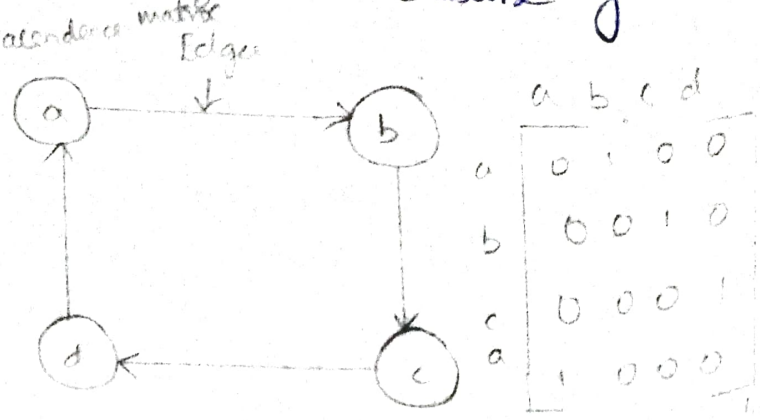
	0	1	2	3	4	5....	k-1...	k
0	1							
1	1	1						
2	1	2	1					
3	1	3	3	1				
4	1	4	6	4	1			
5 ⋮	1	5	10	10	5	1		
k								
n-1								
n								

Warshall's Algorithm:

Constructs transitive closure T as the last matrix in the sequence of n-by-n matrices $R^{(0)} \ldots R^{(k)} \ldots R^{(n)}$ where

It is used to compute transitive closure of a directed graph

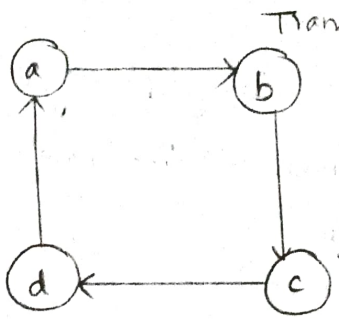


Adjacencies matrix :-

It is a representation of a graph using Matrix. If there exists the edge between vertices v_i and v_j directing from v_i to v_j, if then entry in adjacentie matrix in i^{th} row, j^{th} column, is 1.

Transitive closure :-

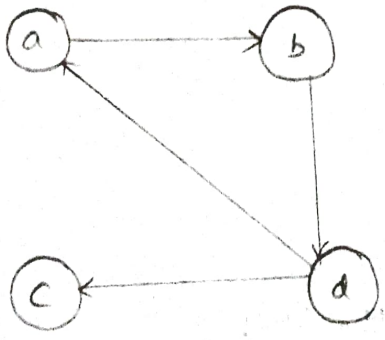
It is basically a boolean matrix in which the existence of directed path of ~~orbitorial~~ orbitary length between vertices is mentioned.



Transitive closure

	a	b	c	d
a	1	1	1	1
b	1	1	1	1
c	1	1	1	1
d	1	1	1	1

4 x 4



	a	b	c	d
a	0	1	0	0
b	0	0	0	1
c	0	0	0	0
d	1	0	1	0

4 x 4

	a	b	c	d
a	1	1	1	1
b	1	1	1	1
c	0	0	0	0
d	1	1	1	1

4 x 4

There is no outgoing edge

27/1/22

Minimum spanning Tree:

Prims Algorithm:

Spanning Tree:

Minimum Spanning Tree.

Prims algorithm used to find out minimum spanning Tree.

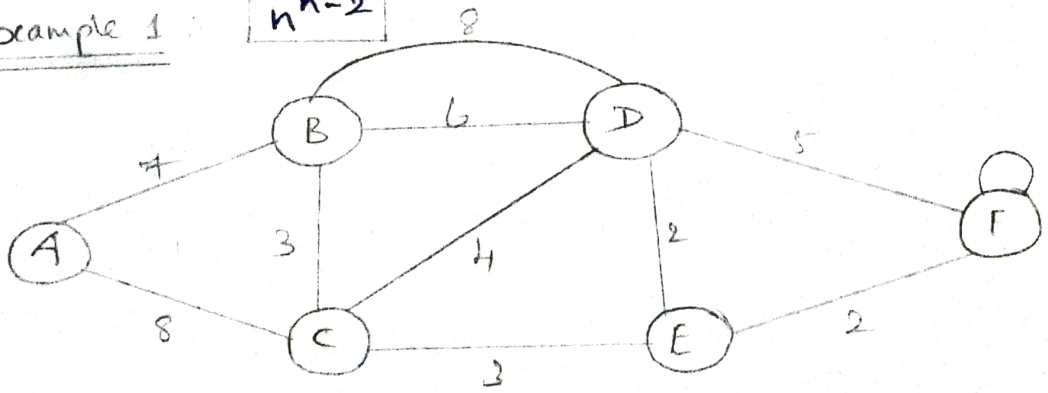
Minimum Spanning Tree:

It is spanning tree where sum of weight of the edges is as minimum as possible.

Spanning Tree:

A spanning Tree is a subgraph of an undirected connected graph, which includes all the vertices with a minimum possible no of edges.

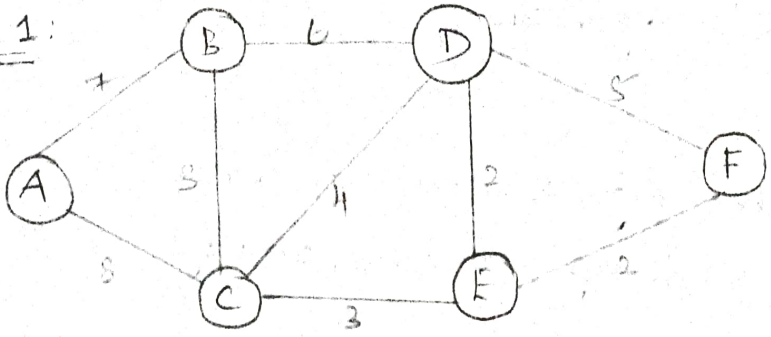
Example 1 n^{n-2}



Step 1: Remove all the loops

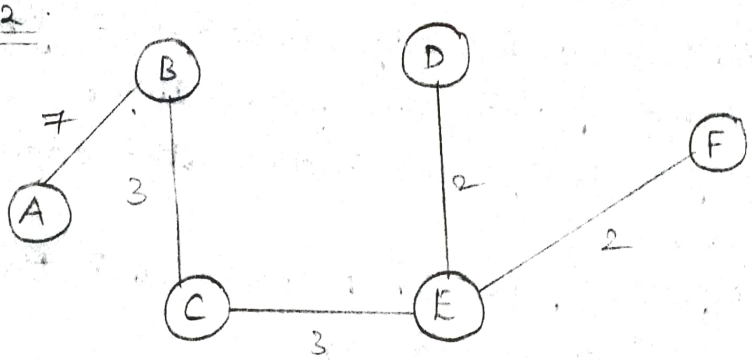
Step 2: Remove parallel edges.

Rule 1:



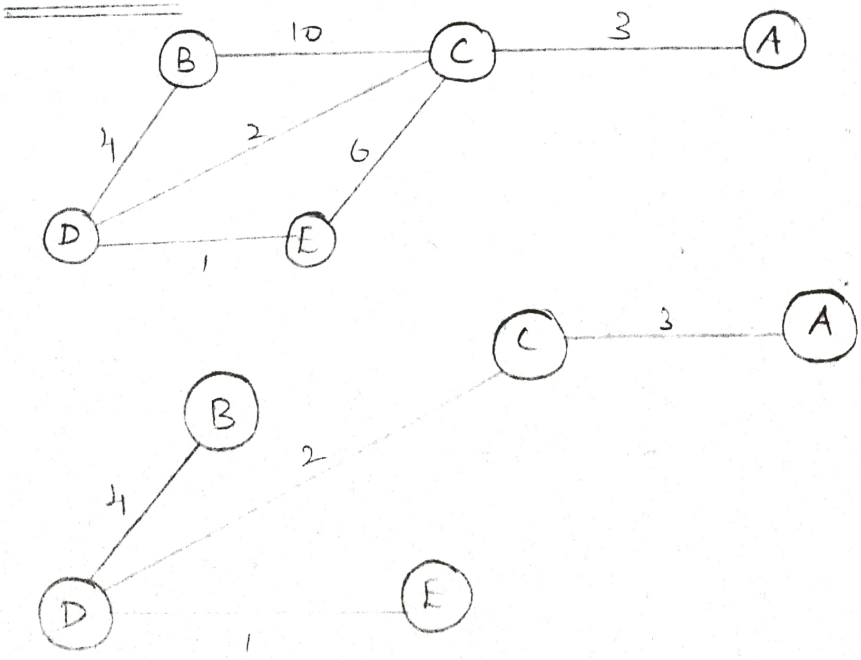
Remove all the Loops

Rule 2:



Remove parallel edges

Example 2:



Kruskal's Algorithm:

NOTE:

Prims algorithm based on vertex.

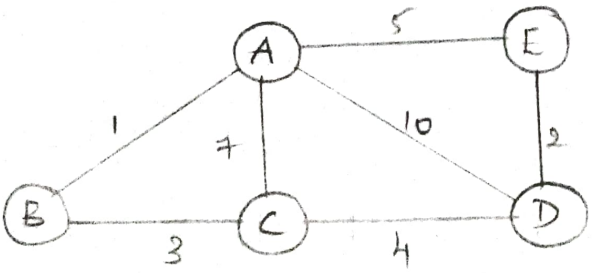
Kruskal's algorithm based on edges.

Rule 1: Sort all the edges in non decreasing order of their weight.

Rule 2: (i) Pick smallest edge. check if it is form a cycle with a spanning tree formed so far.

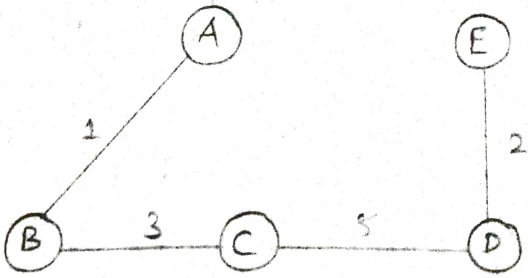
(ii) if cycle is not formed include the edges else discord it.

(iii) Repeat step no 2 (or) Rule no 2 until there are V-1 edges in the spanning Tree



Edge	BA	BC	AC	AD	CD	AE	AD
weight	1	3	7	10	4	5	2

Edge	BA	ED	BC	CD	AE	AC	AD
weight	1	2	3	4	5	7	10

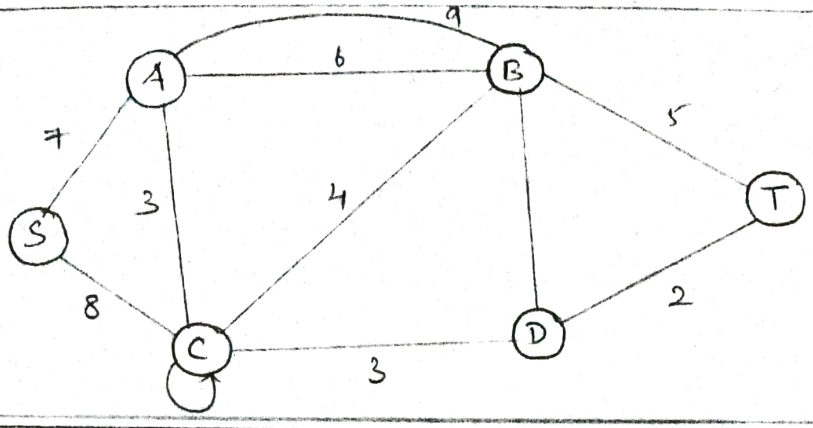


AB + BC + CD + DE

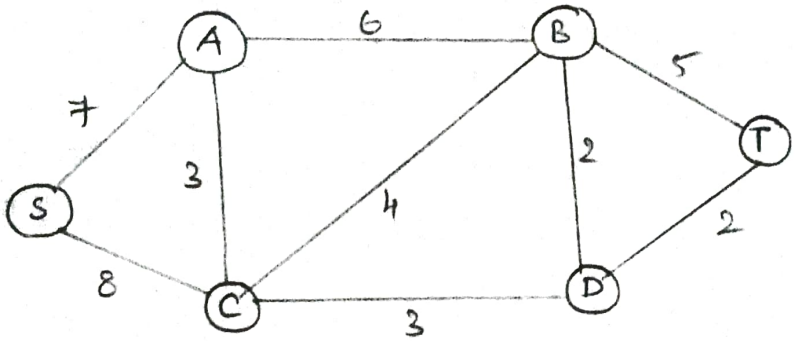
= 1 + 3 + 5 + 2

= 11

Runtime complexity $O(E \log E)$ or $O(V \log V)$

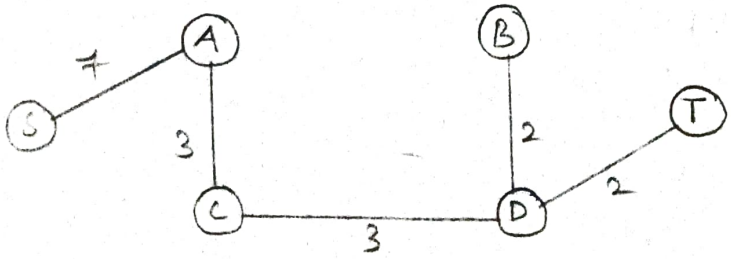


Find the minimum spanning Tree using kruskal's Algorithm:



Edge	SA	SC	AC	AB	CB	CD	DT	BT	BD
weight	7	8	3	6	4	3	2	5	2

Edge	BD	DT	CD	AC	CB	BT	AB	SA	SC
weight	2	2	3	3	4	5	6	7	8



SA + AC + CD + DB + DT = 7 + 3 + 3 + 2 + 2 = 17

Huffman Trees :-

Huffman Trees is binary tree. that minimize the weight path time from root to leaf of predefined weights The most important application of Huffman of tree is Huffman code.

Huffman code :-

It is an optimal prefix free variable length encoding scheme that assigns bit strings to symbols based on their frequencies in a given text.

This is accomplished by a greedy represent the alphabet symbols and whose edges are labelled with zero's and one's

2/3/22 ABBCD BCC DAA BBCCC BEAB

Soln :-

A B B C D B C C D A A B B C C C B E A B = 20

Fixed length

variable length

ASCII

$20 \times 8 = 160$

Encrypt, Encoder

01000001 = 8bit

COMPRESSION

Lossy — Lossless

Step 1 :-

char	count
A	4
B	7
C	3
D	2
E	4

Step 2:-

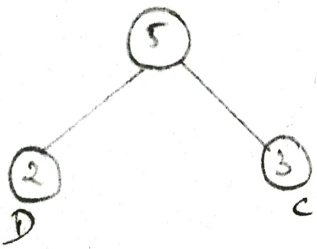
Arranging Ascending order

Char	Count
D	2
C	3
A	4
E	4
B	7

Step 3:-

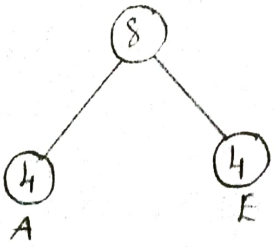
Heap

D	2
C	3
A	4
E	4
B	7



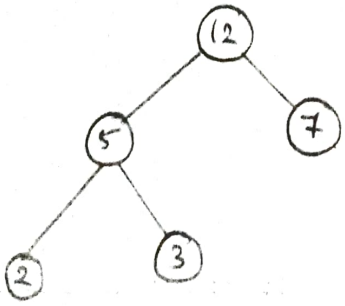
Step 4:-

Char	Count
A	4
E	4
Internal node	5
B	7



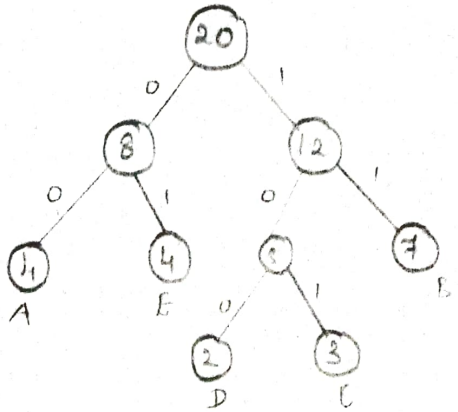
Step 5 :-

Char	Count
Internal node	5
B	7
Internal node	8



Step 6 :-

Char	Count
Int	8
Int	12



Step 7:-

Char	Count	Encode	
A	4	00	8
B	7	11	14
C	3	101	9
D	2	100	6
E	4	01	8
			45

15/22

Unit- IV

Iterative Improvement:-

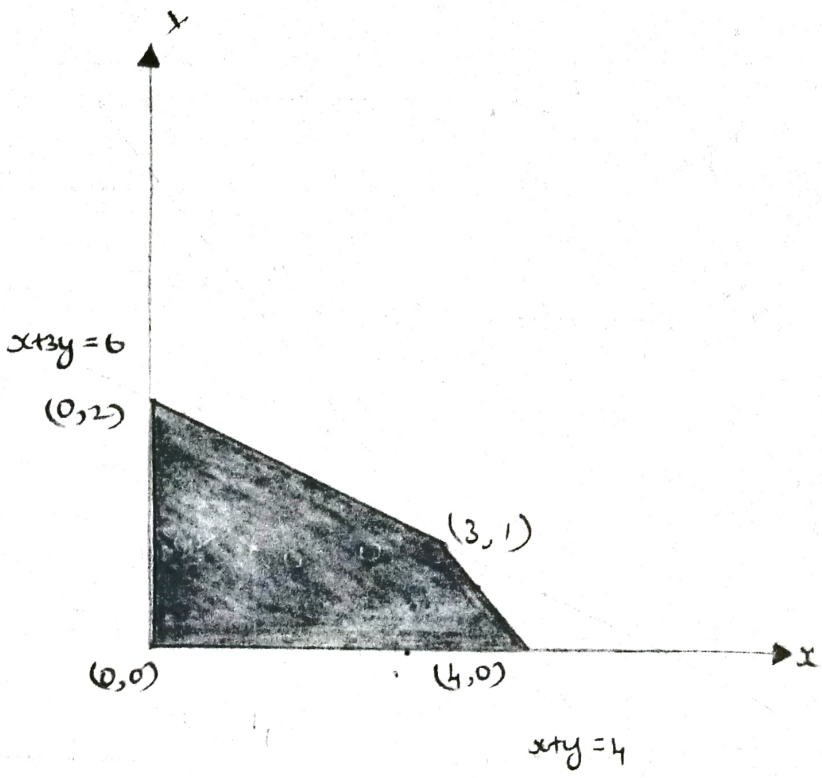
Algorithm design techniques for solving optimization Problems start with a feasible solution

Linear programming:

Linear programming problem is to optimize a linear function of several variables subjects to linear constraints.

Maximum (or) minimize

$x + y \leq 4 \longrightarrow$ In equality



#5/22

Simplex Method:-

* It is an method to solve Linear programming.
* we can used the simple Method
* It is used to solve LP problem having two or more variables.

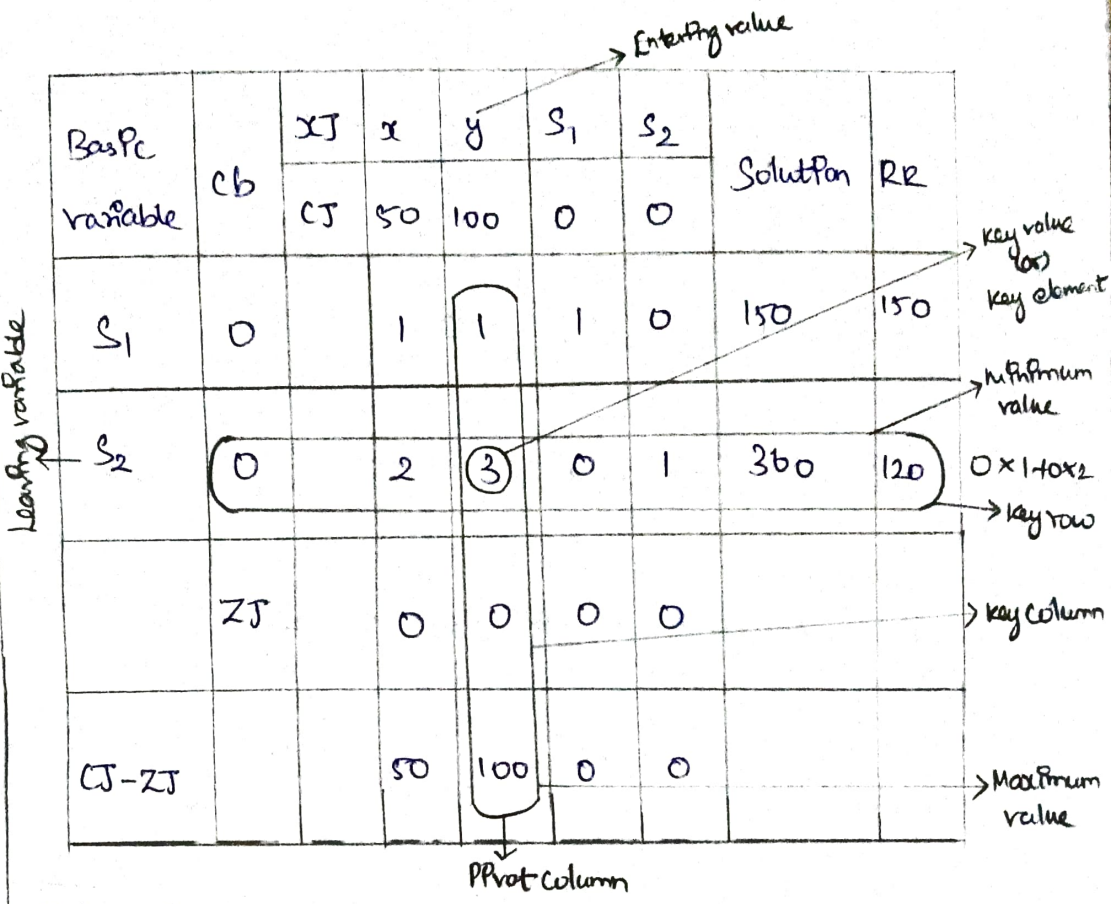
Ex: 1 :-

Maximum $z = 50x + 100y$

Constraints $x + y \leq 150$

$2x + 3y \leq 300$

$x, y \geq 0$



$x+y+S_1=150,\ 2x+3y+S_2=360,\ Z=50x+100y+0S_1+0S_2$

Step 1: Introduce the slack variable given constraints to make inequality to equality.

Step 2: Iteration Table

Basic operation = refers to S_1, S_2.

cb = Coefficient of basic variables.

CJ = Coefficient of given Maximum variable Z.

xJ = The variable of given Maximum.

$ZJ = \sum cb \times aij$

CJ-ZJ = Opportunities (or) Net Evolution Row

TO Find out the maximum opportunities cost

RR - Replacement Ratio → To Find out RR divided by corresponding Pivot column.

non key column →

Basic variable	Cb	x_1	x	y	S_1	S_2	Solution	RR
		CJ						
S_1	0		1/3	0	1	-1/3	30	
y	100		2/3	1	0	1/3	120	
	ZJ		200/3	100	0	100/3		
CJ-ZJ			-50/3	0	0	-100/3		

0 ≤ 0

Formula :

New number = old number − [key column number × corresponding key row number / key Element]

$= 1 - \left[\frac{1 \times 2}{3}\right]$

$= \frac{1}{3}$

$= 1 - \left[\frac{1 \times 3}{3}\right]$

$= 0$

$= 1 - \left[\frac{0 \times 1}{3}\right]$

$= 1$

$= 0 - \left[\frac{1 \times 1}{3}\right]$

$= \frac{-1}{3}$

$= 150 - \left[\frac{1 \times 360}{3}\right] = 150 - \left[\frac{360}{3}\right]$

$= 30.$

$y = 120$

$x = 0$

$s_1 = 30$

$Z = 50x + 100y$

$= 50 \times 0 + 100 \times 120$

$= 12000.$

$Z = 50x + 100y = 12000.$

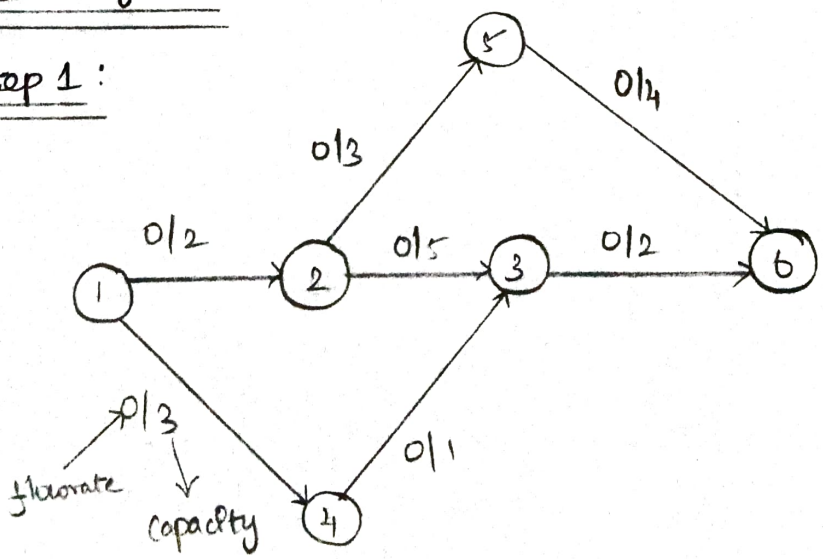
H/W
7/5/22

Solve the linear programming equations using simplex method

Maximum $z = 3x + 5y$ subject to $x + y \leq 4$ and $x + 3y \leq 6$ where $x, y \geq 0$.

Maximum flow

Step 1 :



Augumented Method (or) ford flukerson Method

10/5/22

Source node is called 1.

Destination node is called 6.

(or)

Sink node

Step2:

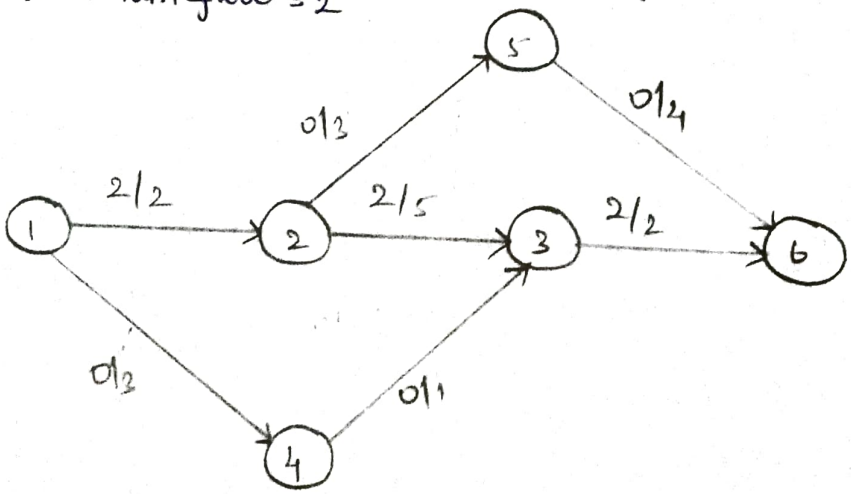
$C_{ij} - f_{ij}$

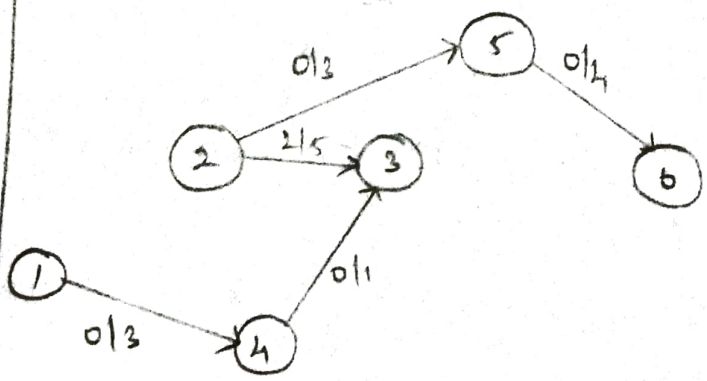
$2-0=2$

$5-0=5$

$2-0=2$

Minimum flow = 2





$3-0=3$

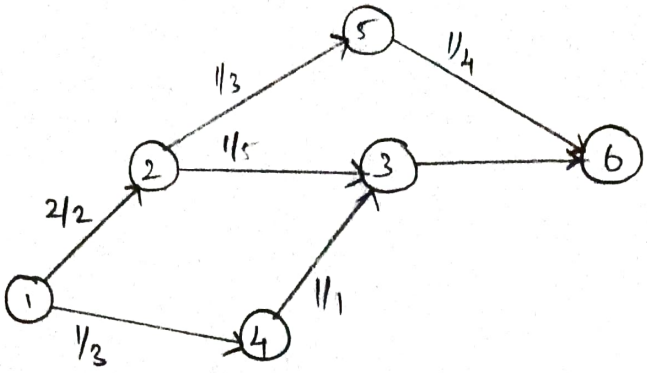
$1-0=1$

$3-0=3$

$3-0=3$

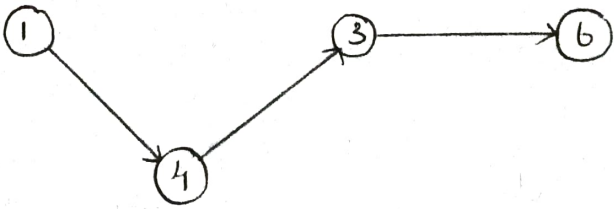
$4-0=4$

Minimum value = 1



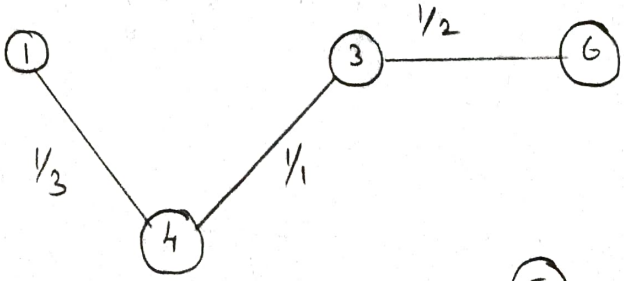
Maximum flow is

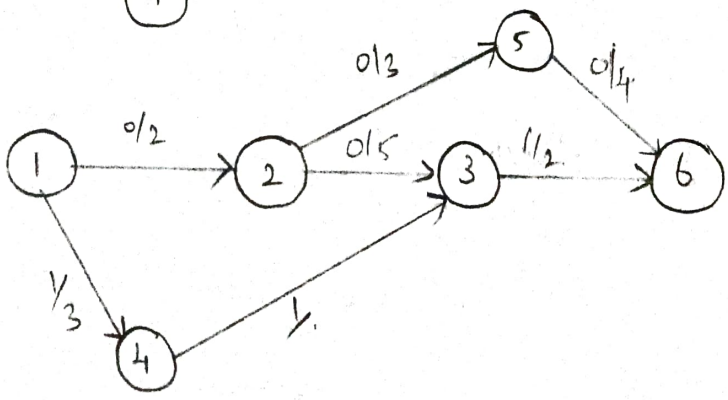
For backward edge selection = c[i,j] > f[i,j] > 0



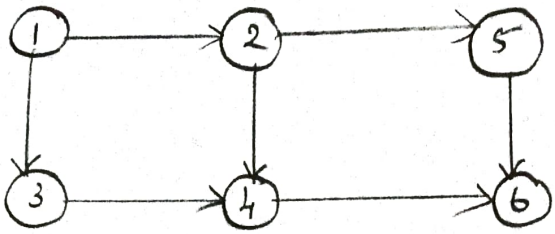
2 - 0 = 2
3 - 0 = 3
1 - 0 = 1
2 - 0 = 2

Minimum flow = 1





H/W



A set of edge whose removal divides network into two halfes X and Y.

Where S belongs to X

Capacity of cut = Max flow

Maximum Matching in bipartite (bigraph) :.

It is graph whose vertices can be divided into two disjoined and independent sets. and that is every edge connect vertex into 1 in.

17/5/22

Unit - V

Coping with limitation of algorithm Power

Limitation :.

* Same problem have algorithm.
* Algorithm which are solved by the lower ~~bond~~ bound.

Lower bound :→ Destination of number of Task

Lower bound

- → Trivial Lower bound
- → Information Theoretical arguments eg: Decision Tree
- → Adversary Arguments eg: when playing a guessing game to determine a number between 1 and n.
- → Problem reduction.

* Deterministic

* Non-Deterministic

P:-

Class of problem that can be Solved in polynomial.

NP:-

Class of problems that can be solved in non deterministic Polynomial.

NP complete and NP Hard:-

NP complete:-

Let c is a decision tree problem.

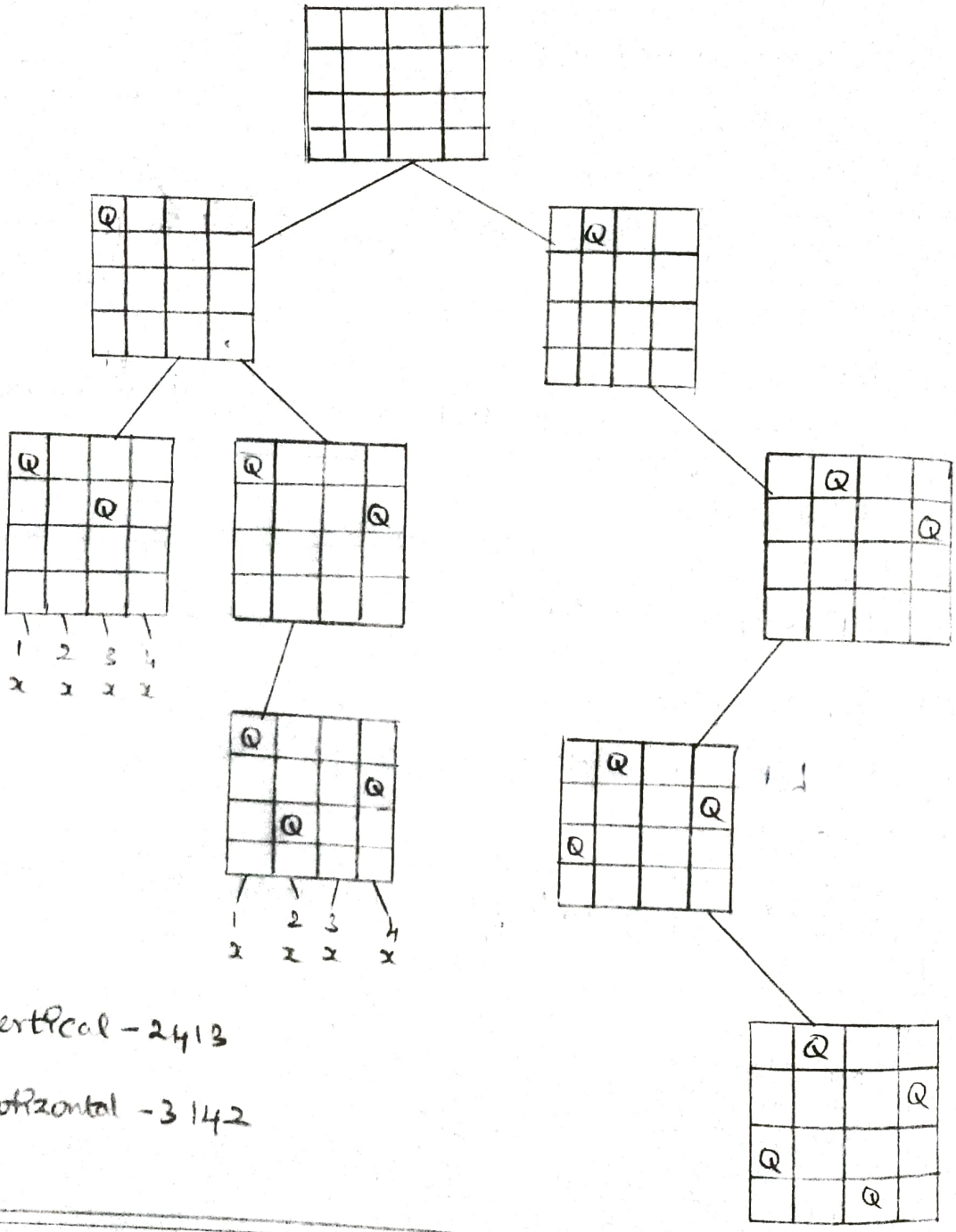
(i) c is in NP.

(ii) Every problem in np is Reducable to c at polynomial time.

NP Hard:-

(i) A problem H is np Hard

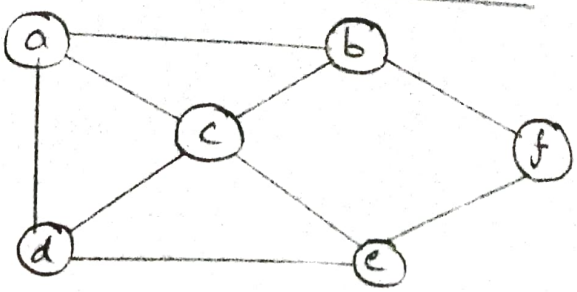
if and only if there is an np complete problems L Reducable to H.



Vertical - 2413

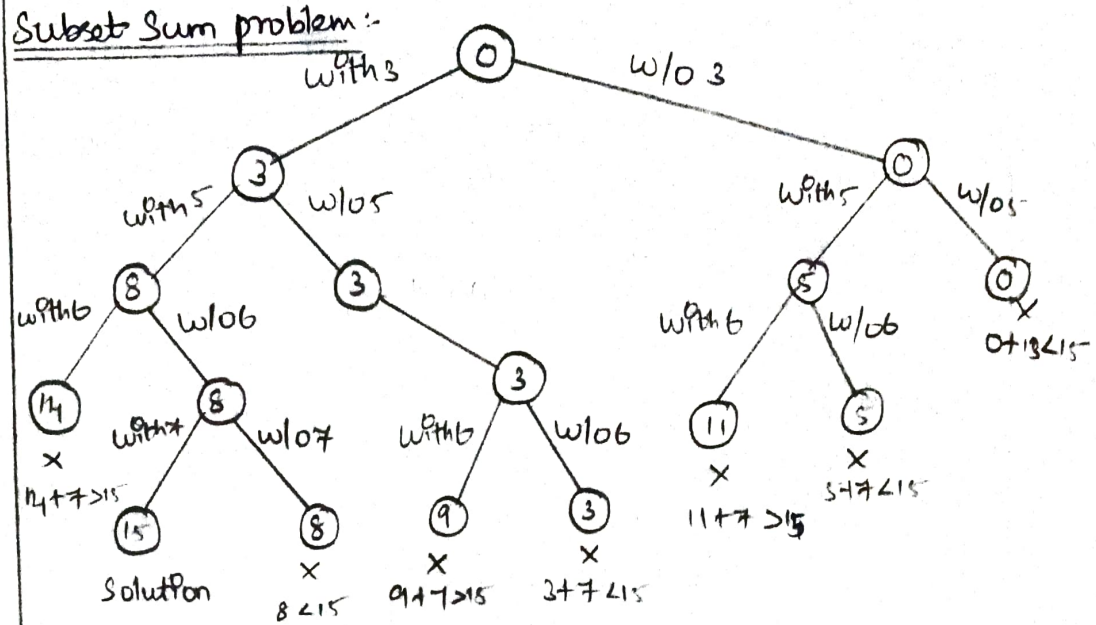
Horizontal - 3142

Hamiltonian Circuit Problem :-



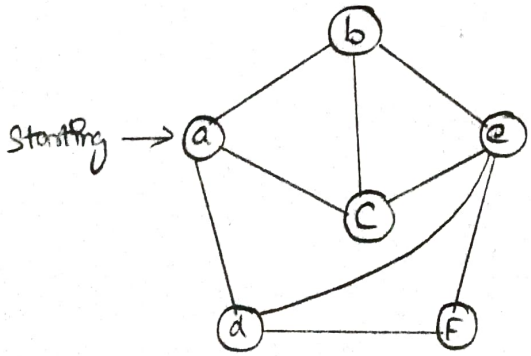
Starting to a and formed cycles is known Hamiltonian.

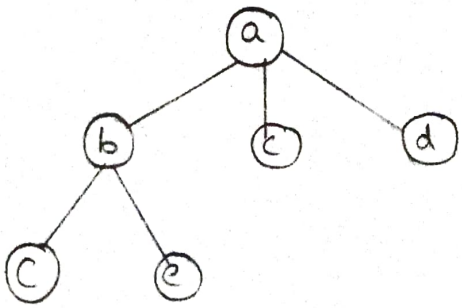
Subset Sum problem :-

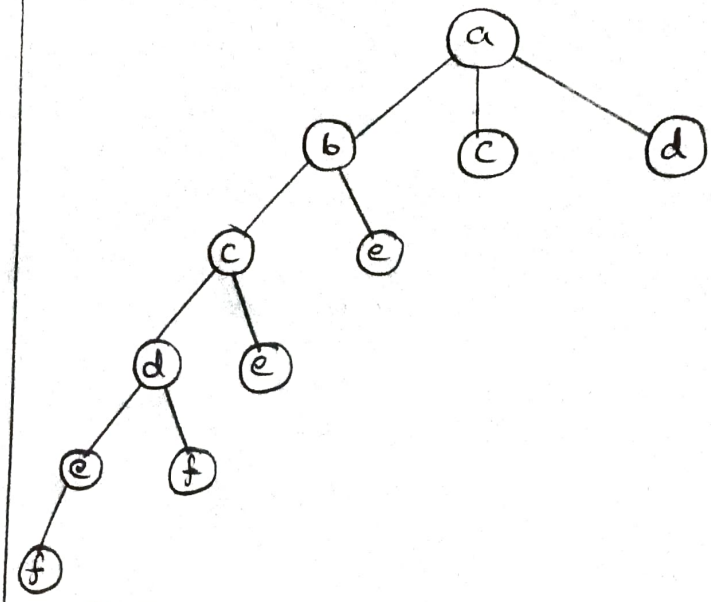


Vertex 2(or) graph cycle vertex :-

It is a cycle that visits each vertex exactly ones. a graph that contains hamiltonian cycle is known as hamiltonian graph.



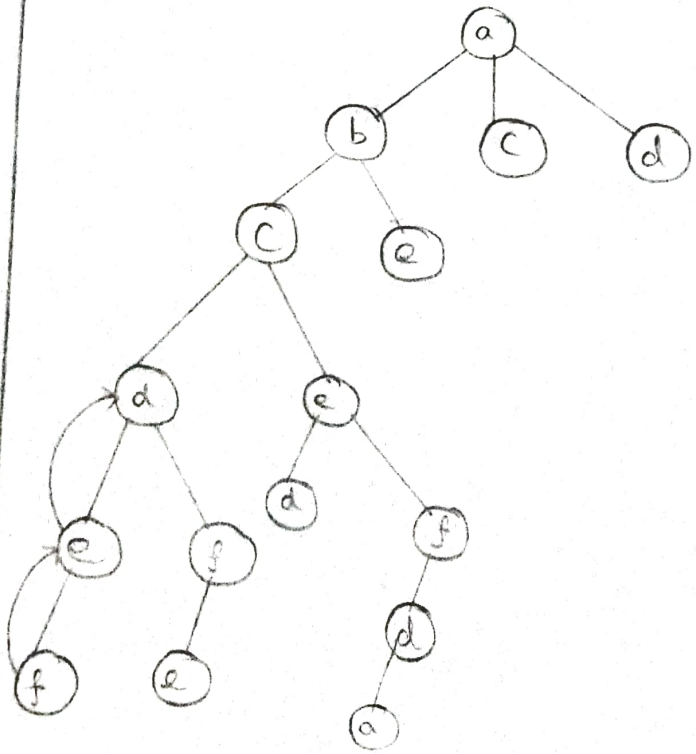




A	B	C	E	F	D	A

A	C	B	E	F	D	A

A	D	F	E	C	B	A



Subset Sum problem :

(small ↓ s, capital ↓ S)

It is to find subset s of given $S = \{s_1, s_2 \ldots s_n\}$ where elements of S (or) n positive integers, in such a manner that (s belongs to S $s \in S$) and sum of the subset s = Sum positive integers X

Example :

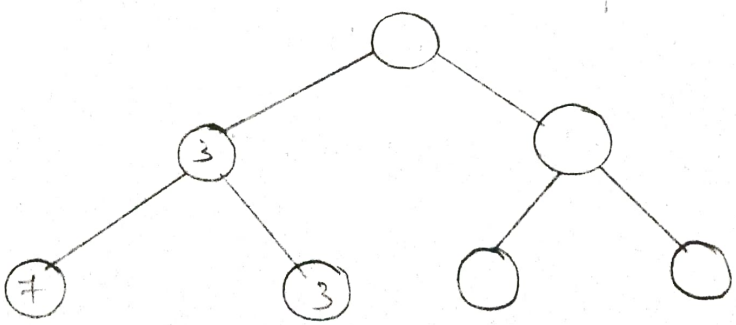
Given a set $S = \{3, 4, 5, 6\}$ and $X = 9$. obtain the subset sum using backtracking approach.

$S_1 = \{3\}$, $S_2 = \{4,5\}$

ϕ, 3, 4, 5, 6, (3,4) (3,5) (3,6) (4,5) (4,6) (5,6)

(3,4,5), (3,4,6), (4,5,6) (5,6,3) (3,4,5,6)

Tree → state space Tree



2/5/22

Branch and Bound :

It is one of the techniques used for problem solving. It is similar to back tracking since it uses state space tree. It is used for solving optimization problems, minimization and maximization problems

Example 1 :

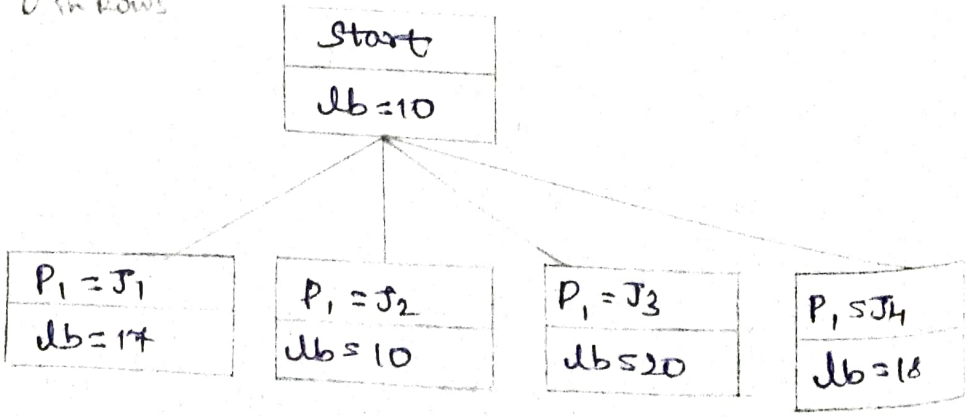
Assignment problem :

$$C = \begin{array}{c} \begin{matrix} J_1 & J_2 & J_3 & J_4 \end{matrix} \\ \begin{bmatrix} 9 & 2 & 7 & 8 \\ 6 & 4 & 3 & 7 \\ 5 & 8 & 1 & 8 \\ 7 & 6 & 9 & 4 \end{bmatrix} \end{array} \begin{matrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{matrix}$$

Try out all posible Job assignment with person minimum cost using branch and bound

lb (lower bound) $= 2 + 3 + 1 + 4 = 10$

↓ Taking min value in Rows



J_1	J_2	J_3	J_4	
(9)	2	7	8	P_1
6	4	3	7	P_2
5	8	1	8	P_3
7	6	9	4	P_4

$P_1 \rightarrow J_1$

$lb = 9 + 3 + 1 + 4 = 17$

J_1	J_2	J_3	J_4	
9	(2)	7	8	P_1
6	4	3	7	P_2
5	8	1	8	P_3
7	6	9	4	P_4

$lb = 2 + 3 + 1 + 4 = 10$

$P_1 \rightarrow J_2$

J_1	J_2	J_3	J_4	
9	2	(7)	8	P_1
6	4	3	7	P_2
5	8	1	8	P_3
7	6	9	4	P_4

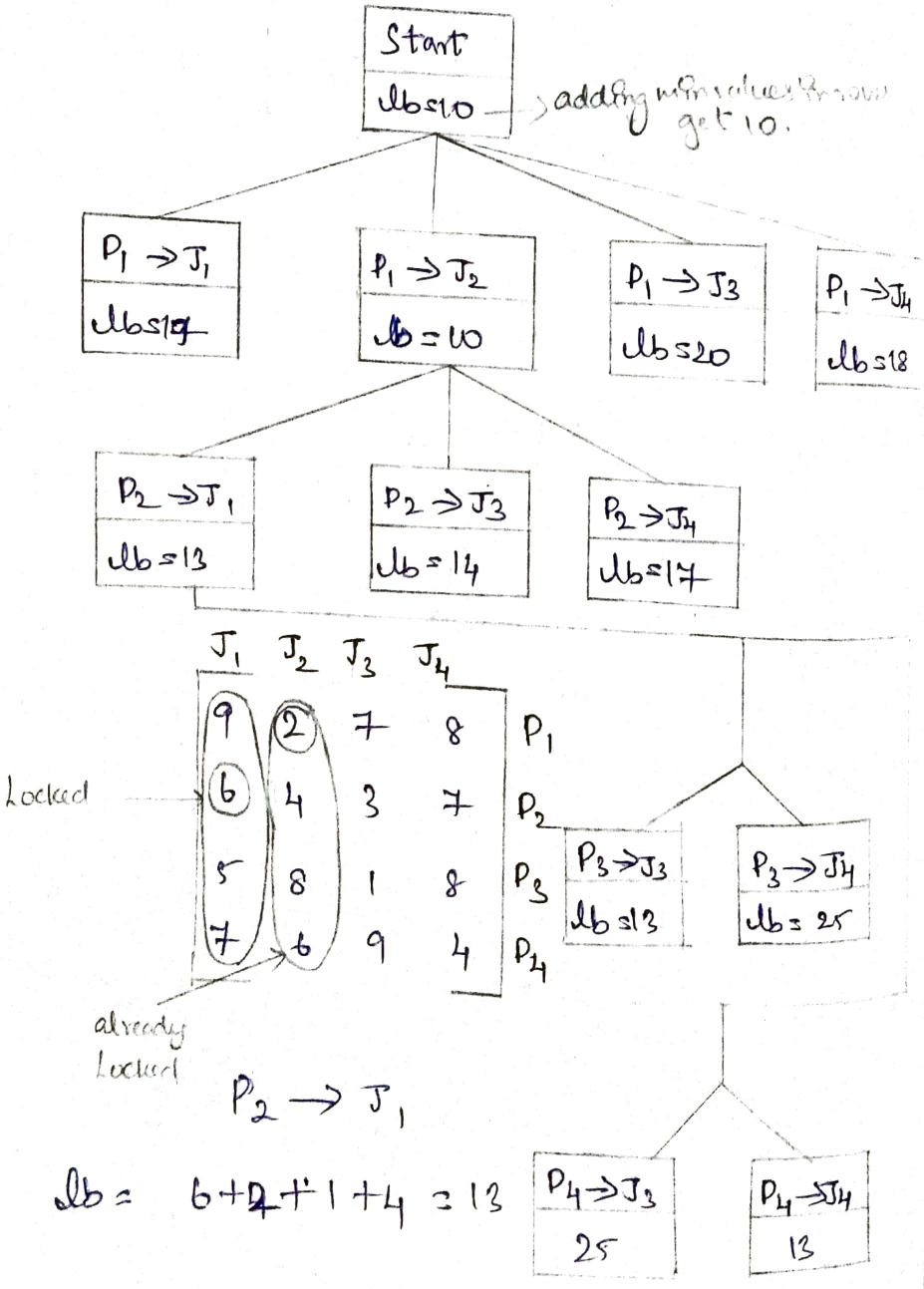
$P_1 \rightarrow J_3$

$lb = 7 + 4 + 5 + 4 = 20$

J_1	J_2	J_3	J_4	
9	2	7	8	P_1
6	4	3	7	P_2
5	8	1	8	P_3
7	6	9	4	P_4

$P_1 \rightarrow J_4$

$lb = 8 + 3 + 1 + 6 = 18$



$P_2 \rightarrow J_1$

$lb = 6 + 2 + 1 + 4 = 13$