Unit 6

Graph Theory

Graph Theory

* Graph :

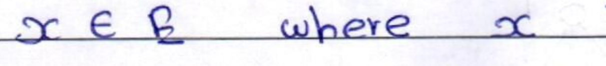
A graph $G = \langle V, E, \phi \rangle$ consist of nonempty set V called set of nodes. [points, vertices] of the graph. E is said to be edges of graph and ϕ is mapping from set of edges E to a set of ordered or unordered pairs of elements of V.

Set of points vertices (nodes) are represented by v_1, v_2 or 1, 2 & shown by dots or circles.

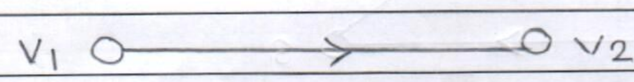
Every arc or edge strat at one point and end at another point.

Both the set V and E are finite.
If $x \in E$ where x is any node
$\langle u, v \rangle$ nodes

If x is edge the u and v are called adjacent nodes.

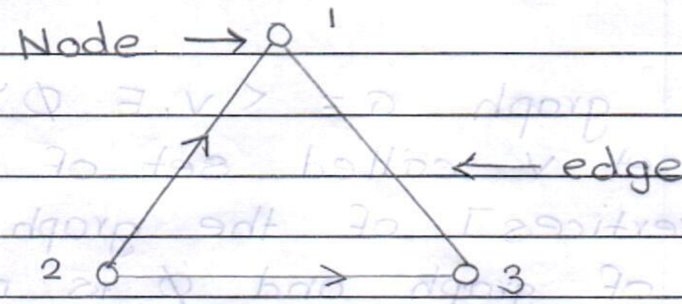


(a)

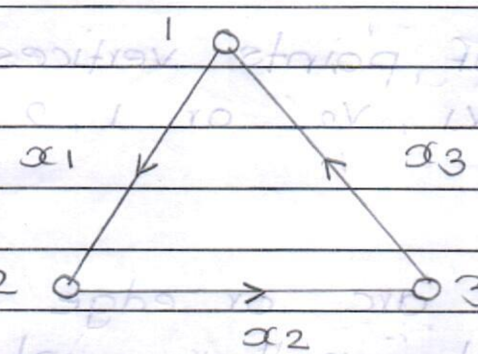


(b)

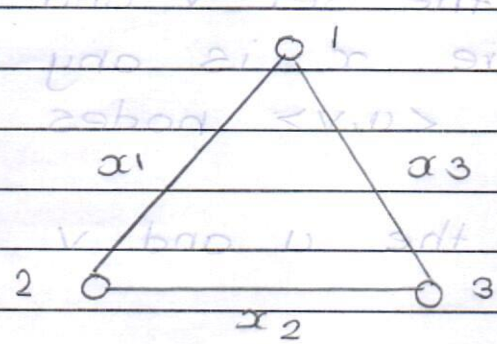
(c)



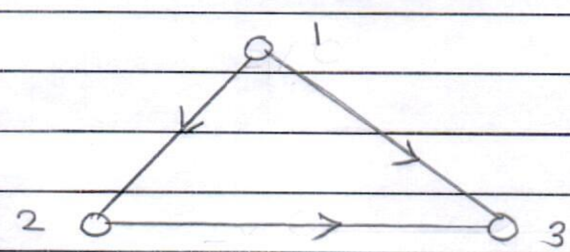
(d)



(e)



(f)



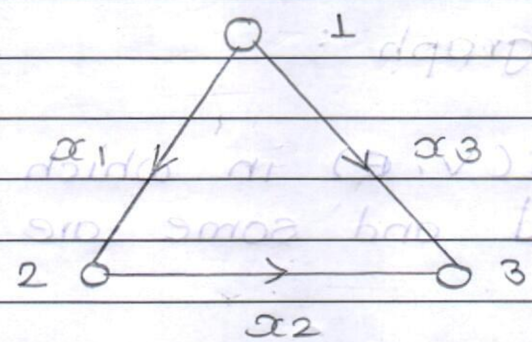
(g)

Rajdhani
DATE / /

A graph G < V, E > is with ordered pair V x V called Directed ~~graph~~ edge of G. While an edge which is associated with an unordered pair of nodes called undirected edge.

* Digraph / Directed graph :

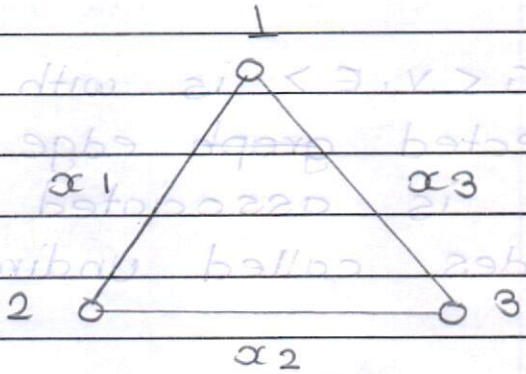
A graph in which every edge is directed



x_1 having ordered pair < 1, 2 >
x_2 having ordered pair < 2, 3 >
x_3 having ordered pair < 1, 3 >

* Undirected graph :

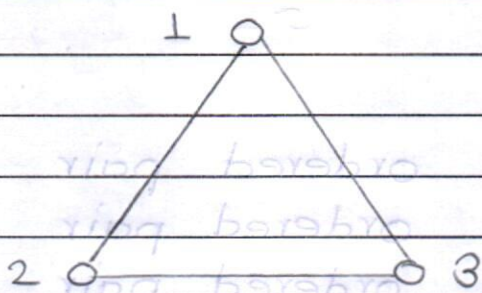
A graph G (V, E) in which every edge is undirected



x_1 unordered pairs $(1, 2)$
x_2 unordered pairs $(2, 3)$
x_3 unordered pairs $(3, 1)$

* Mixed graph :

A graph $G(V, E)$ in which some edges are directed and some are undirected.



If $G(V, E)$ be graph with $x \in E$ be directed edge and ordered pair $\langle u, v \rangle$

u = initiating or originating edge

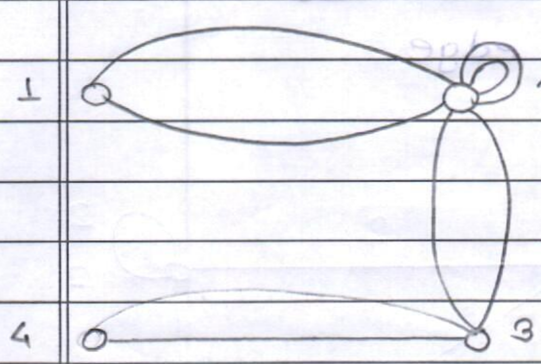
v = terminiting or ending edge.

Edge of graph which joins a node called loop [string]

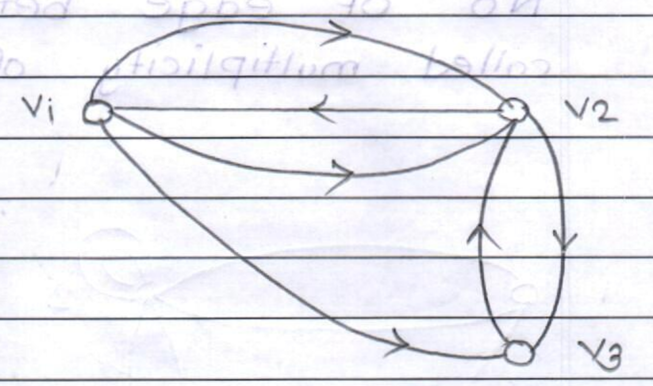
* Parallel edge :

The edge which are joined a nodes and more than one called parallel edge. The graph contain some parallel edges called multigraph.

It may be directed or undirected.



a



b

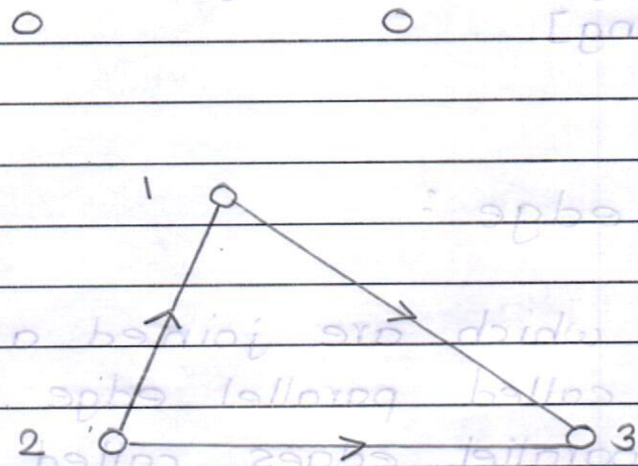
2 parallel edges

3 parallel edges

* simple graph :

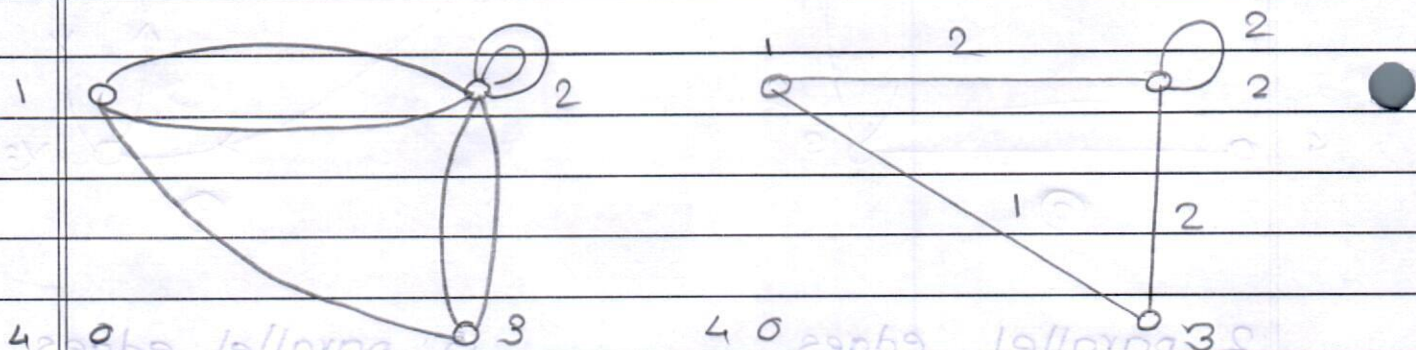
If graph contain nodes which not having more than one edge

Ex:



* Multiplicity of the edge :

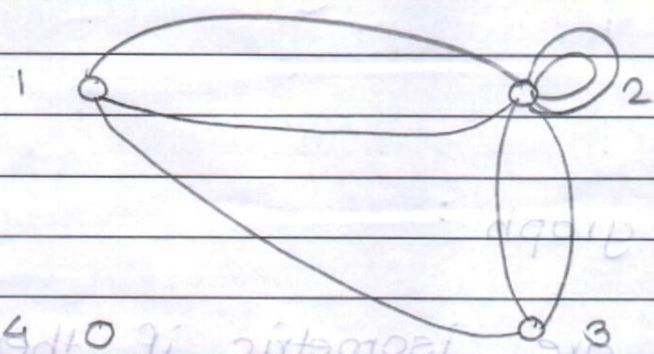
No of edge between the two nodes is called multiplicity of the edge.



The graph will be represented as

* Isolated node

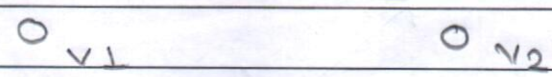
In a graph a node which is not adjacent to any other node called isolated node.



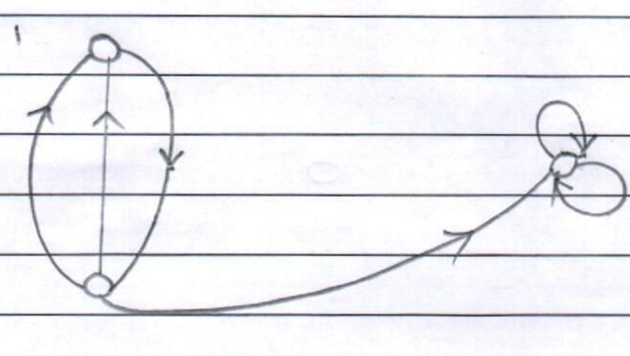
* Null graph :

The set of edges in null graph is empty.

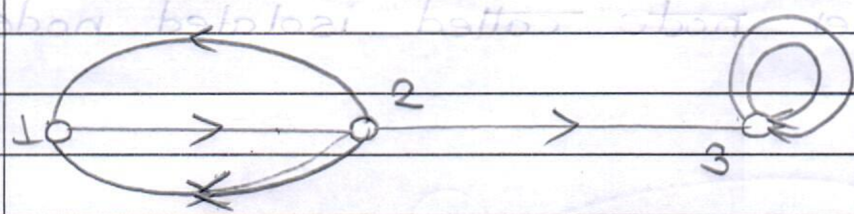
Ex :



Representation of graph :

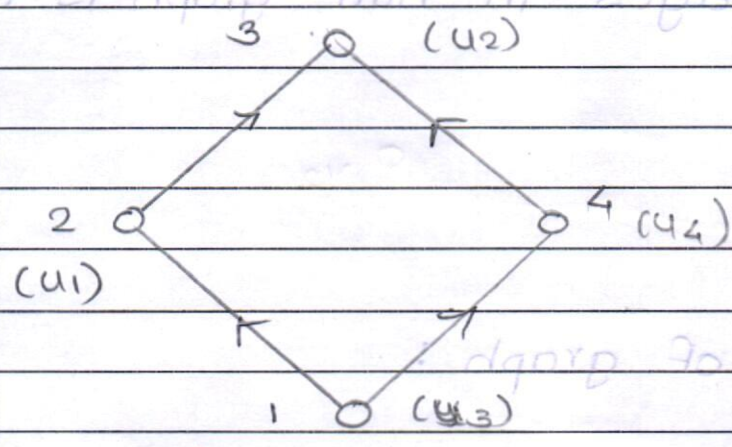


The original graph will represent as follow without change the nodes

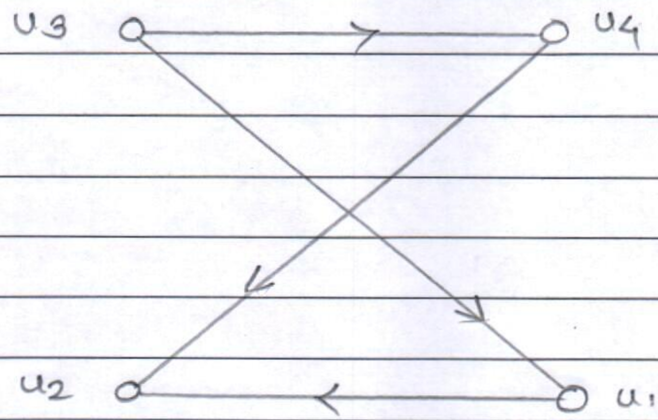


* Isometric graph :

Two graphs are isometric if there exists a one to one correspondence between the nodes of the two graphs which preserves adjacency of the nodes as well as directions of edges if any



Rajdhani
DATE / /



$1 \to u_3$

$2 \to u_1$

$3 \to u_4$

$4 \to u_2$

The ordered pairs of original pair of graph to new will be

$\langle 1, 2 \rangle$ will $\langle u_3, u_1 \rangle$

$\langle 1, 3 \rangle$ will $\langle u_3, u_4 \rangle$

$\langle 3, 4 \rangle$ will $\langle u_4, u_2 \rangle$

$\langle 2, 4 \rangle$ will $\langle u_1, u_2 \rangle$

Rajdhani
DATE / /

* Indegree, outdegree, total degree :

Indegree :

If r is the node then no of edges which having terminals node as r called indegr

Outdegree :

If r is the node then the edges which having r as initial node called outdegree

Totaldegree :

indegree + outdegree at node r

The total degree of isolated node is 0

* Converse of Graph / Reversal

A simple diagraph $G = \langle V, E \rangle$ is called as reflexive, transitive, antisymmetric and symmetric if the relation E is reflexive, transitive, antisymmetric and symmetric.

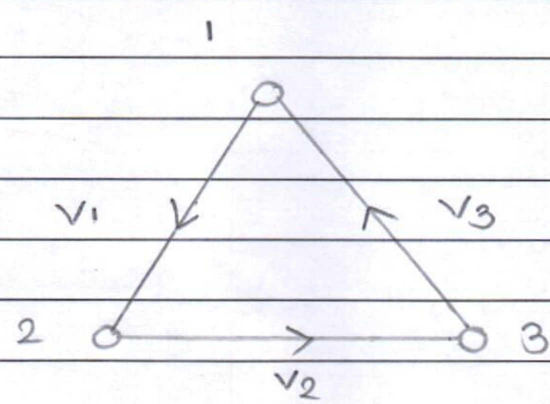
If $G = \langle V, E \rangle$ is graph then converse.
$G' = \langle V, \tilde{E} \rangle$

$\tilde{E}$ is converse of E

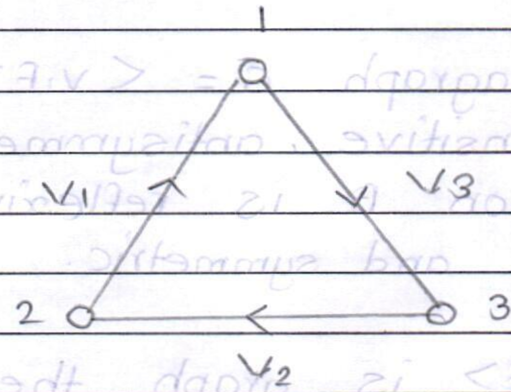
The diagram can obtain by simply remove the directions

$\tilde{G}$ called reversal as directional dual of diagraph G

Ex :



The reversal of given graph is as follow :


1
V1
V3
2
3
V2

Rajdhani
DATE / /

* Path Rechability and connectedness

If $G = \langle V, E \rangle$ be a simple diagraph with E are edges. The sequence of edges from initial node to next node is given as

$$(\langle v_1, v_2 \rangle \ \langle v_{12}, v_{13} \rangle \ \text{-------} \ \langle v_{ik-1}, v_{ik} \rangle)$$

Then we can write it as

$$(v_{i1}, v_{i2}, v_{i3}, \text{----------} \ v_{ik-1}, v_{ik})$$

Except first node v_{i1} and v_{ik} the every noc should be in sequence.

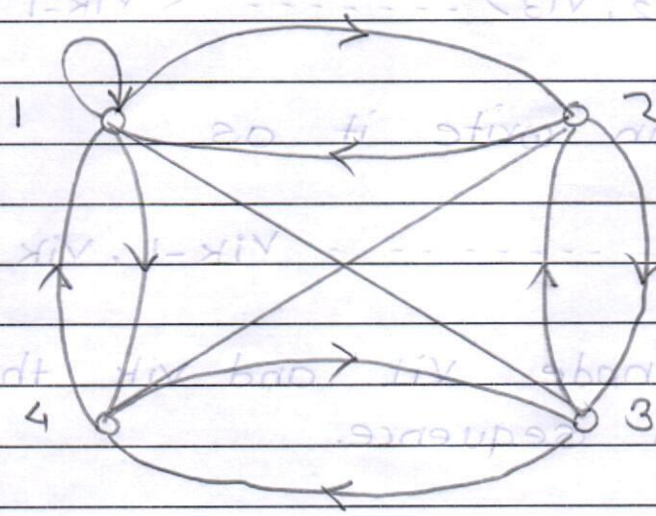
* Path of the graph:

Any sequence of edges of a diagraph such tha the terminal node of any edge is initial node of the other edge if any appearing then called as path.

Path is tran verse threw the nodes appearing in sequence originating in the initial node of first edge and ending in terminal node of last edge of sequence.

* Length of the path :-

The no of edges appearing in the sequence of the path called length of the path



some path originating from 1 and ends at 3

$P_1 = (\langle 1,2\rangle, \langle 2,3\rangle)$ length = 2

$P_2 = (\langle 1,4\rangle, \langle 4,3\rangle)$ length = 2

$P_3 = (\langle 1,1\rangle, \langle 1,4\rangle, \langle 4,1\rangle, \langle 1,2\rangle, \langle 2,3\rangle)$ length = 5

$P_4 = (\langle 1,2\rangle, \langle 2,4\rangle, \langle 4,3\rangle)$ length = 3

* Simple path :

A path in a diagraph in which edges are all distinct is called simple path (edge simple) P_1, P_2, P_4

A path in which all nodes throu-gh which it tranverse are distinct called an elementary path (node simple) P_3

* cycle :

The path which originates and ends in the same node is called cycle (circuit)

A cycle is simple if path is simple.

A cycle called elementry if it does not traverse through any node more than once

$C_1 = (\langle 1,1\rangle)$ elementry cycle

$C_2 = (\langle 1,2\rangle \langle 2,1\rangle)$

$C_3 = (\langle 1,2\rangle \langle 2,3\rangle \langle 3,1\rangle)$

$C_4 = (\langle 1,4\rangle \langle 4,3\rangle , \langle 3,1\rangle)$

$C_5 = (\langle 1,4\rangle , \langle 4,3\rangle , \langle 3,2\rangle , \langle 2,2\rangle)$

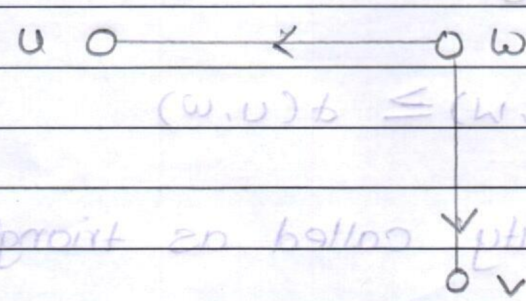
Rajdhani
DATE / /

A simple diagraph which doesn't have any cycle called cyclic, it does't contain any loops.

* Reachable [Accessible]

A node v in the simple diagraph is said to be reachable from node u of same diagraph if there exists path from u to v

reachable



Not reachable

Not necessary which path is used

Every node is reachable from itself.

If node u is reachable from v with minimum distance then that the distance called geodesic

denoted by $d\langle u,v\rangle$. $d\langle u,v\rangle = 0$

Reachability having binary relation property transitive

If node u is reachable from v and v from w then u is also reachable from w

It doesn't have reflexive, symmetric, and antisymmetric property.

If reachability from u to v then not necessary from u to v is reachable.

The distance between u to v can be denoted as,

$$d(u,v) \geq 0$$
$$d(u,u) = 0$$

$$d(u,v) + d(v,w) \geq d(u,w)$$

The last inequality called as triangle inequality

If u is not reachable from v then
$d(u,v) = \infty$

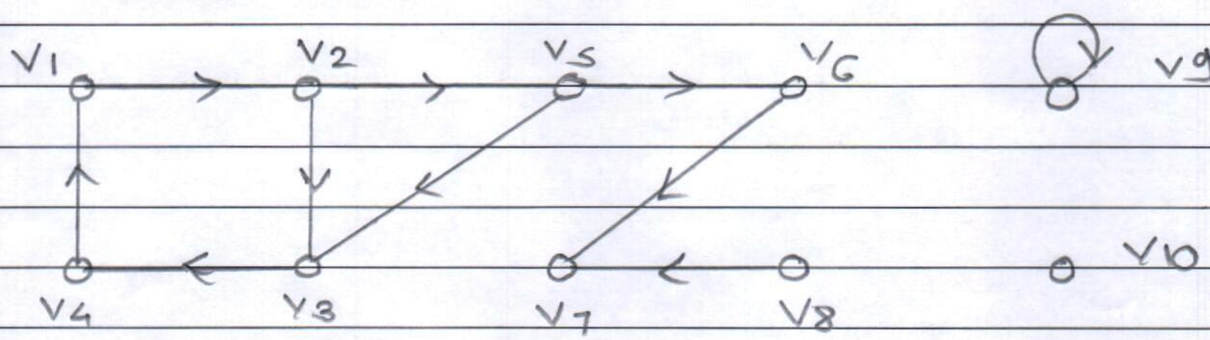
If u is reachable to v then $d(u,v)$ if v is reachable from u then $d(v,u)$

Not necessary to show

$$\boxed{d(u,v) = d(v,u)}$$

They are equals

Reachability



The reachability of any node is denoted by $R(v)$

Where, v = Node

$R(V1) = \{V1, V2, V3, V4, V5, V6, V7\}$

$= R(V2) = R(V3) = R(V4) = R(V5)$

$R(V6) = \{V6, V7\}$

$R(V7) = \{V7\}$

* Connected graph :

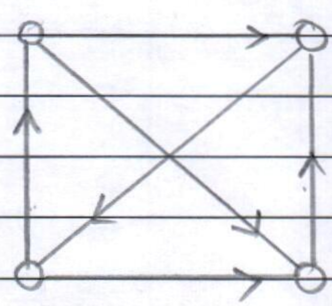
Any pairs of nodes of the graph two nodes are reachable from one another

* Weakly connected :

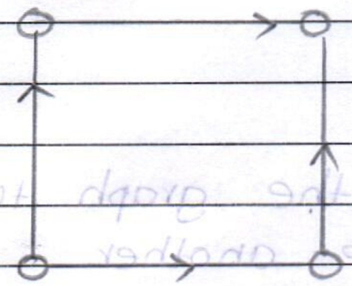
The node u is reachable from v but v is not reachable from u.

* strongly connected

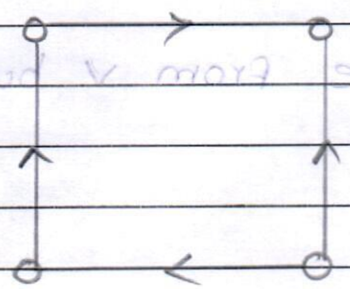
For any pair of nodes of the graphs both the nodes of the pair are reachable from one another then given graph is strongly connected.



strongly connected graph is also ,
① weakly connected
② unilaterally connected



Weakly Not unlateraly



unjlateraly with weakly

[At least one of the node reachable From another]

* Representation of simple Graph Allocation in operating system :

In multiground computer system multiple program in are used at some time like hard disk, disk drives, the processor, main memory, compilers.

These all processors are controlled by operating system and opearting system ensure that weather all process tanjent acquire resource or not.

If A and B are two program. A access resource r_1 and at same time B also need r_1 that time deadlocks is created. By using deadlock we can detect deadlock and recover it

To avoid deadlock waiting of process upto an available the resource

Let

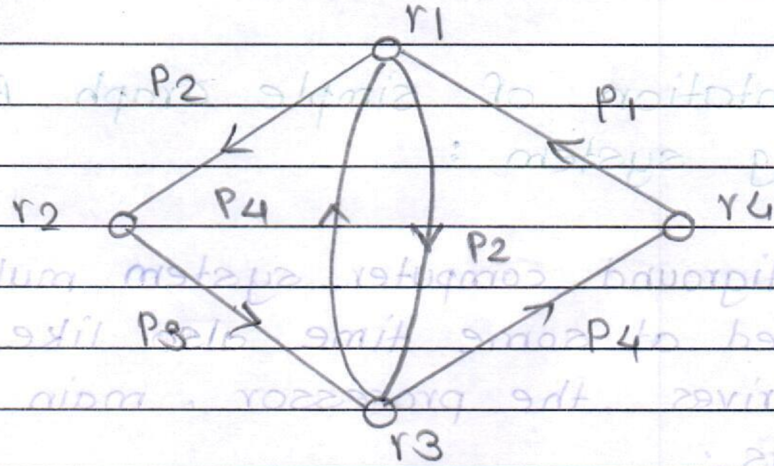
$P_t = \{P_1, P_2, ------ P_m\}$ all programs at time t

$R_t = \{r_1, r_2, r_3, ----- r_m\}$ resources

Eg. for eg.

$R_t = \{r_1, r_2, r_3, r_4\}$

$A_t = \{P_1, P_2, P_3, P_4\}$



P1 has resource r4 and require r1

P2 has resource r1 and require r2, r3

P3 has resource r2 and require r3

P4 has resource r3 and require r1, r4

Deadlock is occurs if the graph contain strongly connected components so avoid it

1. No cycle
2. wait

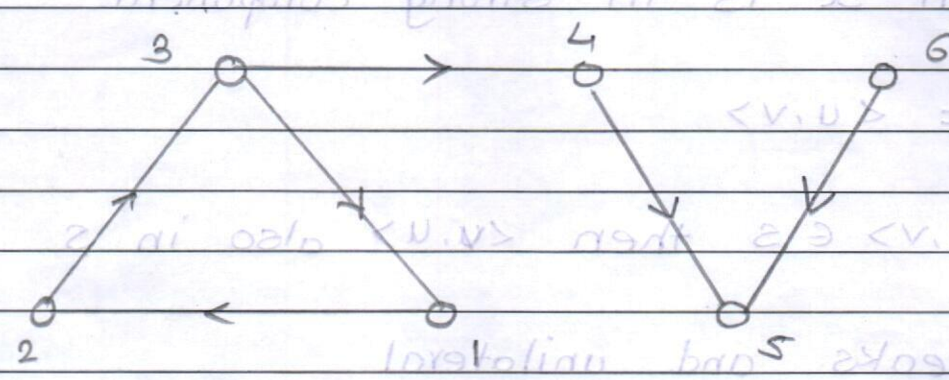
These some techniques to avoid deadlock.

Rajdhani
DATE / /

* Strong Components :

A maximal strongly connected subgraph called strong components

Maximum unilaterally connected or maximal weakly connected subgraph called unilateral or weak components respectively.



strong components ⇒ {1, 2, 3} {4}, {5} {6}

Unilatrol components ⇒ {1, 2, 3, 4, 5, 6} {6}

Weak components ⇒ {1, 2, 3, 4, 5, 6}

Theorem :

In a simple diagraph $G = \langle V, E \rangle$ every node of diagraph lies in exactly one strong components

Rajdhani
DATE / /

Proof :

$v \in V$ set of node which mutually reachable

Now Assume v having 2 strong components which are reachable from one node to another

Now note that $x \in E$ be any edge which may or may not contain any strong component s then x is in strong component.

$x = \langle u, v \rangle$

if $\langle u,v \rangle \in S$ then $\langle v, u \rangle$ also in S

for weaks and unilateral

Every node and edge of simple diagraph is contained in exactly one weak components

Every edge and node of simple diagraph lies in at least one uniletral component

* Matrix Representation of graphs

Alternative method to graph represention well known operations of matrix algebra can be used to calculate paths, cycle etc

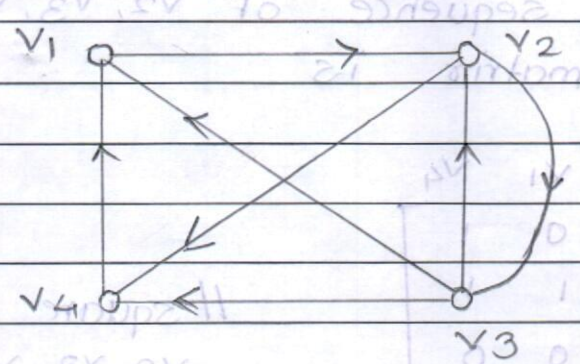
Adjacency Matrix :

Let $G = \langle V, E \rangle$ be a simple diagraph in which $V = \{v_1, v_2, \ldots v_n\}$ and notes to be ordered from v_1 to v_n

An nxn matrix A whose elements a_{ij} are given by,

$$a_{ij} = \begin{cases} 1 & \text{if } \langle v_i, v_j \rangle \in E \\ 0 & \text{otherwise} \end{cases}$$

is called the adjacency matrix of graph G.



Rajdhani
DATE / /

Element of adjacency matrix is either 0 or 1

Such matrix is called bit matrix or a Boolean matrix

An adjacency matrix completly defines a simple diagraph

The same graph G can be obtained by using different sequence of notes

i.e., $v_1, v_2, v_3, \ldots\ldots v_n$ or
$v_1, v_4, v_1 \ldots\ldots\ldots v_n$

Adjacency matrix

$$A1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

// The sequence of nodes for A1 is v_1, v_2, v_3, v_4

If we take the sequence of v_2, v_3, v_1, v_4 then adjacency matrix is

$$A2 = \begin{array}{c c} & \begin{array}{cccc} v_2 & v_3 & v_1 & v_4 \end{array} \\ \begin{array}{c} v_2 \\ v_3 \\ v_1 \\ v_4 \end{array} & \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

// square of v_2, v_3, v_1, v_4

Rajdhani
DATE / /

Reflexive ⇒ Diagonal elements are 1

symmetric ⇒ $a_{ij} = a_{ji}$

Antisymmetric ⇒ if $a_{ij}=1$ then $a_{ji}=0$

$a_{ij}=0$ then $a_{ji}=1$

- The matrix represention can also used in,
  1. Multigraphs
  2. Weighted graphs.

In the case of multigraph or weighted graph we can write,

$$a_{ij} = w_{ij}$$

Where W_{ij} = Multiplialy or weight of edge $<v_i, v_j$

- if $<v_i, v_j> \notin E$ then $w_{ij}=0$

* Null graph :

All elements of matrix are 0 i.e., adjacency matrix is null matrix

If there are loop at every node but no other edges in the graph. then the adjacency matrix is adjacency or unit matrix.

Rajdhani
DATE / /

If $G = \langle V, E \rangle$ is a simple diagraph whose adjacency matrix is A then $\tilde{G}$ is the converse of G.

$\tilde{G}$ is a transpose of G. such that A^T for A

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$A^T = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

$$A \cdot A^T = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 2 & 1 & 0 \\ 1 & 1 & 3 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

Rajdhani
DATE / /

$$A^T A = \begin{bmatrix} 2 & 1 & 0 & 1 \\ 1 & 2 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix}$$

$$A^2 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 2 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

$$A^3 = \underset{A_2}{\begin{bmatrix} 0 & 0 & 1 & 1 \\ 2 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}} \underset{A}{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}} = \begin{bmatrix} 2 & 1 & 0 & 1 \\ 1 & 2 & 1 & 1 \\ 2 & 2 & 1 & 2 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

$$A^4 = \begin{bmatrix} 1 & 2 & 1 & 1 \\ 2 & 2 & 2 & 3 \\ 3 & 3 & 2 & 3 \\ 2 & 1 & 0 & 1 \end{bmatrix} \qquad B = A \cdot A^T$$

Let A be the adjacency of matrix of diagraph G, the element in ith row and jit column n of A^2 is equal to the number of paths of length n from the ith node to jit node

By using adjacency matrix we can determine whether there exists an edge from v_i to v_j in G by using

Rajdhani
DATE / /

$$Br = A + A^2 + A^3 + \dots + A^4$$

The Br represents the no of paths of length less than or equals to r from vi to vj.

It will be denoted by for elementry cycle or equal to n-1. In the case of ri = vj and the path is cycle, we need to examine all posible an elementry cycle of length less than or equals to n.

such cycles or paths are determined by,

$$Bn = A + A^2 + A^3 + \dots + A^n$$

* Path Matrix (Reachability Matrix)

Let $G = \langle V, E \rangle$ be a simple diagraph in which $|V| = n$ and the nodes of G are assumed to be ordered. An $n \times n$ matrix P whose elements are given by

$$P_{ij} = \begin{cases} 1 & \text{Path between } v_i \,\&\, v_j \\ 0 & \text{otherwise} \end{cases}$$

is called path / reachability matrix of Graph G

Path matrix shows presence / absence of cycle at any node.

$$B_4 = \begin{bmatrix} 3 & 4 & 2 & 3 \\ 5 & 5 & 4 & 6 \\ 7 & 7 & 4 & 7 \\ 3 & 2 & 1 & 2 \end{bmatrix} \quad \text{Hence} \quad P = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Rajdhani
DATE / /

* Matrix Representation of Graph

Diagramatic representation of graph have limited usefulness

Possible only at no of nodes and edges are reasonably small

useful in paths cycle and other characteristics of graphs.

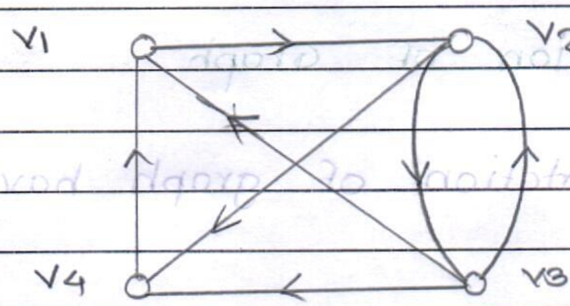
Let $G = \langle V, E \rangle$ be simple digraph with $V = \{v_1, v_2, \ldots\ldots v_n\}$ nodes having ordering from v_1 to v_n then $n \times n$ matrix given as

$$a_{ij} = \begin{cases} 1 & \text{if } \langle v_i, v_j \rangle \in E \\ 0 & \text{otherwise} \end{cases}$$

It is called adjecency matrix of graph G.

Adjecency matrix is either 0 or 1 it is called as bit matrix or boolean matrix.

From the graph we can draw matrix as follow



Graph

Adjecency matrix

$$A_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Reflexive = If diagonal elements are identical

Symmetric = $a_{ij} = a_{ji}$

original matrix and transpose of matrix should be identical

Antisymmetric = If $a_{ij} = 1$ then $a_{ji} = 0$

if $a_{ji} = 1$ then $a_{ij} = 0$

Rajdhani

DATE / /

Now transpose of adjacency matrix A will be replace column by row and viceversa.

$$A^T = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

Now

$$A.A^T = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 2 & 1 & 0 \\ 1 & 1 & 3 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \qquad A^T.A = \begin{bmatrix} 2 & 1 & 0 & 1 \\ 1 & 2 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix}$$

We can simply take it as

$$B = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 2 & 1 & 0 \\ 1 & 1 & 3 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

We can find $A^2, A^3, A^4 - - - A^n$

where $n = 2, 3, 4, -$

There are 2 path from v_2 to v_1 of length 1 $\Rightarrow A^2$

and also 4 path from v_2 to v_1 $\Rightarrow A^4$

Rajdhani
DATE / /

If B is the matrix then path from it will be

$$B_r = A + A^2 + A^3 + \cdots\cdots A^r$$

Path matrix / Reachability matrix :

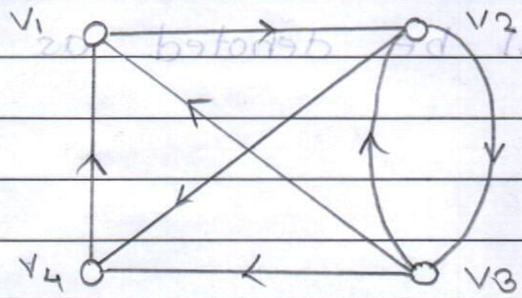
Let $G = \langle V, E \rangle$ be a simple diagraph in which $|V| = n$ and node of G are assumed to be ordered. An $n \times n$ matrix P whose elements are given by

$$P_{ij} = \begin{cases} 1 & \text{if path from } v_i \text{ to } v_j \\ 0 & \text{otherwise} \end{cases}$$

is called path matrix (reachability) matrix of graph G.

The matrix B_n by choosing $P_{ij} = 1$
If we have to find B_n
$$B_n = A^1 + A^2 + \cdots\cdots + A^n$$
$\therefore$ if $B_n = A^1 + A^2 + A^3 + A^4$

If the node contain at least one path then the path matrix will contain 1 otherwise 0



$$\text{Adjacency matrix} = A^1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$A^2 = A \cdot A = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 2 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

$$A^3 = A \cdot A^2 = \begin{bmatrix} 2 & 1 & 0 & 1 \\ 1 & 2 & 1 & 1 \\ 2 & 2 & 1 & 2 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

$$B^3 = A^1 + A^2 + A^3$$

$$= \begin{bmatrix} 2 & 2 & 1 & 1 \\ 3 & 3 & 2 & 2 \\ 4 & 4 & 2 & 4 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

path matrix of B^3 will be denoted as P

$$P = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Rajdhani
DATE / /

* Trees :

A directed tree is an acyclic digraph which has one node as rootnode with indigree 0 . while other all nodes have indegree 1

Every directed tree have at least one node.

Terminal node/ leaf :

The node which having outdegree 0 called leaf.

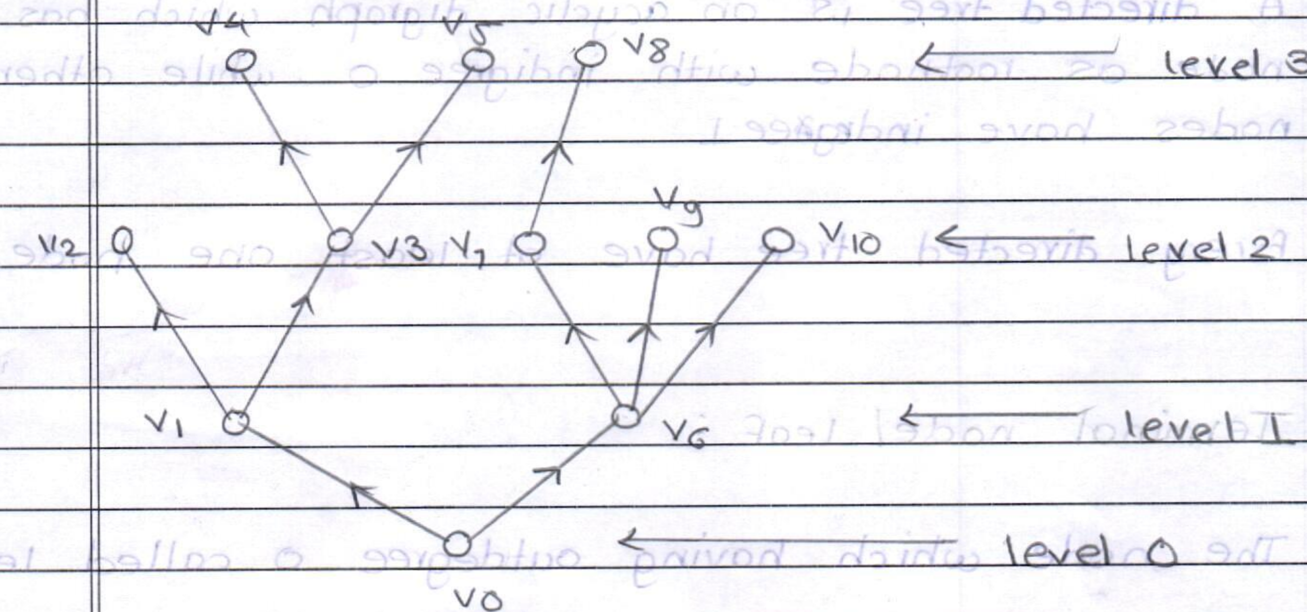
Branch node -
other than terminal and root node called branch node

level -
root 0 level
any node distance from root.

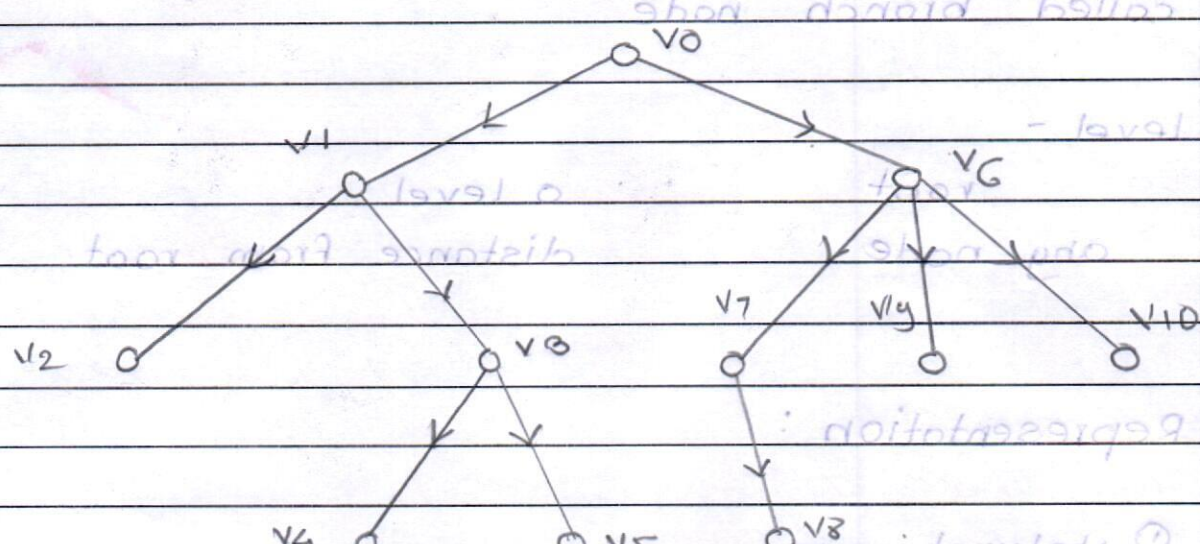
Representation :

① Natural array
② upside array
③ Root at original and nodes from left to right.

Natural away :

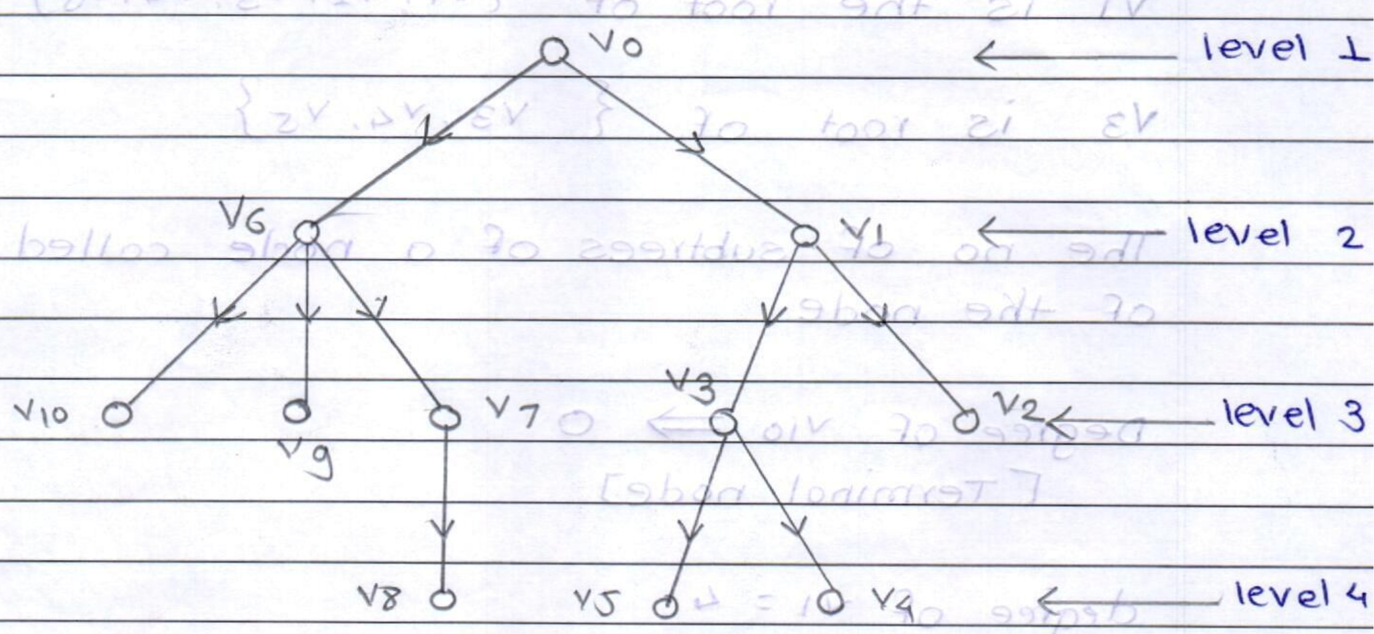


Upside down



Rajdhani
DATE / /

upside down from left to right



Directed tree is acyclic

In directed tree ordering of tree or if each node contain order then called as ordered tree

If the tables of nodes from 1, 2 ...... n from left to right called as canonically labeled

The root node may be at top or bottom

Now at level 1 node V6 having the root of subtree.

Rajdhani
DATE / /

$\{ V_6, V_7, V_8, V_9, V_{10} \}$

V_1 is the root of $\{V_1, V_2, V_3, V_4, V_5\}$

V_3 is root of $\{V_3, V_4, V_5\}$

The no of subtrees of a node called subtree of the node

Degree of $V_{10} \Rightarrow 0$
[Terminal node]

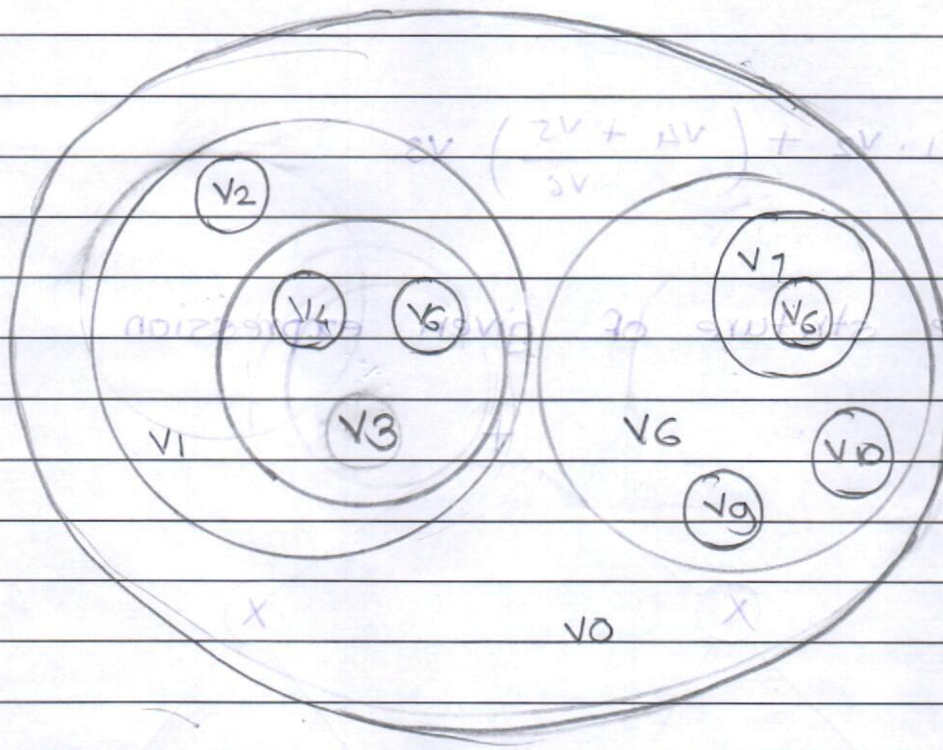
degree of $V_1 = 4$
degree of $V_3 = 2$

If we delete root node and edges joined to level 1 node we get the forest any node of directed tree is the root of same subtree

subtree below a node from a forest

Representotion of tree :

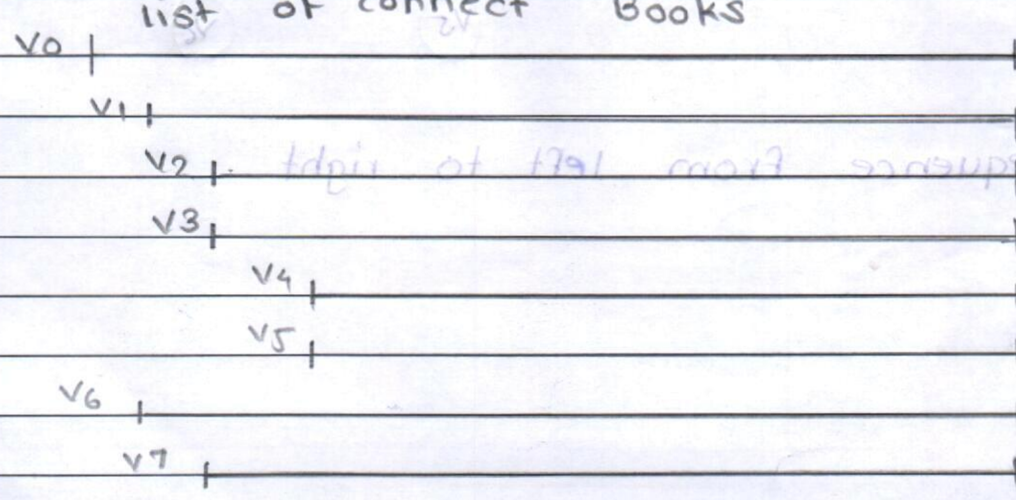
① venn diagram

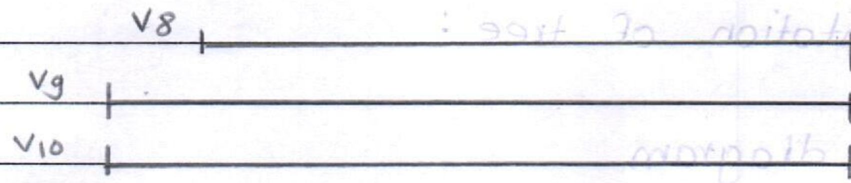


② conversion of nesting paronthesis

② $(V_0 (V_1 (V_2) (V_3 (V_4) V_5))) (V_6 (V_7 (V_8)) (V_9) (V_{10})))$

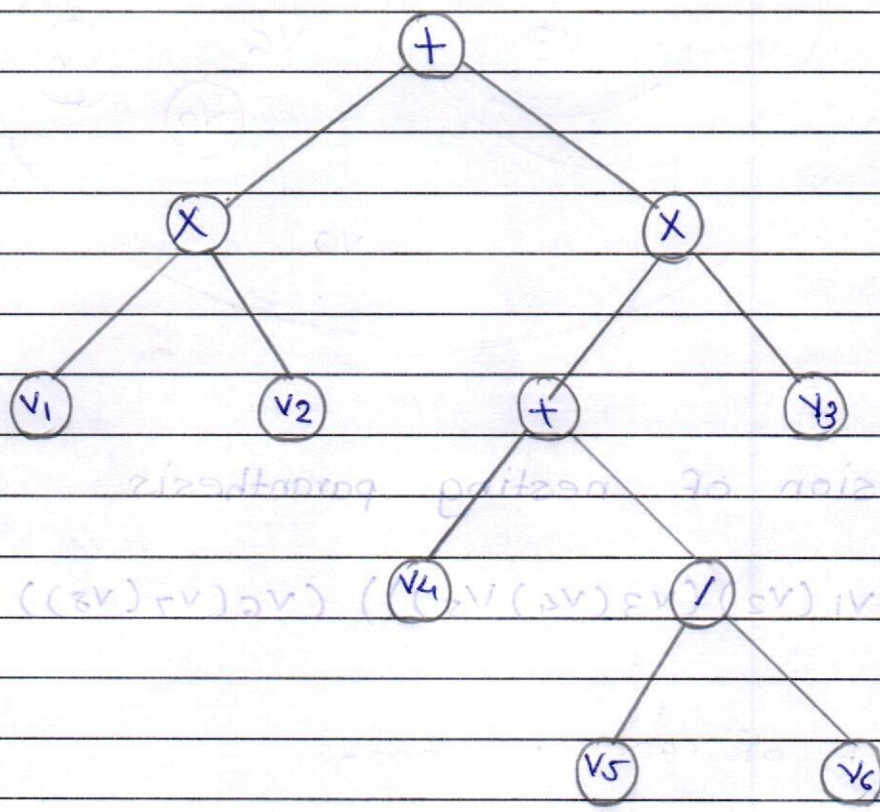
③ list of connect Books





$$v_1 \cdot v_2 + \left(v_4 + \frac{v_5}{v_6} \right) v_3$$

Tree strcture of given expression



sequence from left to right

Rajdhani
DATE / /

* Descendent of u

Every node reachable from u called descendent of u.

Ex: v_{10}-v_9, v_7, v_8 are descendent of v_6

Sons of u :

A node which are directly reachable from u with single edge.

v_{10} is child of v_6

No restrication in outdegree only indegree is having restrication.

In directed tree outdegree of every node is less than or equal to m then tree called as m any tree

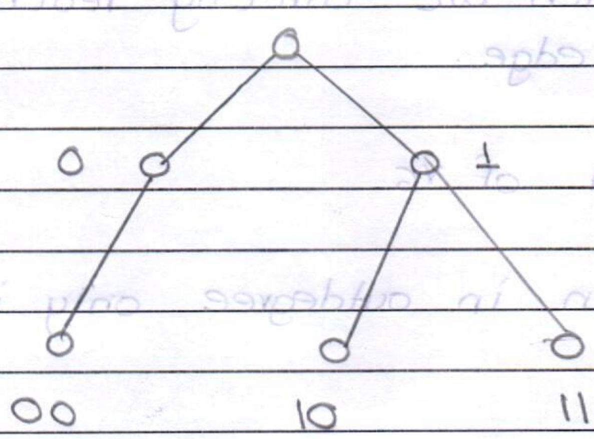
outdegree

0 or m	full or complete m-any tree
m = 2	binary or full binary

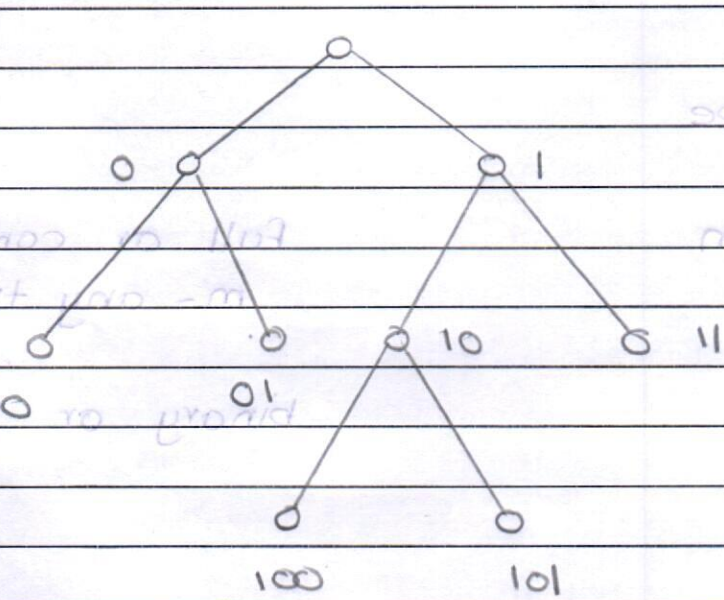
- Binary tree :
  Tree having node at most two outdegree

- full binary tree :
  Every node have 2 outdegree

Binary tree :

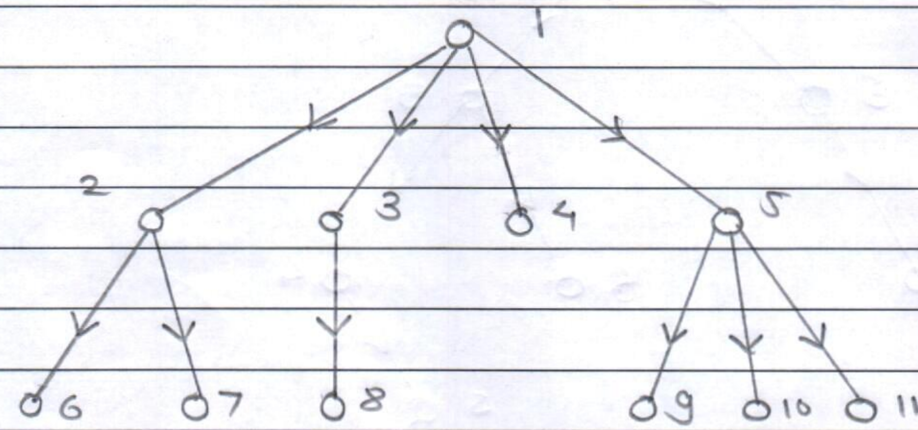


full binary tree :



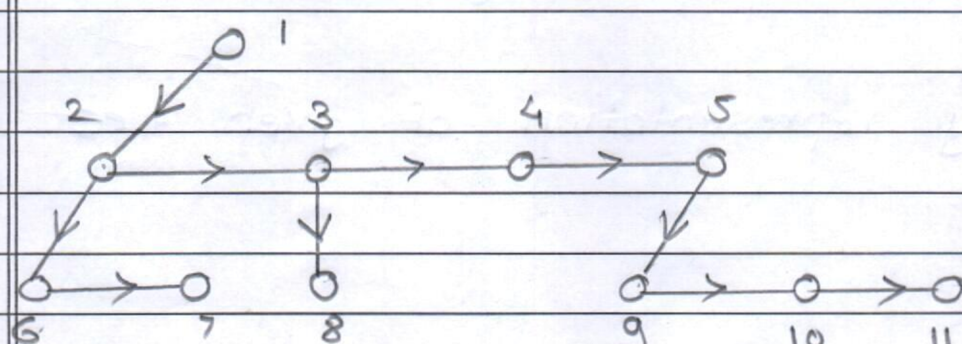
Rajdhani
DATE / /

The above fully binary tree can be represented as {0.1} string and denoted as {0,1,2 .... m-1}

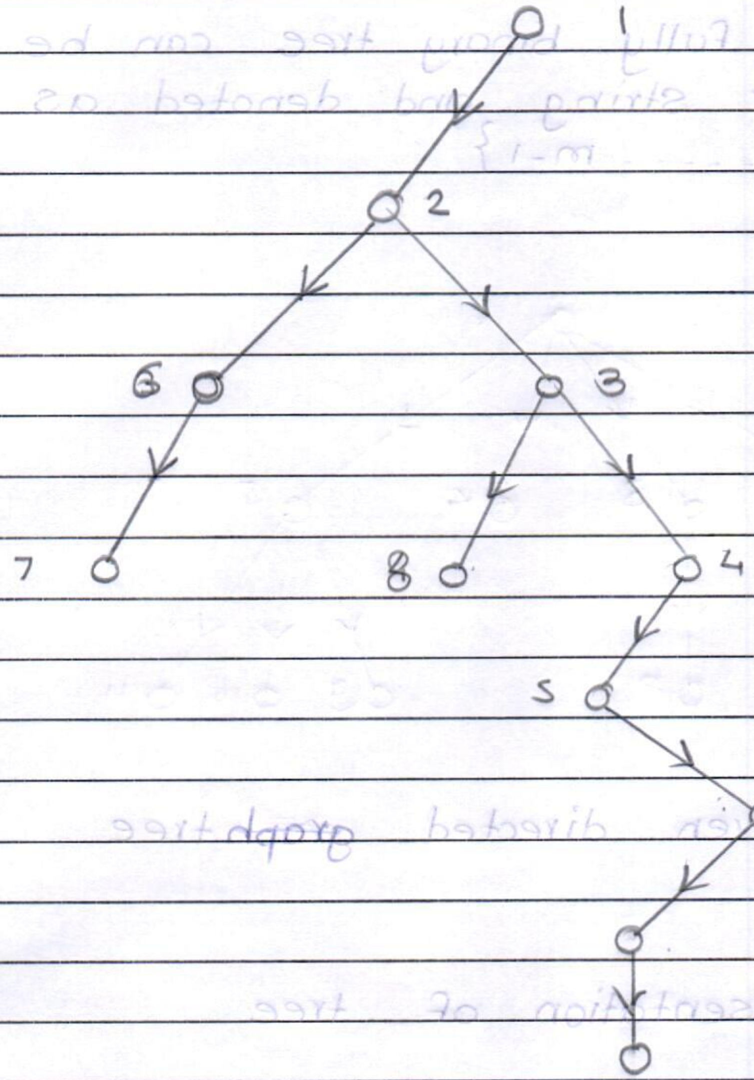


Given directed tree

Binary representation of tree



Stage 1



Binary representation of given tree

* Binary representation of a forest

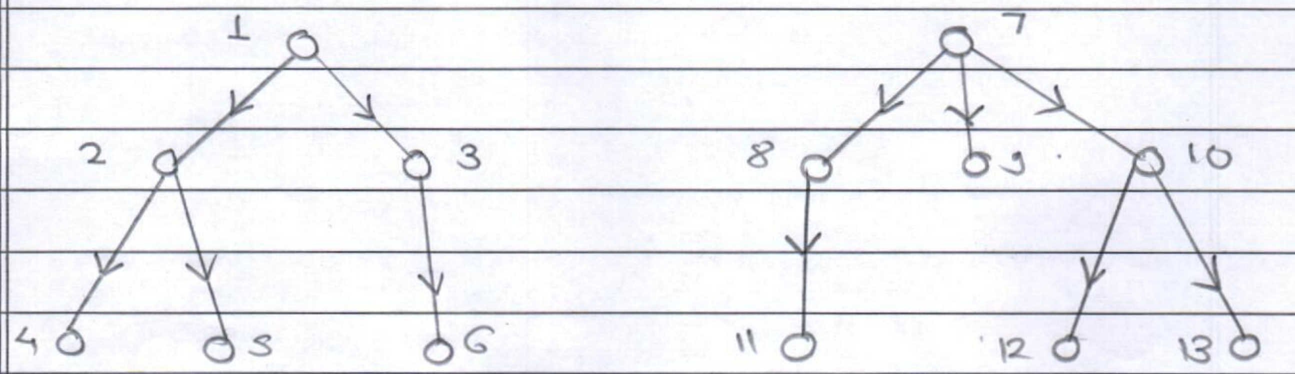
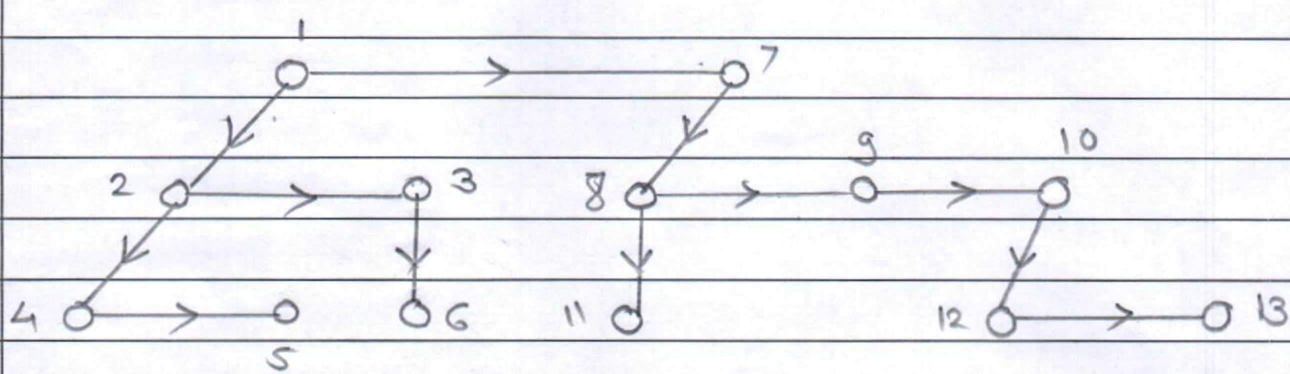
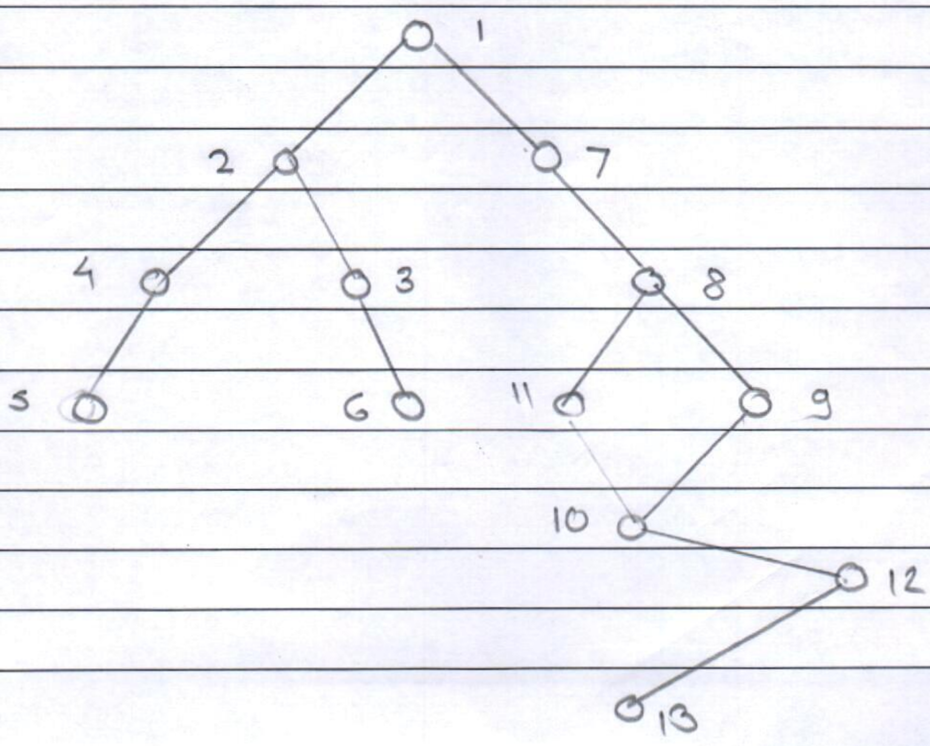


Fig : Forest





Rajdhani
DATE / /

* Pert & Related Techniques

By using directed graph we can design, describ represets and analyze the many interrelated an activities

Design & construction of power dam apartment building

In this example we are interested in determining critical path of diagraph

such critical path is managed by techniques like, PERT [Program evaluation and Review Technique] and CPM [critical path method]

By using critical path we have to find longest path between two nodes of weighted diagraph

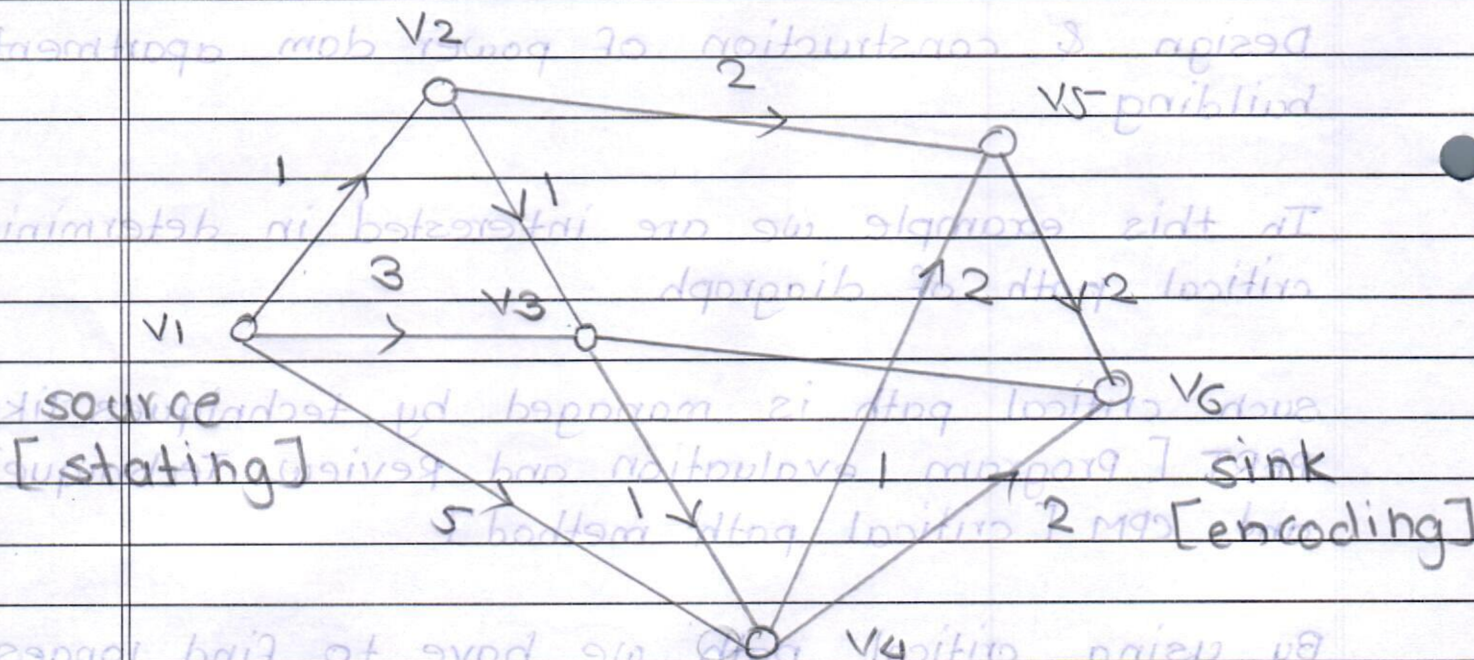
Definition :

A PERT graph is a finite — — — — — — diagraph with no parallel edges or cycles in which there is exactly one source (i.e, node which having indegree 0) and one sink (i.e, having outdegree 0) & each edge in graph is assigned a weight (time) value

Directed edge meant to represent the start time and end time between two edges.

The weight value represent the time required to complete activity

* A PERT Graph



The diagram contain 10 activities with $V_1, V_2, ---- V_6$ i.e., 6 nodes, Each Node called as event

The edges represent No of days to complete given tests.

The <activities <V2, V3>, <V3, V6> and <V3, V4> should complete before <V1, V2>, <V1, V3> and <V1, V4>

Before going to Next activity every activity should complete at initial level

At source node V1 we should assign the value 0 (time required)

i.e. $TE(V1) = 0$ [Time early]

$TE(V_j) = \max\{t(p)\}$ $j \neq 1$

where $t(p) \Rightarrow$ sum of time from V_i to V_j

From figure

V2 having one incoming edge so

TE at $V2 = 0+1 = 1$ $TE(V2) = 1$

V3 having two incoming edge

$(V1, V2, V3) = 2$

$(V1, V3) = 3$

chose $TE(V3) = 3$

similarly

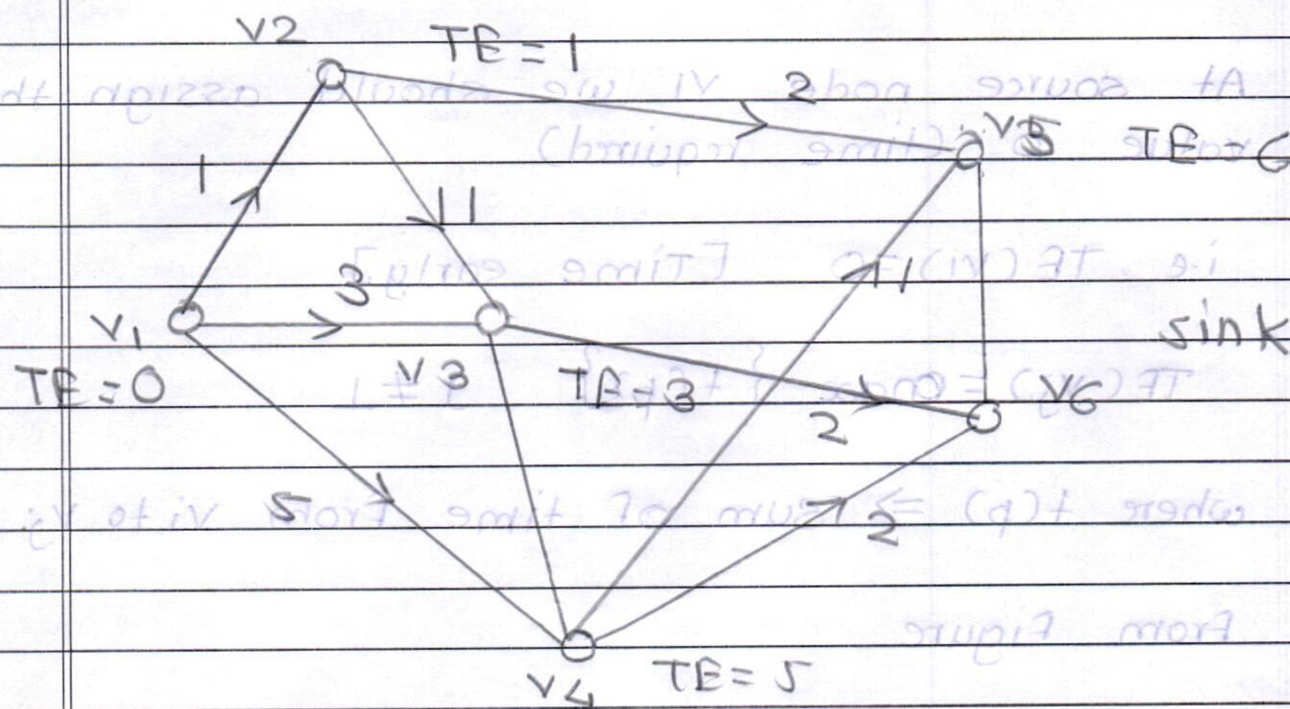
$TE(V4) = 5$

$TE(V5) = 6$

$TE(V6) = 8$

Rajdhani
DATE / /

A PERT with early complition time



Latest time completion :

It is calculated from sink to source denoted by TL

① TL (Vn) = TE (Vn)
At sinks Node

② TE (vj) = TE (vn) - max{ t (p) }

t(p) = time duration from vj to vn

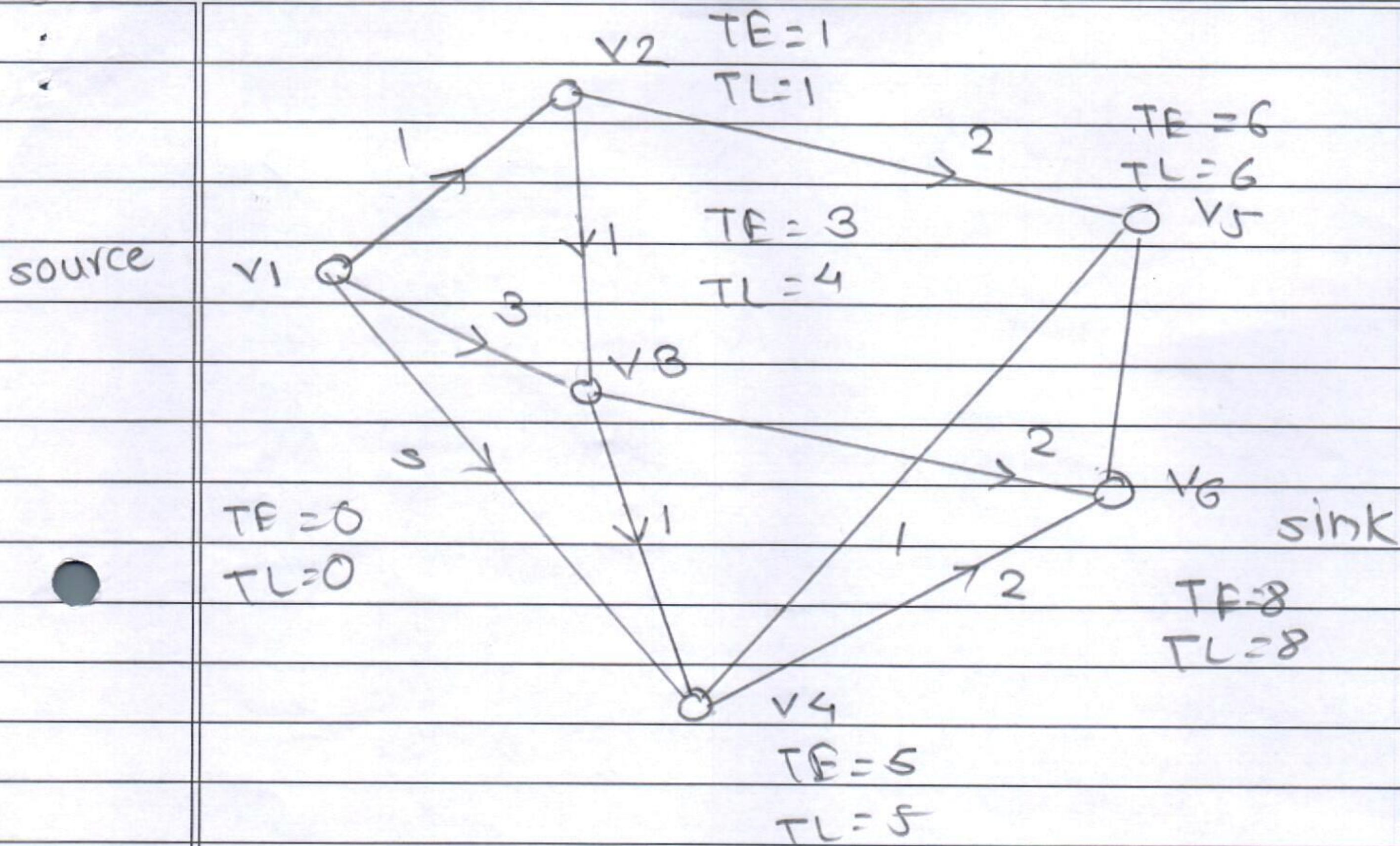


fig : A PERT graph with earlist & latest compietion time

stack time = The time different between earlist and latest time called stack time

stack time = |TE − TL|

critical path :

The path which consist of equal TL and TE values called original path

Ex : (V1, V4, X5, V6)

Rajdhani
DATE / /

TE method :

Find maximum values of every node travelling from source to sink

TL method :

Find maximum value of every node travelling from sink to source.

Rajdhani
DATE / /

* fault detection in condinational switching circuit

* Faults in combinational circuit

In 32 bit computer system if we have to execute add instruction need 1 uses (10^{-6} sec) extra time.

The no of possible operations are 1019, which need (8×10^{5}) years to execute in such cases the failure could be occur.

To avoid failure we may done :

① Make procedure as fast as possible so it may involve few tests in it.

② Result of procedure should be as specific as possible

The system may contain two types of failure :

① Transient / intermittent fault

② Per manent.

The transient fault is not effective as permanen in case of transient their is no ~~stads~~ - at fault. if the system contain

Rajdhani
DATE / /

open circuit problem, short circuit component Failure then they called as ~~stacks~~ stack at fault.

In stack at fault the input or o/p could be changed by 0 or 1

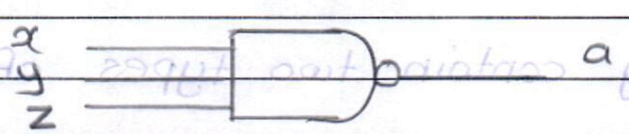
If i/p or o/p changes by 0 then it called as stack at zero i.e. (s-a-0)

IF it changes to one then (s-a-1)

Ex : In following example three i/p x, y, z with one o/p - a

stack at one (s-a-1)
stack at zero (s-a-0)

By using NAND gate we can draw table as follow



* Notations of Fault Detection

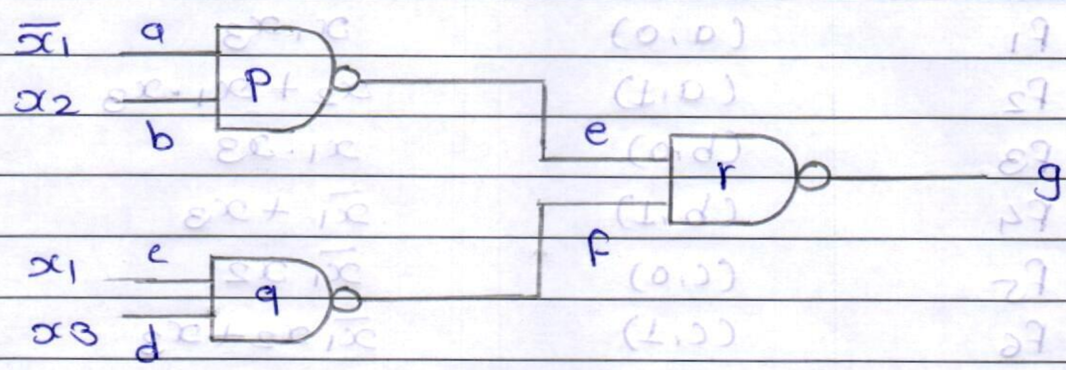


fig : A switching circuit

In given switching circuit 4 i/p and 3 NAND gates are there, consider all stack at zero (s-a-0) & (s-a-1) at every i/p we will get $f_1, f_2, \ldots\ldots f_{14}$ functions in it.

By putting the values of i/p we find out the table of faults.

Test :

Test in a combinational circuit is the application of a combination of i/p symbols to the circuit.

There are 3 i/p so it will contain test will be 8

i.e.,

$t_0, t_1, \ldots\ldots t_7$

Fault Designation	Description	Function at output
F_1	(a,0)	$x_1 \cdot x_3$
F_2	(a,1)	$x_2 + x_1 \cdot x_3$
F_3	(b,0)	$x_1 \cdot x_3$
F_4	(b,1)	$\bar{x}_1 + x_3$
F_5	(c,0)	$\bar{x}_1 x_2$
F_6	(c,1)	$\bar{x}_1 x_2 + x_3$
F_7	(d,0)	$\bar{x}_1 \cdot x_2$
F_8	(d,1)	$x_1 + x_2$
F_9	(e,0)	1
F_{10}	(e,1)	$x_1 \cdot x_3$
F_{11}	(f,0)	1
F_{12}	f(1)	$\bar{x}_1 x_2$
F_{13}	g(0)	0
F_{14}	g(1)	1

Fig : single fault for circuit

From table F_1, F_3, F_{10} are identical o/p as well as F_5, F_7, F_{12} and F_9, F_{11}, F_{14} having same o/p so remove them and take only initial function F_1, F_5, F_9

Rajdhani
DATE / /

Test	x_1	x_2	x_3	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}
t_0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1	0	0	1
t_1	0	0	1	0	0	0	0	1	0	1	0	0	1	0	1	0	0	1
t_2	0	1	0	1	0	0	1	1	1	1	1	1	1	0	1	1	0	1
t_3	0	1	1	1	0	0	0	1	1	1	1	1	1	0	1	1	0	1
t_4	1	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1
t_5	1	0	1	1	1	1	1	1	0	1	0	1	1	1	1	0	0	1
t_6	1	1	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1
t_7	1	1	1	1	1	1	1	1	0	1	0	1	1	1	1	0	0	1

fig : Results of Tests for circuit

Indistionguishable faults :

In combinational circuit two haults f_i & f_j are said to be indistinguishable faults if they have same value presence in f_i and f_j

Ex. ① f_1, f_3, f_{10}

② f_5, f_7, f_{12}

③ f_9, f_{11}, f_{14} are indistinguishable

From above table remove such faults we will get faults matrix.

Distinguishable :

If f_i & f_j are different called distinguishable.

Test	x_1	x_2	x_3	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}
t_0	0	0	0	0	0	0		1	0	0		0	1	0
t_1	0	0	1	0	0	0		1	0	1		0	1	0
t_2	0	1	0	1	0	1		1	1	1		1	1	0
t_3	0	1	1	1	0	1		1	1	1		1	1	0
t_4	1	0	0	0	0	0		0	0	0		1	1	0
t_5	1	0	1	1	1	1		1	0	1		1	1	0
t_6	1	1	0	0	0	1		0	0	0		1	1	0
t_7	1	1	1	1	1	1		1	0	1		1	1	0

Fig : Fault matrix

				All value of x are 0	All value are 1
x	y	z	a	s-a-0	s-a-1
0	0	0	1	1	1
0	0	1	1	1	1
0	1	0	1	1	1
0	1	1	1	1	0
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	1	1	1
1	1	1	0	1	0

Fig : table of stack at faults

In given table original o/p a and s-a-0 and s-a-1 are different so it is called stack at fault.

* Algorithm for Generating fault matrix

To convert combinational switching circuit we convert fault matrix

NAND gates into nodes

convert connection between two NAND gate edge between two nodes

Primary input : The nodes or gates which an i/p called primary i/p

Method to convert :

1. Initially assign successive integers starting at 1 to each of the primary i/p

2. Assign successive integer to those gates all of those i/p are derived from get already numbered

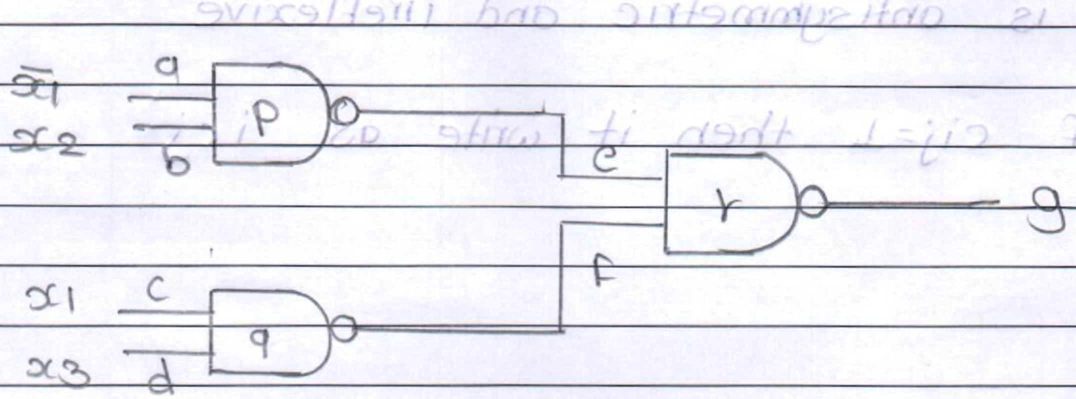


fig : A switching circuit

Rajdhani
DATE / /

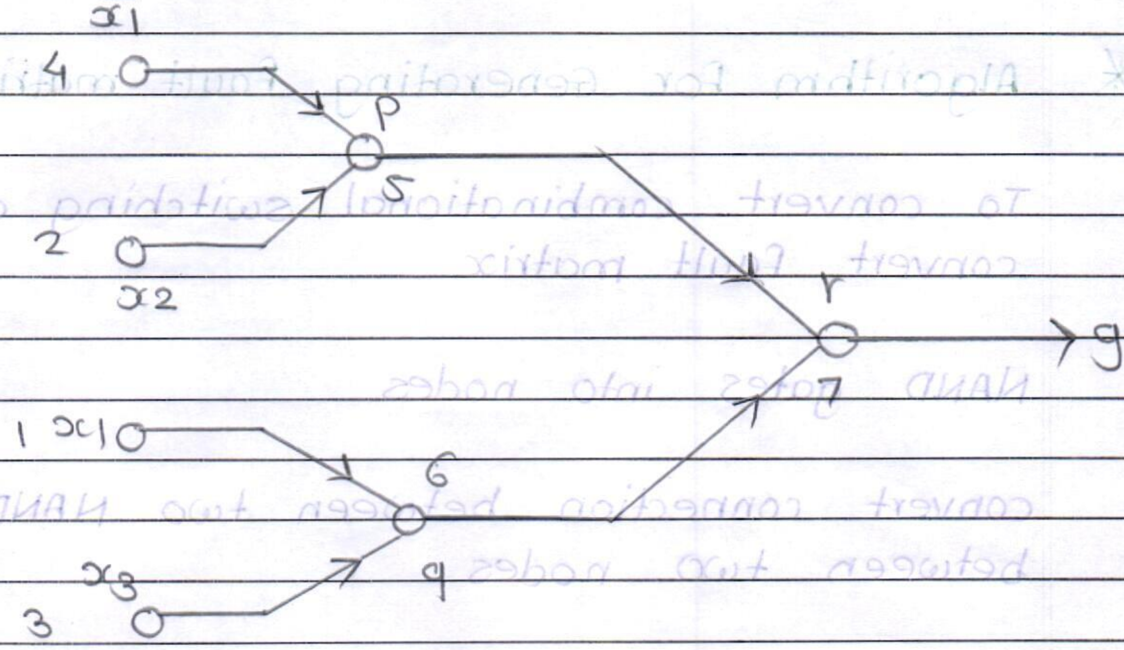


fig : Directed graph representation

In above diagram 1, 2, 3, 4 are primary i/p
5 & 6 intermediate gate
7 final o/p gate

Adjacecy matrix of combinational circuit called connectivity matrix and denoted by c

Naturally $c_{ij} = 1$ if o/p of gate is i/p to gate j otherwise $c_{ij} = 0$

c is antisymmetric and irreflexive

If $c_{ij} = 1$ then it write as icj

Input			Intermediate o/p		Final o/p
x_1	x_2	x_3	p	q	g
0	0	0	1	1	0
0	0	1	1	1	0
0	1	0	0	1	1
0	1	1	0	1	1
1	0	0	1	1	0
1	0	1	1	0	1
1	1	0	1	0	0
1	1	1	1	0	1

If No of i/p = n then o/p will be 2^n tuples

Here n = 3 , tuples = 2^3 = 8 tuple

Now, output of g is 8 tuple

<0,0,1,1,0,1,0,1> as shortly 00110101

Here 8 tuples or 2^n tuples minterms or sum of product canonical form

for g $\Rightarrow \Sigma(2,3,5,7)$

for $x_2 \Rightarrow \Sigma(2,3,6,7)$

By using this algorithm we can find out product of sum canonical form. i.e, maxterm

Denoted by π

for $g \Rightarrow \pi(0,1,4,6)$

$x_2 \Rightarrow \pi(0,1,4,5)$

If given example doesn't contain any example then maxterms then it will simply denoted by (x)

on given set of i/p NAND gate perform s-a-0 and s-a-1 operation

Maxterm representation of s-a-1 fault is always having value 0

If the given circuit consist of r inputs then by using NAND gate we get result y as

$$(y) = \overline{(x_2) \cup (x_2) \cup \cdots \cup (x_r)}$$

o/p maxterms set - maxterm set produced at o/p of gate

i/p maxterms set - maxterm set produced at i/p of gate

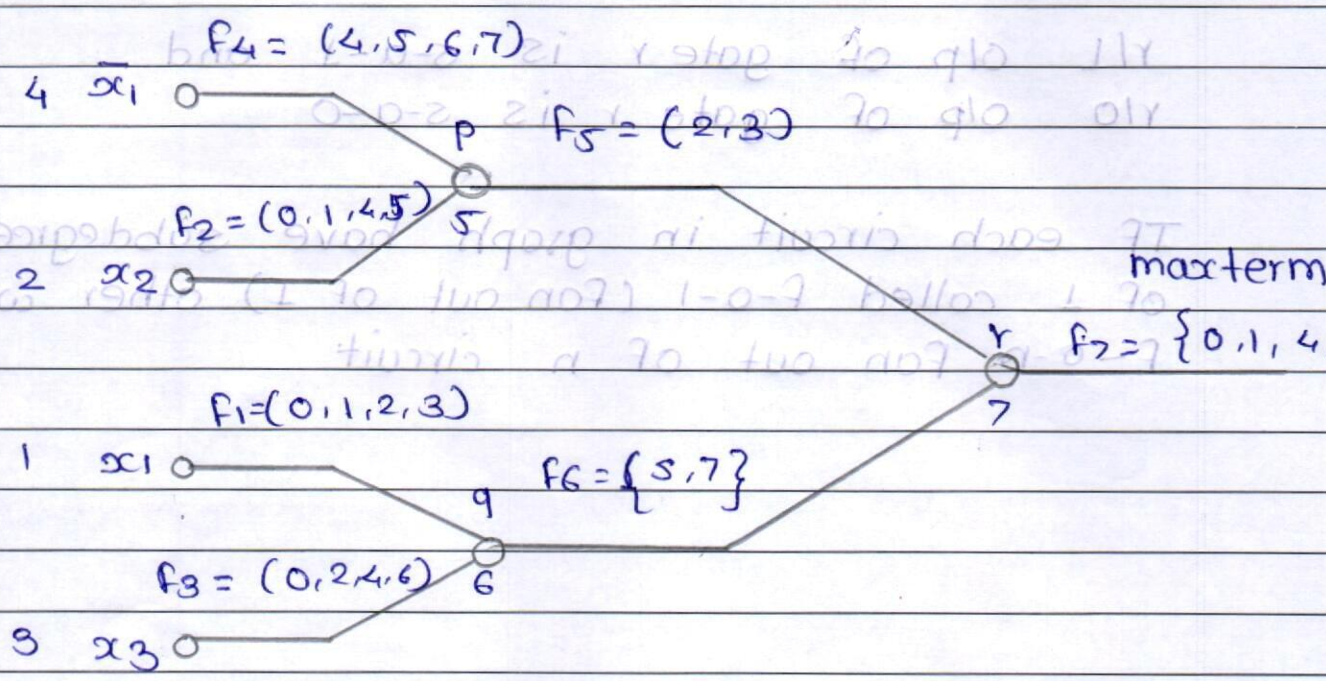


Fig : circuit representation with maxterm sets

The sets are denoted by f_i for i gate

maxterm in set are denoted by $f_1, f_2, \ldots\ldots f_m$

* Notation for stack at fault

$p-q/1$ denote that i/p to gate q from p is s-a-1 and

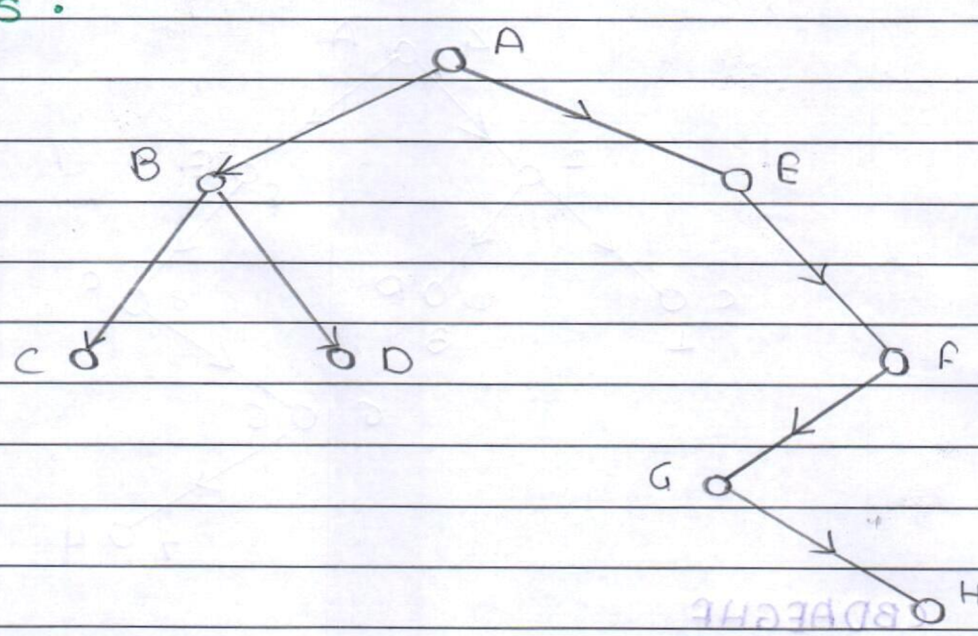
$p-q/0$ i/p to gate q from p is s-a-0

Rajdhani
DATE / /

r/1 o/p of gate r is s-a-1 and
r/0 o/p of gate r is s-a-0

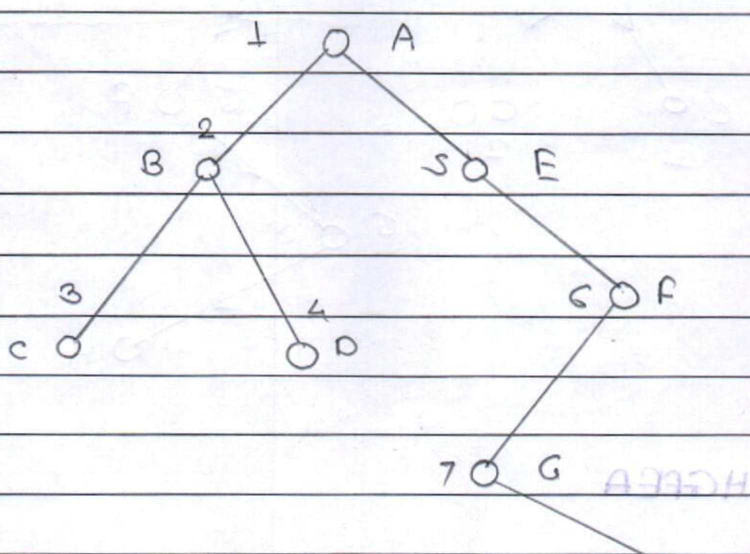
If each circuit in graph have subdegree of 1 called f-o-1 (fan out of 1) other wise f-o-n fan out of n circuit

* Storage Representation and manipulation of Graphs :



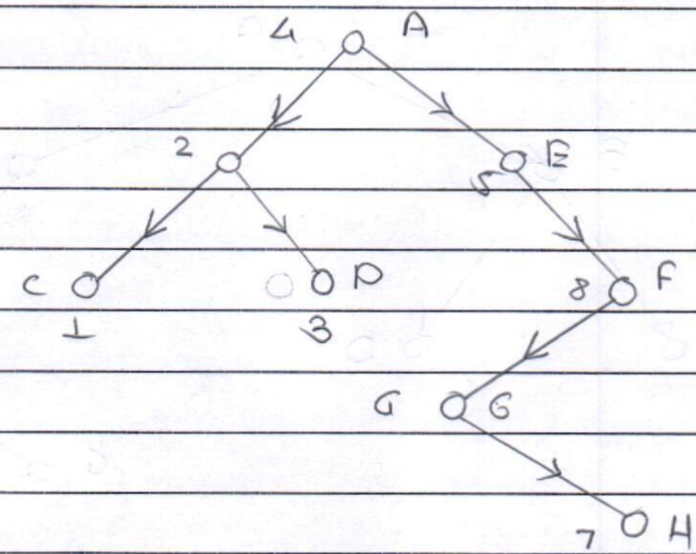
Preorder Travel :

ⓐ Process the root node
ⓑ Traverse the left subtree in preorder
ⓒ Traverse the right subtree in preorder



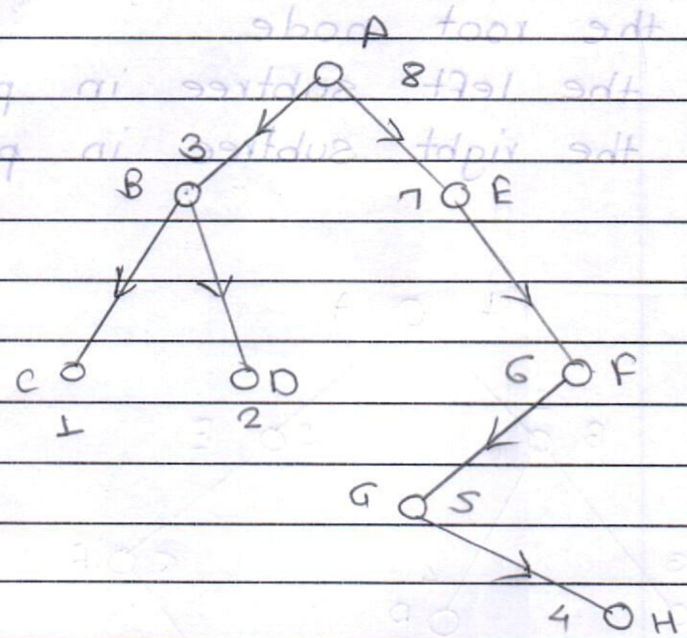
$\overline{\text{ABCDEFGH}}$

② Inorder Traversal :



CBDAEGHF

③ Postorder :



CDBHGFEA

* Storage Representation and manipulation of graphs

① Tree : their representation and operation

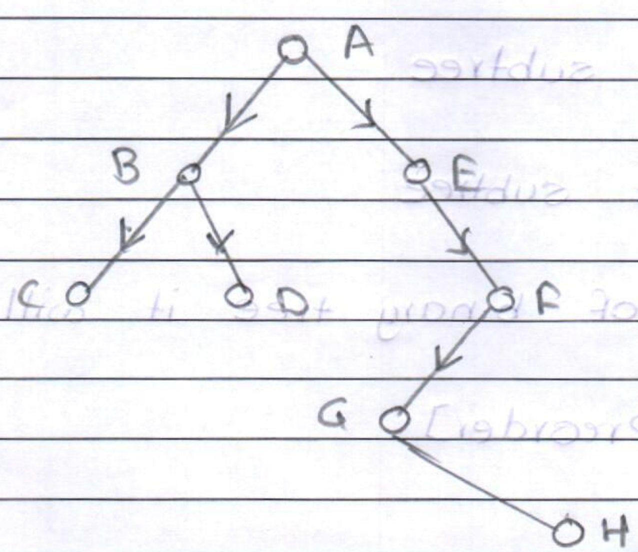
Tree strcture allocation is simple to insert and delete node on it.

By using tree strcture & allocation we can convert any binary tree into binary tree

Binary representation :

L LINK	DATA	R LINK

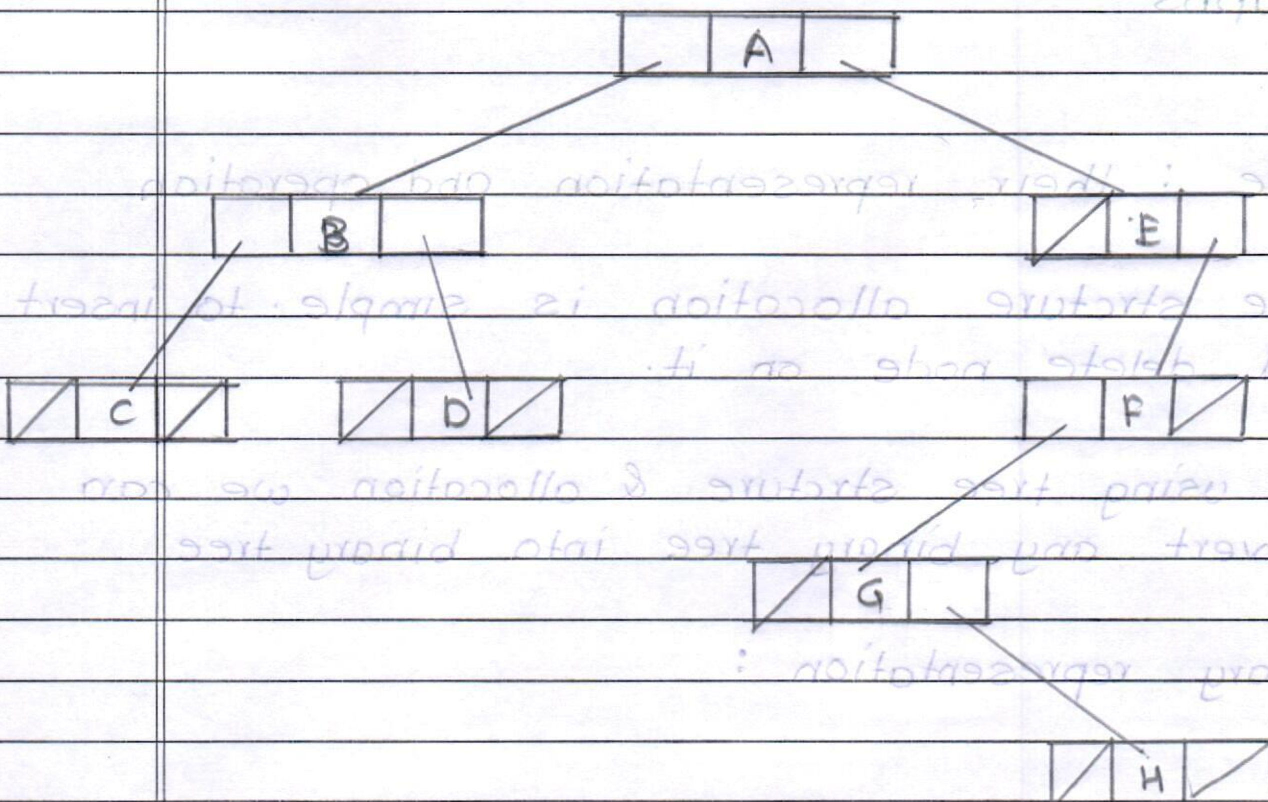
Left subtree pointer   Node   Right Point



Simple tree

Rajdhani
DATE / /

Graph & Linked Representation of binary tree



① Preorder Traversal

- Process the root node
- Traverse left subtree
- Traverse Right subtree

From diagram of binary tree it will

ABCDEFGH [Preorder]

Rajdhani
DATE / /

② Inorder

Traverse left subtree
Process root node
Traverse right subtree

Ex. CBDAEGHF

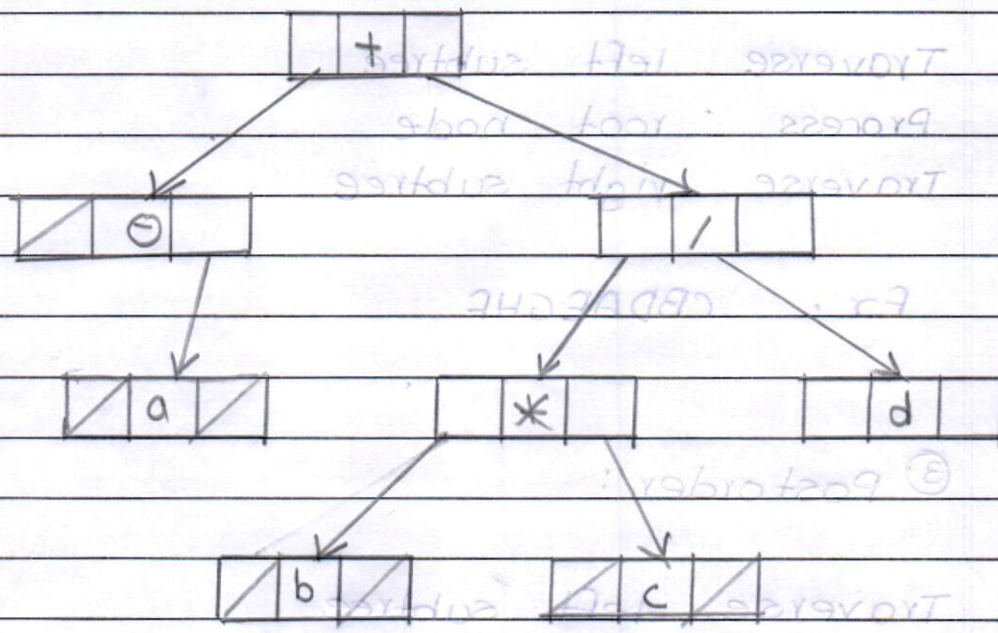
③ Postorder :

Traverse left subtree
Traverse right subtree
Process root node

Ex. CDBHGFEA

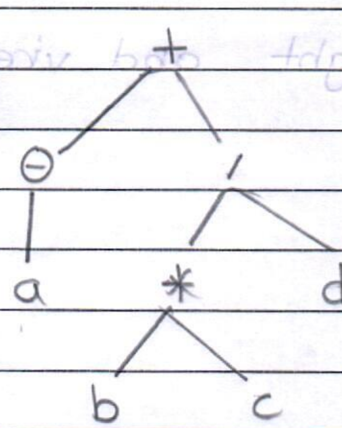
If we replace left by right and viceversa we get converse of it.

convert following formula into binary tree

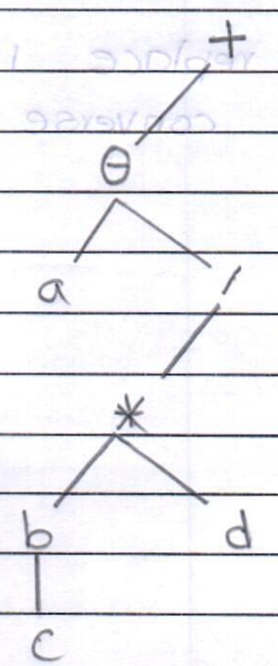


a + b * c / d

Fig linked tree



General tree



Binary tree

Rajdhani
DATE / /

* List strctures and graphs

A list strctures are used for represent directed graph.

List:
Any finite sequence of zero or more atoms or lists. Where atom is taken to be any object [string of symbol] Which is distinguishable from a list by treating the atoms are structurally indivi sible.

We enclose atoms or list with parothesis or as follow.

Ex: ① (a, (b, c), d (e, f, g))
② ()
③ (((a)))

In example 1st list of four elements Namely.

a ⇒ atom
(b, c) ⇒ list contain atoms b, c
d ⇒ atom
(e, f, g) ⇒ list of atoms e, f, g

second example contain no elements

third example contain element (a) and a is atom.

Rajdhani
DATE / /

A list is directed graph with one source node (indegree 0)

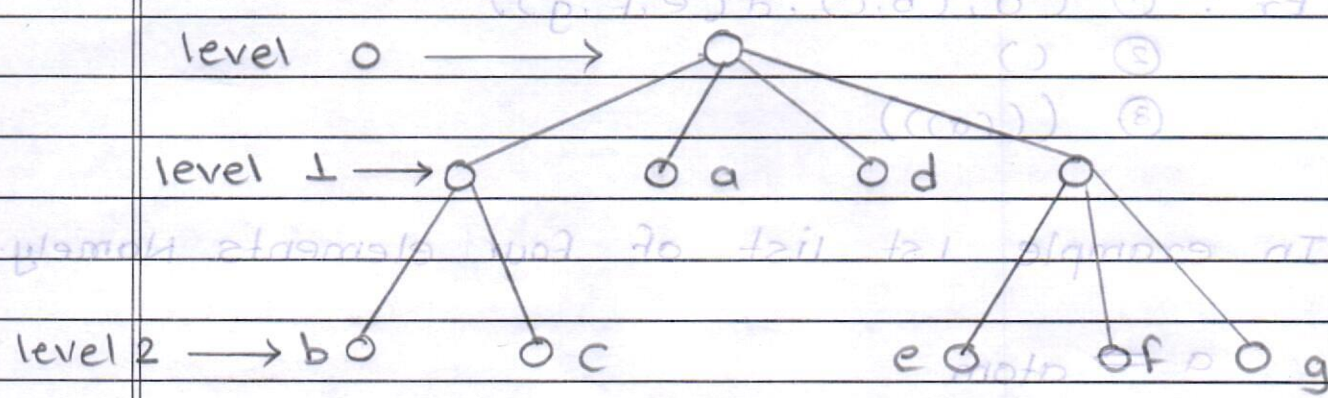
For atoms outdegree is 0

Every node except source node having indegree of 1

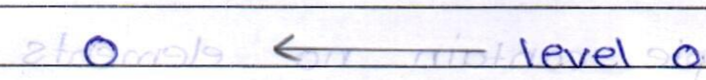
Every tree can be represent list.

Ex : Graph of list strcture

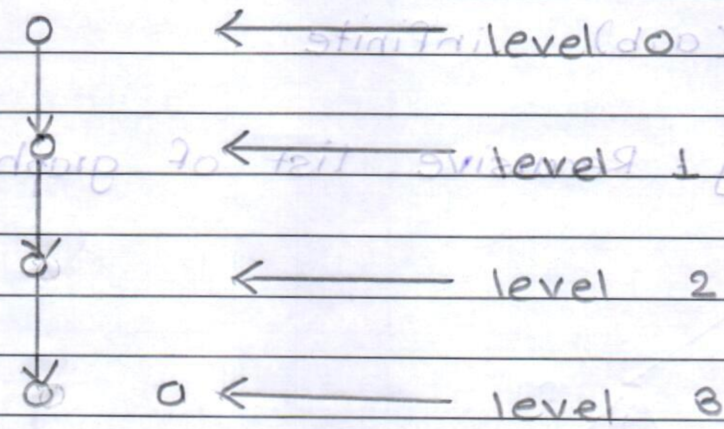
① (a, (b, c), d, (e, f, g))



② ( )



③ (((a)))



Linked allocation technique is used to represen[t]
Lists in memory of computer

Two types of node : ① one for atom

② one for list

① Atom : value of atom or string of symbol

② List : Pointer to list value atoms.

Depth :

No of paranthesis in given atom called depth

Ex : (a (b,c), d)

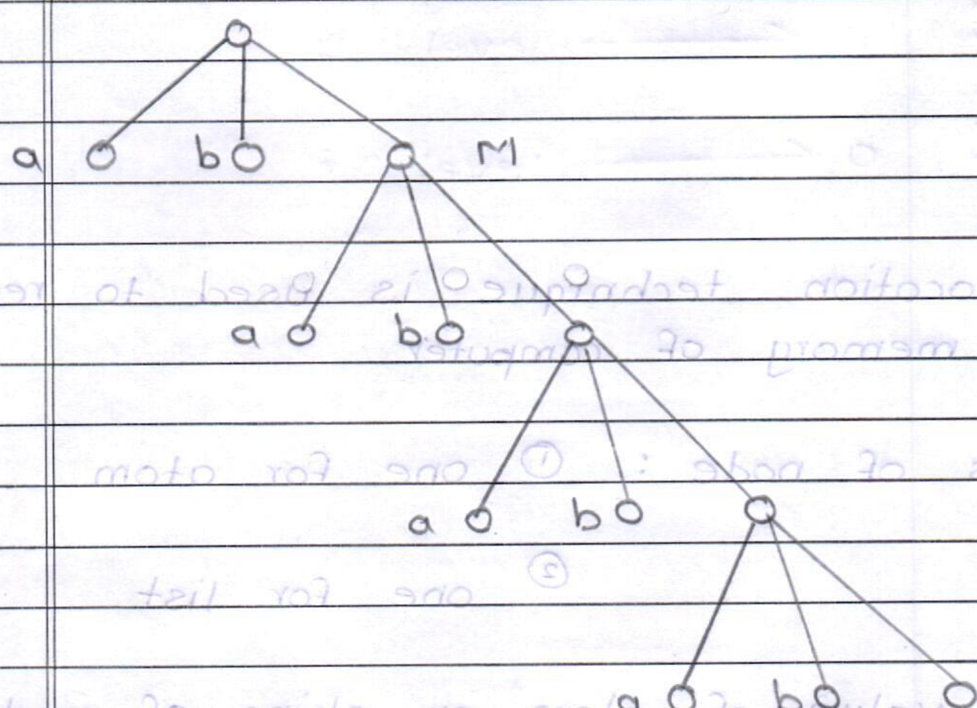
depth ⇒ 2 for b & c
depth ⇒ 1 for a & d

Rajdhani
DATE / /

Draw list (a, b, M)

M = (a, b) infinite

i.e, fig : Recursive list of graph



Recursion means repeatation or simply in storage representation.

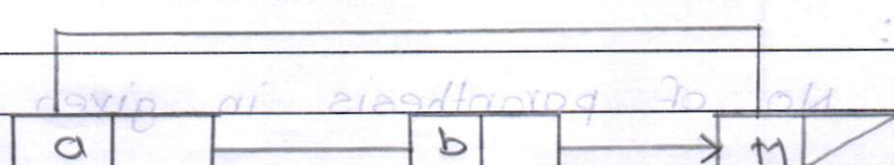


fig : storage representation

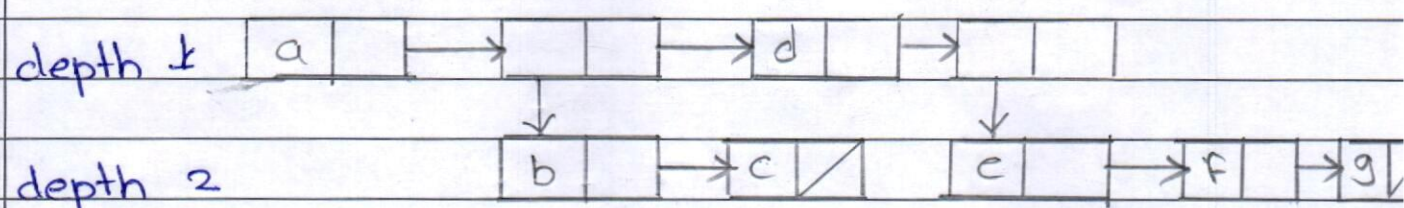
Storage Representation :

Ex : (a, (b, c), d, (e, f, g))

depth of a, d = 1

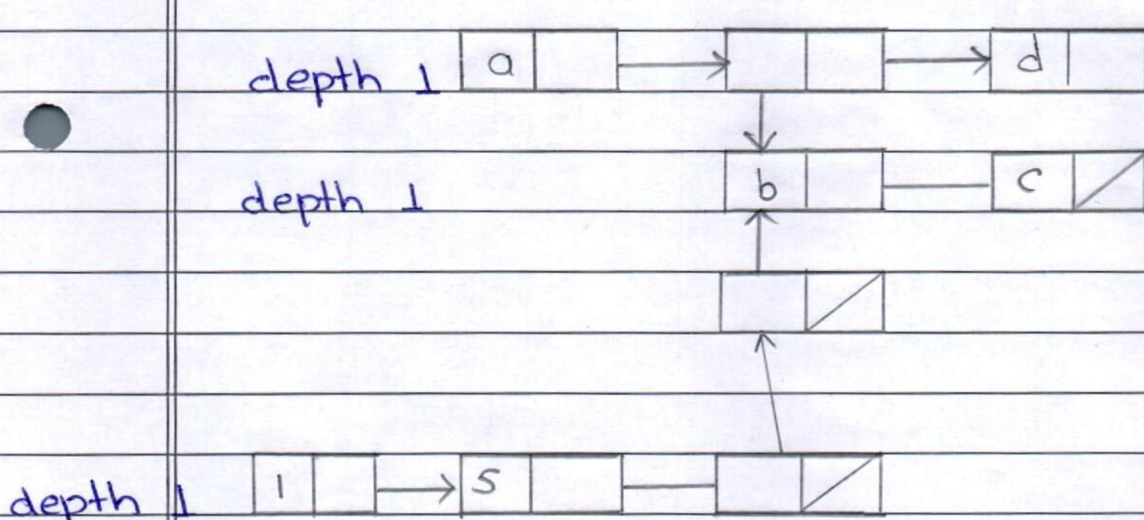
b, c = 2

e, f, g = 2



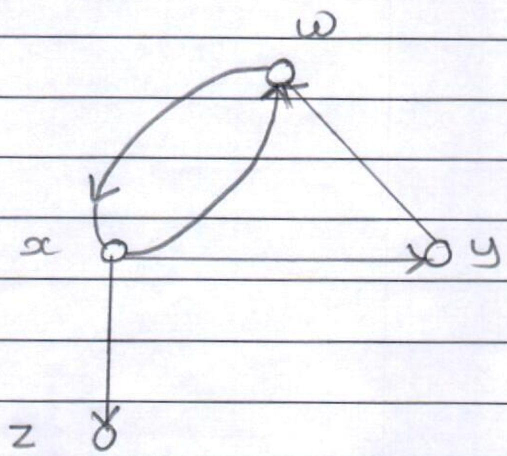
Ex : List share common sublist

(a, (b, c), d) and (1, 5, (b, c))

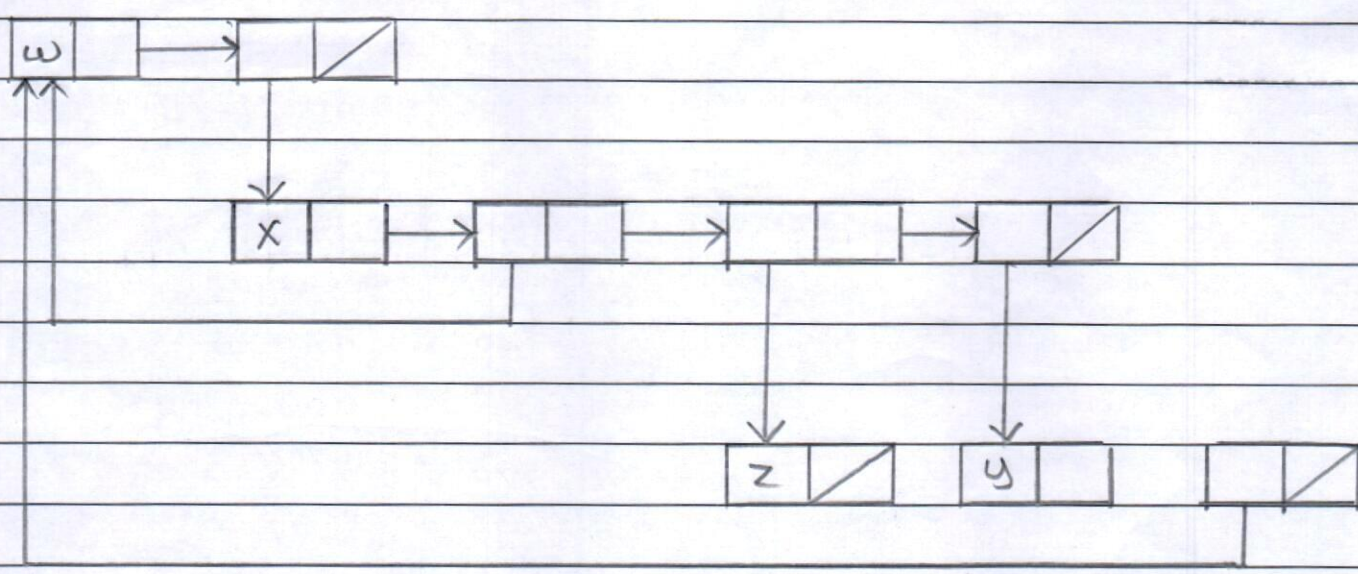


Rajdhani
DATE / /

* Diagraph and its list strcture representation



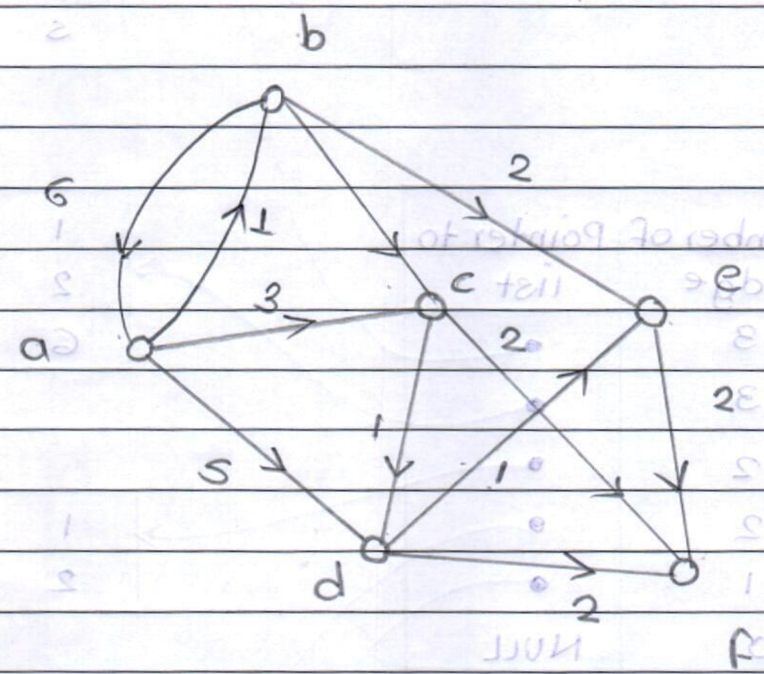
List strcture of above digraph



Rajdhani

DATE / /

Storage Representation of a weighted diagraph



Weight	Destination Node
1	2
3	3
5	4

1	3
2	5
6	1

Node #	Node data	Number of edge	Pointer to list
1	a	3	•
2	b	3	•
3	c	2	•
4	d	2	•
5	e	1	•
6	f	0	NULL

1	4
2	6

1	5
2	6

2	6