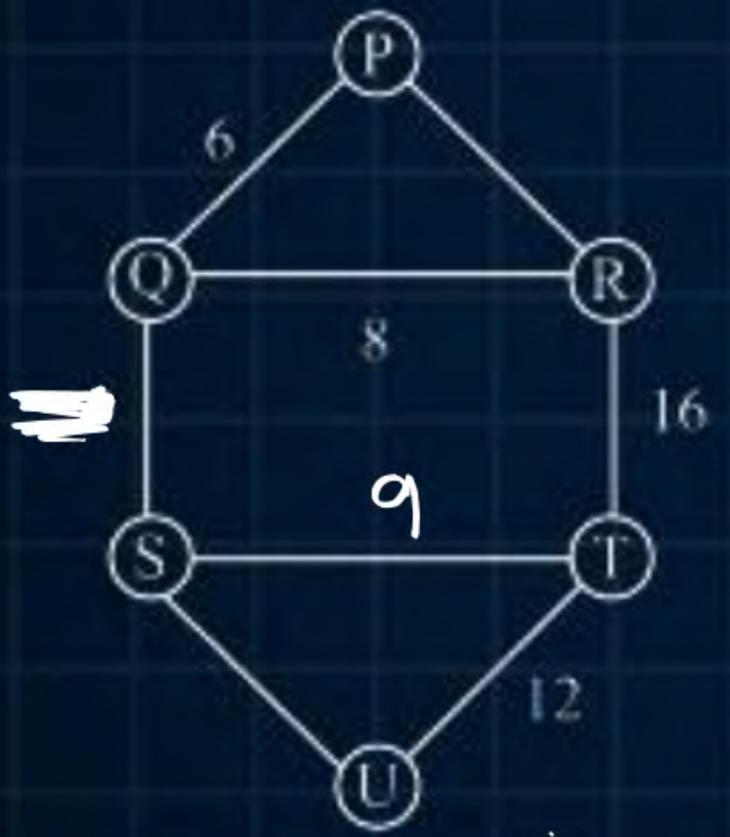


cost $= 1 + 4 + 3 + 4$
$= 12$

Question

#Q. Consider the following graph:

G:



in G
(all edges distinct wt)

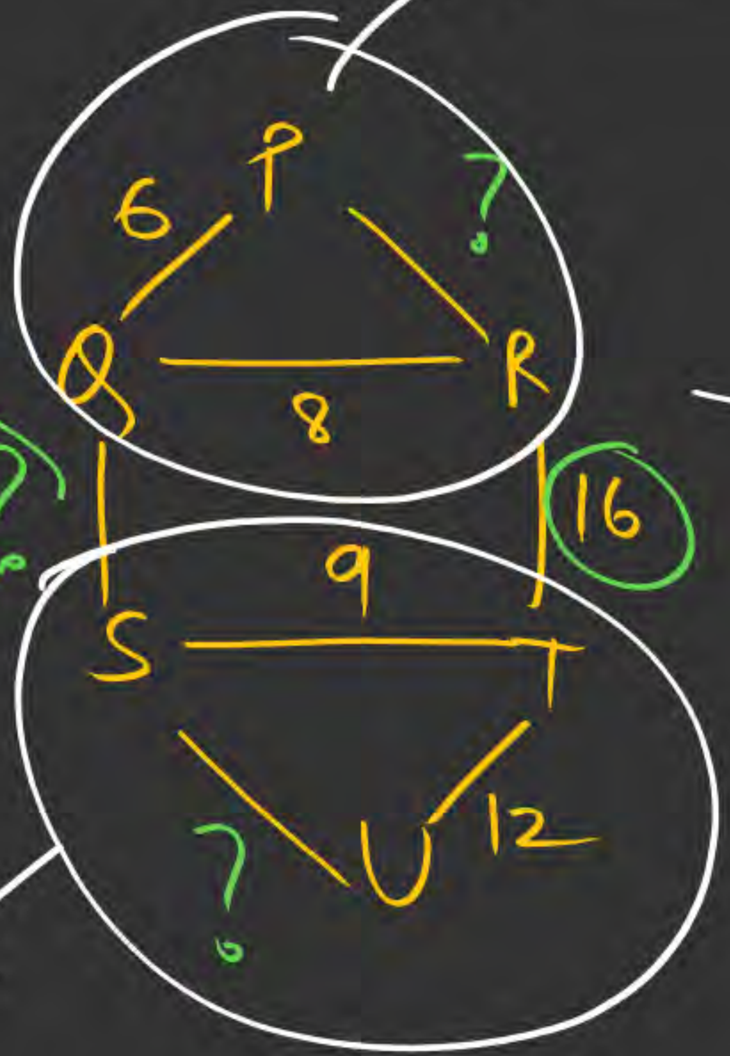
MCST marked with edge weight of 51.

What is the ~~sum of~~ minimum weight of all edges of graph G:

sum of

Sdn :-

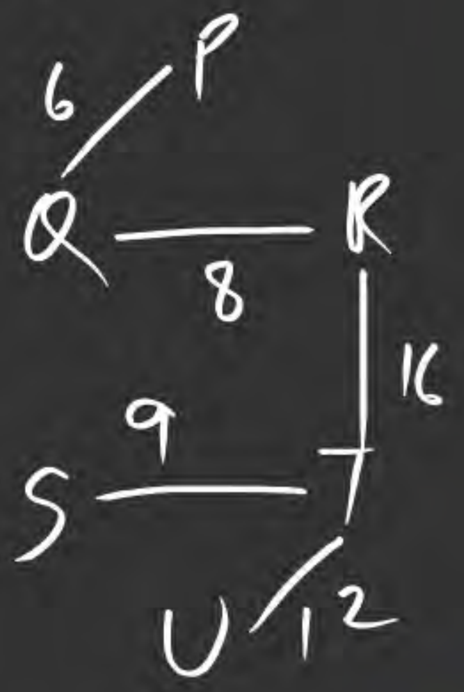
PR > 8



QS > 16

SU > 12

mcst



PR > 8 → 9, 10, 11, 12 . . .

SU > 12 → 13, 14, 15 . . .

QS > 16 → 17, 18, 19 . . .

$\sigma \Rightarrow 51 + (10 + 13 + 17)$

$= 51 + 40 = 91$

Question

#Q. Consider a machine which needs a minimum of 100 seconds to sort 4096 names by quick sort best case , then what is the minimum time required to sort 512 names (approximately)is____(round off to 2 decimal)

Quick Sort:

$n = 4096, \quad t = 100$ sec

$n = 512 \quad ?$

Best Case Time $= O(n \log_2 n)$

$t = \underline{c * n \log_2 n}$ sec

$n = 4096$, 100 sec

$c \times n \times \log_2 n = 100$

$c \times 4096 \times \log_2(4096) = 100$

$c \times 2^{12} \times \log_2(2^{12}) = 100$

$12 \times c \times 2^{12} = 100$

$c = \left[\dfrac{100}{2^{12} \times 12}\right]$ ✓

given

for $n = 512$, $t = c \times 512 \times \log_2(512)$

$$t = c \times 2^9 \times \log_2(2^9)$$

$$t = c \times 2^9 \times 9$$

$$= \frac{100}{\cancel{2^{12}} \times \cancel{12}} \times \cancel{2^9} \times \cancel{9} = \frac{\overset{25}{\cancel{100}} \times 3}{2^3 \times \cancel{4}} = \frac{25 \times 3}{8} = \frac{75}{8}$$

$$= 9.4$$

Question

#Q. Which of the following algorithm can be used to sort n integers in the range $[1, \ldots 10^3]$ in O(n) time?

BC : $O(n^2)$
WC : $O(n^2)$

A. Selection sort

BC : $O(n)$
WC : $O(n^2)$

B. Bubble sort

C. Radix sort

$\hookrightarrow$ TC : $O(n * d)$

D. Quick sort

BC : $O(n \log n)$
WC : $O(n^2)$

$[1 \longrightarrow 1000]$ n elems

TC of Radix Sort $\longrightarrow O(n*d)$

1000

$d = 4$

$n \rightarrow$ no. of elems

$d \rightarrow$ no. of digits in max elm.

$$TC = O(n*4)$$

$$= O(n)$$

Thank

THANK

Keep Hustling!