what is algorithm?

= An algorithm is any well defined computational procedure that takes some input and produce some output. An algorithm is thus a sequence of computational steps that transform the input into the output.

what kinds of problems are solved by algorithms?

(i) Human genome mapping.

(ii) Routing of data through internet

(iii) Electronics commence security concern.

(iv) gndustrial manufacturing sector

(v) And many more.

Efficiency of algorithm

(i) computer speed is not infinite and memory is not free so we need efficiency algorithm.

(ii) Algorithm efficiency can be determined by time and space it need.

(iii) For example, insertion sort takes time equal to $c_1 n^2$ to sort n items while merge sort takes $c_2 n \log n$.

(iv) Which one would work faster depends on size of n.

Example of efficiency:-

computer A is 100 times faster than B.

→ A takes $2n^2$ time to sort n items using insertion sort.

$$\frac{2(10^6)^2 \text{ instructions}}{10^8 \text{ instructions/second}} = 2000 \text{ second}$$

→ B takes $50 \log n$ time to sort n items using merge sort

$$\frac{50 \cdot 10^6 \log 10^6 \text{ instructions}}{10^7 \text{ instructions/second}}$$

= 100 second.

Analysis of algorithms:-

⇒• Analysis is performed with respeet to a computationa model

- we will usually use a generic uniprocessor random access machine.
- All memory equally expensive to access.
- No concurrent operations.
- All reasonable instructions take unit time
- constant word size.

Asymptotic performance

- Running time
- memory / storage requirments
- Remember that we use ~~th~~ the all RAM model.
- All memory equally expensive to access.
- All reasonable instruction take unit time.
- Constant word size.

An example → insertion sort

```
insertionsort (A,n){
    for i = 2 to n{
        key = A[i]
        j = i-1
      while (j<0) and (A[j] > key)){
              A[j+1] = A[j]
              j = j-1
           }
         A[j+1] = key
      }
   }
```

Job sequence PRoblem:-

Job id:- 1 2 3 4 5

PRofit :- 10 5 20 1 15

deadlines :- 1 3 2 3 2

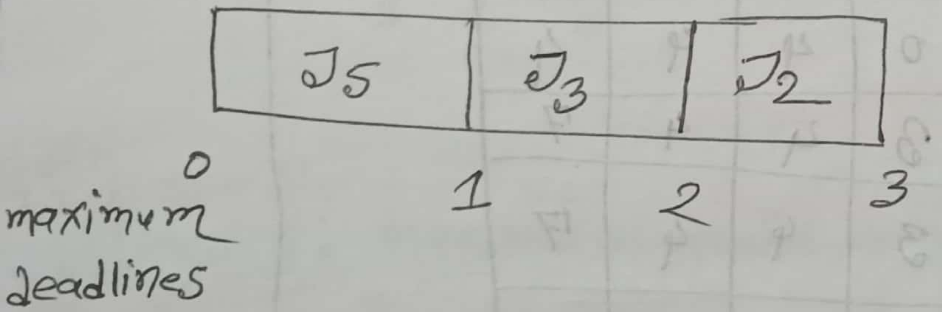
solⁿ:- Job id :- 3 5 1 2 4 [PRofit অনুযায়ী সাজাতে হবে]

PRofit :- 20 15 10 5 1

deadlines :- 2 , 2 1 3 3

1 → Rejected

3 → Rejected



Job id	PRofit	deadlines	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 : J_3, J_5, J_2

total profit = 20+15+5 = 40

0/1 knapsack problem :-

Dynamic programming

youtube
Farhan Hossain

object : 1 2 3 4

weight : 3 2 5 4

profit : 4 3 6 5

maximum weight = 5

↓ object / → weight	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	0	4	4	4
2	0	0	3	4	4	7
3	0	0	3	4	4	7
4	0	0	3	4	5	7

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

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

= 4, B = ~~(3, 0, 5)~~ (3,0) + 5

= ~~3, 5)~~ = 0+5

= 5

= max (4, 5) = 5

$B(4.5) = \max(3,5) \quad B = [3.[5-4]+5$

$= \max(B[3,5)], B = [3,1]+5$

$= \max(7, \quad B = 0+5)$

$= \max(7.5)$

$= 7$

maximum profit :-

item:	1	2	3	4
	1	1	0	0

7-3 =4
4-4 =0

∴ maximum profit = 4+3 = 7

knapsack problem : Greedy method

object :-	1	2	3	4	5	6	7
profit :	12	5	16	7	9	11	6
weight :	3	1	4	2	9	4	3
$\max(\frac{p}{w})$:	4	5	4	3.5	1	2.75	2

n = 7

m = 15

object	profit	weight	Remaining weight
2	5	1	15-1=14
1	12	3	14-3=11
3	16	4	11-4=7
4	7	2	7-2=5
6	11	4	5-4=1
7	$\frac{6}{3}=2$	1	1-0=0
	56	15	

~~maximum~~

Define greedy method.

⇒ A greedy algorithm is an algorithmic stategy that makes the best ~~choice~~ optimal choice at each small stage with the goal of this eventually leading to a globally optimum solution.

Define Dynamic programming.

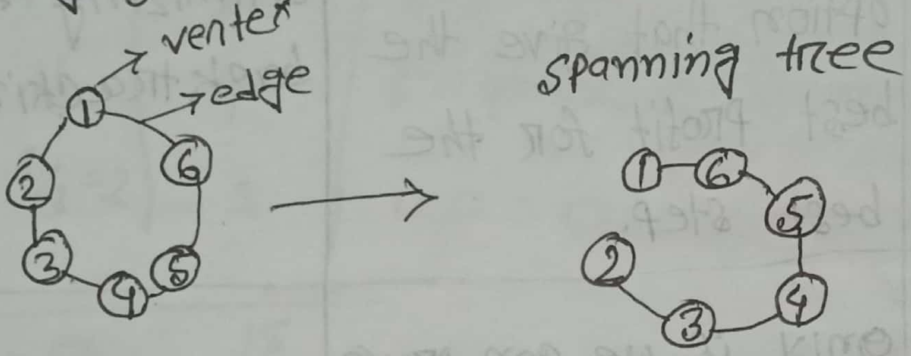
⇒ Dynamic programming is a technique in computer programming that helps to efficiently solve a class of problems that have overlapping subproblems and optimal substructure property

what is the difference between greedy method and dynamic programming.

	Greedy method	Dynamic programming
Main concept	choosing the best option that give the best profit for the best step.	optimizing the recursive backtracking solution.
optimality	only if we can prove that local optimality leads to global optimality.	Gives an optimal solution
Time complexity	polynomial	polynomial, but usually worse that the greedy approach.
memory complexity	more efficiently.	less efficiently
Examples	prim's algorithm	0/1 knapsack.

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.

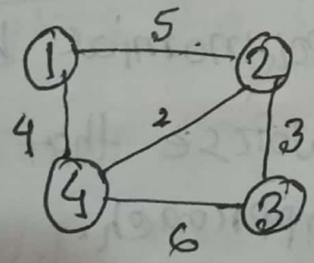


$E' = V-1$

কতটো spanning tree হবে তা বের করার সূত্র.

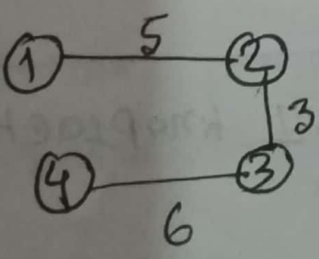
$EC_{\cancel{V}E'} = 6C_5 = 6$

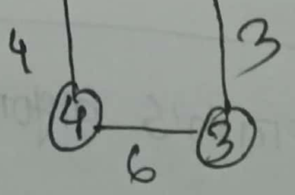
weighted graph.



$5C_3 - 2 = 8$

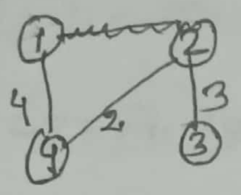
spanning tree





$4+6+3 = 13$

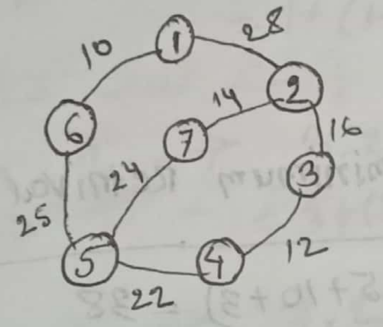
$5+3+6 = 14$



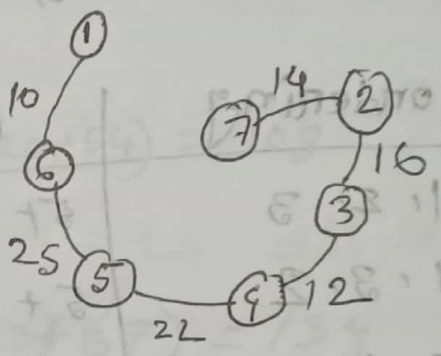
$4+2+3 = 9$

prims algorithm

[1st G minimom cost Related vertex নিতে হবে]

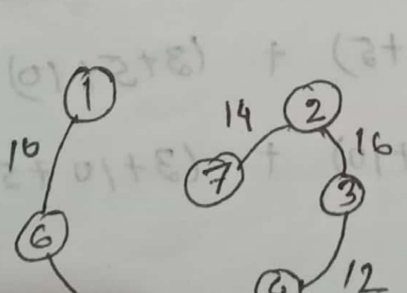


soln:-



cost = 99

kruzkal's algorithm



cost = 99

time

$\Theta(|V|\,|E|)$

$=(n.e)$

$=\Theta(n^2)$

Optimal storage on tapes:-

$n = 3$

$L_1 = 5$

$L_2 = 10$

$L_3 = 3$

L_1	L_2	L_3
5	(5+10)	(5+10+3)

ordering	MRT → minimum Retnival 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	$3 + (3+5) + (3+5+10) = \boxed{29}$
3, 2, 1	$3 + (3+10) + (3+10+5) = 34$

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

Solⁿ :- 1, 3, 7, 10, 20, 23, 45, 54, 67, 70

Tape 0 → 1, 10, 45, 70

$= 1 + (1+10) + (1+10+45) + (1+10+45+70) = 194$

Tape 1 → 3, 20, 54,

$= 3 + (3+20) + (3+20+54) = 103$

Tape 2 → 7, 23, 67,

$7 + (7+23) + (7+23+67) = 134$

$\therefore$ Average time $= \dfrac{194 + 103 + 134}{3} = 143.67$

state the 0/1 knapsack problem.

The 0/1 knapsack problem means that the items are either completely or no items are filled in a knapsack. for example;

we have two items having weights 2kg and 3kg respectively. If we pick the 2kg item then we cannot pick 1 kg item from the 2kg item. we have to pick the 2kg item completely.

Difference between prim'ss algorithm and kruskal algorithm.

prim's algorithm	kruskal Algorithm
① A greedy algorithm that finds a minimum spanning tree for a weighted undirected graph.	① A minimum spanning tree algorithm which finds an edge of the least possible weight that connects any two trees in the forest.
② It start to built the MST from any of the Node.	② It start to built the MST from minimum weighted vertex in the graph.
③ Adjacency matrix, Binary heap or Fibonacci Heap is used in prim's algorithm	③ Disjoint set is used in kruskal algorithm.
④ prim's algorithm run faster in dense graphs.	④ kruskal algorithm rus faster in sparse graph.
⑤ Greedy algorithm	⑤ Greedy algorithm.

Scanned with CamScanner

Backtracking algorithm

Brute force approach.

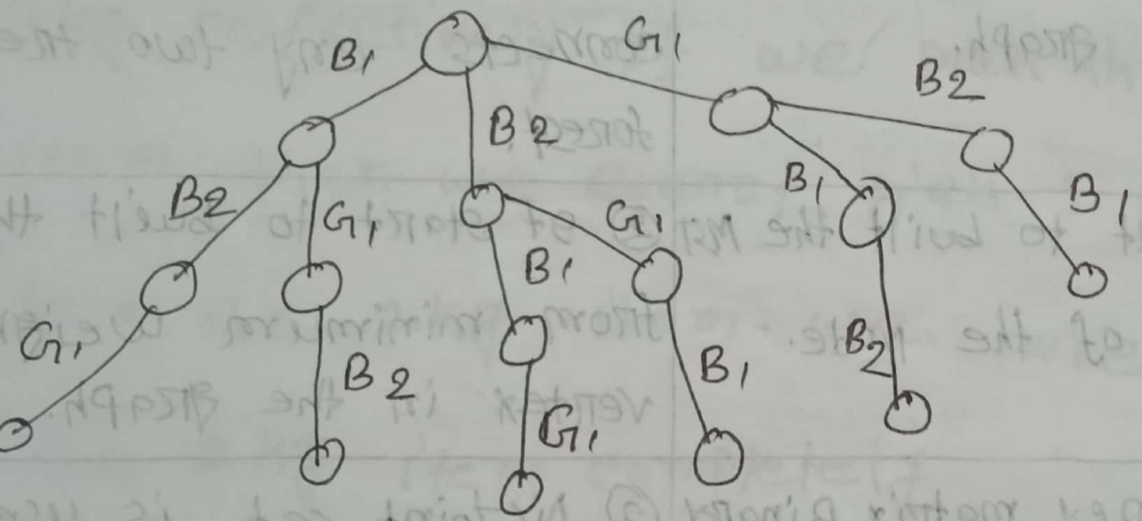
Abdul Bari
youtube

all possible value বের করা যায়।

State space tree

$B_1 \cdot B_2 G_1$

$n = 3.$



✓ B_1 B_2 G_1

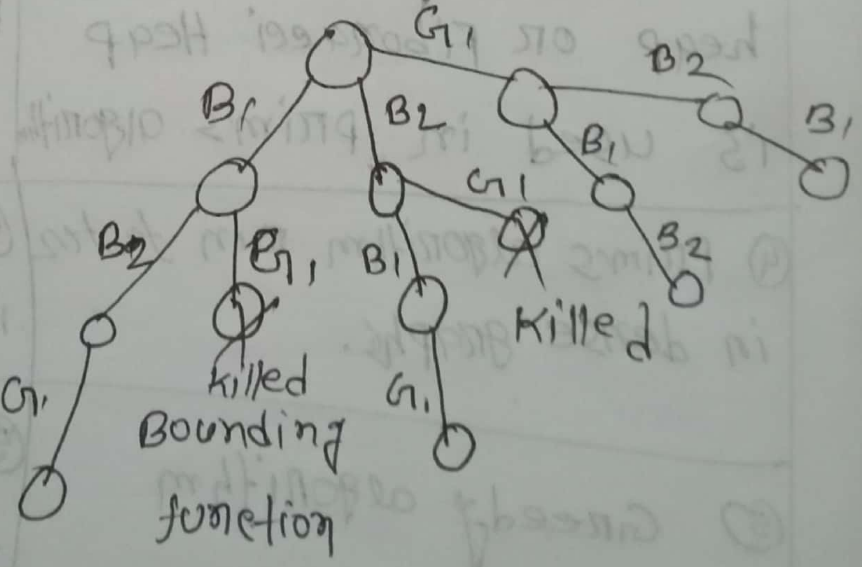
B_1 G_1 B_2

✓ B_2 B_1 G_1

B_2 G_1 B_1

✓ G_1 B_1 B_2

✓ G_1 B_2 B_1

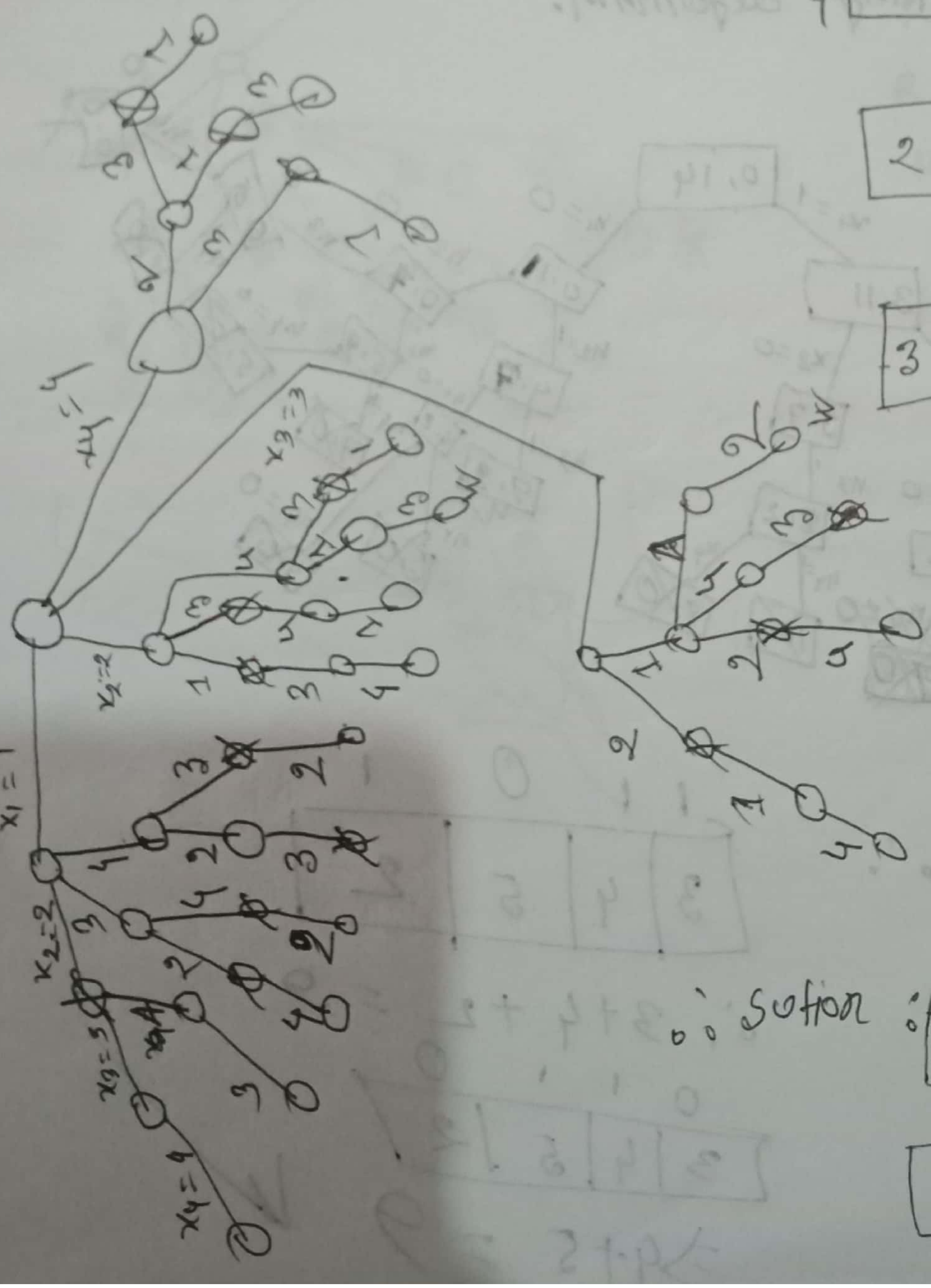


Backtracking algorithm (N-queens problem)

শর্ত :- queen → same row
same column
diagonal এ থাকতে পারবে না।

queen n = 4.

	1	2	3	4
1				
2				
3				
4				



2	4	1	3

3	1	4	2

∴ Soliton :

2	4	1	3

3	1	4	2

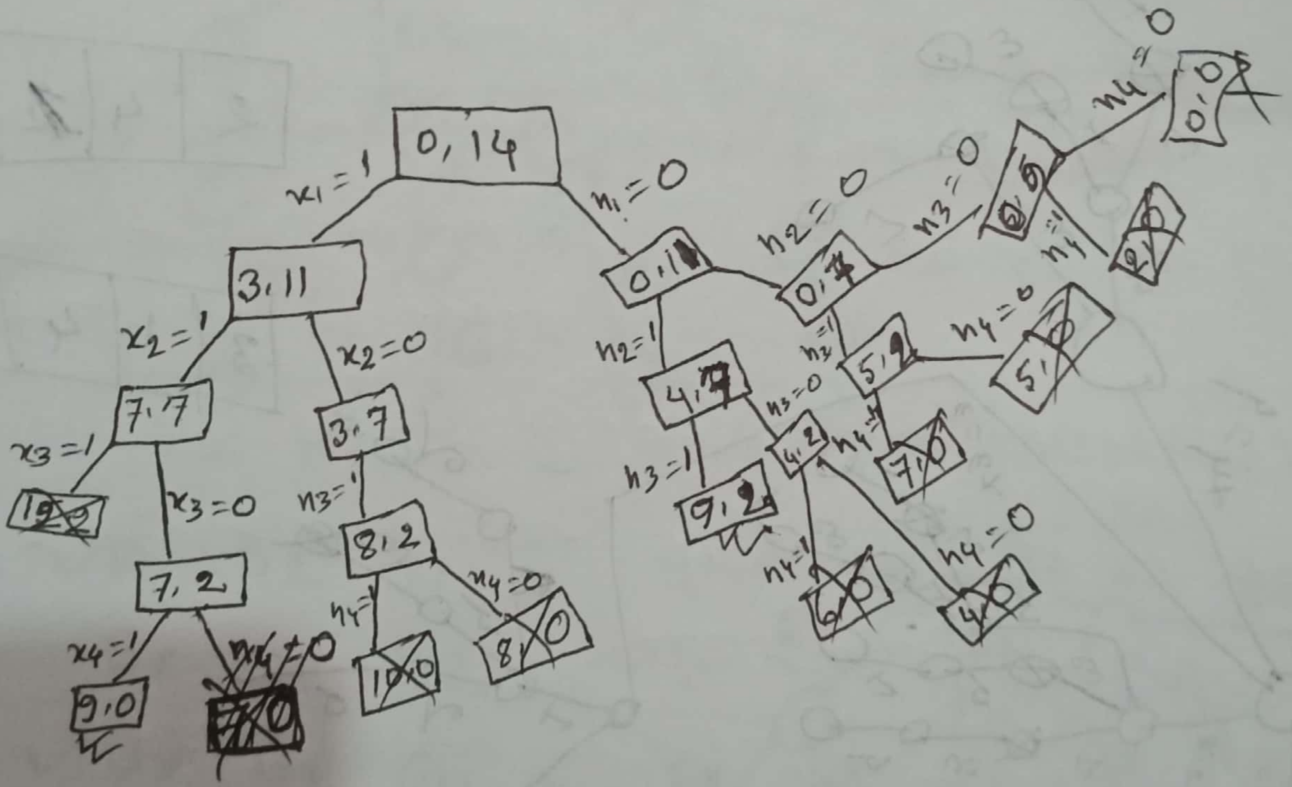
sum of subsets

~~W[1:6] = {5, 10, 12, 13, 15, 18}~~

~~n = 6, m = 30 total = 73~~

W = {3, 4, 5, 2}

find sum of subsets for m = 9 using backtraking algorithm.



∴

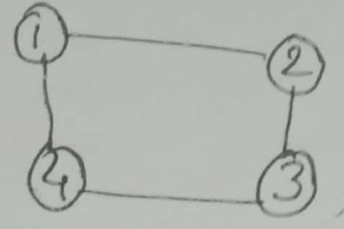
1	1	0	1
3	4	5	2

∴ 3 + 4 + 2 = 9

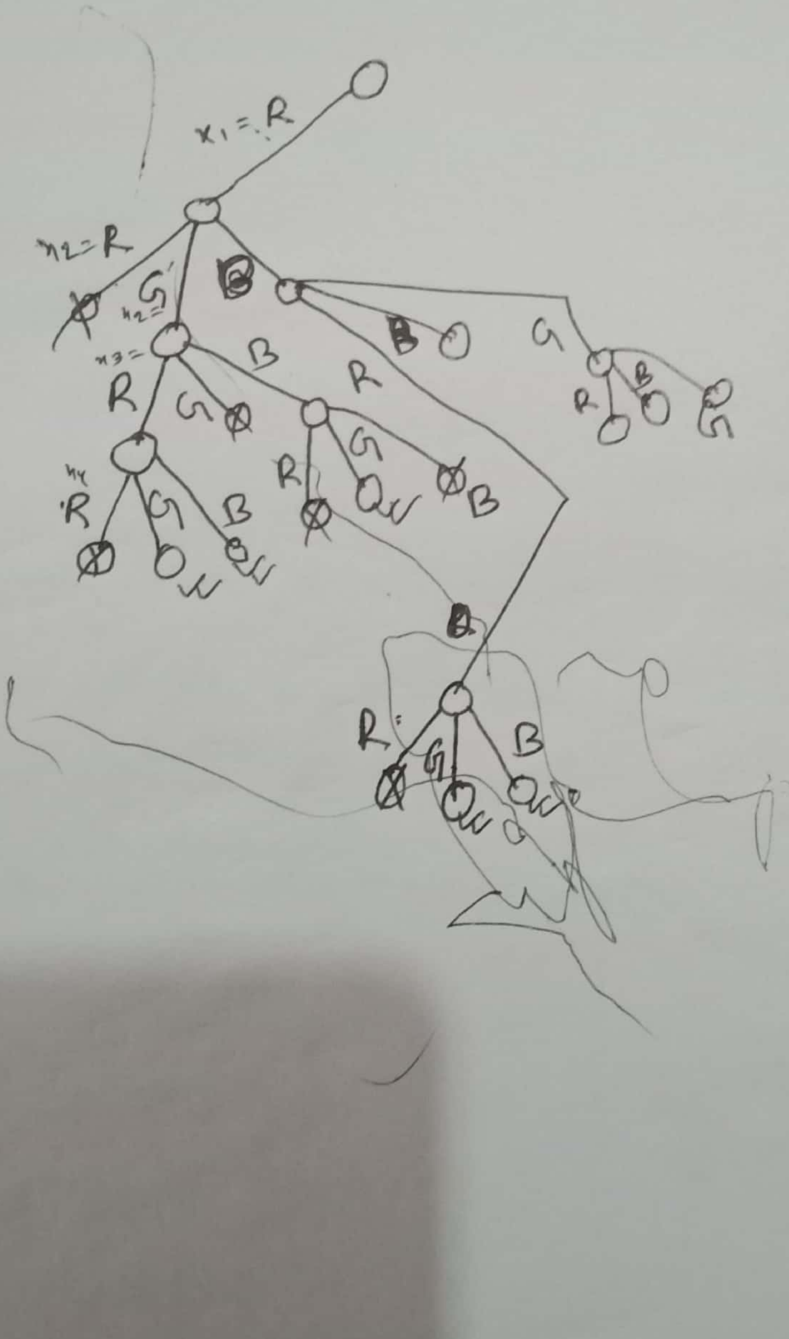
0	1	1	0
3	4	5	2

4 + 5 = 9

Graph colouring



m = 3, {R, G, B}



R G R G

R G R B

R G B G

R B R G

R B R B