Class - 3 → Algorithms

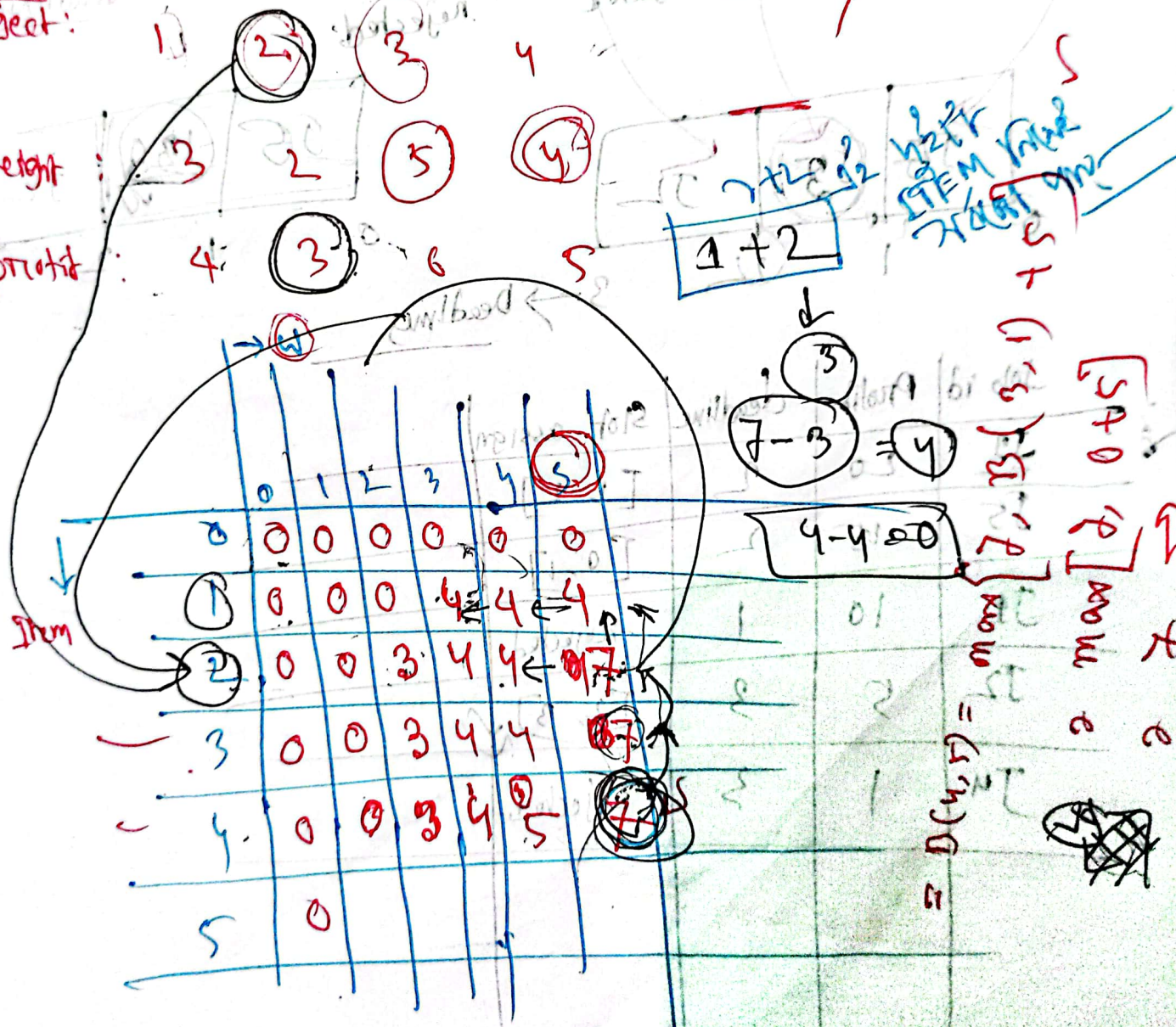
Greedy algorithms:

0/1 Knapsack problem: → All নিবে নাকি কিছু নিবে না

Item / object:

weight:

profit:



$B[i, w] = \max(B[i-1, w], B[i-1, w-w[i]] + P[i])$

$B(4,5) = \max[B(3,5), B(3,(5-4)) + 5]$

Job sequencing problem

Job id:	1	2	3	4	5
profit:	10	5	20	1	15
deadlines:	1	3	2	3	2

maximum deadlines

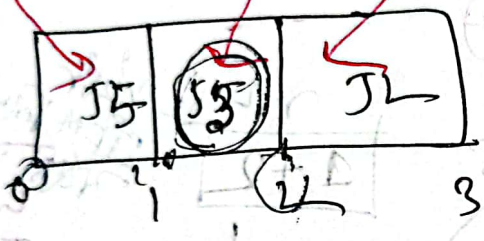
highest profit

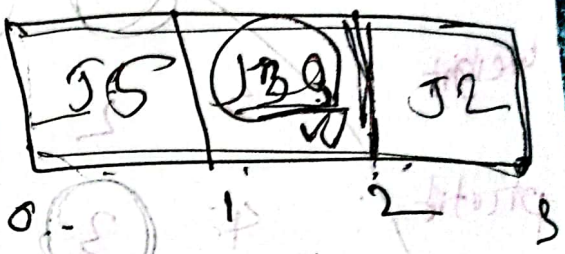
job id: 3 5 1 2 4

profit: 20 15 10 5 1

deadlines: 2 2 1 3 3 → rejected

rejected



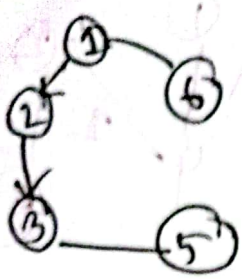


Job id	Profit	deadline	Slot assign
J3	20	2	[1, 2]
J5	15	2	[0, 1]
J1	10	1	rejected
J2	5	3	[2, 3]
J4	1	3	rejected

Solution: J3 + J5 + J2

Total = 40 profit

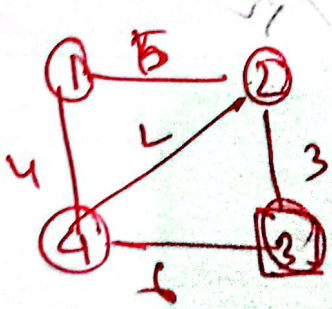
Minimum cost spanning tree:



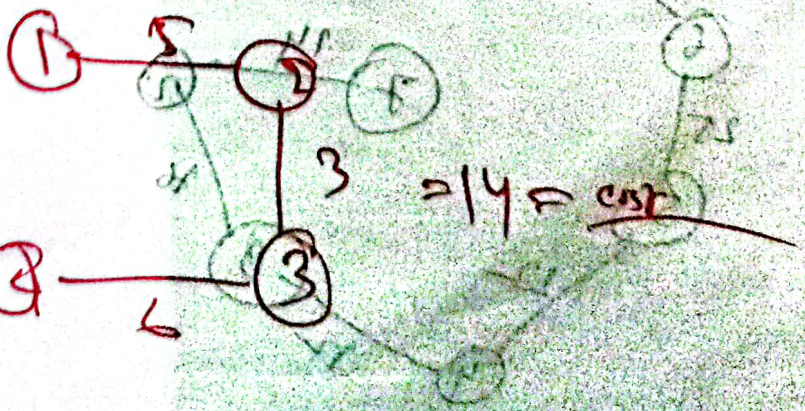
Spanning tree: Spanning tree is a subgraph of a graph where we should take all vertices and only (n-1) edges.

No. of cycle = $^{e}C_{(v-1)}$ — no. of cycle

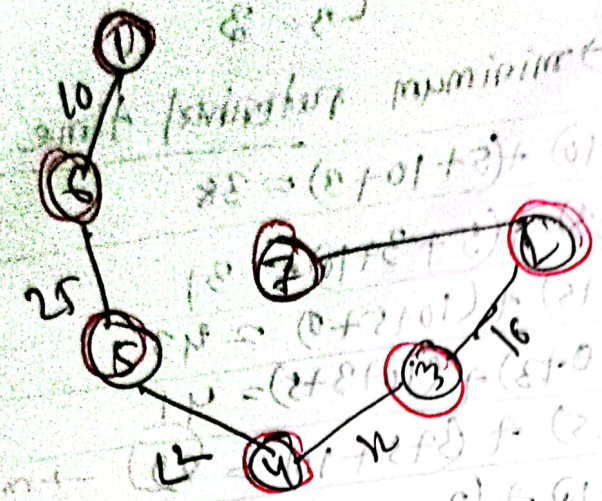
Minimum spanning tree.



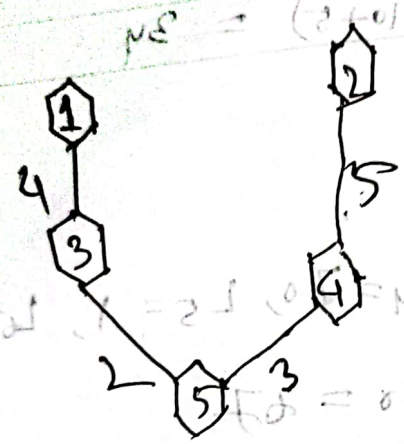
Spanning tree →



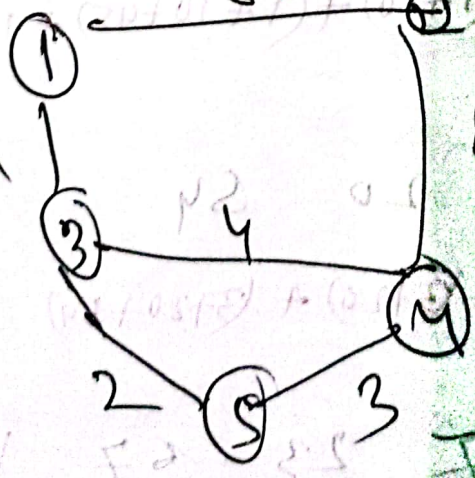
Kruskal's algorithm:



Q.



main tree



Optimal storage tapes:

$n = 3$
$L_1 = 5$
$L_2 = 10$
$L_3 = 3$

ordering	MRT → minimum retrieval time
1, 2, 3	$5 + (5+10) + (5+10+3) = 38$
1, 3, 2	$5 + (5+3) + (5+3+10) = 31$
2, 1, 3	$10 + (10+5) + (10+5+3) = 43$
2, 3, 1	$10 + (10+3) + (10+3+5) = 41$
3, 1, 2	$3 + (3+5) + (3+5+10) = 29$ → minimum
3, 2, 1	$3 + (3+10) + (3+10+5) = 34$

Q. $L_1 = 10$, $L_2 = 20$, $L_3 = 45$, $L_4 = 70$, $L_5 = 1$, $L_6 = 3$
$L_7 = 7$, $L_8 = 54$, $L_9 = 23$, $L_{10} = 67$

ascending order —
1, 3, 7, 10, 20, 23, 45, 54, 67, 70

Tape → 1 → 1 10 45 70 = 12

$1 + (1+10) + (1+10+45) + 1+45+10+70$

Tape → 2 → 3 20 54 = 77

$3 + (3+20) + (3+20+54)$

Tape → 3 → 7 23 67 = 97

Dynamic programming →

Class Date
22.03.23
TNT

Difference between greedy method and D.P →

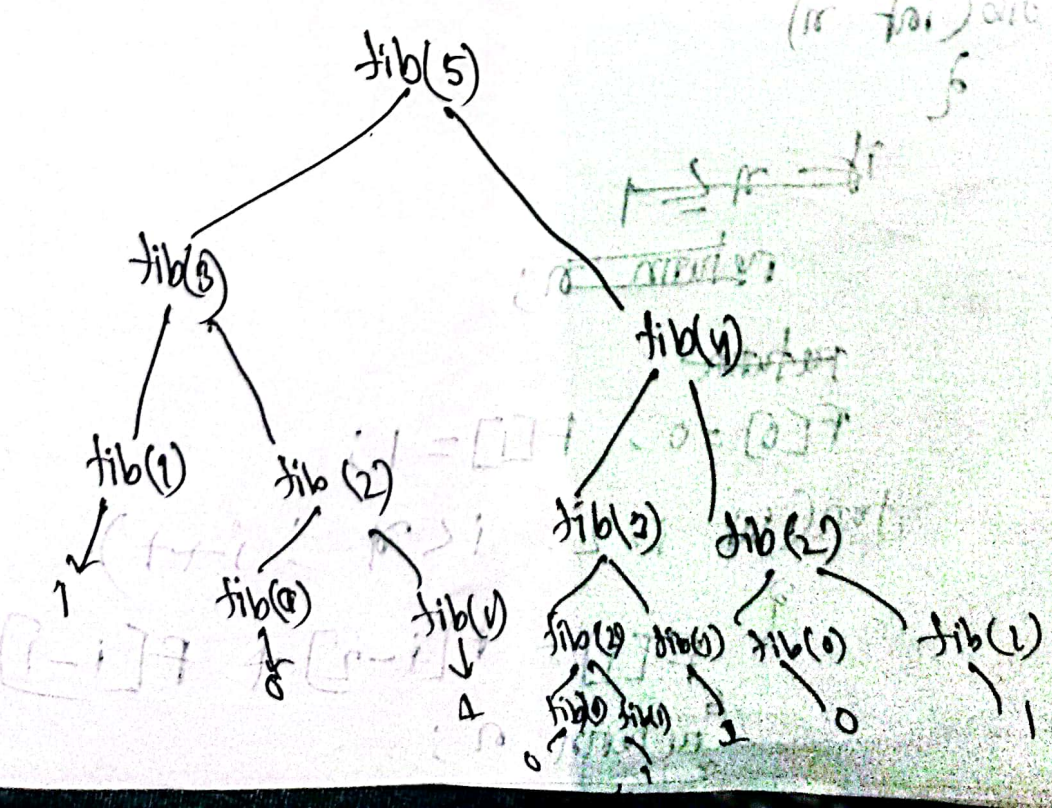
Principle of optimality :

$$fib(n) = \begin{cases} 0 & \text{if } n=0 \\ 1 & \text{if } n=1 \\ fib(n-2)+fib(n-1) & \text{if } n>1 \end{cases}$$

```
int fib(int n)
{
    if(n<=1)
    return n;
    return fib(n-2) + fib(n-1);
}
```

fib(5)

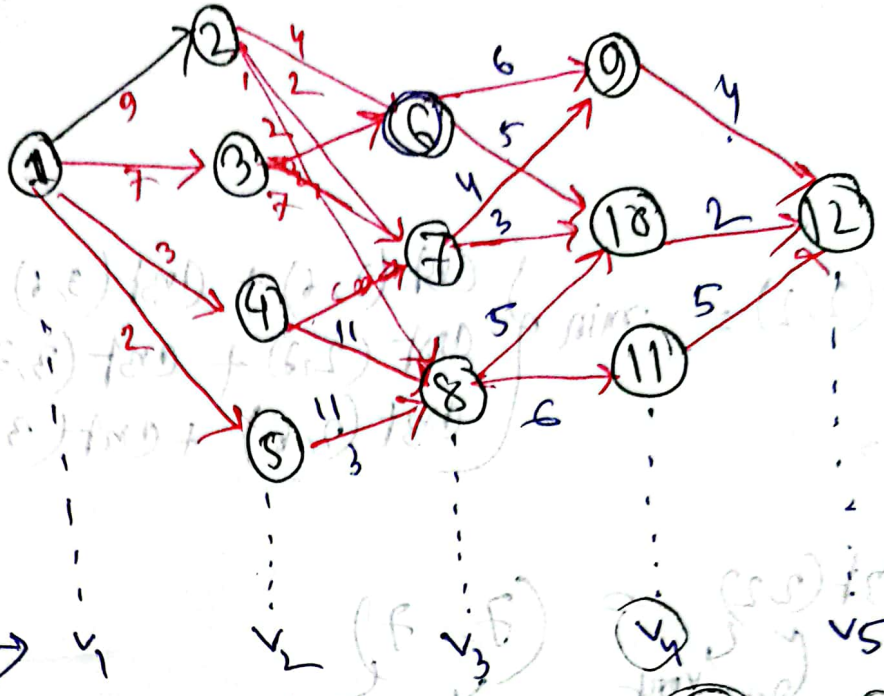
Tracing tree



Multistage graph : (Directed + weighted) condition

(a)

Question



⇒ Cost(i, j) = min { c(j, l) + Cost(j+1, l) }

Vertex

V	1	2	3	4	5	6	7	8	9	10	11	12
Cost	16	7	9	18	15	7	5	7	4	2	5	0
d	3/3	7	6	8	8	10	10	10	12	12	12	12

Cost(5, 12) = 0

Stage → Vertex

Cost(4, 9) = 4

Cost(4, 10) = 2

Cost(4, 11) = 5

Cost(3, 6) = 6 min { Cost(6, 9) + Cost(4, 9) = 6 + 4 = 10
Cost(6, 10) + Cost(4, 10) = 5 + 2 = 7 }

Vertex → min

Cost min

Cost(3, 7) = min { 10, 7 }

Cost(3, 7) → min { Cost(7, 9) + Cost(4, 9)
Cost(7, 10) + Cost(4, 10) }

Cost(3, 8) = min { 10, 5 }

cost(3, 8) = min {

cost(2, 2) = min { cost(2, 6) + cost(3, 6) = 11 ; cost(2, 7) + cost(3, 7) = 7 ; cost(2, 8) + cost(3, 8) = 8 }

cost(2, 2) = (7, 7)

stage, vert, vertex, cost, path

Path →

d(1, 1) = 2 (stage, vertex)

d(2, 2) = (7)

d(3, 7) = 10

d(4, 11) = 12

d(1, 3) = (3)

d(2, 6) = 6

d(3, 6) = 10

d(4, 10) = 12

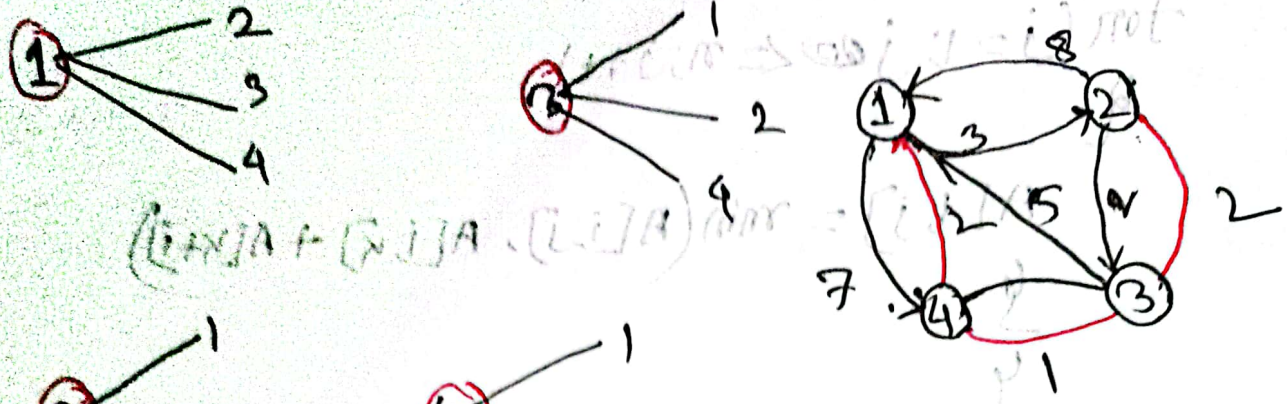
→ 9 + 2 + 3 + 2 = 16

= 7 + 2 + 5 + 2 = 16

Shortest path → distance

Shortest path distance

All pair shortest path (Floyd Warshall's algorithm)



$$A^0 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 3 & \infty & 7 \\ 8 & 0 & 2 & \infty \\ 5 & \infty & 0 & 1 \\ 2 & \infty & \infty & 0 \end{bmatrix} \end{array}$$

$$A^1 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 3 & \infty & 7 \\ 8 & 0 & 2 & 5 \\ 8 & 8 & 0 & 1 \\ 2 & 8 & \infty & 0 \end{bmatrix} \end{array}$$

$A[i,j] = \min\{A^0[i,j],\ A^0[i,v] + A^0[v,j]\}$

v = vertex

$A[2,3] = \min\{A^0[2,3],\ A^0[2,1] + A^0[1,3]\}$

$= 2$

$$A^2 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 3 & 5 & 7 \\ 8 & 0 & 2 & 15 \\ 5 & 8 & 0 & 1 \\ 2 & 5 & 7 & 0 \end{bmatrix} \end{array}$$

$$A^3 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 3 & 5 & 6 \\ 7 & 0 & 2 & 3 \\ 5 & 8 & 0 & 1 \\ 2 & 5 & 7 & 0 \end{bmatrix} \end{array}$$

$$A^4 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 3 & 5 & 6 \\ 5 & 0 & 2 & 3 \\ 3 & 6 & 0 & 1 \\ 2 & 5 & 7 & 0 \end{bmatrix} \end{array}$$

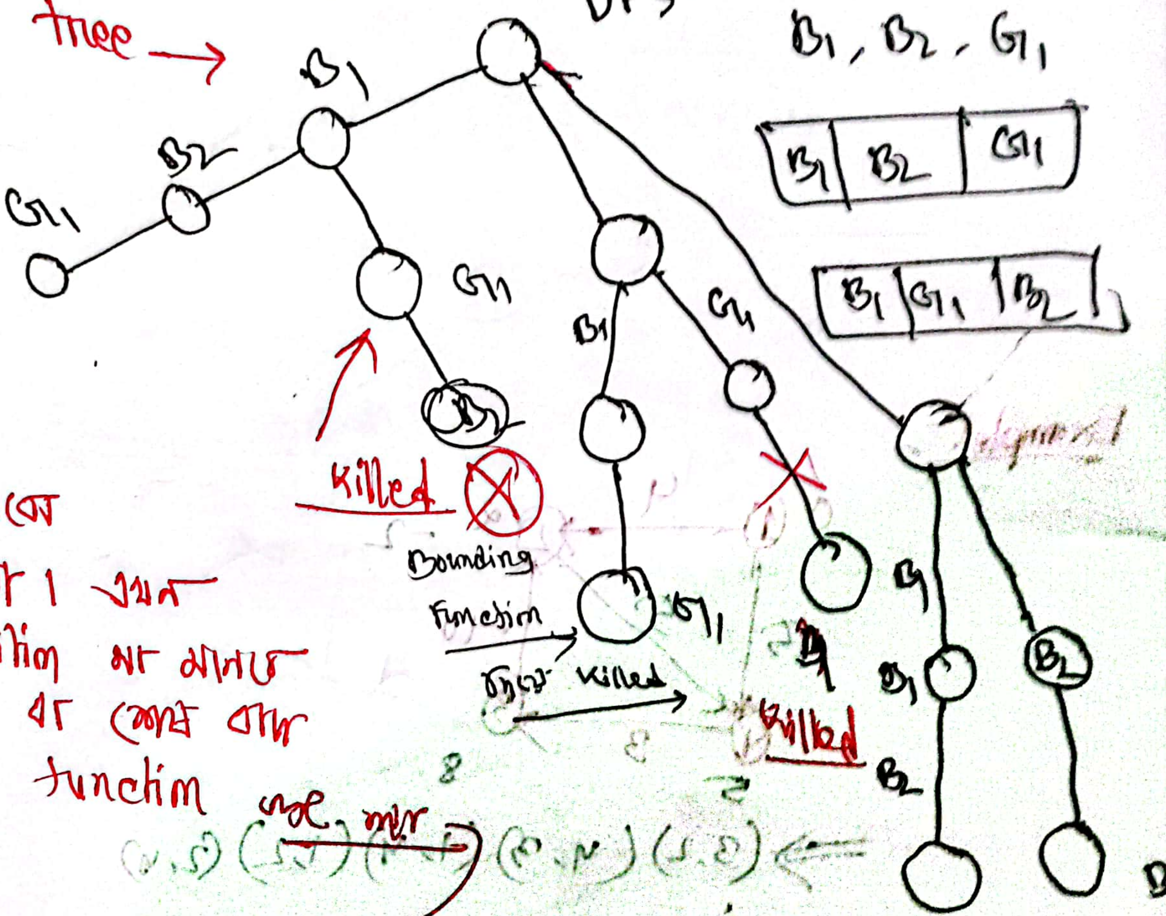
Backtracking algorithm →

→ Brute force approach →

→ State space tree →

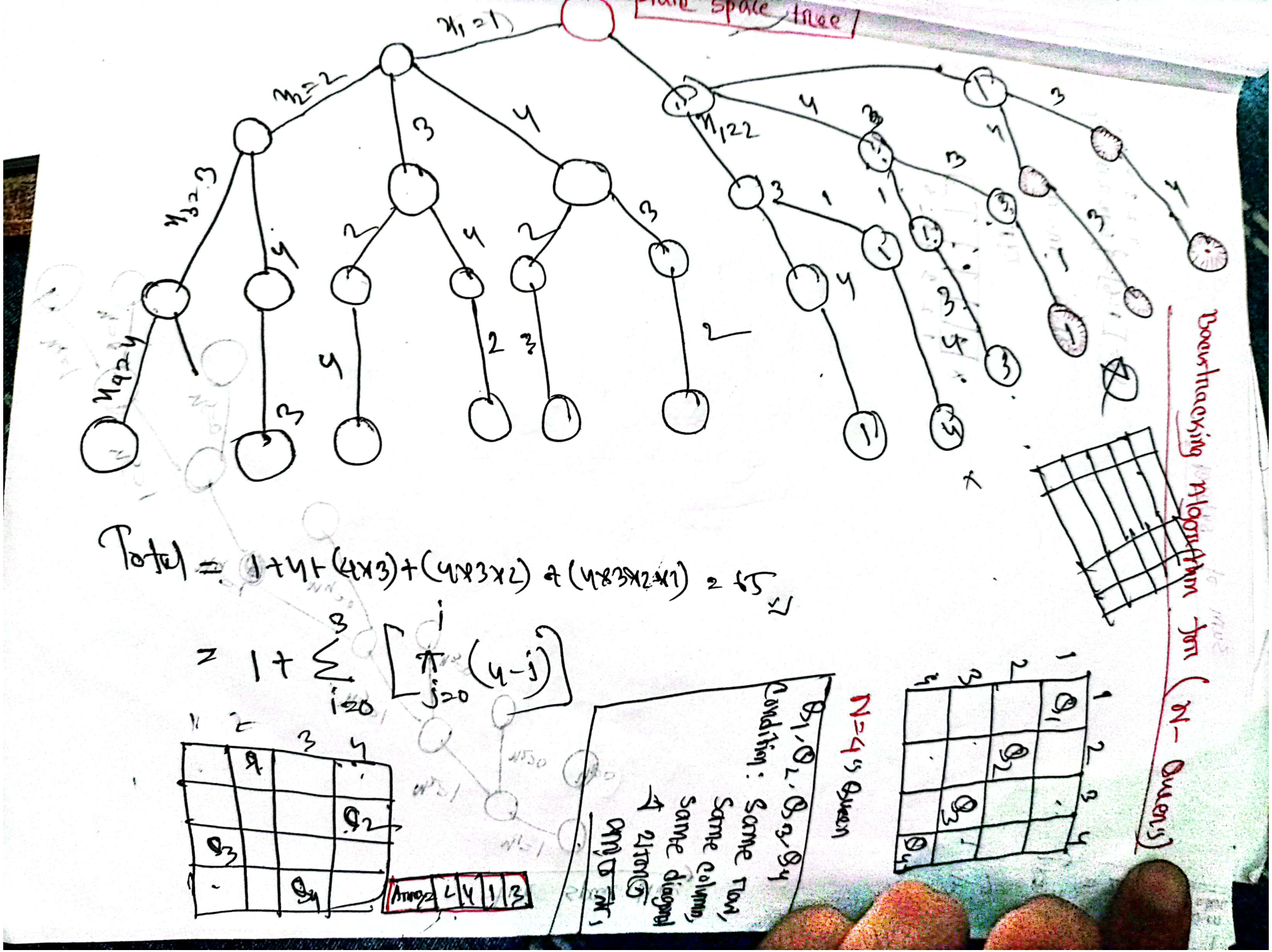
DFS — Searching হয়

B_1, B_2, G_1



Condition G_1 মানে
যাদের পাবে না। এখন
যে যে condition না মানে
সেগুলো killed হবে কোনো বাদ
এর Bounding function

Branch and Bound →


State space tree
Backtracking algorithm for (N- Queens)
N=4 Queen
Total = 1+4+(4x3)+(4x3x2)+(4x3x2x1) = 65
$= 1 + \sum_{i=0}^{3} \left[\prod_{j=0}^{i} (4-j) \right]$
Condition: Same row, Same column, same diagonal
Array 2 4 1 3