Branch and Bound (Introduction)

Branch and Bound

Job sequencing

Jobs = $\{J_1, J_2, J_3, J_4\}$

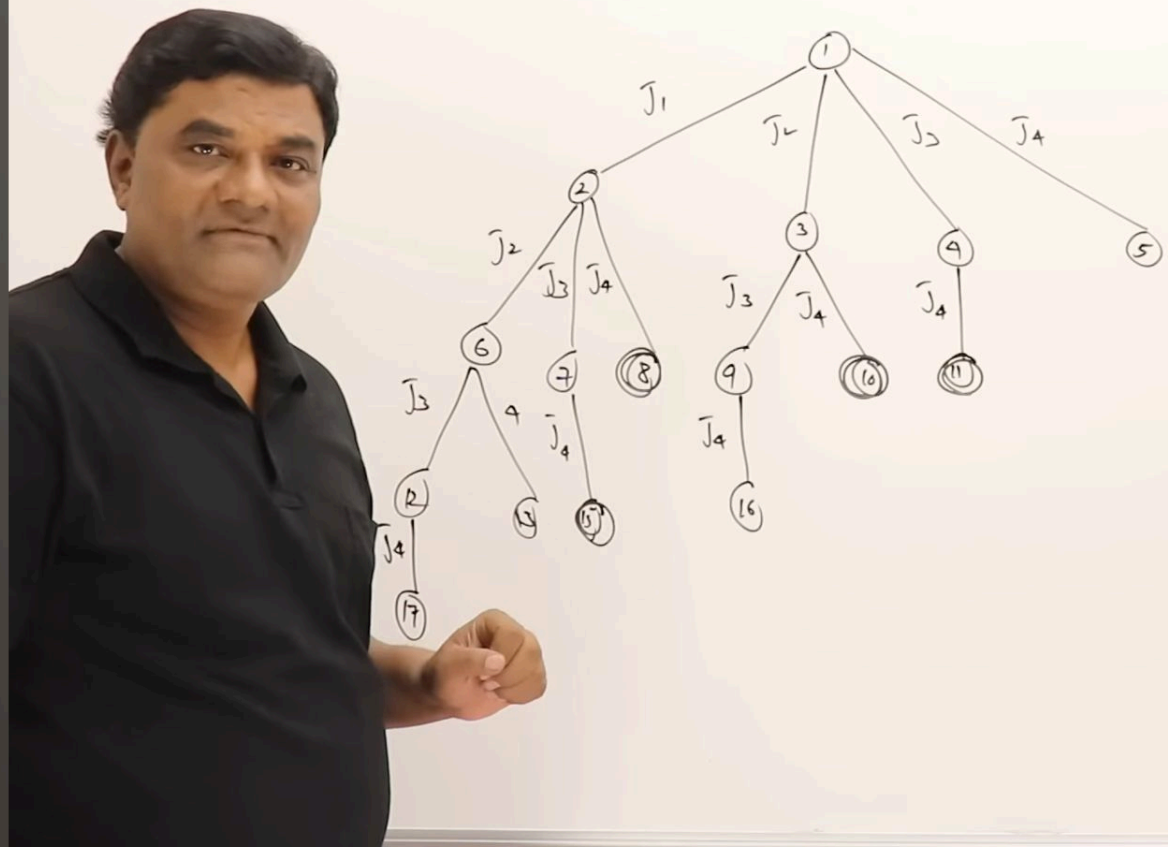
$p = \{10, 5, 8, 3\}$

$d = \{1, 2, 1, 2\}$

$S1 = \{J_1, J_4\}$

$S2 = \{1, 0, 0, 1\}$

Branch and Bound



Job sequencing

$Jobs = \{J_1, J_2, J_3, J_4\}$

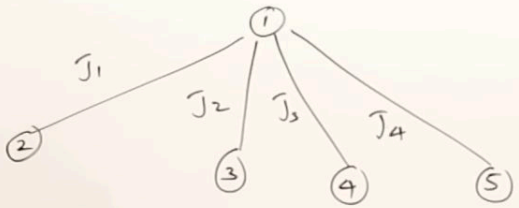
$P = \{10, 5, 8, 3\}$

$d = \{1, 2, 1, 2\}$

$S1 = \{J_1, J_4\}$

$S2 = \{1, 0, 0, 1\}$

Branch and Bound



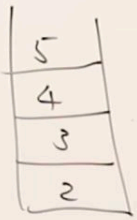
Job sequencing

Jobs = {J1, J2, J3, J4}

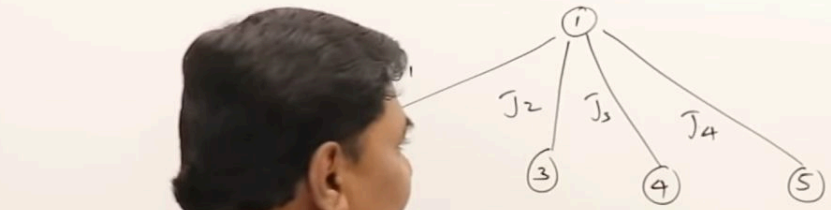
P = {10, 5, 8, 3}

d = {1 2, 1, 2}

S1 = {J1, J4}



Branch and Bound



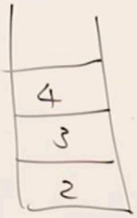
Job sequencing

Jobs = {J1, J2, J3, J4}

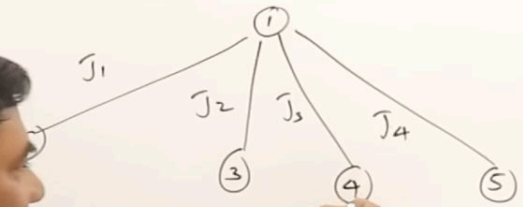
P = {10, 5, 8, 3}

d = {1 2, 1, 2}

S1 = {J1, J4}



Branch and Bound



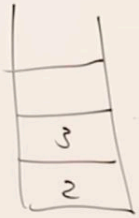
Job sequencing

Jobs = {J1, J2, J3, J4}

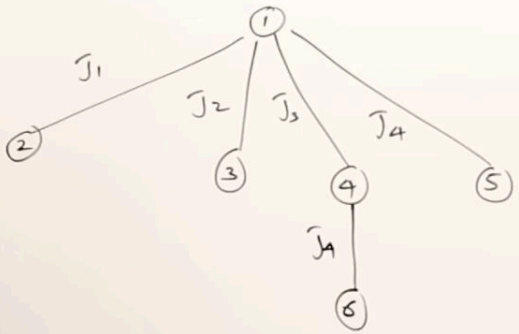
P = {10, 5, 8, 3}

d = {1 2, 1, 2}

S1 = {J1, J4}



Branch and Bound



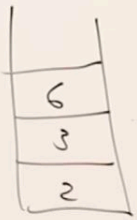
Job sequencing

$Jobs = \{J_1, J_2, J_3, J_4\}$

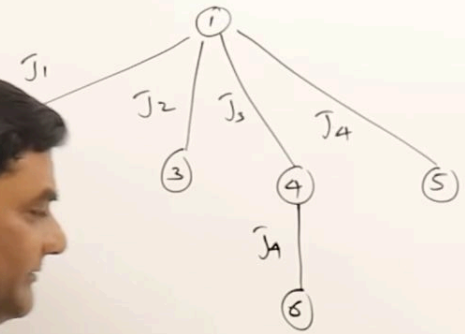
$P = \{10, 5, 8, 3\}$

$d = \{1, 2, 1, 2\}$

$S_1 = \{J_1, J_4\}$



Branch and Bound



Job sequencing

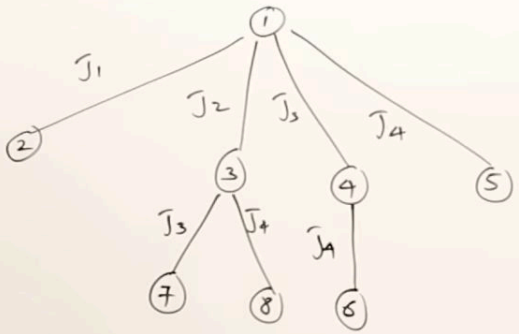
$Jobs = \{J_1, J_2, J_3, J_4\}$

$P = \{10, 5, 8, 3\}$

$d = \{1, 2, 1, 2\}$

$S_1 = \{J_1, J_4\}$

Branch and Bound



Job sequencing

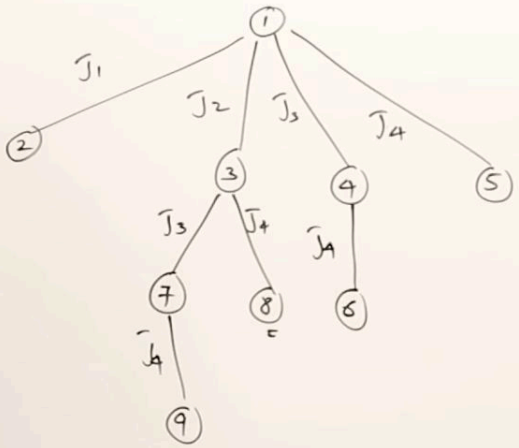
$Jobs = \{J_1, J_2, J_3, J_4\}$

$P = \{10, 5, 8, 3\}$

$d = \{1, 2, 1, 2\}$

$S_1 = \{J_1, J_4\}$

Branch and Bound



Job sequencing

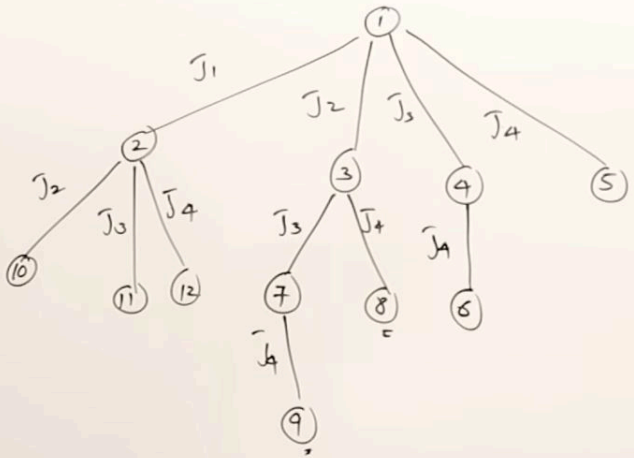
Jobs = {J1, J2, J3, J4}

P = {10, 5, 8, 3}

d = {1 2, 1, 2}

S1 = {J1, J4}

Branch and Bound



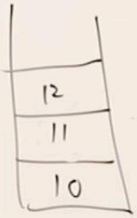
Job sequencing

Jobs = $\{J_1, J_2, J_3, J_4\}$

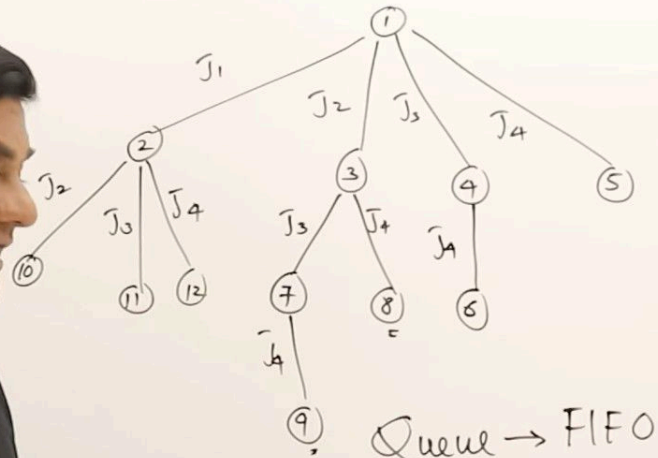
$P = \{10, 5, 8, 3\}$

$d = \{1\ 2, 1, 2\}$

$S_1 = \{J_1, J_4\}$



Branch and Bound



Queue → FIFO

Stack → LIFO

Least cost → LC-BB

Job sequencing

Jobs = {J1, J2, J3, J4}

P = {10, 5, 8, 3}

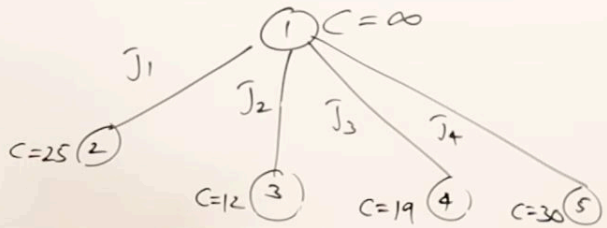
d = {1 2, 1, 2}

S1 = {J1, J4}

Remember B&B is always BFS, whereas recursion is DFS

Branch and Bound

Least-Cost - BB



Job sequencing

Jobs = {J1, J2, J3, J4}

P = {10, 5, 8, 3}

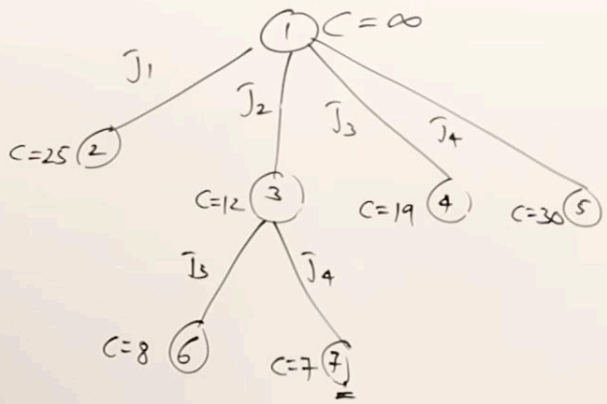
d = {1 2, 1, 2}

S1 = {J1, J4}

1. FIFO - BB
2. LIFO - BB
3. LC - BB

Branch and Bound

Least Cost - BB



Job sequencing

Jobs = {J1, J2, J3, J4}

p = {10, 5, 8, 3}

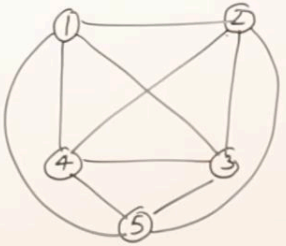
d = {1 2, 1, 2}

S1 = {J1, J4}

1. FIFO - BB
2. LIFO - BB
3. LC - BB

Traveling salesman problem (Using branch and bound)

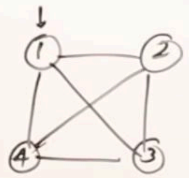
Traveling Salesperson Branch-n-Bound

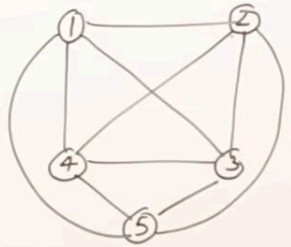


	1	2	3	4	5
1	∞	20	30	10	11
2	15	∞	16	4	2
3	3	5	∞	2	4
4	19	6	18	∞	3
5	16	4	7	16	∞

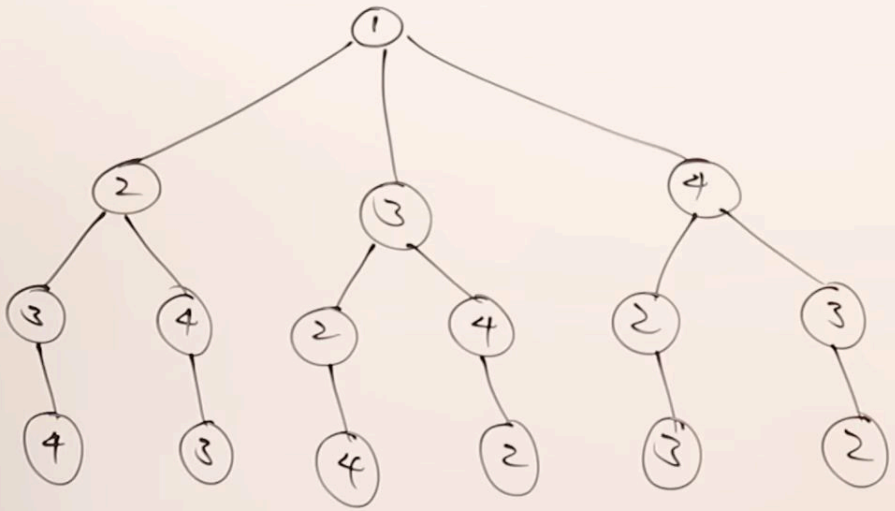
View key concept

Traveling Salesperson Branch-n-Bound

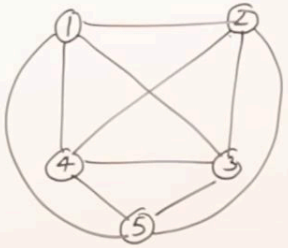




	1	2	3	4	5
1	∞	20	30	10	11
2	15	∞	16	4	2
3	3	5	∞	2	4
4	19	6	18	∞	3
5	16	4	7	16	∞

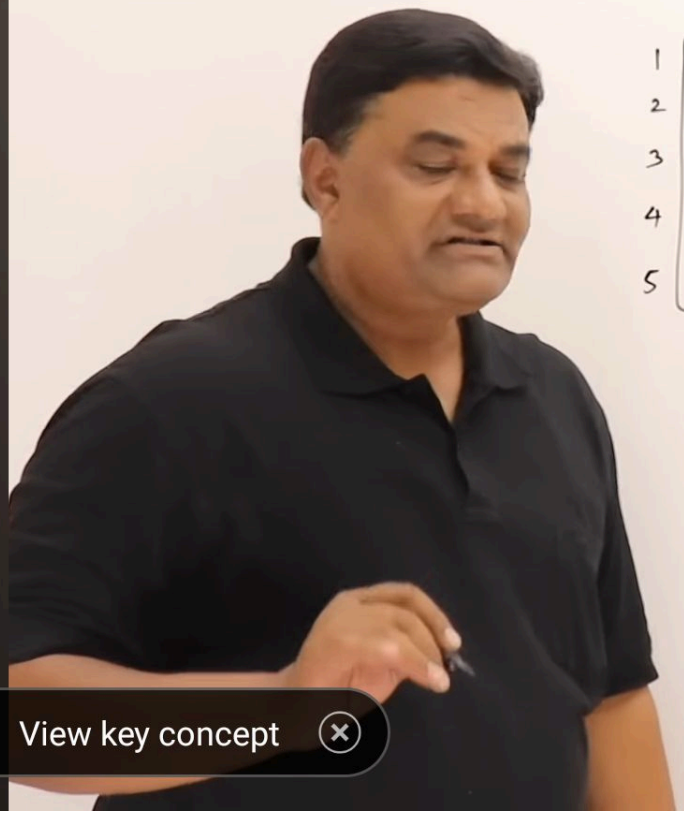


Traveling Salesperson Branch-n-Bound



	1	2	3	4	5
1	∞	20	30	10	11
2	15	∞	16	4	2
3	3	5	∞	2	4
4	19	6	18	∞	3
5	16	4	7	16	∞

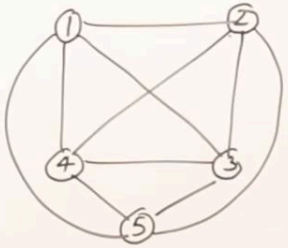
	1	2	3	4	5
1	∞	20	30	10	11
2	15	∞	16	4	2
3	3	5	∞	2	4
4	19	6	18	∞	3
5	16	4	7	16	∞



View key concept

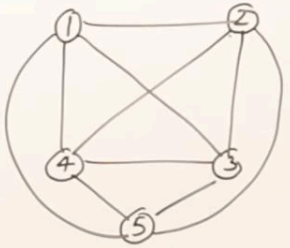
Traveling Salesperson Branch-n-Bound

	1	2	3	4	5	
1	∞	20	30	10	11	10
2	15	∞	16	4	2	2
3	3	5	∞	2	4	2
4	19	6	18	∞	3	3
5	16	4	7	16	∞	4



	1	2	3	4	5
1	∞	20	30	10	11
2	15	∞	16	4	2
3	3	5	∞	2	4
4	19	6	18	∞	3
5	16	4	7	16	∞

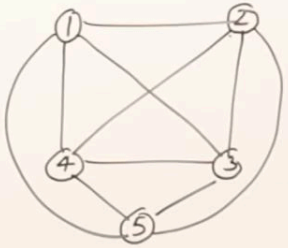
Traveling Salesperson Branch-n-Bound



	1	2	3	4	5	
1	∞	10	20	0	1	10
2	13	∞	14	2	0	2
3	1	3	∞	0	2	2
4	16	3	15	∞	0	3
5	12	0	3	12	∞	4
						21

	1	2	3	4	5
1	∞	20	30	10	11
2	15	∞	16	4	2
3	3	5	∞	2	4
4	19	6	18	∞	3
5	16	4	7	16	∞

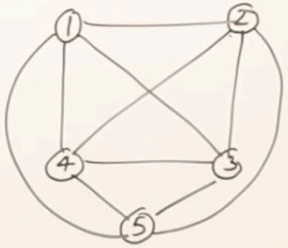
Traveling Salesperson Branch-n-Bound



	1	2	3	4	5	
1	∞	10	20	0	1	10
2	13	∞	14	2	0	2
3	1	3	∞	0	2	2
4	16	3	15	∞	0	3
5	12	0	3	12	∞	4
	1	0	3	0	0	21

	1	2	3	4	5
1	∞	20	30	10	11
2	15	∞	16	4	2
3	3	5	∞	2	4
4	19	6	18	∞	3
5	16	4	7	16	∞

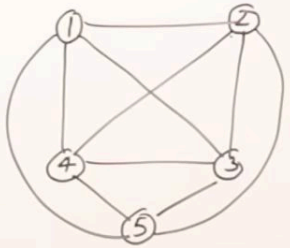
Traveling Salesperson Branch-n-Bound



	1	2	3	4	5	
1	∞	10	17	0	1	10
2	12	∞	11	2	0	2
3	0	3	∞	0	2	2
4	15	3	12	∞	0	3
5	11	0	0	12	∞	4
	1	0	3	0	0	+ 21 + 4 = 25

	1	2	3	4	5
1	∞	20	30	10	11
2	15	∞	16	4	2
3	3	5	∞	2	4
4	19	6	18	∞	3
5	16	4	7	16	∞

Traveling Salesperson Branch-n-Bound

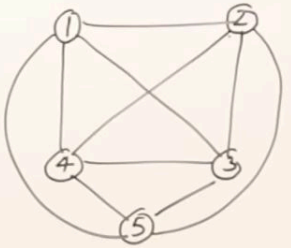


	1	2	3	4	5
1	∞	10	17	0	1
2	12	∞	11	2	0
3	0	3	∞	0	2
4	15	3	12	∞	0
5	11	0	0	12	∞

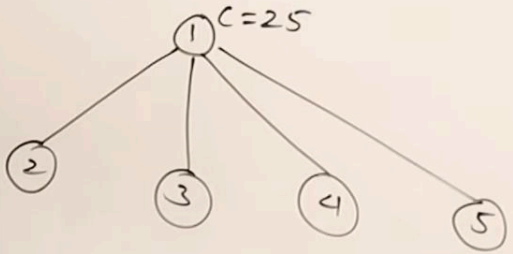
reduced cost = 25

Traveling Salesperson Branch-n-Bound

	1	2	3	4	5
1	∞	10	17	0	1
2	12	∞	11	2	0
3	0	3	∞	0	2
4	15	3	12	∞	0
5	11	0	0	12	∞

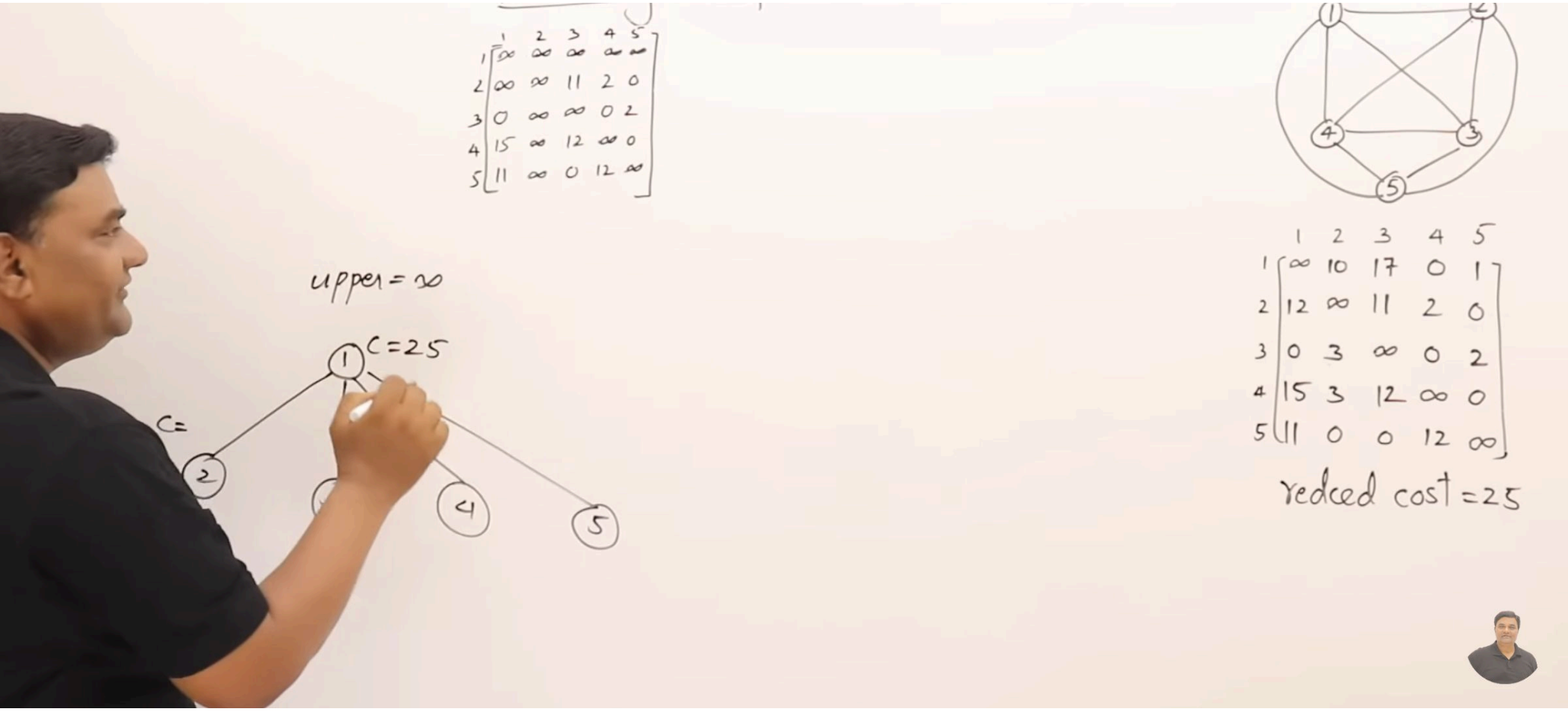


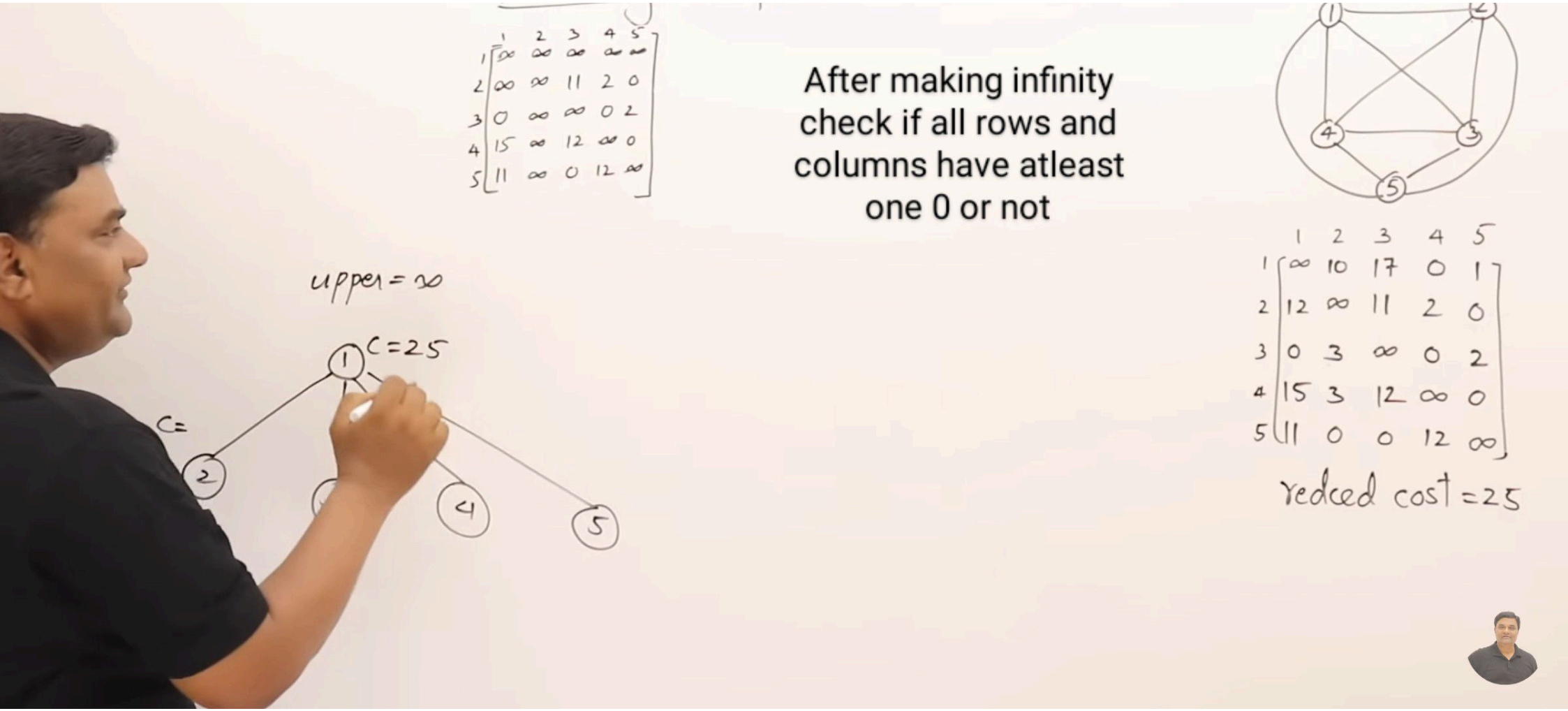
upper = ∞



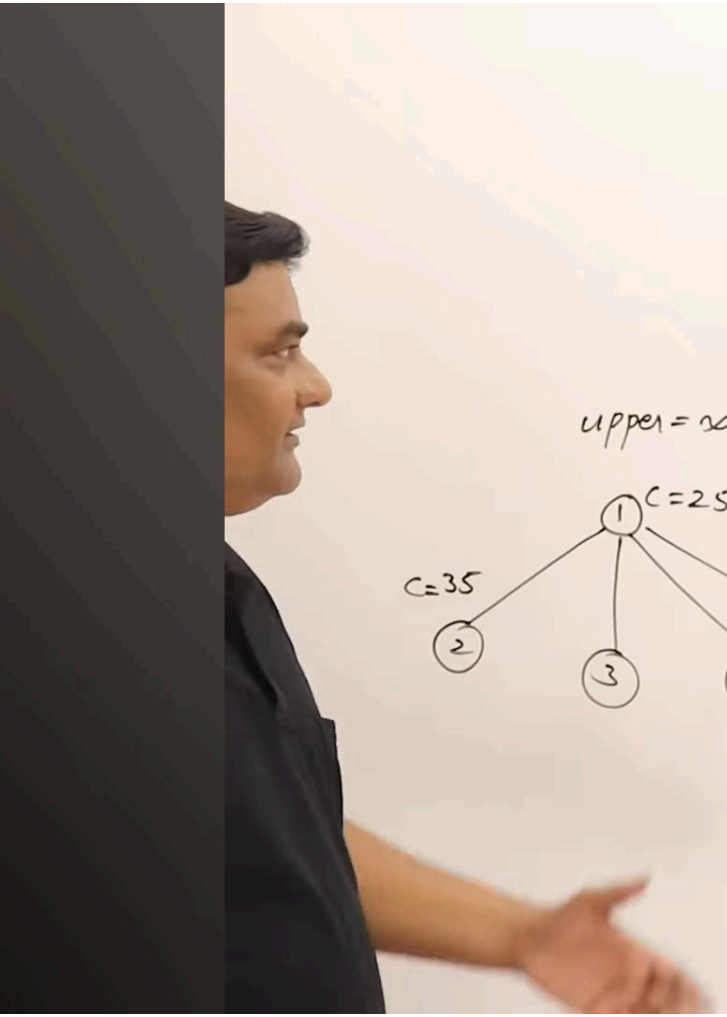
	1	2	3	4	5
1	∞	10	17	0	1
2	12	∞	11	2	0
3	0	3	∞	0	2
4	15	3	12	∞	0
5	11	0	0	12	∞

reduced cost = 25


1 2 3 4 5
1 ∞ ∞ ∞ ∞ ∞
2 ∞ ∞ 11 2 0
3 0 ∞ ∞ 0 2
4 15 ∞ 12 ∞ 0
5 11 ∞ 0 12 ∞
upper = ∞
C=25
C=
1
2
4
5
1 2 3 4 5
1 ∞ 10 17 0 1
2 12 ∞ 11 2 0
3 0 3 ∞ 0 2
4 15 3 12 ∞ 0
5 11 0 0 12 ∞
reded cost =25


After making infinity check if all rows and columns have atleast one 0 or not
upper = ∞
C=25
C=
reded cost =25

Traveling Salesperson Branch-n-Bound

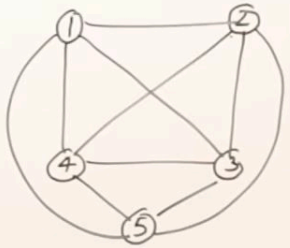


	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	11	2	0
3	0	∞	∞	0	2
4	15	∞	12	∞	0
5	11	∞	0	12	∞

$c(1,2) + r + \hat{r}$

$10 + 25 + 0$

upper = ∞



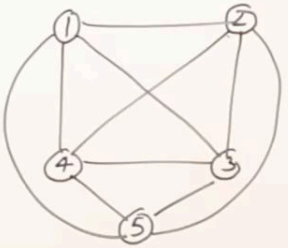
	1	2	3	4	5
1	∞	10	17	0	1
2	12	∞	11	2	0
3	0	3	∞	0	2
4	15	3	12	∞	0
5	11	0	0	12	∞

reduced cost = 25

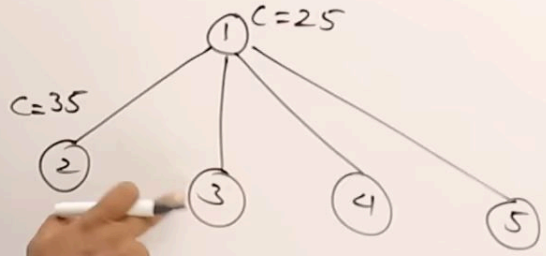
Traveling Salesperson Branch-n-Bound

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	11	2	0
3	0	∞	∞	0	2
4	15	∞	12	∞	0
5	11	∞	0	12	∞

	1	2	3	4	5
1	∞	10	17	0	1
2	12	∞	11	2	0
3	0	3	∞	0	2
4	15	3	12	∞	0
5	11	0	0	12	∞



upper = ∞



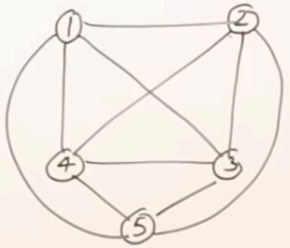
	1	2	3	4	5
1	∞	10	17	0	1
2	12	∞	11	2	0
3	0	3	∞	0	2
4	15	3	12	∞	0
5	11	0	0	12	∞

reduced cost = 25

Traveling Salesperson Branch-n-Bound

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	11	2	0
3	0	∞	∞	0	2
4	15	∞	12	∞	0
5	11	∞	0	12	∞

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	12	∞	∞	2	0
3	∞	3	∞	0	2
4	15	3	∞	∞	0
5	11	0	∞	12	0



	1	2	3	4	5
1	∞	10	17	0	1
2	12	∞	11	2	0
3	0	3	∞	0	2
4	15	3	12	∞	0
5	11	0	0	12	∞

reduced cost = 25

∞

25

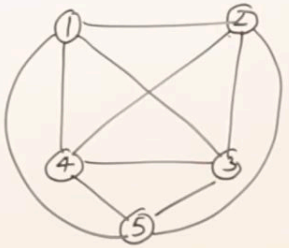
4

5

Traveling Salesperson Branch-n-Bound

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	11	2	0
3	0	∞	∞	0	2
4	15	∞	12	∞	0
5	11	∞	0	12	∞

	1	2	3	4	5	
1	∞	∞	∞	∞	∞	
2	12	∞	∞	2	0	0
3	∞	3	∞	0	2	0
4	15	3	∞	∞	0	0
5	11	0	∞	12	0	0
	11	0	0	0	0	



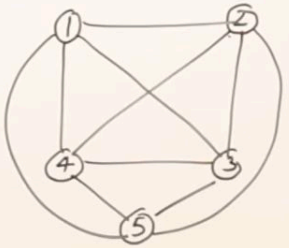
	1	2	3	4	5
1	∞	10	17	0	1
2	12	∞	11	2	0
3	0	3	∞	0	2
4	15	3	12	∞	0
5	11	0	0	12	∞

reduced cost = 25

Traveling Salesperson Branch-n-Bound

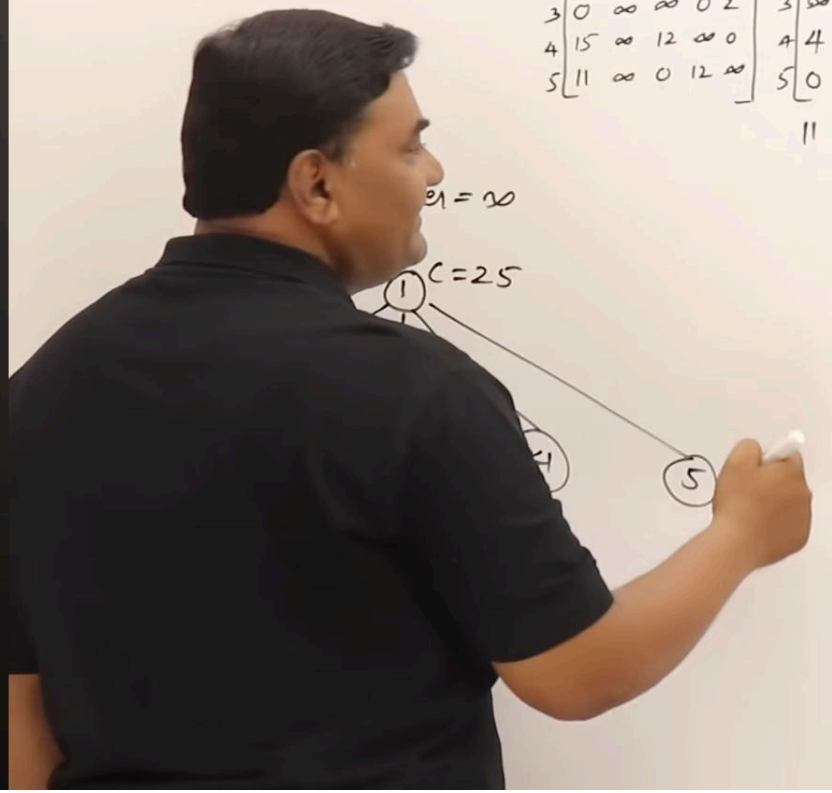
	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	11	2	0
3	0	∞	∞	0	2
4	15	∞	12	∞	0
5	11	∞	0	12	∞

	1	2	3	4	5	
1	∞	∞	∞	∞	∞	
2	1	∞	∞	2	0	0
3	∞	3	∞	0	2	0
4	4	3	∞	∞	0	0
5	0	0	∞	12	0	0
	11	0	0	0	0	



	1	2	3	4	5
1	∞	10	17	0	1
2	12	∞	11	2	0
3	0	3	∞	0	2
4	15	3	12	∞	0
5	11	0	0	12	∞

reduced cost = 25



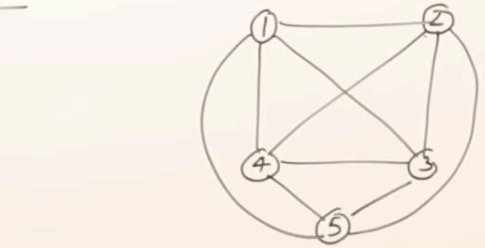
Traveling Salesperson Branch-n-Bound

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	11	2	0
3	0	∞	∞	0	2
4	15	∞	12	∞	0
5	11	∞	0	12	∞

	1	2	3	4	5	
1	∞	∞	∞	∞	∞	
2	1	∞	∞	2	0	0
3	∞	3	∞	0	2	0
4	4	3	∞	∞	0	0
5	0	0	∞	12	0	0
	11	0	0	0	0	

$c(1,3) + r + \hat{r}$

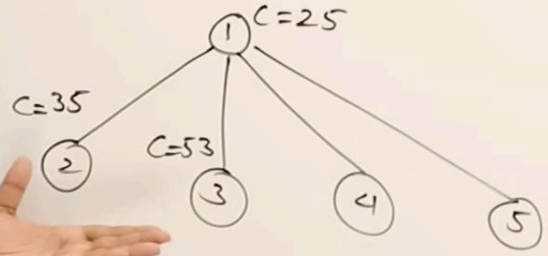
$17 + 25 + 11$



	1	2	3	4	5
1	∞	10	17	0	1
2	12	∞	11	2	0
3	0	3	∞	0	2
4	15	3	12	∞	0
5	11	0	0	12	∞

reduced cost = 25

upper = ∞



Traveling Salesperson Branch-n-Bound

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	11	2	0
3	0	∞	∞	0	2
4	15	∞	12	∞	0
5	11	∞	0	12	∞

(2)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	1	∞	∞	2	0
3	∞	3	∞	0	2
4	4	3	∞	∞	0
5	0	0	∞	12	0

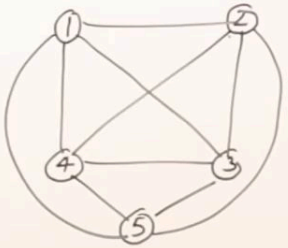
(3)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	12	∞	11	∞	0
3	0	3	∞	∞	2
4	∞	3	12	∞	0
5	11	0	0	∞	∞

(4)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	10	∞	9	0	∞
3	0	3	∞	0	∞
4	12	0	9	∞	∞
5	∞	0	0	12	∞

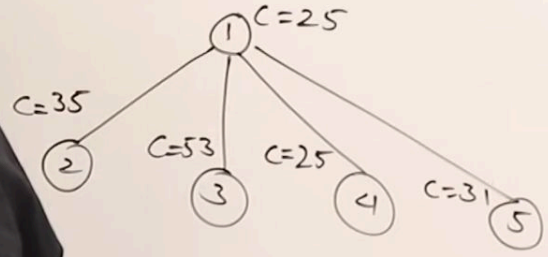
(5)

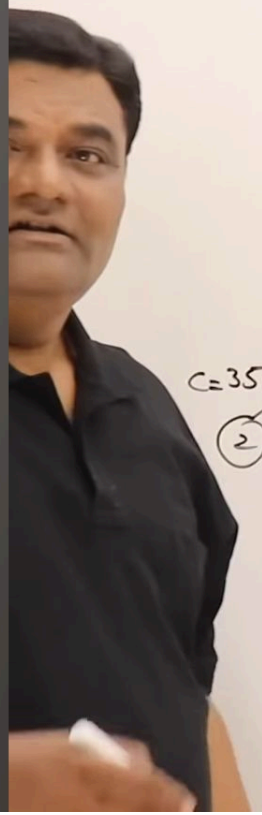


	1	2	3	4	5
1	∞	10	17	0	1
2	12	∞	11	2	0
3	0	3	∞	0	2
4	15	3	12	∞	0
5	11	0	0	12	∞

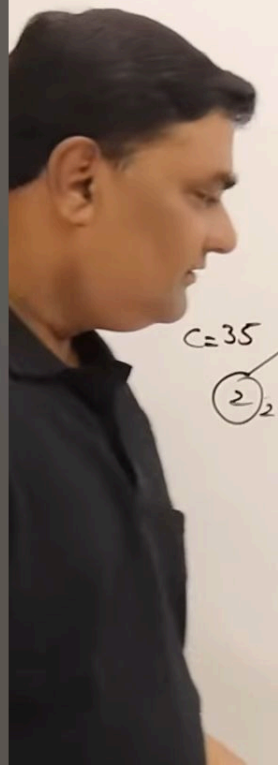
reduced cost = 25

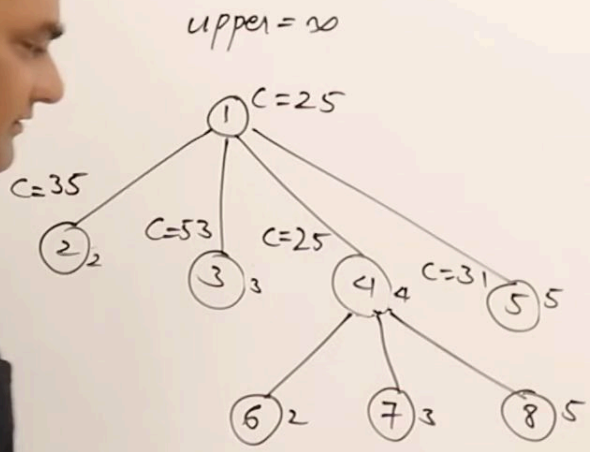
upper = ∞





Traveling Salesperson Branch-n-Bound





(2)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	11	2	0
3	0	∞	∞	0	2
4	15	∞	12	∞	0
5	11	∞	0	12	∞

(3)

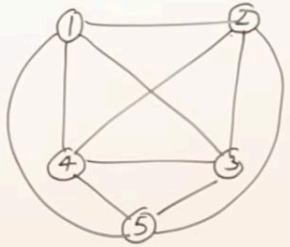
	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	1	∞	∞	2	0
3	∞	3	∞	0	2
4	4	3	∞	∞	0
5	0	0	∞	12	0

(4)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	12	∞	11	∞	0
3	0	3	∞	∞	2
4	∞	3	12	∞	0
5	11	0	0	∞	∞

(5)

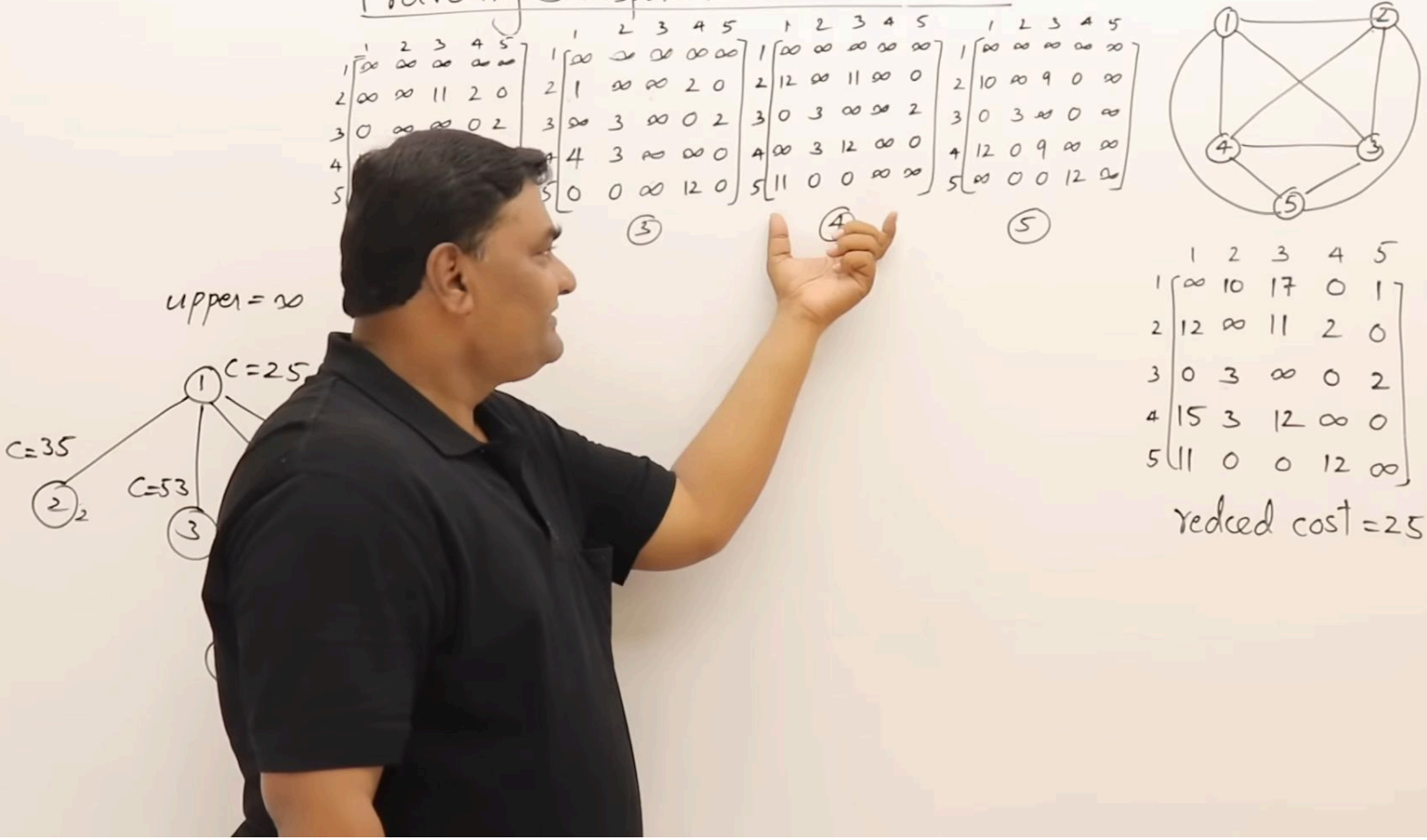
	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	10	∞	9	0	∞
3	0	3	∞	0	∞
4	12	0	9	∞	∞
5	∞	0	0	12	∞



	1	2	3	4	5
1	∞	10	17	0	1
2	12	∞	11	2	0
3	0	3	∞	0	2
4	15	3	12	∞	0
5	11	0	0	12	∞

reduced cost = 25

Traveling Salesperson Branch-n-Bound



Traveling Salesperson Branch-n-Bound

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	11	2	0
3	0	∞	∞	0	2
4	15	∞	12	∞	0
5	11	∞	0	12	∞

(2)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	1	∞	∞	2	0
3	∞	3	∞	0	2
4	4	3	∞	∞	0
5	0	0	∞	12	0

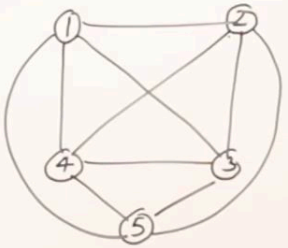
(3)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	12	∞	11	∞	0
3	0	3	∞	∞	2
4	∞	3	12	∞	0
5	11	0	0	∞	∞

(4)

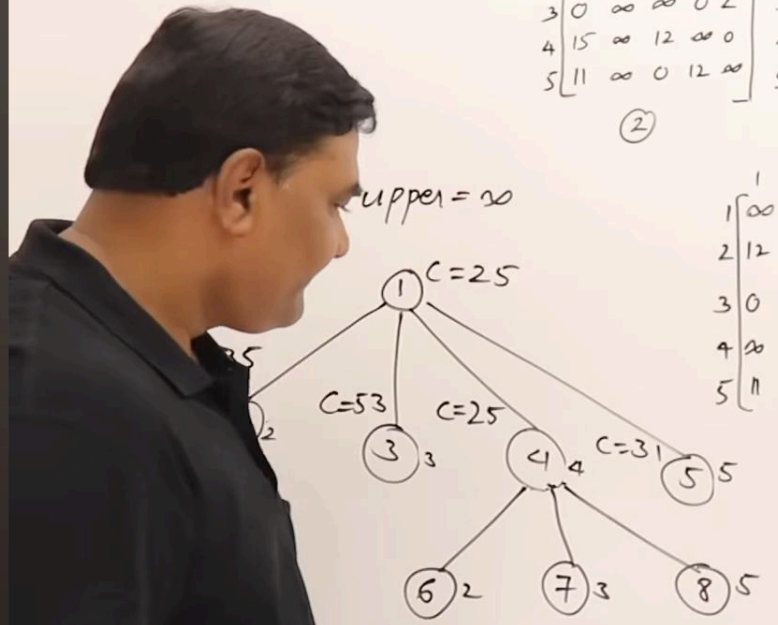
	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	10	∞	9	0	∞
3	0	3	∞	0	∞
4	12	0	9	∞	∞
5	∞	0	0	12	∞

(5)



	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	12	∞	11	∞	0
3	0	3	∞	∞	2
4	∞	3	12	∞	0
5	11	0	0	∞	∞

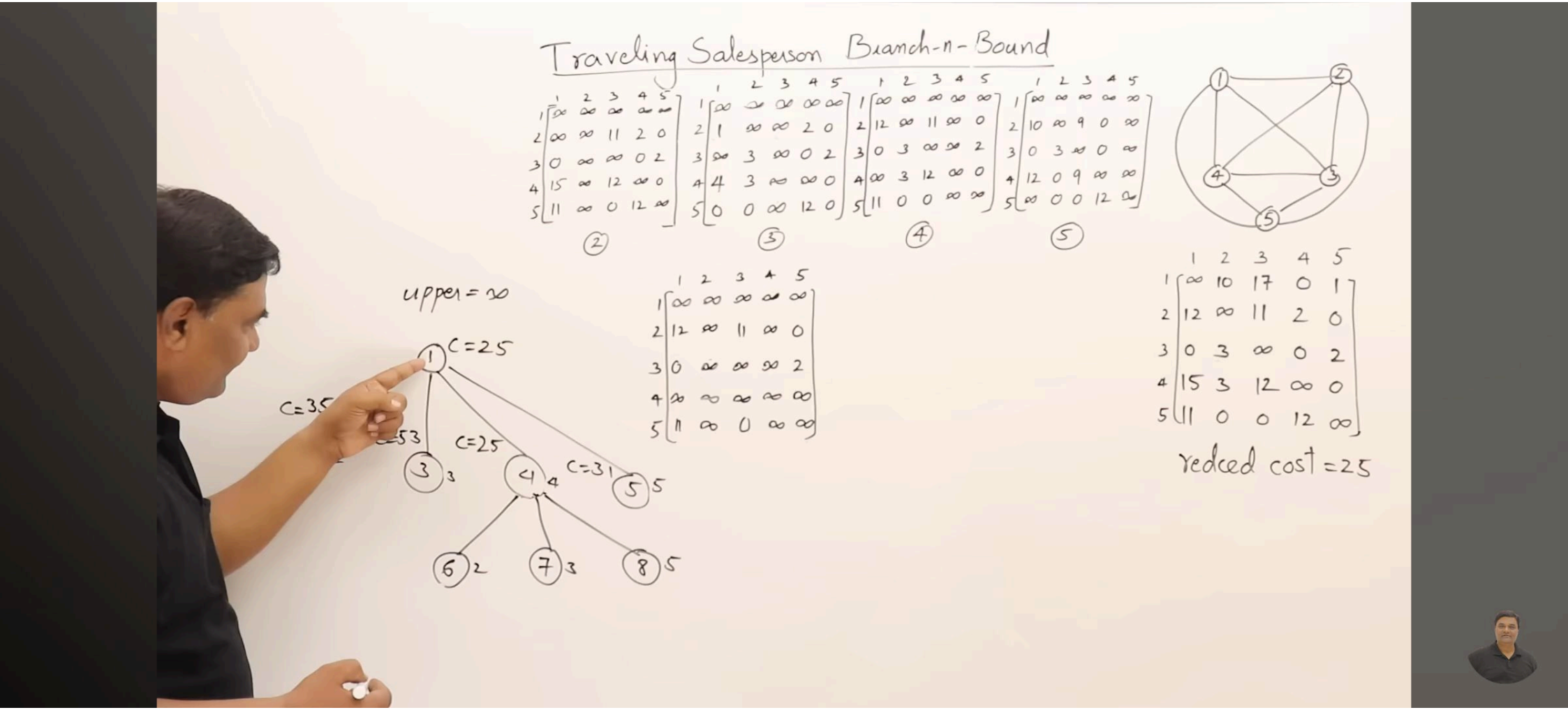
upper = ∞



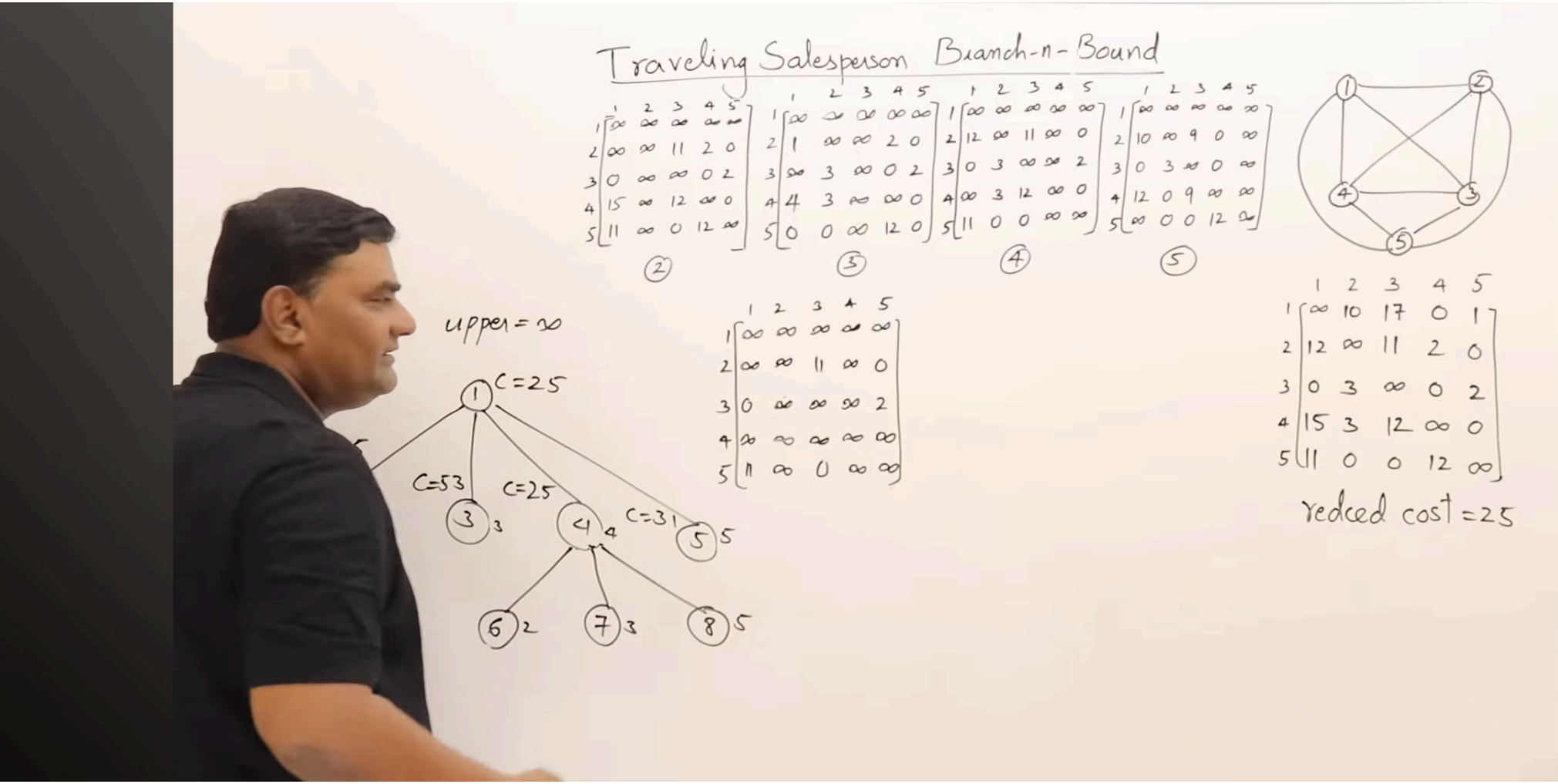
	1	2	3	4	5
1	∞	10	17	0	1
2	12	∞	11	2	0
3	0	3	∞	0	2
4	15	3	12	∞	0
5	11	0	0	12	∞

reduced cost = 25

Traveling Salesperson Branch-n-Bound



Traveling Salesperson Branch-n-Bound



	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	11	2	0
3	0	∞	∞	0	2
4	15	∞	12	∞	0
5	11	∞	0	12	∞

(2)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	1	∞	∞	2	0
3	∞	3	∞	0	2
4	4	3	∞	∞	0
5	0	0	∞	12	0

(3)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	12	∞	11	∞	0
3	0	3	∞	∞	2
4	∞	3	12	∞	0
5	11	0	0	∞	∞

(4)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	10	∞	9	0	∞
3	0	3	∞	0	∞
4	12	0	9	∞	∞
5	∞	0	0	12	∞

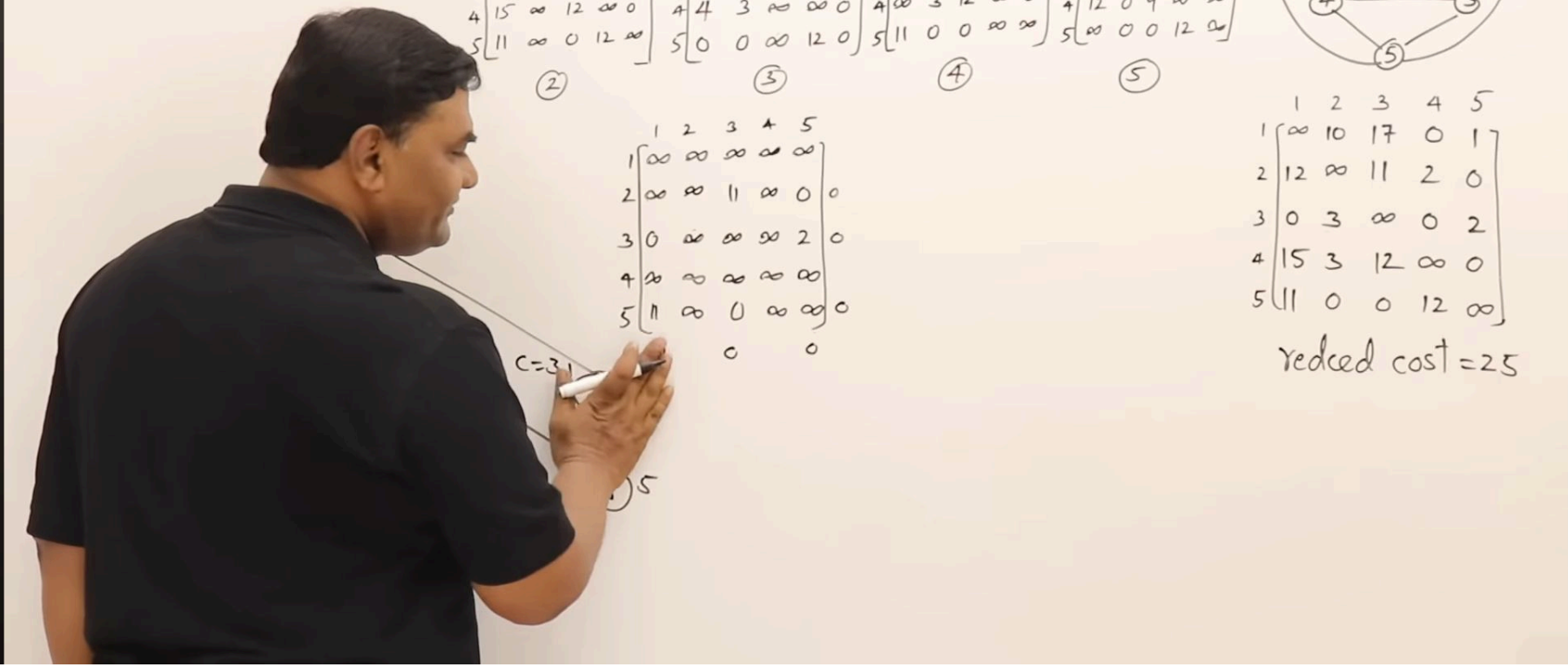
(5)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	11	∞	0
3	0	∞	∞	∞	2
4	∞	∞	∞	∞	∞
5	11	∞	0	∞	∞

	1	2	3	4	5
1	∞	10	17	0	1
2	12	∞	11	2	0
3	0	3	∞	0	2
4	15	3	12	∞	0
5	11	0	0	12	∞

reduced cost = 25

Traveling Salesperson Branch-n-Bound



Traveling Salesperson Branch-n-Bound

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	11	2	0
3	0	∞	∞	0	2
4	15	∞	12	∞	0
5	11	∞	0	12	∞

(2)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	1	∞	∞	2	0
3	∞	3	∞	0	2
4	4	3	∞	∞	0
5	0	0	∞	12	0

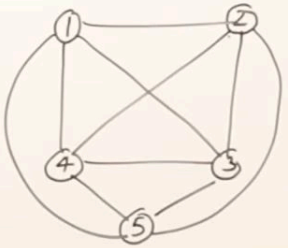
(3)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	12	∞	11	∞	0
3	0	3	∞	∞	2
4	∞	3	12	∞	0
5	11	0	0	∞	∞

(4)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	10	∞	9	0	∞
3	0	3	∞	0	∞
4	12	0	9	∞	∞
5	∞	0	0	12	∞

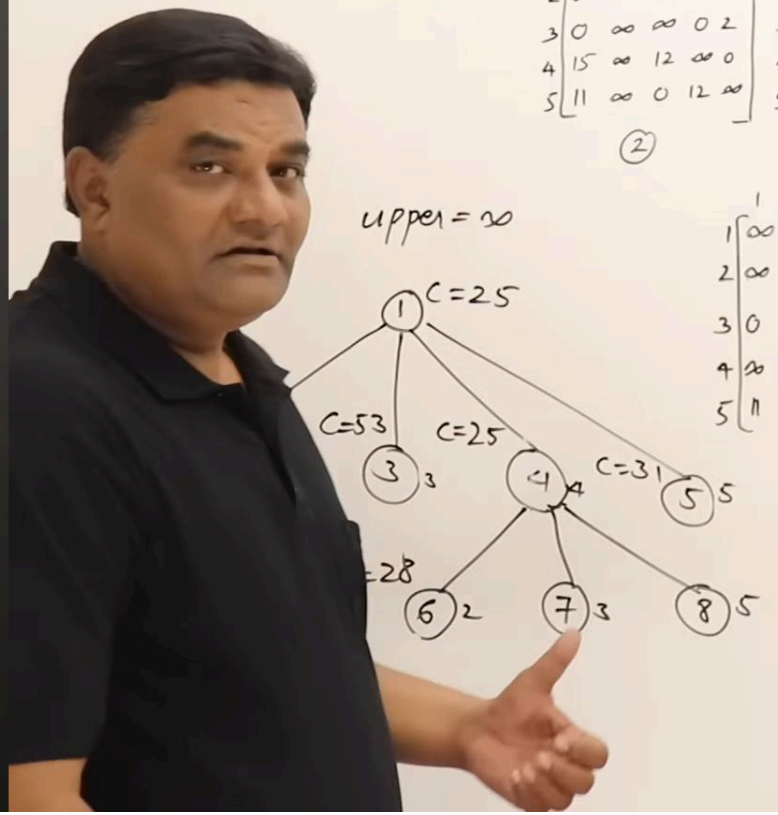
(5)



	1	2	3	4	5
1	∞	10	17	0	1
2	12	∞	11	2	0
3	0	3	∞	0	2
4	15	3	12	∞	0
5	11	0	0	12	∞

reduced cost = 25

upper = ∞

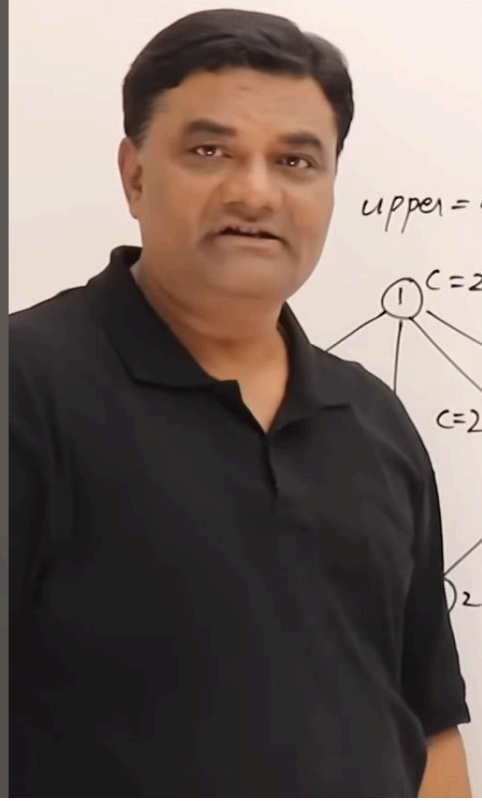


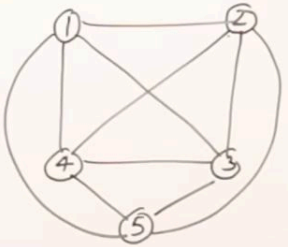
	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	11	∞	0
3	0	∞	∞	∞	2
4	∞	∞	∞	∞	∞
5	11	∞	0	∞	∞

$c(4,2) + c(4) + \hat{r}$

$3 + 25 + 0 = 28$

Traveling Salesperson Branch-n-Bound





	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	11	2	0
3	0	∞	∞	0	2
4	15	∞	12	∞	0
5	11	∞	0	12	∞

(2)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	1	∞	∞	2	0
3	∞	3	∞	0	2
4	4	3	∞	∞	0
5	0	0	∞	12	0

(3)

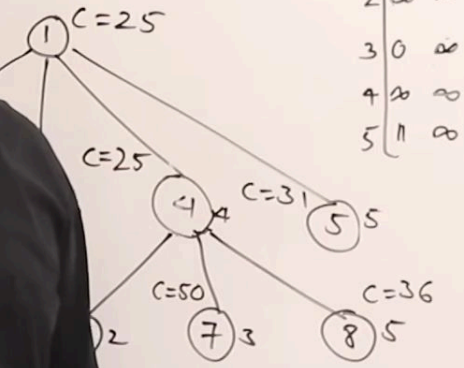
	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	12	∞	11	∞	0
3	0	3	∞	∞	2
4	∞	3	12	∞	0
5	11	0	0	∞	∞

(4)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	10	∞	9	0	∞
3	0	3	∞	0	∞
4	12	0	9	∞	∞
5	∞	0	0	12	∞

(5)

upper = ∞



	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	11	∞	0
3	0	∞	∞	0	2
4	∞	∞	∞	∞	∞
5	11	∞	0	∞	∞

(6)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	1	∞	∞	∞	0
3	∞	1	∞	∞	0
4	∞	∞	∞	∞	∞
5	0	0	∞	∞	∞

(7)

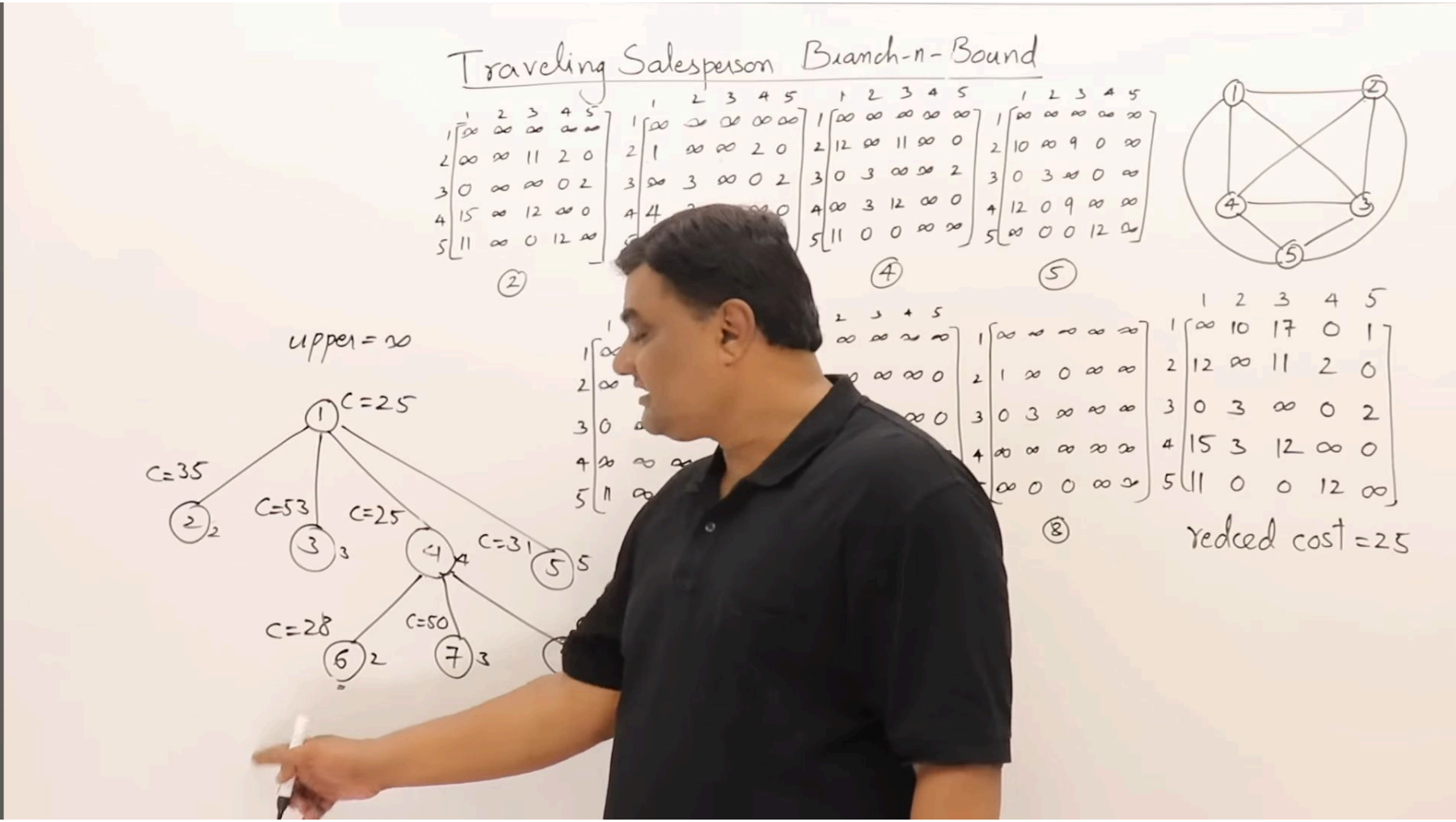
	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	1	∞	0	∞	∞
3	0	3	∞	∞	∞
4	∞	∞	∞	∞	∞
5	∞	0	0	∞	∞

(8)

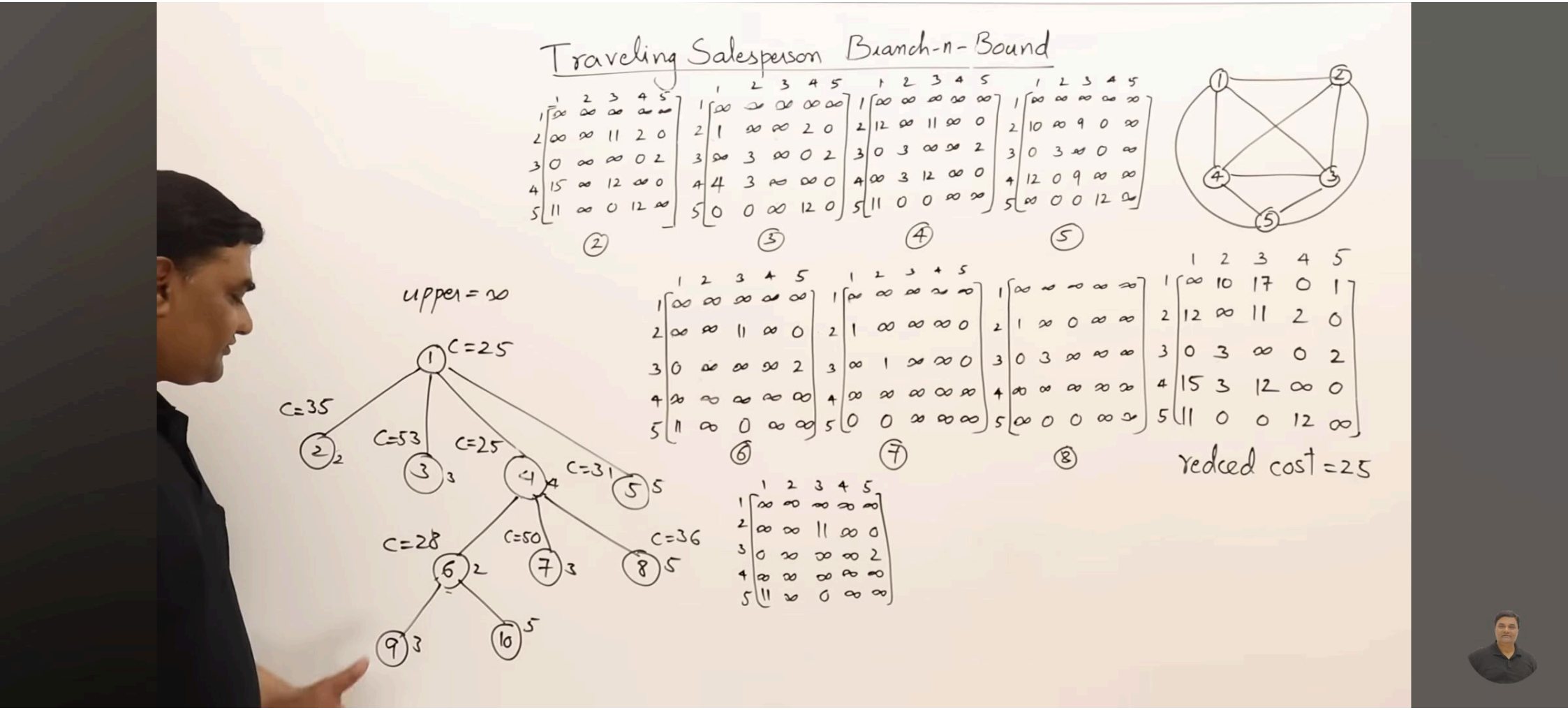
	1	2	3	4	5
1	∞	10	17	0	1
2	12	∞	11	2	0
3	0	3	∞	0	2
4	15	3	12	∞	0
5	11	0	0	12	∞

reduced cost = 25

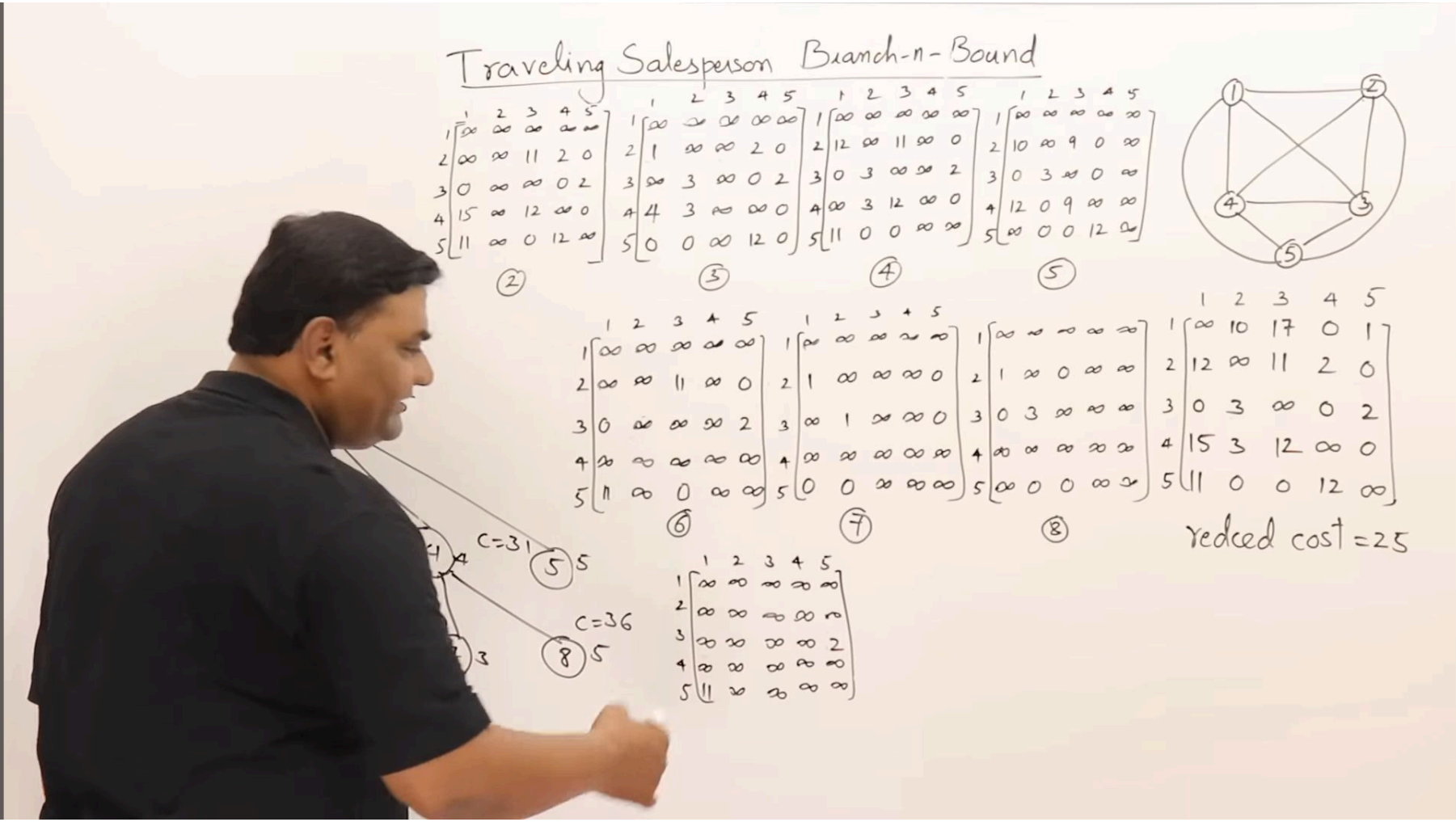
Traveling Salesperson Branch-n-Bound



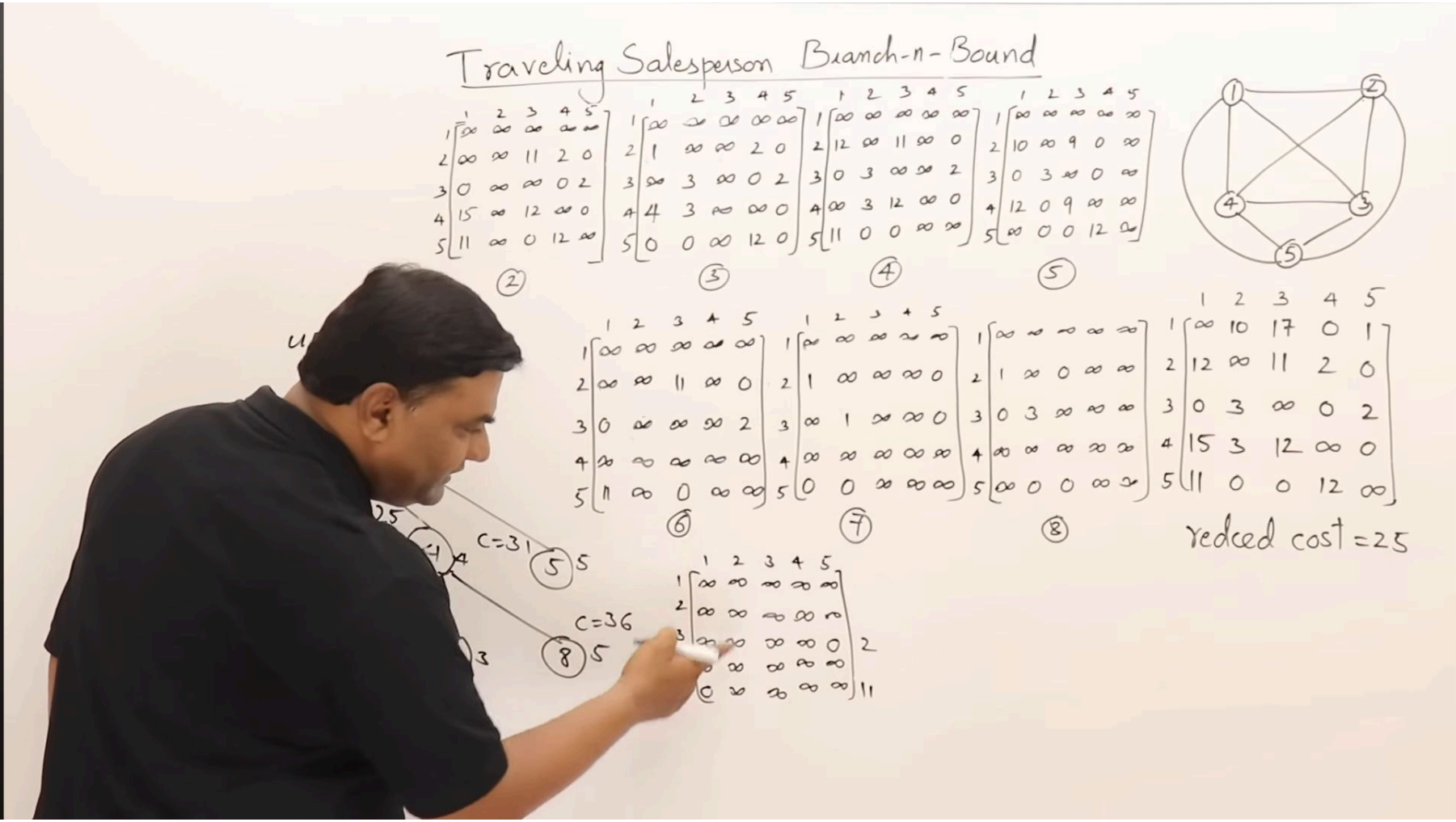
Traveling Salesperson Branch-n-Bound



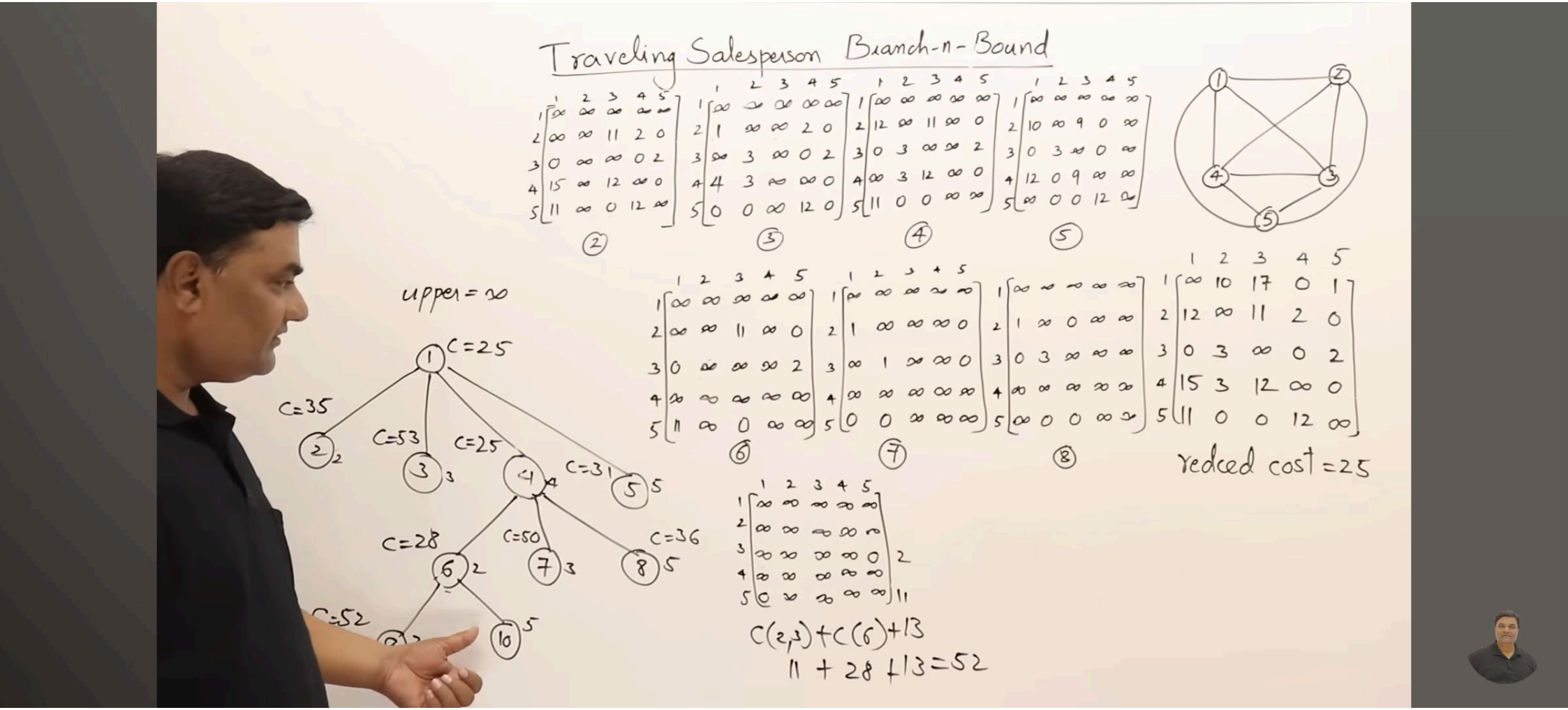
Traveling Salesperson Branch-n-Bound



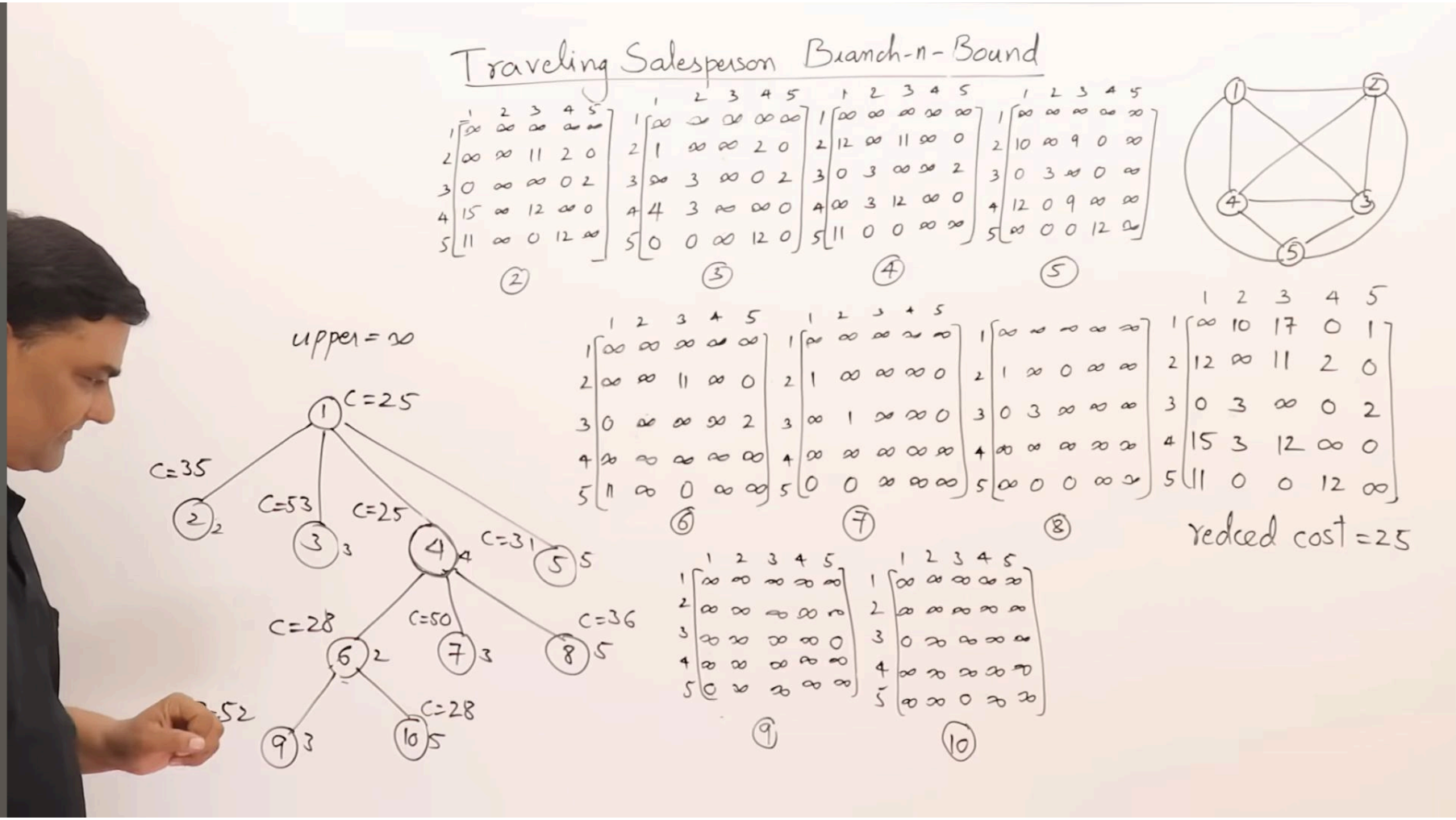
Traveling Salesperson Branch-n-Bound

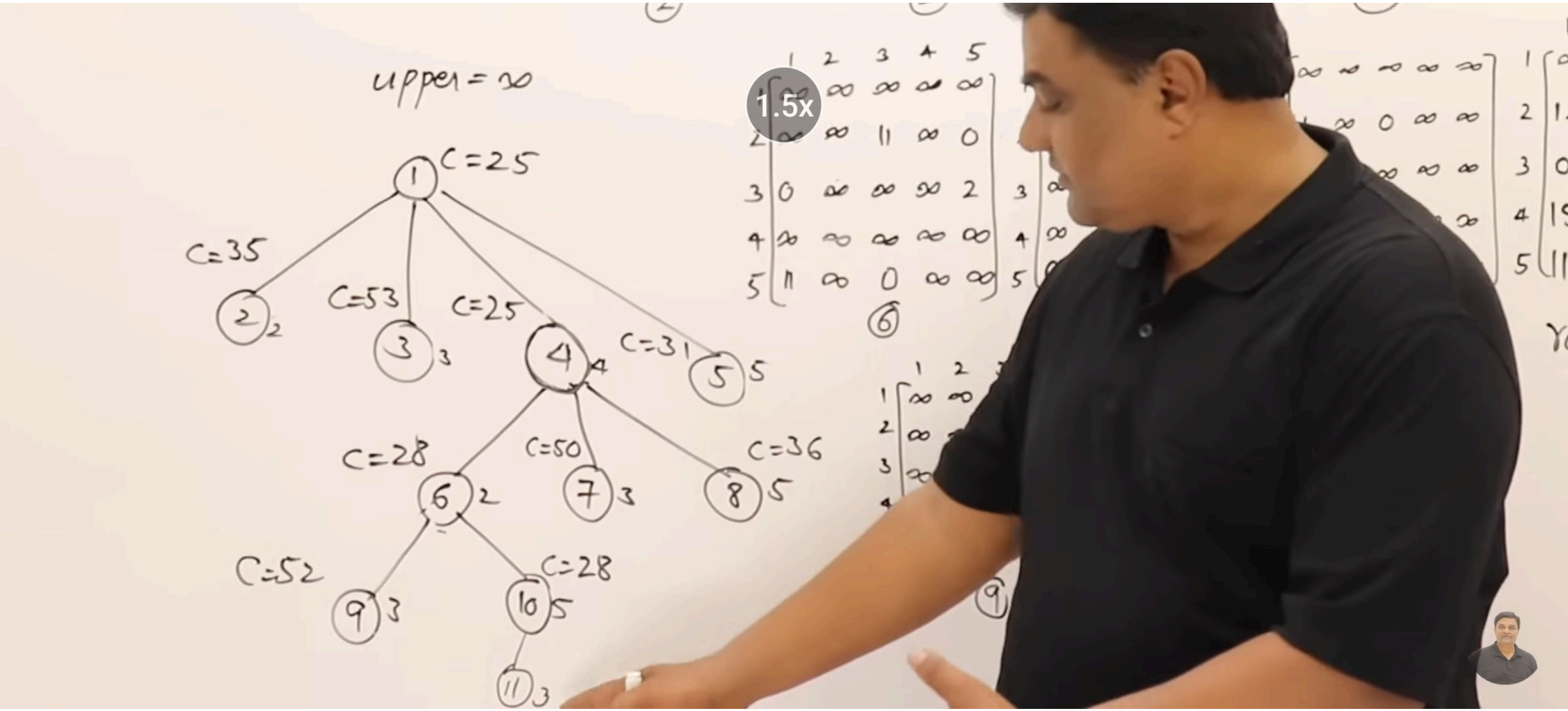


Traveling Salesperson Branch-n-Bound

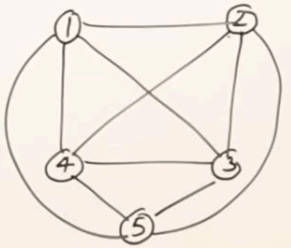


Traveling Salesperson Branch-n-Bound




upper = ∞
C=25
C=35
C=53
C=25
C=31
C=28
C=50
C=36
C=52
C=28
1.5x

Traveling Salesperson Branch-n-Bound



	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	11	2	0
3	0	∞	∞	0	2
4	15	∞	12	∞	0
5	11	∞	0	12	∞

(2)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	1	∞	∞	2	0
3	∞	3	∞	0	2
4	4	3	∞	∞	0
5	0	0	∞	12	0

(3)

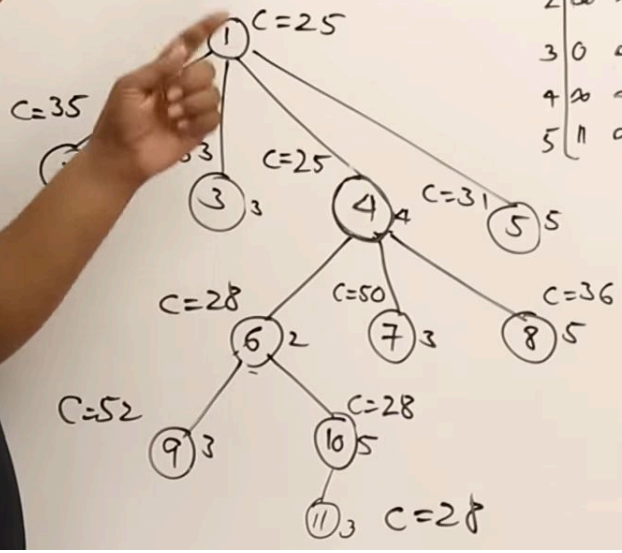
	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	12	∞	11	∞	0
3	0	3	∞	∞	2
4	∞	3	12	∞	0
5	11	0	0	∞	∞

(4)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	10	∞	9	0	∞
3	0	3	∞	0	∞
4	12	0	9	∞	∞
5	∞	0	0	12	∞

(5)

upper = ∞



	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	11	∞	0
3	0	∞	∞	∞	2
4	∞	∞	∞	∞	∞
5	11	∞	0	∞	∞

(6)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	1	∞	∞	∞	0
3	∞	1	∞	∞	0
4	∞	∞	∞	∞	∞
5	0	0	∞	∞	∞

(7)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	1	∞	0	∞	∞
3	0	3	∞	∞	∞
4	∞	∞	∞	∞	∞
5	∞	0	0	∞	∞

(8)

	1	2	3	4	5
1	∞	10	17	0	1
2	12	∞	11	2	0
3	0	3	∞	0	2
4	15	3	12	∞	0
5	11	0	0	12	∞

reduced cost = 25

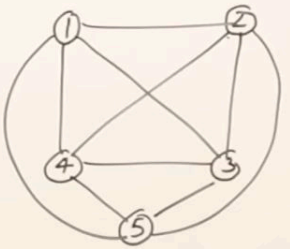
	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	∞	∞	∞
3	∞	∞	∞	∞	0
4	∞	∞	∞	∞	∞
5	0	∞	∞	∞	∞

(9)

	1	2	3	4	5
1	∞	∞	∞	∞	∞
2	∞	∞	∞	∞	∞
3	0	∞	∞	∞	∞
4	∞	∞	∞	∞	∞
5	∞	∞	0	∞	∞

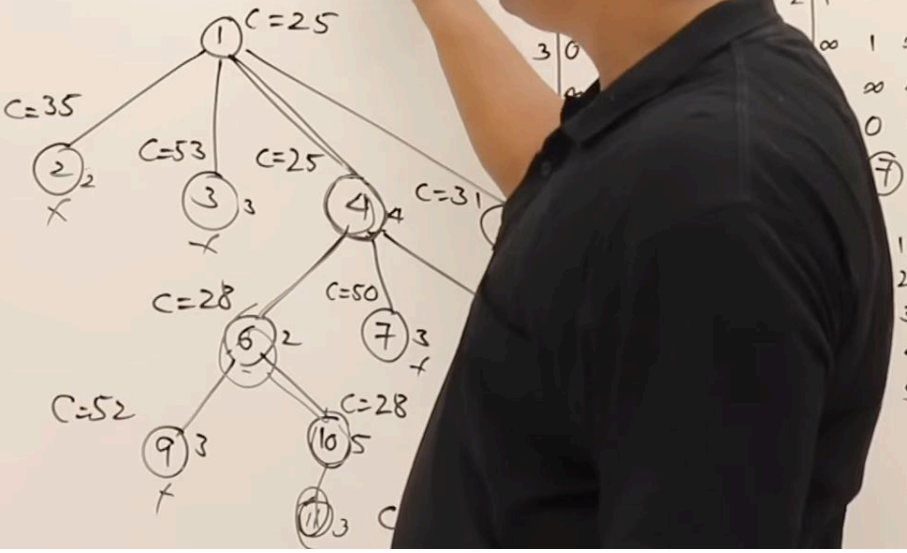
(10)

Traveling Salesperson Branch-n-Bound



1 - 4 - 2 - 5 - 3 - 1

upper = ∞ 2



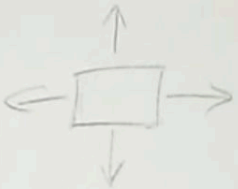
reduced cost = 25

15 Puzzle problem (Using branch and bound)

15 - Puzzle Problem

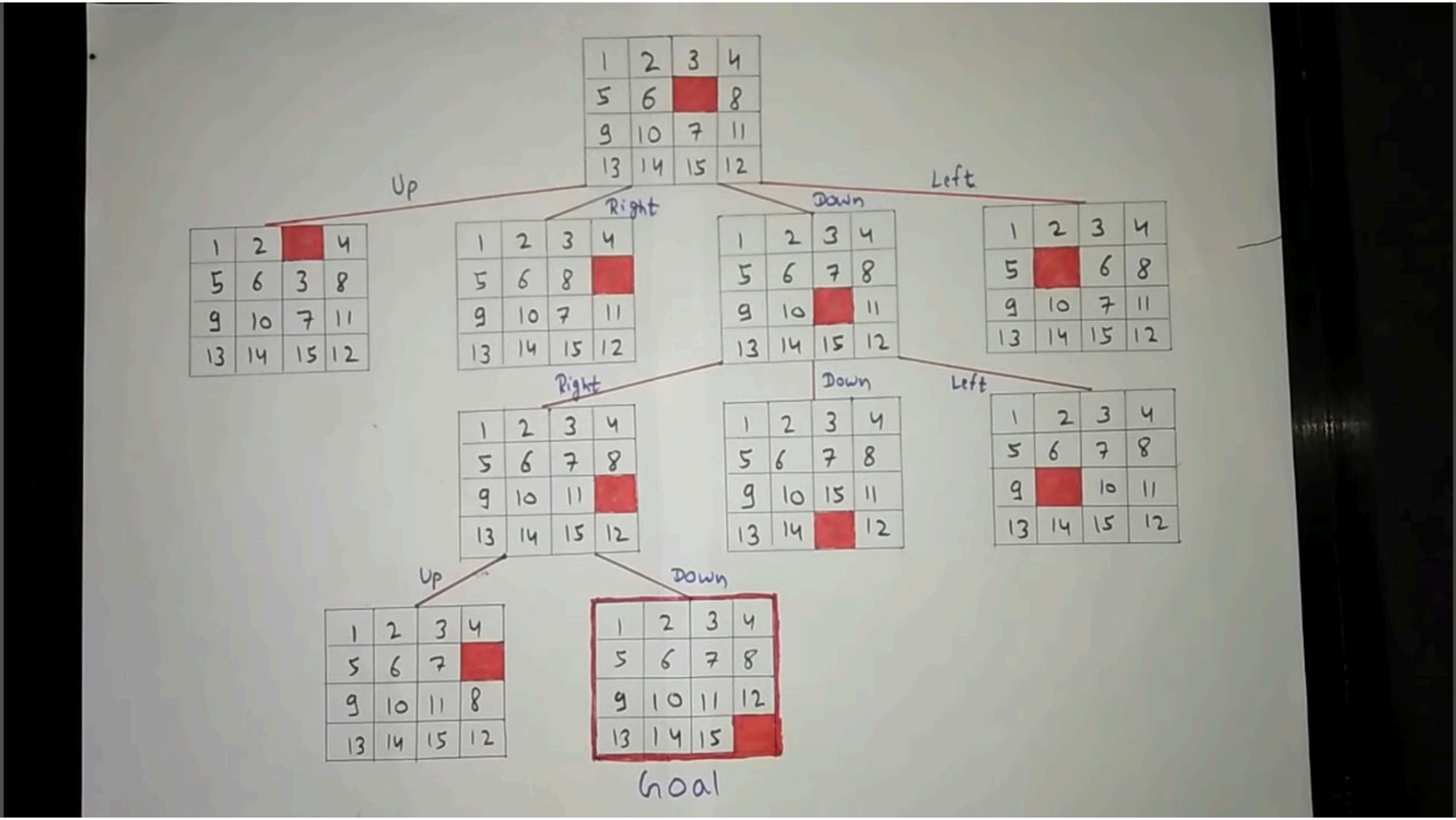
1	2	3	4
5	6		8
9	10	7	11
13	14	15	12

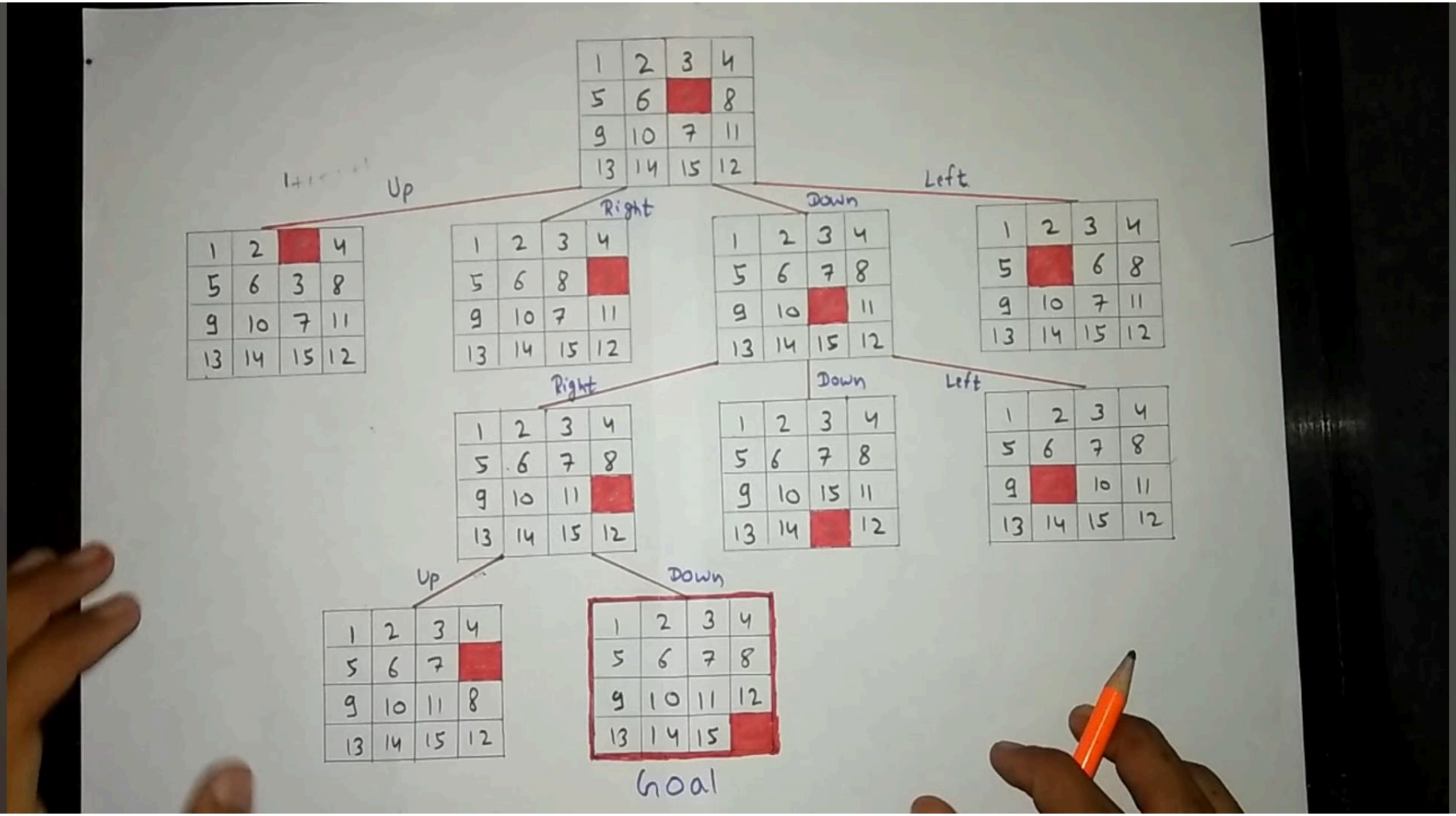
An arrangement

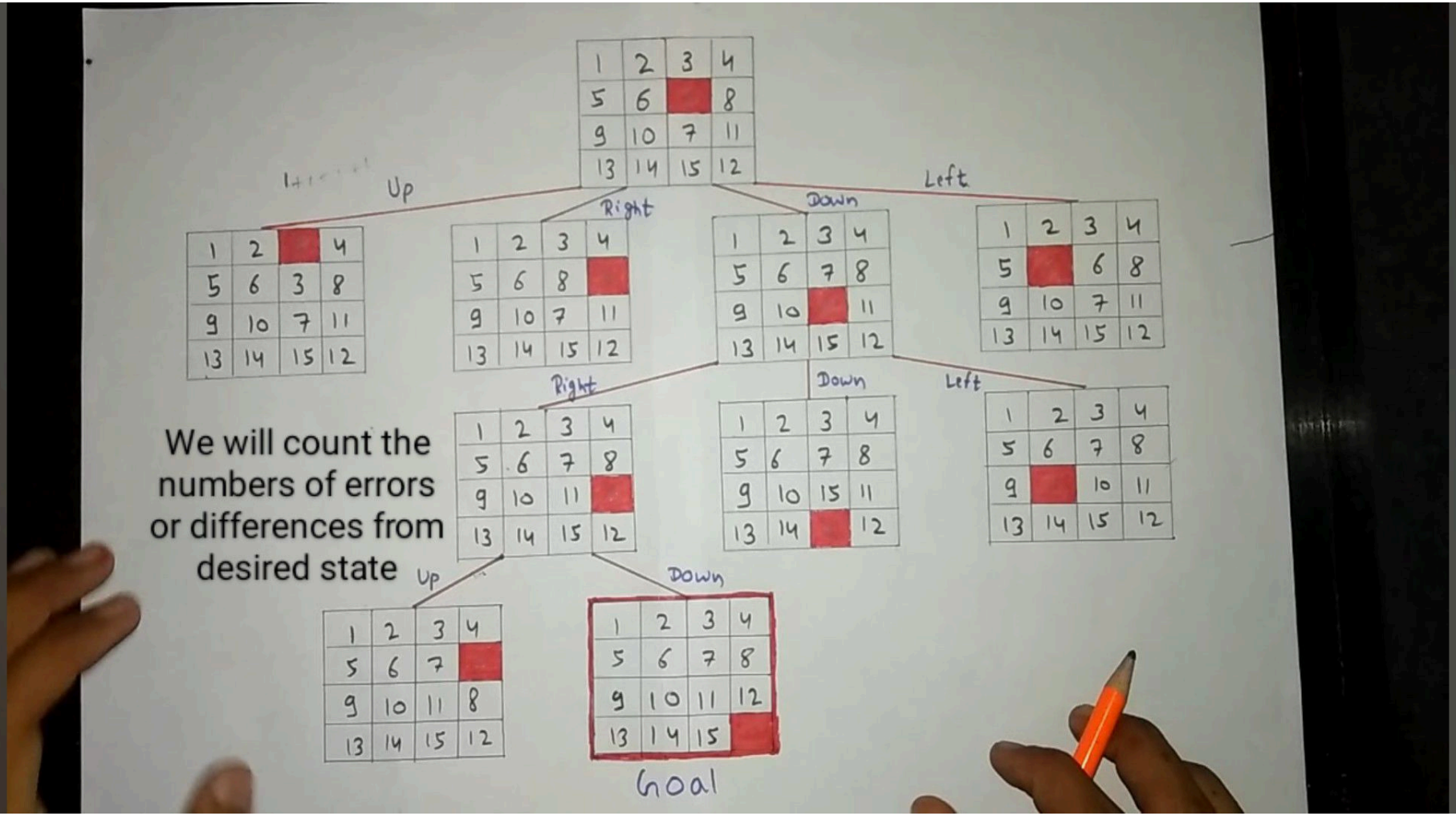


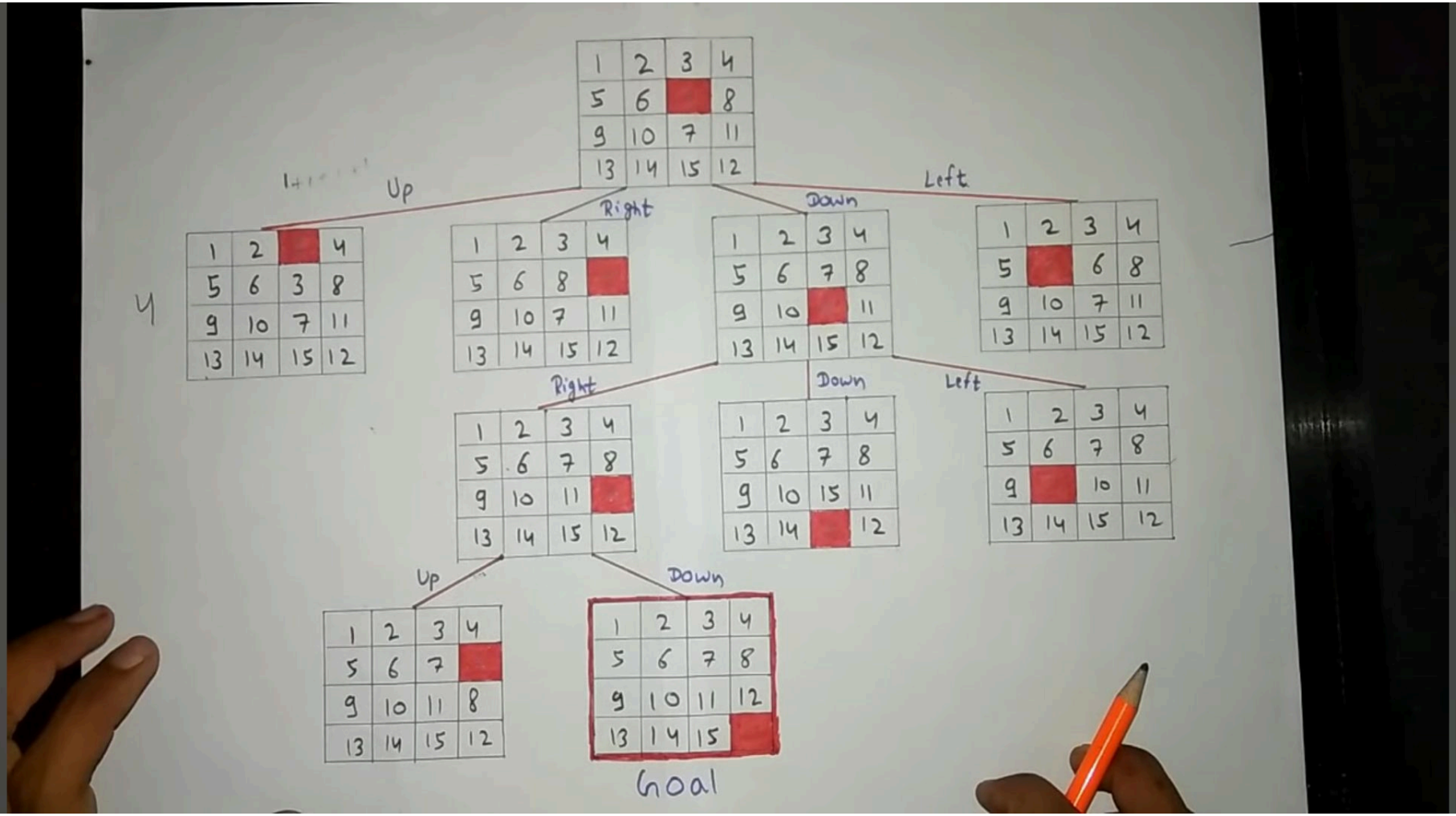
1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

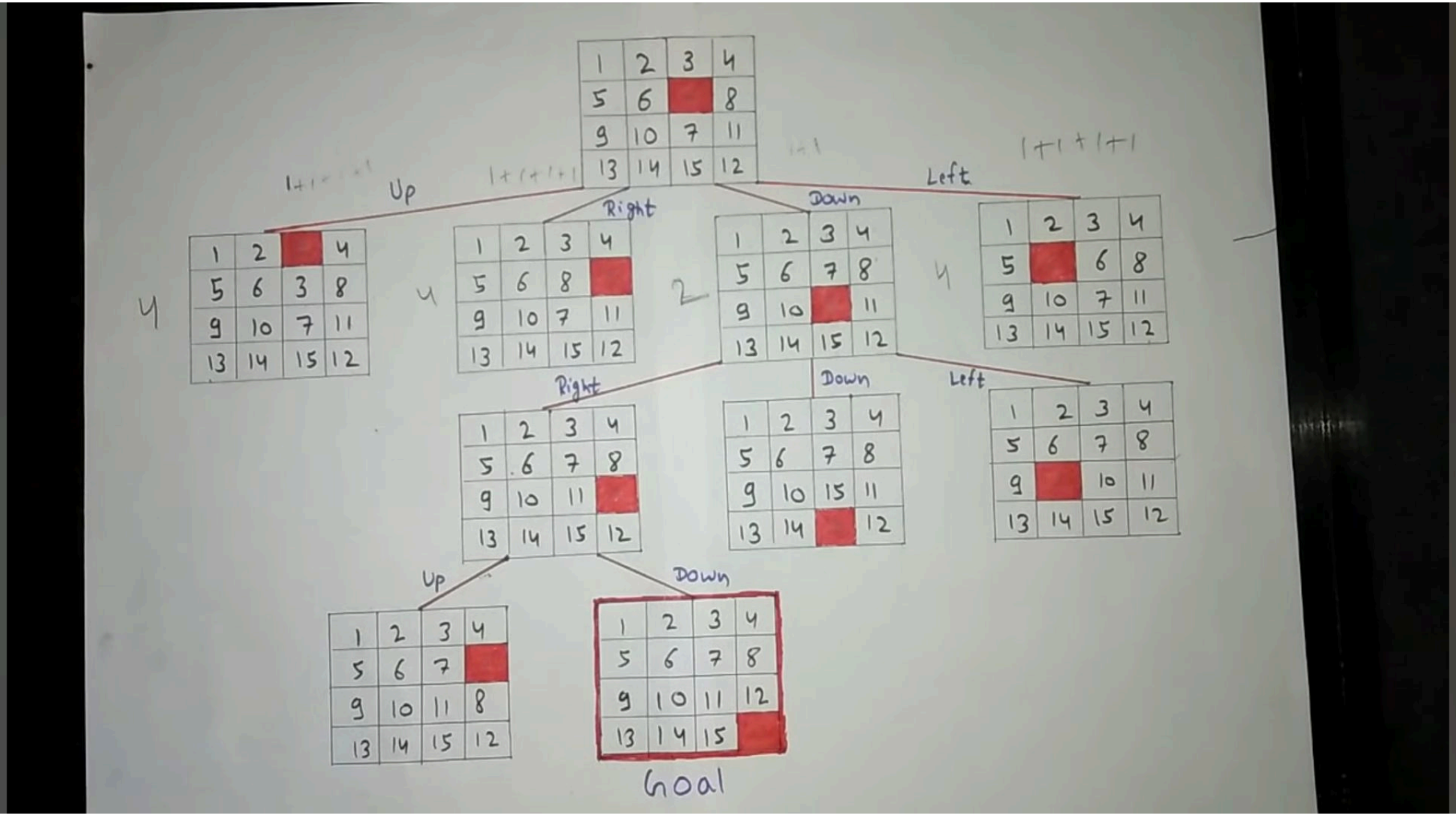
Goal arrangement

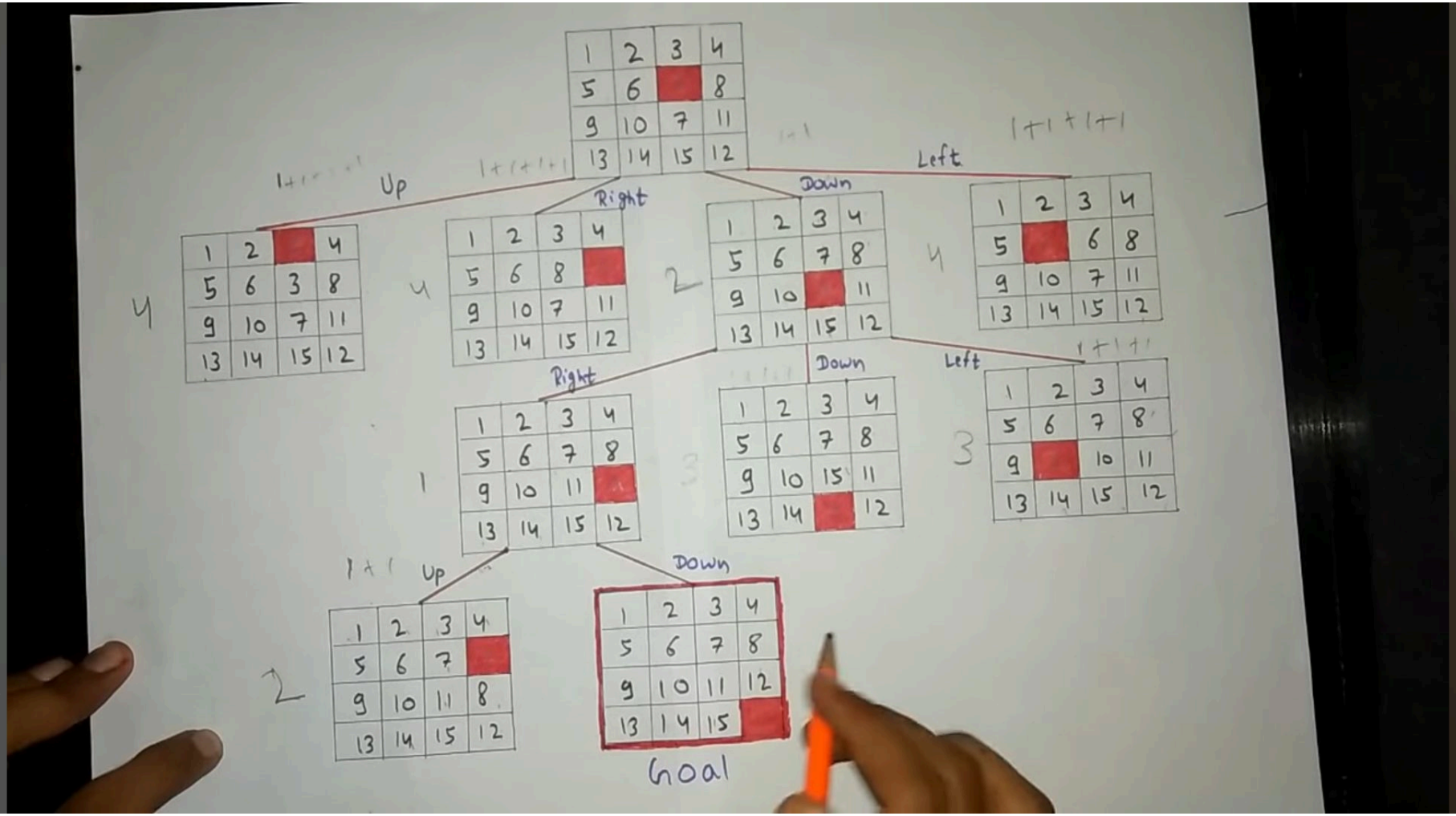

Up
Right
Down
Left
Right
Down
Left
Up
Down
Goal


Up
Right
Down
Left
Right
Down
Left
Up
Down
Goal


Up
Right
Down
Left
Right
Down
Left
Up
Down
Goal
We will count the numbers of errors or differences from desired state


Up
Right
Down
Left
Right
Down
Left
Up
Down
Goal


Up
Right
Down
Left
Up
Right
Down
Down
Left
Down
Goal


Up
Right
Down
Left
Up
Right
Down
Left
Down
Goal