CHAPTER :- GRAPHS

A graph $G = (V, E)$ consists of two things :-

(i) A set V of elements called nodes (or points or vertices)

(ii) A set E of edges such that each edge e in E is identified with a unique (unordered) pair $[u, v]$ of nodes in V.
denoted by $e = [u, v]$

If $e = [u, v]$
then nodes u and v are called endpoints of e.

u and v are said to be adjacent nodes or neighbours.

THINGS TO DO

Degree of a node u, $\deg(u)$ is the number of edges containing u.

If u does not belong to any edge then u is called an isolated node.

Path A path P of length n from a node u to a node v is defined as a sequence of n+1 nodes.

$$P = (v_0, v_1, v_2, \dots v_n)$$

such that $u = v_0$; v_{i-1} is adjacent to v_i (for $i = 1, 2, \dots n$)

$v_n = v$

The path P is said to be **closed** if $v_0 = v_n$.

A Graph G is said to be **connected** if there is a path b/w any two of its nodes.

A Graph G is said to be labeled if its edges are assigned data.

An edge e is called a loop if it has identical endpoints

$e = [u, u]$

THINGS TO DO

MARCH '13

M	T	W	T	F	S	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

APRIL '13

M	T	W	T	F	S	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

Directed graph :- It is also called digraph. or graph in which each edge e in G is assigned a direction.

Outdegree of a node u in G written as outdeg(u), is the number of edges beginning at u.

Indegree of u written as indeg(u) is the number of edges ending at u.

A node u is called a source if it has a positive outdegree but zero indegree.

u is called sink if it has a zero outdegree but a positive indegree.

Sequential Representation of Graphs :-

• Adjacency matrix :-

G is a simple directed graph with m nodes suppose the nodes of G have been ordered (directions are given) and are called $V_1, V_2 - - V_m$

Then the adjacency matrix $A = (a_{ij})$ of the graph G is the m x m matrix defined as follows :-

THINGS TO DO

IMPORTANT

WEEK-08 | 054-311

SATURDAY 23

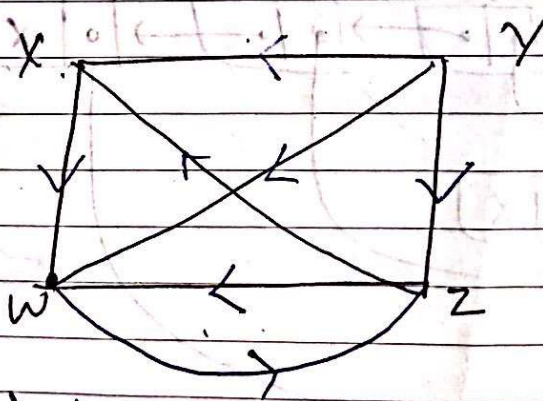
$a_{ij} = \begin{cases} 1 & \text{if } v_i \text{ is adjacent to } v_j \text{ that is if there is an edge } (v_i, v_j) \\ 0 & \text{otherwise.} \end{cases}$

Such a matrix A which contains entries of only 0 and 1 is called bit matrix or Boolean matrix

Path Matrix

Eg:-

$$A = \begin{array}{c} \\ X \\ Y \\ Z \\ W \end{array} \begin{array}{c} \begin{array}{cccc} X & Y & Z & W \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$



* Linked Representation of Graphs:- SUNDAY 24

Sequential Representation of Graphs has number of drawbacks :-

THINGS TO DO

① It may be difficult to insert and delete nodes in G.

MARCH '13

M	T	W	T	F	S	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

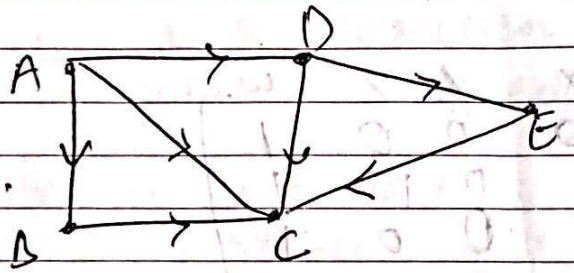
APRIL '13

M	T	W	T	F	S	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

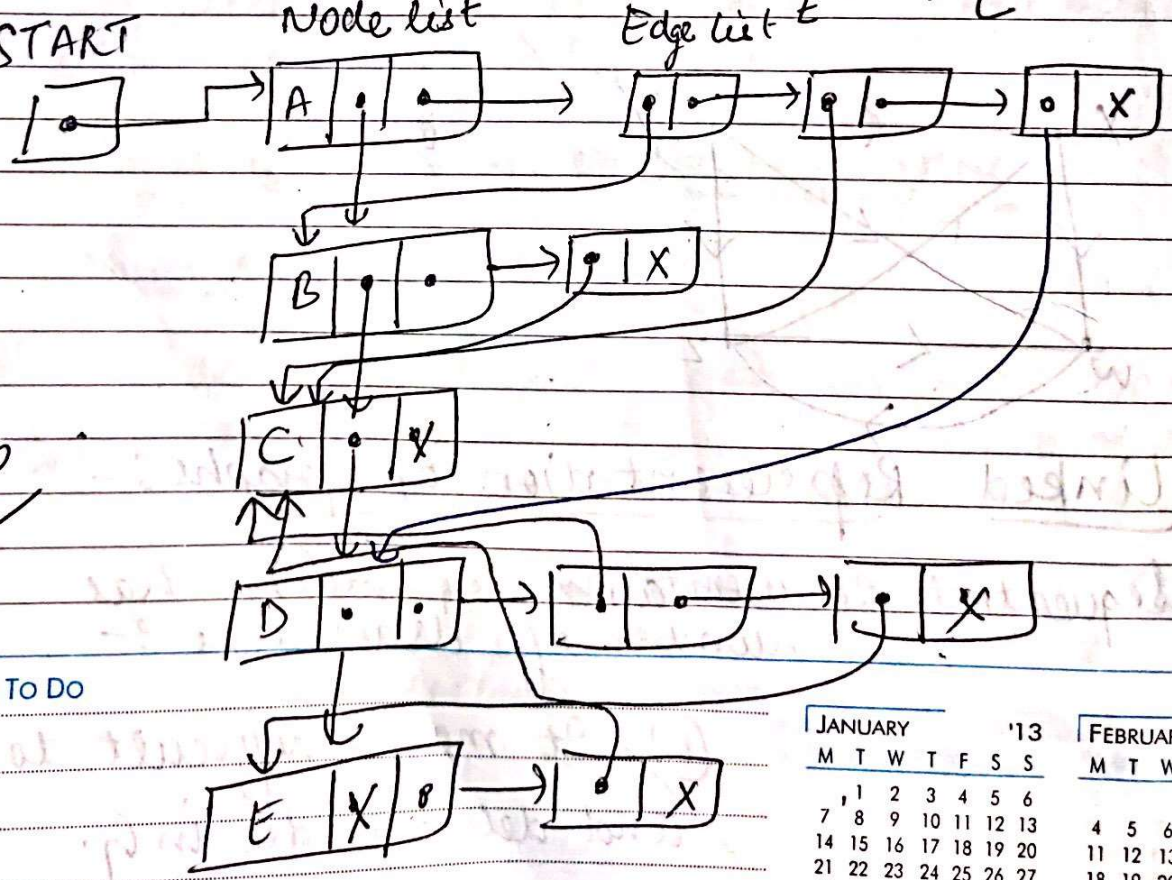
IMPORTANT

This is because the size of A may need to be changed and the nodes may need to be reordered, so there may be many changes in the matrix A.

Q is represented by linked representation also called an adjacency structure.



Node	Adjacency list
A	B, C, D
B	C
C	—
D	C, E
E	C



Pg 8.13

THINGS TO DO

* Traversing a Graph:-

Breadth-first search — use queue

Depth-first search — use stacks

During the execution of our algorithms, each node N of G will be in one of three states called the status of N, as follows.

STATUS = 1 (Ready state): The initial state of the node N.

STATUS = 2 (waiting state):- The node N is on the queue or stack, waiting to be processed.

STATUS = 3 (Processed state) The node N has been processed.

* Breadth-First Search:-

8021

Graphs

Breadth-first Search (BFS)

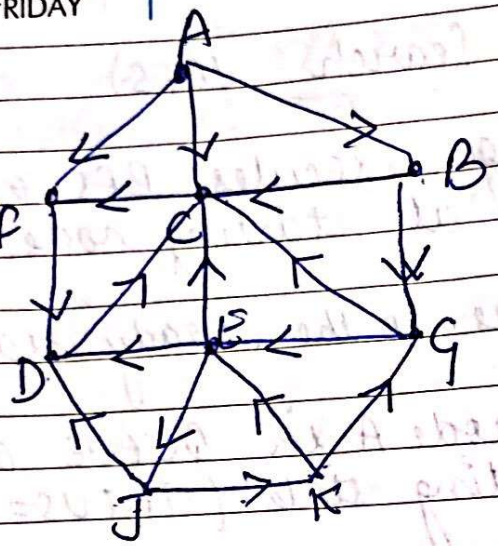
Algorithm : This algo executes BFS on a graph G beginning at starting node A.

1. Initialize all nodes to the ready state (STATUS = 1)
2. Put the starting node A in QUEUE and change its status to the waiting state (STATUS = 2)
3. Repeat steps 4 and 5 until QUEUE is empty:
4. Remove the front node N of QUEUE. Process N and change the status of N to the processed state (STATUS = 3)
5. Add to the rear of QUEUE all the neighbours of N that are in the ready state and change their status to the waiting state.

[End of step 3 loop]

6. Exit

eg



Adjacency list.

A: F, C, B
B: G, C
C: F
D: C
E: D, C, J
F: D
G: C, E
J: D, K
K: E, G

for Path A → J (A to J)

(1) Initially add A to QUEUE and add NULL to ORIG as follows:-

THINGS TO DO

F = 1 QUEUE = A
R = 1 ORIG = ϕ

(2) Remove the front element A from QUEUE by setting FRONT := FRONT + 1 and add to QUEUE the neighbors of A

F = 2 QUEUE = A, F, C, B
R = 4 ORIG = ϕ, A, A, A

(3) Remove the front element F from QUEUE by setting FRONT := FRONT + 1 and add to QUEUE the neighbors of F as follows :-

F = 3 QUEUE : A, F, C, B, D
R = 5 ORIG : ϕ, A, A, A, F

(4) Remove the front element C from QUEUE and add to QUEUE the neighbors of C (there are in ready state) as follows :-

F = 4 QUEUE : A, F, C, B, D
R = 5 ORIG : ϕ, A, A, A, F

(5) Remove the front element B from QUEUE and add to QUEUE the neighbors of B as follows :-

SUNDAY 28

F = 5 QUEUE : A, F, C, B, D, G
R = 6 ORIG : ϕ, A, A, A, F, B

MAY '13

M	T	W	T	F	S	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

JUNE '13

M	T	W	T	F	S	S
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

THINGS TO DO

Only G is added as C is already in QUEUE Not in ready state

MAY JUN

29 | 119-246 | WEEK-18 | MONDAY

IMPORTANT

(6) Remove the front element D from QUEUE and add to QUEUE the neighbors of D as follows:-

F=6 QUEUE: A, F, C, B, D, G
R=6 ORIG: Φ, A, A, A, F, B

(7) Remove the front element G from QUEUE and add to QUEUE the neighbors of G as follows:-

F=7 QUEUE: A, F, C, B, D, G, E
R=7 ORIG: Φ, A, A, A, F, B, G

(8) Remove the front element E from QUEUE and add to QUEUE the neighbors of E as follows:-

F=8 QUEUE: A, F, C, B, D, G, E, J
R=8 ORIG: Φ, A, A, A, F, B, G, E

Stop as soon as J is added to QUEUE since J is final destination.

J ← E ← G ← B ← A] Reqd Path.

THINGS TO DO

MARCH '13 M T W T F S S

APRIL '13 M T W T F S S

Depth-First Search

(DFS)

This algo executes a DFS on graph G beginning at a starting node A.

(1) Initialize all nodes to the ready state (STATUS = 1)

(2) Push the starting node A onto STACK and change its status to the waiting state (STATUS = 2)

(3) Repeat steps 4 and 5 until STACK is empty.

(4) Pop the top node N of STACK. Process N and change its status to the processed state (STATUS = 3)

(5) Push onto STACK all the neighbors of N that are still in the ready state (STATUS = 1) and change their status to the waiting state (STATUS = 2)

(End of Step 3 loop)

(6) Exit