Algorithm

d[s]=0

for each v= V-{s}

do d[v]=∞

for i= 1 to | v| - 1 do

for each edge (u, v) ∈ E do

if d[v] > d|u] + w(u, v) then

d[v] = d[u] + w(u, v)

π [v] = u

for each edge (u, v) ∈ E do

if d[v] > d[u] + w(u, v)

then report that a negative-weight cycle exists At the end,

n-1 edges shortest

1 2 3 4

(DE) (CD) (BC) (AB)

X

AB

d[B] > d[A] + w(A, B)

d[C] > d[B] + w(B, C)

∞ > 1 + 1

Relax

d[u] = 5

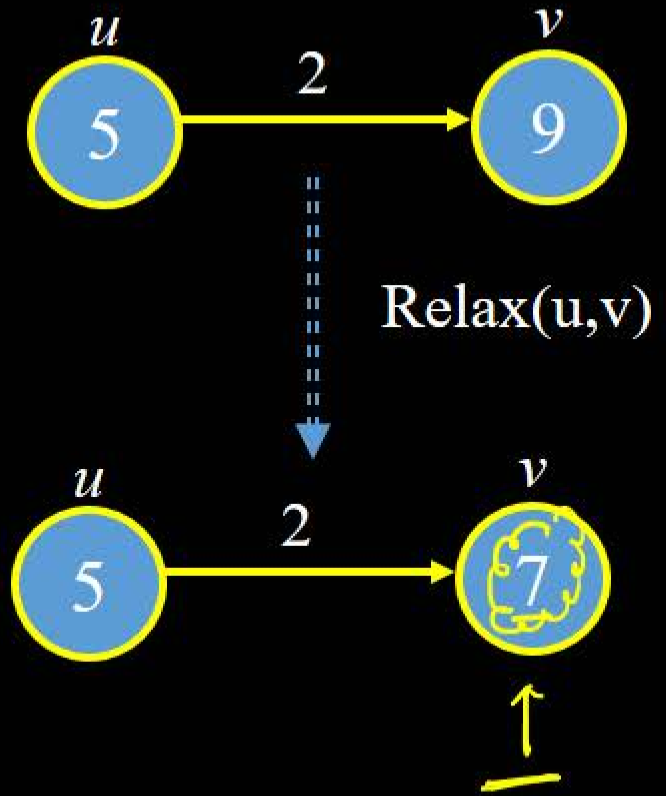
d[v] = 9

(u,v)

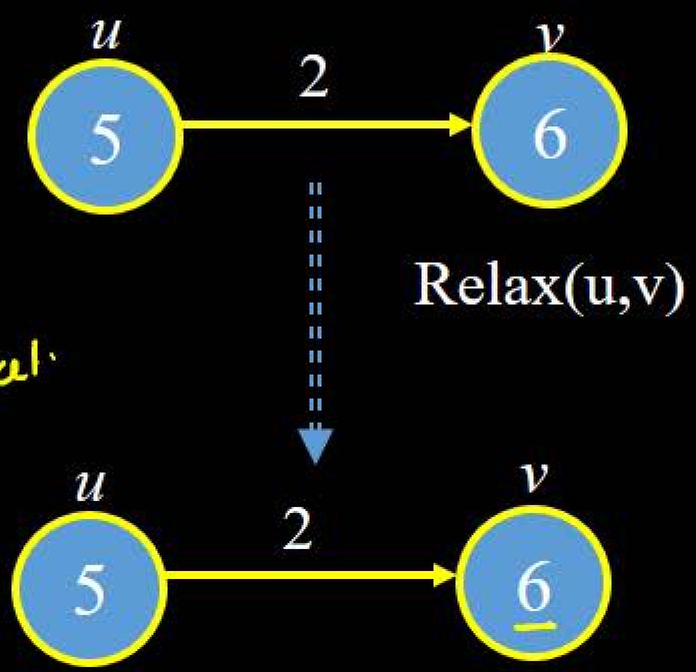
d[v] > d[u] + w(u,v)

9 > 5+2

↑ shorter pat.



Clear or Not



d[u] = 5

w(u,v) = 2

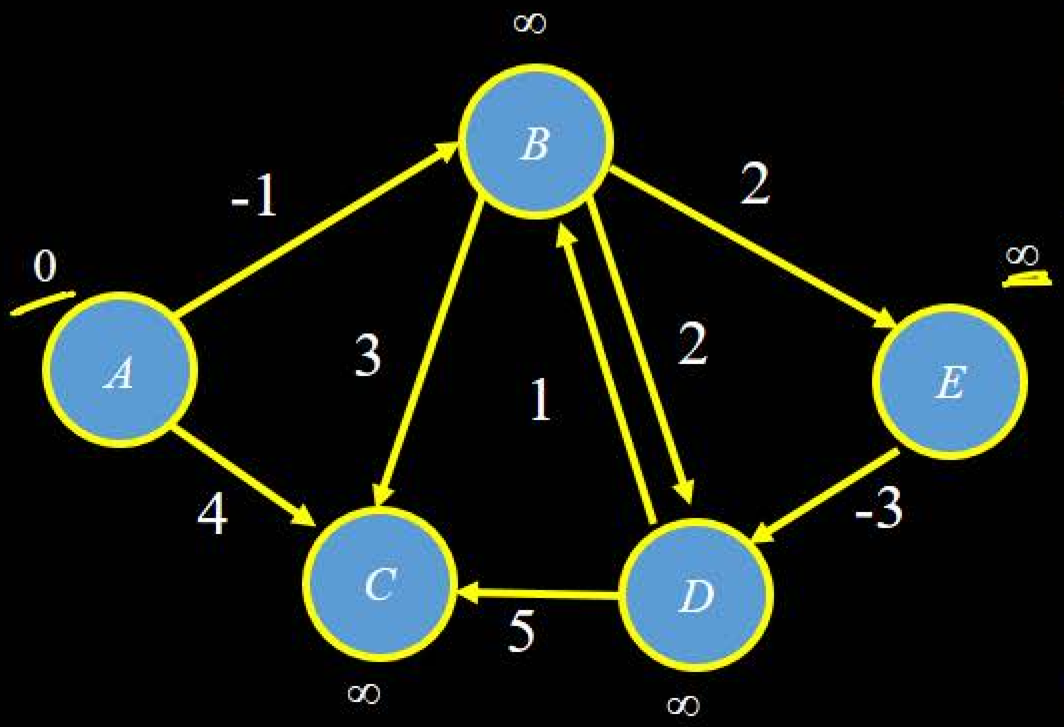
d[v] > d[u] + w(u,v)

6 > 5+2 = 6>7

No

Example

Order of the edges (B, E), (DB), (BD), (AB), (AC), (DC), (BC), (ED)



A	B	C	D	E
0	∞	∞	∞	∞

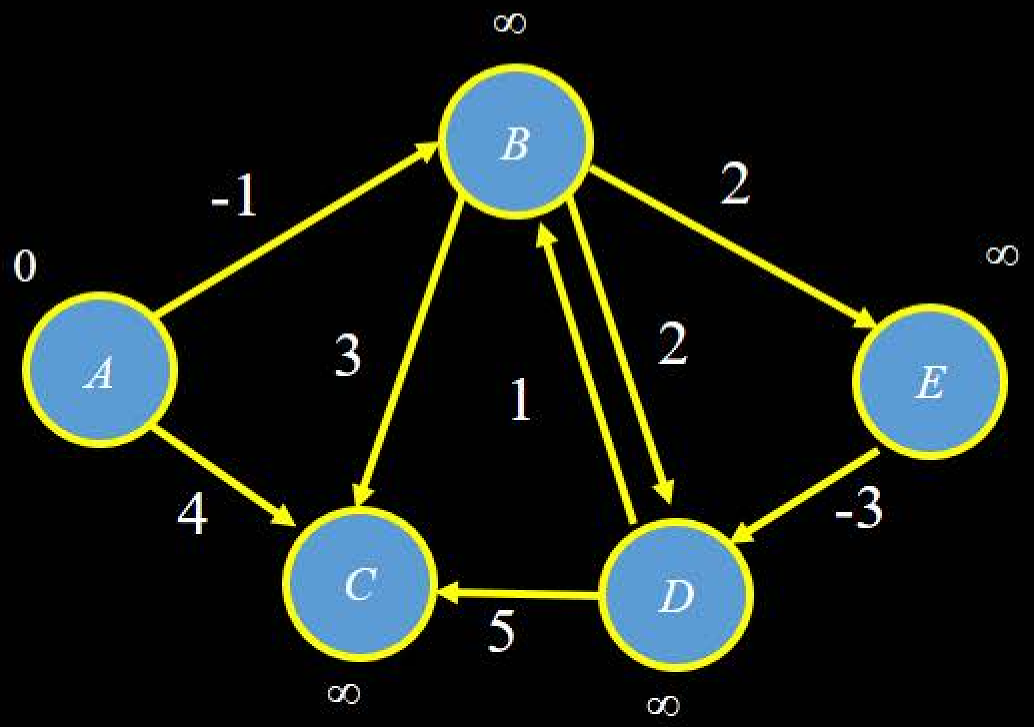
$\boxed{\infty > \infty + 2}$

(DB) $\infty > \infty + 1$

(BD) $\infty > \underset{B}{\infty} + 2$

Edge (B,E) , (D,B) ,(BD)

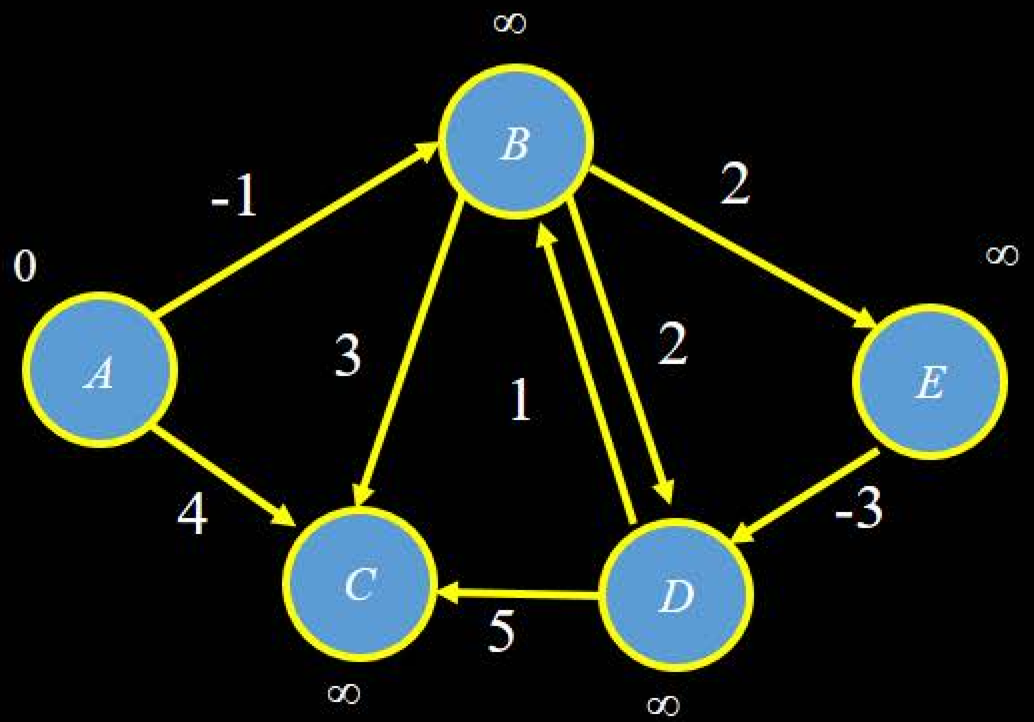
Order of the edges (B, E), (DB), (BD), (AB), (AC), (DC), (BC), (ED)



A	B	C	D	E
0	∞	∞	∞	∞

Edge (AB)

Order of the edges (B, E), (DB), (BD), (AB), (AC), (DC), (BC), (ED)



A	B	C	D	E
0	∞	∞	∞	∞

(A B)

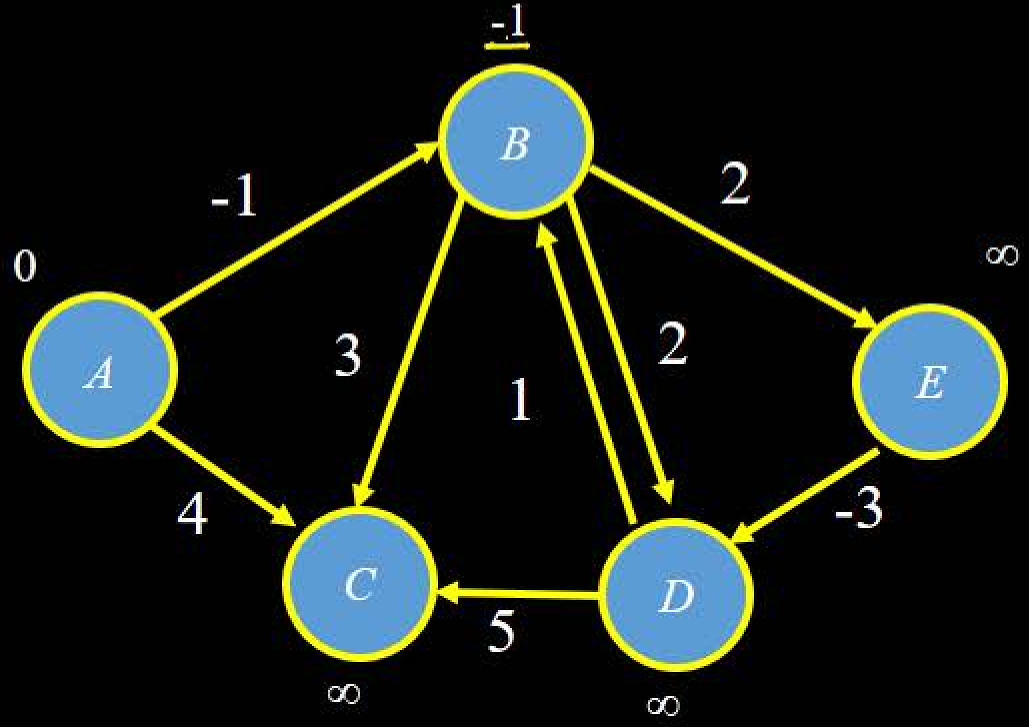
$d[B] > d[A] + w(A, B)$

$\infty > 0 + (-1)$

$\infty > -1$

Edge (AB)

Order of the edges (B,E), (DB), (BD), (AB), (AC), (DC), (BC), (ED)



A	B	C	D	E
0	-1	∞	∞	∞

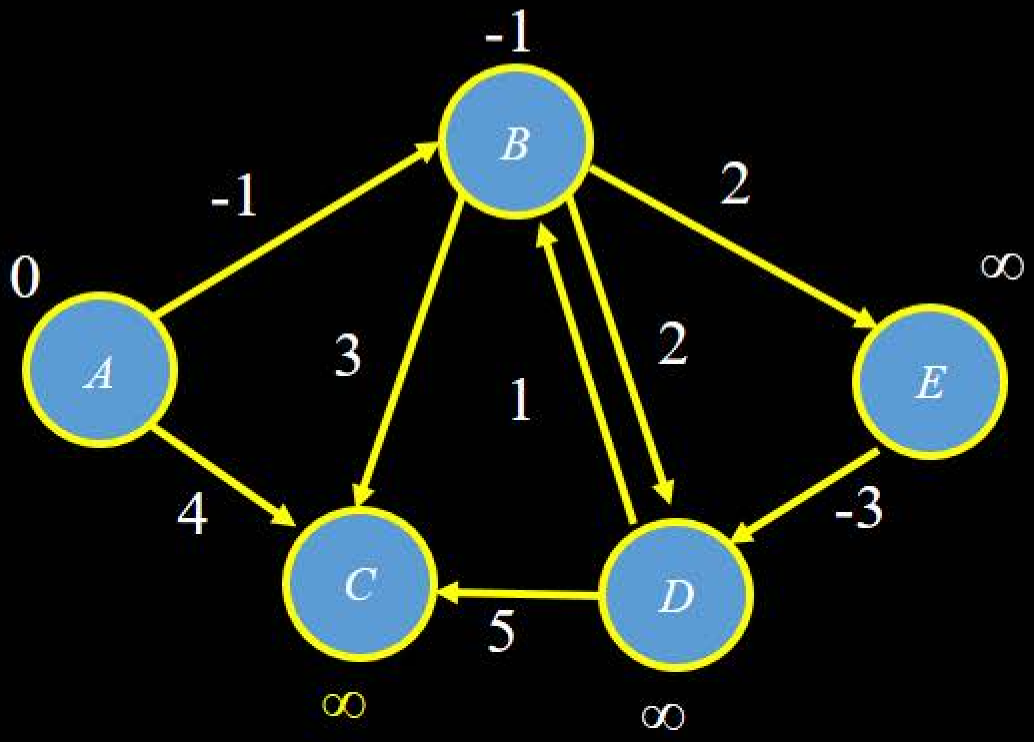
(AC)

$d[c] > d[A] + w(a,c)$

$\infty > 0 + 4$

Edge (A C)

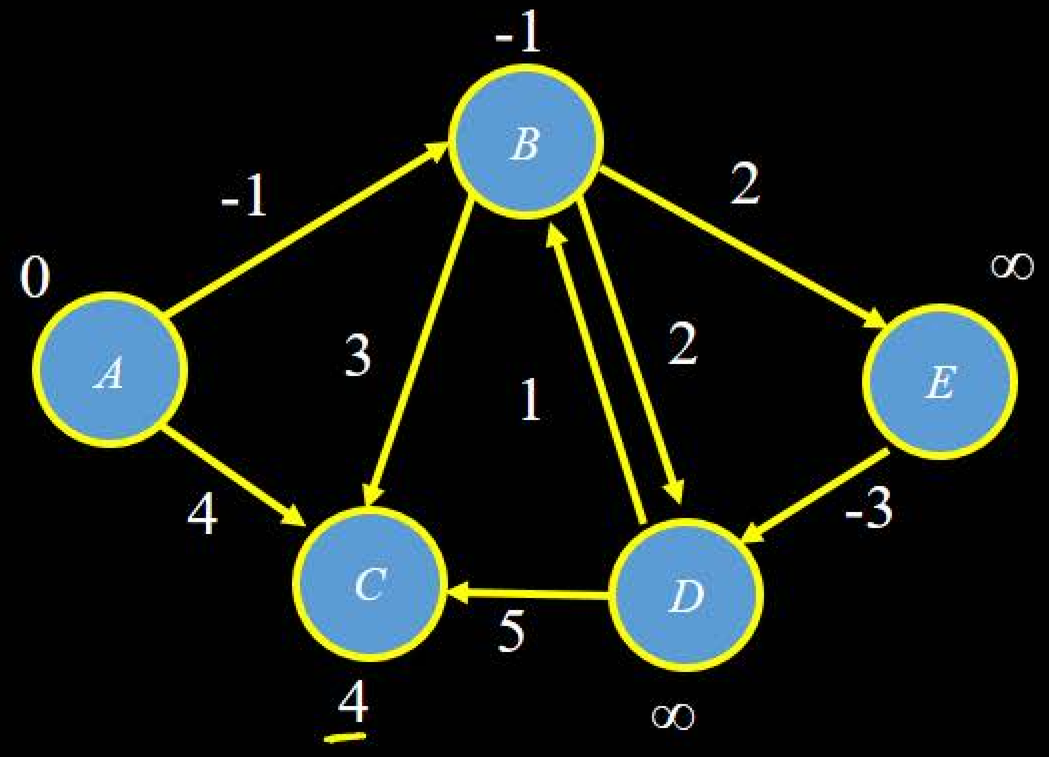
Order of the edges (B, E), (DB), (BD), (AB), (AC), (DC), (BC), (ED)



A	B	C	D	E
0	-1	∞	∞	∞

Edge (A C)

Order of the edges (B, E), (DB), (BD), (AB), (AC), (DC), (BC), (ED)

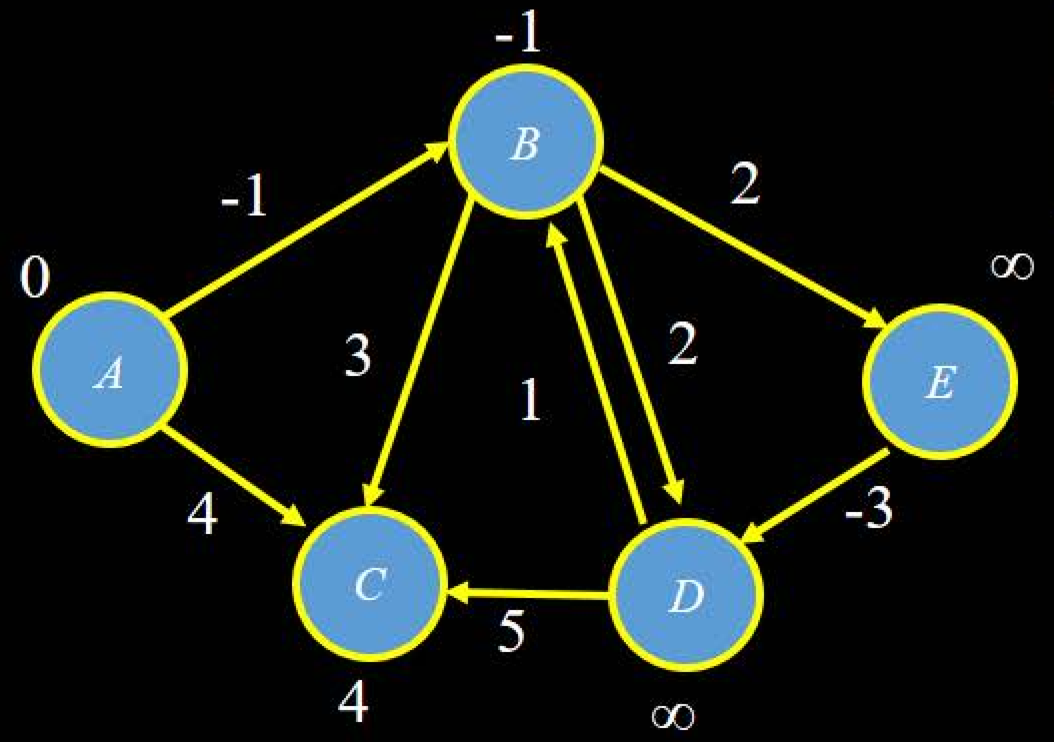


A	B	C	D	E
0	-1	4	∞	∞

d shortest path consists of 1 edge

Edge (D C)

Order of the edges (B, E), (DB), (BD), (AB), (AC), (DC), (BC), (ED)



A	B	C	D	E
0	-1	4	∞	∞

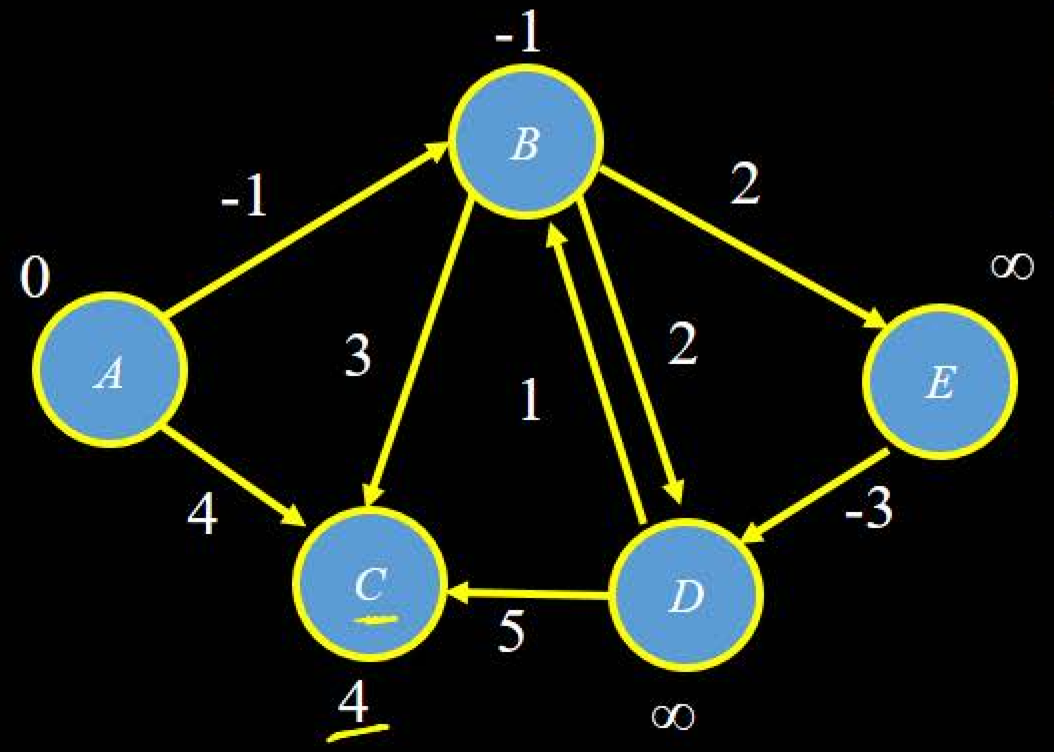
(D C)

$d[C] > d[D] + w(D, C]$

$4 > \infty + 5$

Edge (B C)

Order of the edges (B, E), (DB), (BD), (AB), (AC), (DC), $\underline{(BC)}$, (ED)



A	B	C	D	E
0	-1	4	∞	∞

(BC)

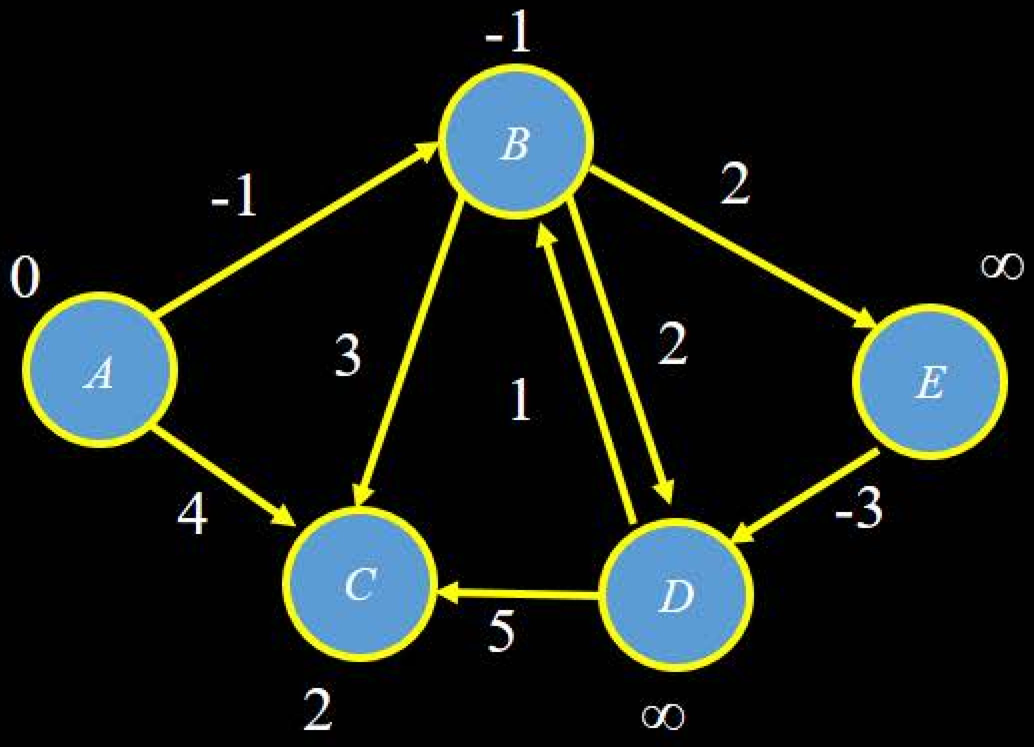
$d[C] > d[B] + w(B,C)$

$4 > -1 + 3$

$4 > 2$

Edge (B C)

Order of the edges (B, E), (DB), (BD), (AB), (AC), (DC), $\underline{(BC)}$, $(\underline{ED})$



A	B	C	D	E
0	-1	$\underline{2}$	∞	∞

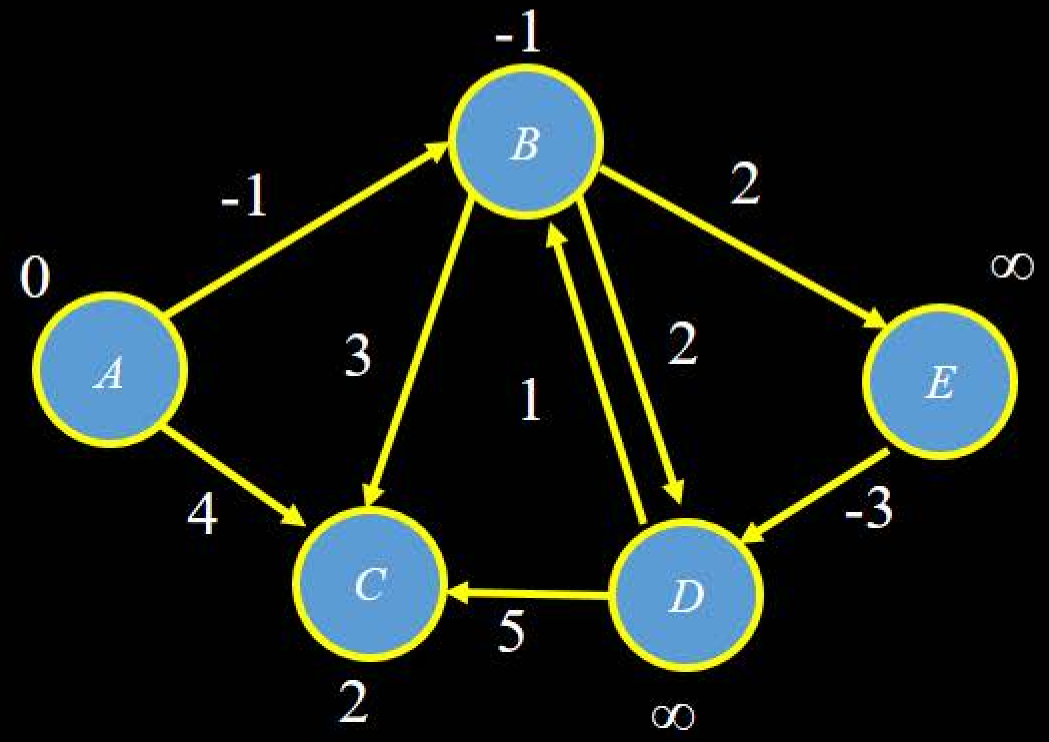
ED

$\underline{D[D]}\ \{\boxed{\infty > \infty - 2}\}$

One pass Complete

Second Pass Edge (B E)

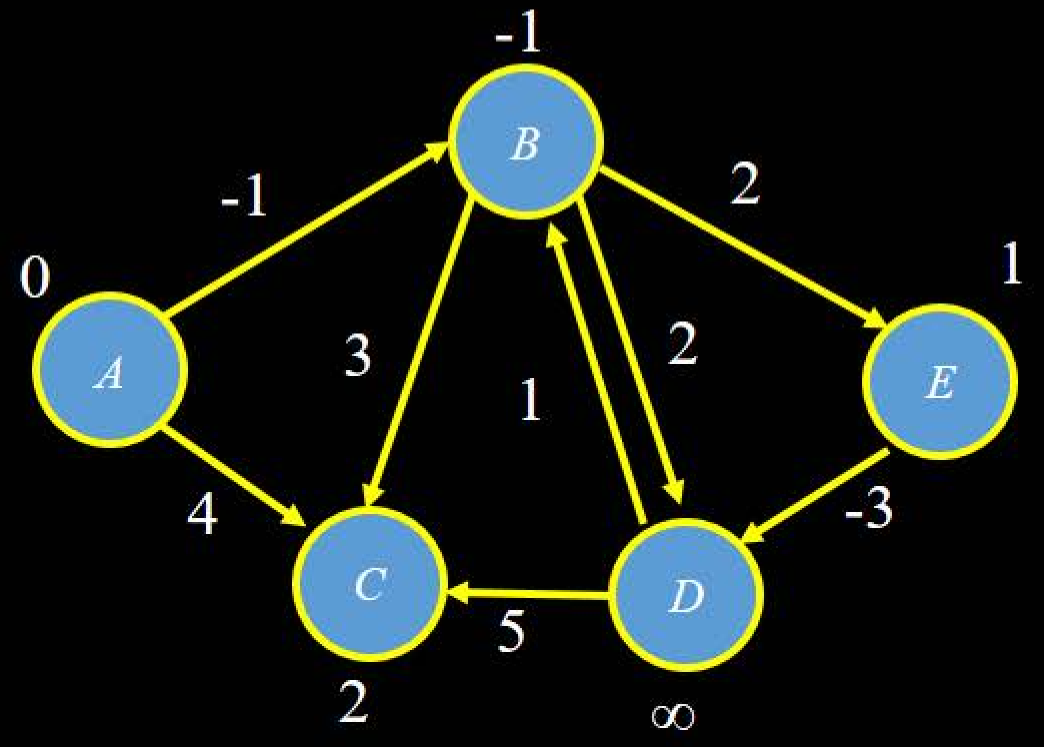
Order of the edges (B, E), (DB), (BD), (AB), (AC), (DC), (BC), (ED)



A	B	C	D	E
0	-1	2	∞	∞

Second Pass Edge (B E)

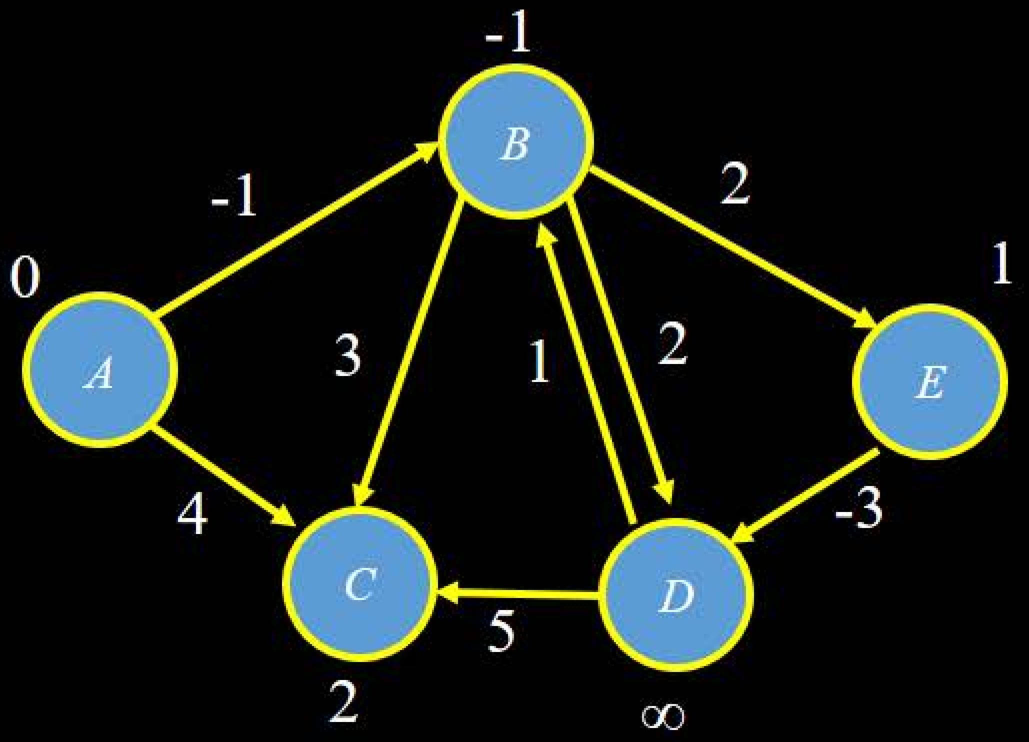
Order of the edges (B, E), (DB), (BD), (AB), (AC), (DC), (BC), (ED)



A	B	C	D	E
0	-1	2	∞	1

Second Pass Edge (D B) (B D)

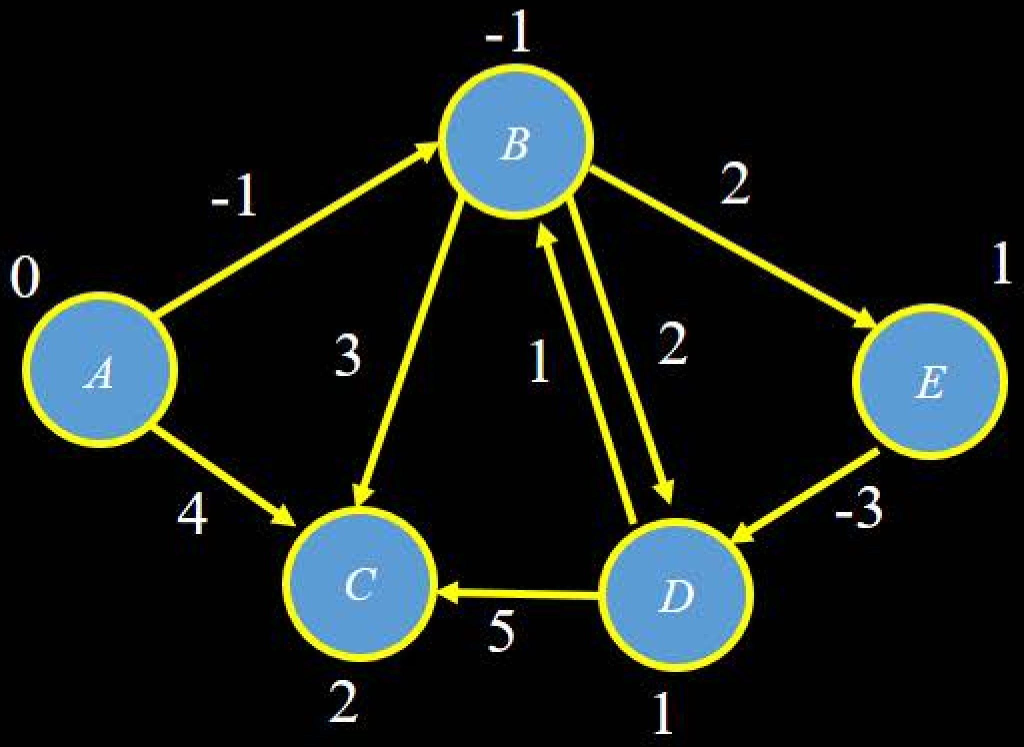
Order of the edges (B, E), (DB), (BD), (AB), (AC), (DC), (BC), (ED)



A	B	C	D	E
0	-1	2	∞	1

Second Pass Edge (D B) (B D)

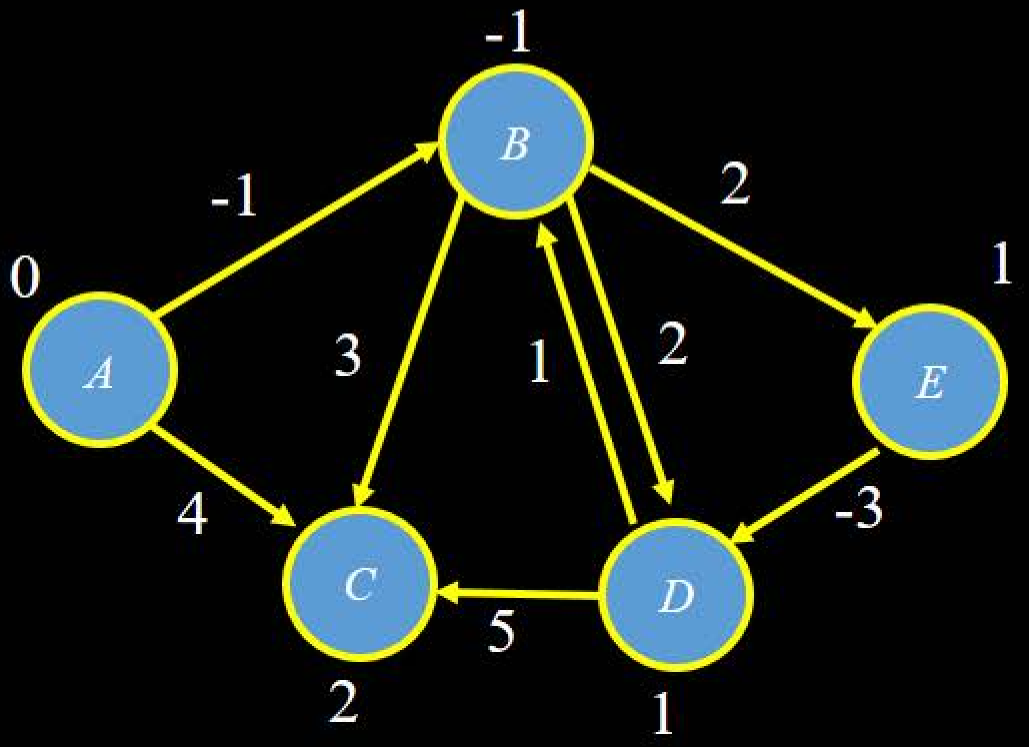
Order of the edges (B, E), (DB), (BD), (AB), (AC), (DC), (BC), (ED)



A	B	C	D	E
0	-1	2	∞	1

Second Pass Edge (ED)

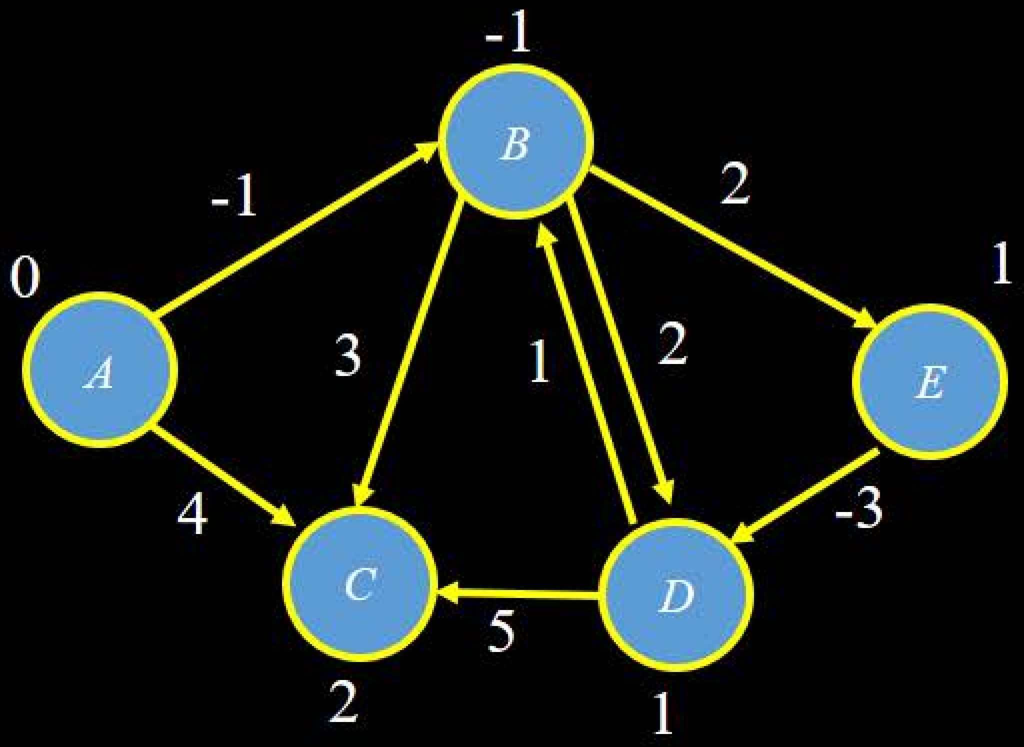
Order of the edges (B, E), (DB), (BD), (AB), (AC), (DC), (BC), (ED)



A	B	C	D	E
0	-1	2	1	1

Second Pass Edge (ED)

Order of the edges (B, E), (DB), (BD), (AB), (AC), (DC), (BC), (ED)



A	B	C	D	E
0	-1	2	-2	1

GATE 2008 | 2 Marks Question

The subset-sum problem is defined as follows. Given a set of n positive integers, $S = \{a_1, a_2, a_3, \ldots, a_n\}$, and positive integer W, is there a subset of S whose elements sum to W? A dynamic program for solving this problem uses a 2-dimensional Boolean array, X, with n rows and W+1 columns. $X[i, j], 1 \le i \le n, 0 \le j \le W$, is TRUE if and only if there is a subset of $\{a_1, a_2, \ldots, a_i\}$ whose elements sum to j.

Which of the following is valid for $2 \le i \le n$ and $a_i \le j \le W$?

(A) $X[i, j] = X[i-1, j] \vee X[i, j-a_i]$

(B) $X[i, j] = X[i-1, j] \vee X[i-1, j-a_i]$

(C) $X[i, j] = X[i-1, j] \wedge X[i, j-a_i]$

(D) $X[i, j] = X[i-1, j] \wedge X[i-1, j-a_i]$

GATE 2008 | 2 Marks Question

Which entry of the array X, if TRUE, implies that there is a subset whose elements sum to W?

(A) $X[1, W]$ (B) $X[n, 0]$ (C) $X[n, W]$ (D) $X[n-1, n]$

GATE 2021 Set-1 | 2 Marks Question

Define R_n to be the maximum amount earned by cutting a rod of length n meters into one or more pieces of integer length and selling them. For i> 0, let p[i] denote the selling price of a rod whose length is 1 metres. Consider the array of prices:

$$p[1]=1, p[2]=5, p[3]=8, p[4]=9, p[5]=10, p[6]=17, p[7]=18$$

Which of the following statements is/are correct about R_7 ?

(A) R_7 is achieved by three different solutions

(B) $R_7 = 19$

(C) R_7 cannot be achieved by a solution consisting of three pieces

(D) $\mathrm{R}_7 = 18$

Graph Traversal

Graph Traversal

The process of visiting all vertices

- Depth first Search
- Breadth first Search

Binary traversal

In order traversal

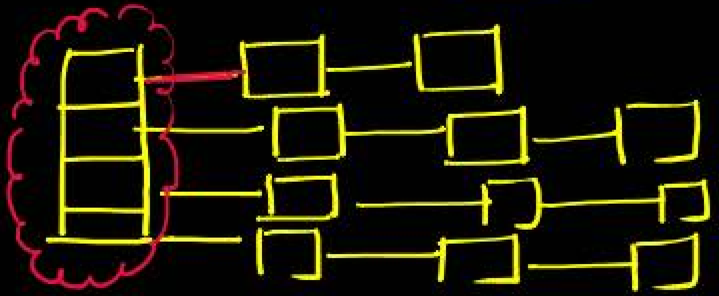
pre-order traversal

post-order traversal

G (V, E)

Space

1. Adjacency List — $\Theta(V+E)$
2. Adjacency matrix. — $|V|^2$ —



Length of the list equal to degree of vertices

& total Length of List = $\sum d_i = 2E$

Breadth First Search

data structure – Queue

FIFO

Array
Linked
stack
queue

Inserting ~~an~~ element 1, 2, 3
in order 1, 2, 3

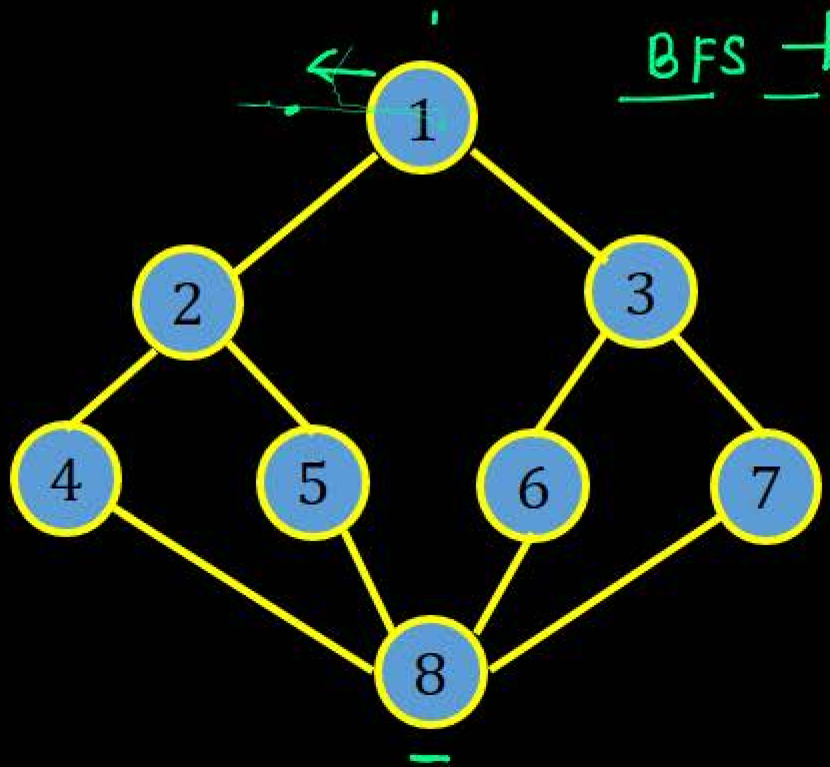
. Linear data structure

upop deletion – 3 LIFO stack

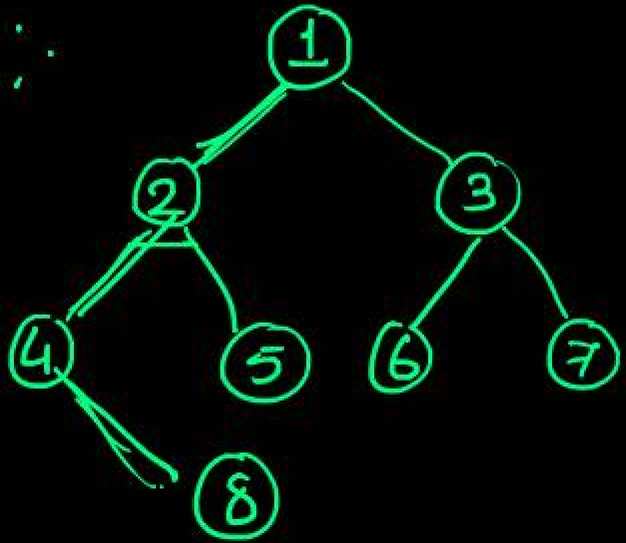
upon deletion – 1 ~~Quue~~ Queue

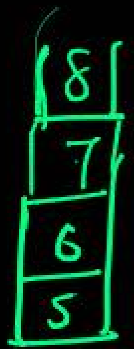
FIFO

Breadth First Search

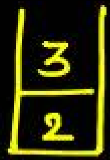


BFS tree:





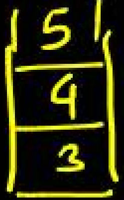
for



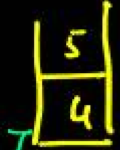
Queue

deletion from queue

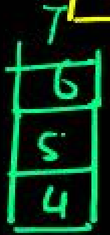
2



Insert 4,5



deletion - 3



Insert in queue

deletion - 4

Application of BFS

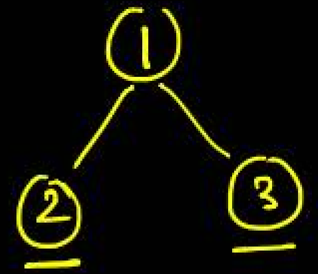
BFS is considered as Single Source shortest path for unweighted graph where cost is No. of edges between pair of vertices.

Breadth First Search

```
Algorithm BFS(v){
    u := v;
    visited[v] := 1;
    repeat {
        for all vertices w adjacent from u do
            if (visited[w] = 0) then {
                Add w to q;
                visited[w] := 1;
            }
        if q is empty then return;
        Delete u from q
    } until(false);
}
```

Adjacent vertex

visited is a global variable



All

~~adjca~~ adjacency list

$deg(u)$

$\sum deg(u) = 2E$

Every vertex adjacency list

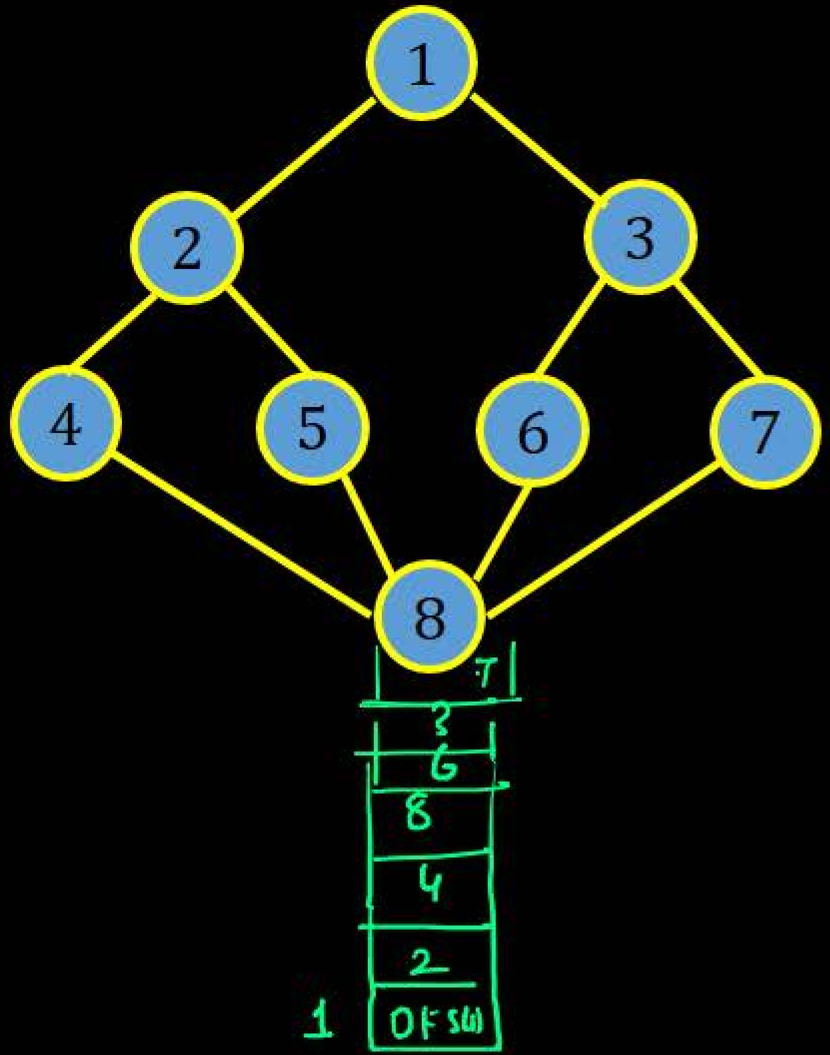
BFS complexity $= O(V+E)$

Cumulative

Depth First Search

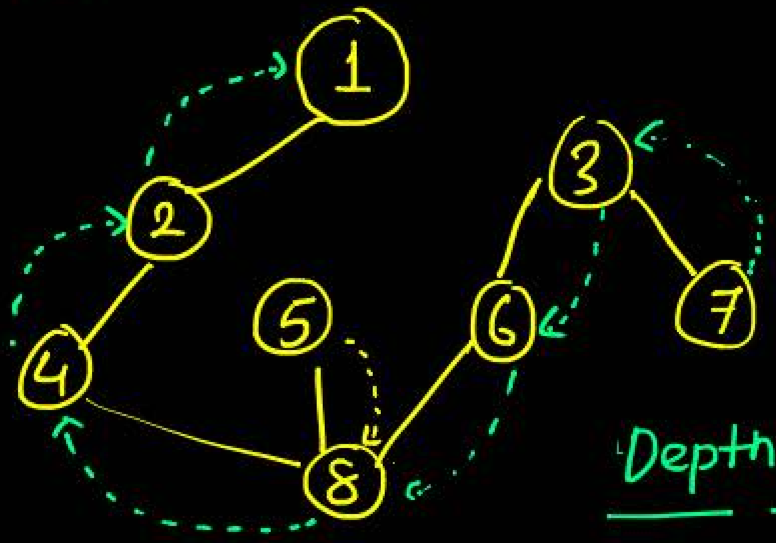
Given an undirected (directed) graph G = (V, E) with n vertices and an array visited [] initially set to zero, this algorithm visits all vertices reachable from v. G and visited[] are global.

Depth First Search



Exploration of a vertex = Visiting adjacent Node of the given vertex.

"Exploration of vertex is Suspended as soon as a New vertex is discovered"



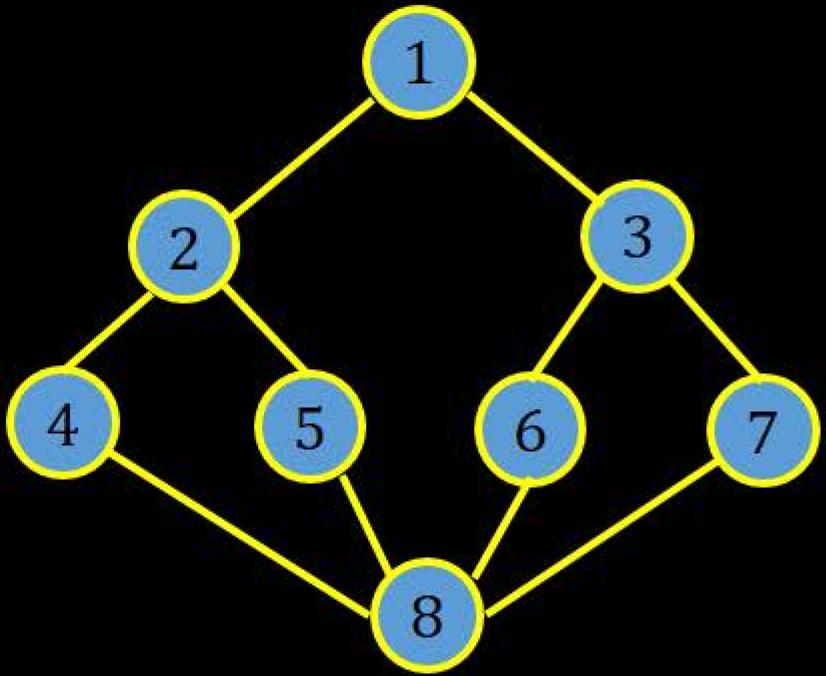
DFS $\leq$ tree

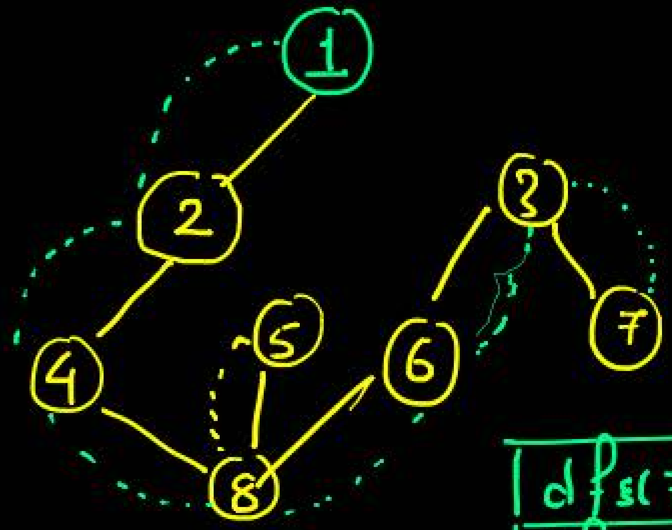
Recursion

Depth First Tree

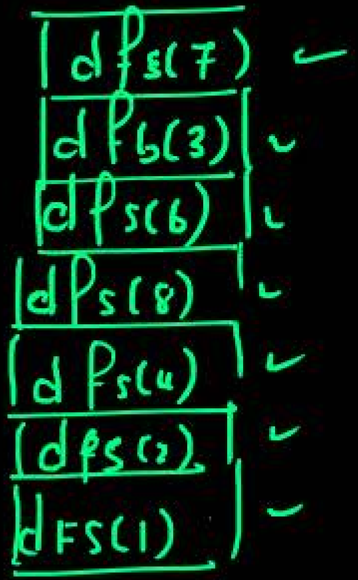
Depth First Search

```
Algorithm DFS(v) {
visited[v] := 1;
for each vertex w adjacent from v do {
if (visited[w] = 0) then DFS(w);
 }}
```





depth of Recursion = 7



GATE CSE 2014 Set 2 | Question: 14

Consider the tree arcs of a BFS traversal from a source node W in an unweighted, connected, undirected graph. The tree T formed by the tree arcs is a data structure for computing

a) the shortest path between every pair of vertices.

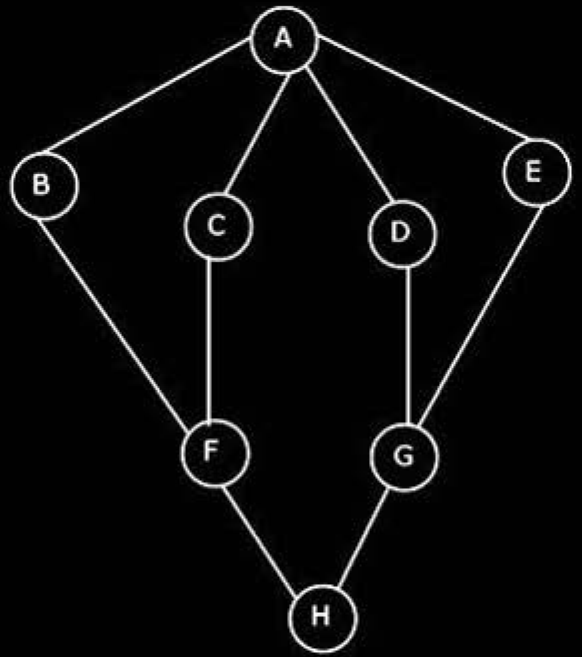
b) the shortest path from W to every vertex in the graph.

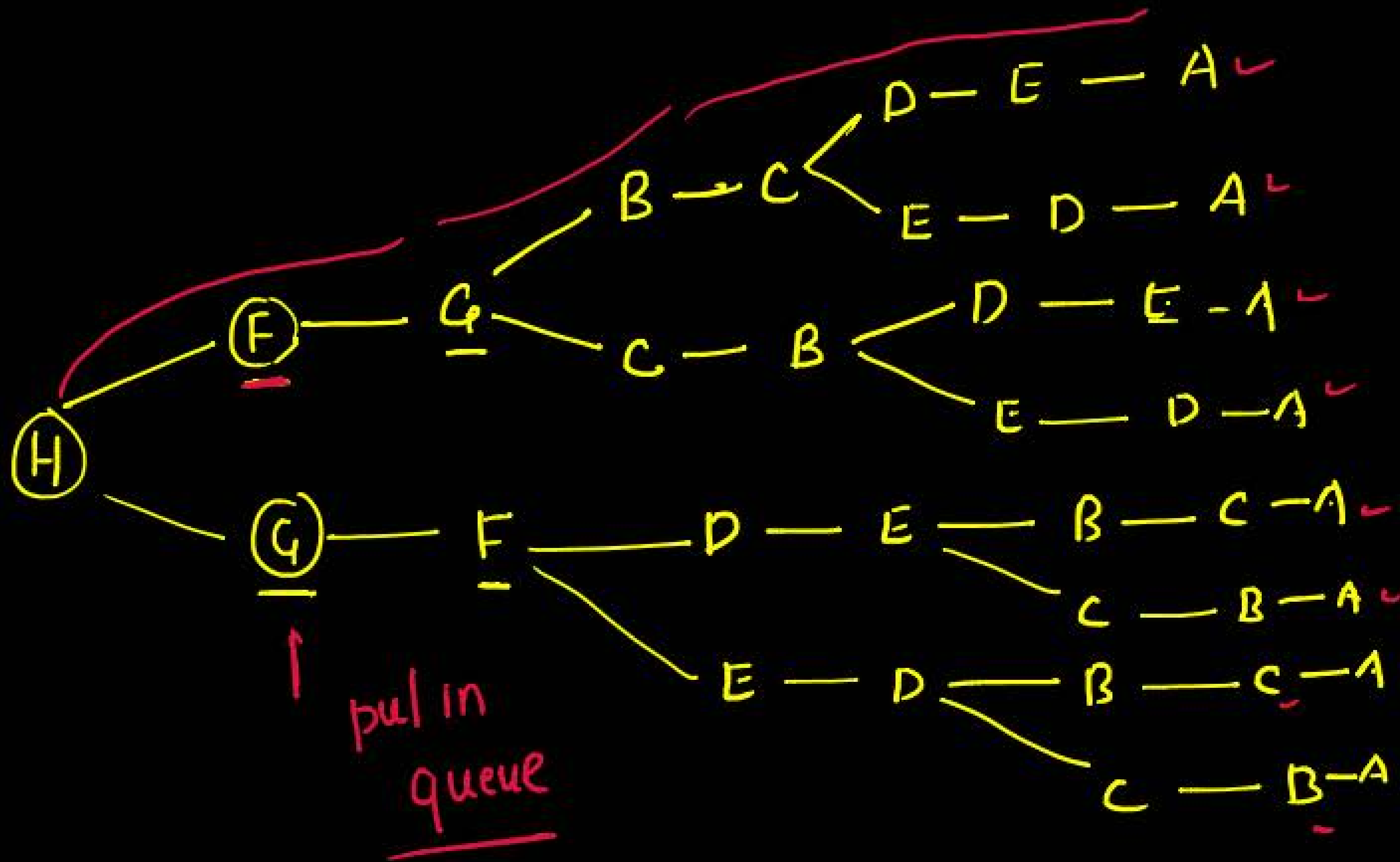
c) the shortest paths from W to only those nodes that are leaves of T.

d) the longest path in the graph.

Questions

Q. Consider the following graph





How many different breadth-first search traversals are possible considering H as a source vertex?

(A) 1 (B) 4 (C) 16 (D) 8

GATE 2001

Q. Consider an undirected unweighted graph G. Let a breadth-first traversal of G be done starting from a node r. Let d(r, u) and d(r, v) be the lengths of the shortest paths from r to u and respectively in G. If u is visited before v during the breadth-first traversal, which of the following statements is correct?

(a) $d(r,u) < d(r,v)$

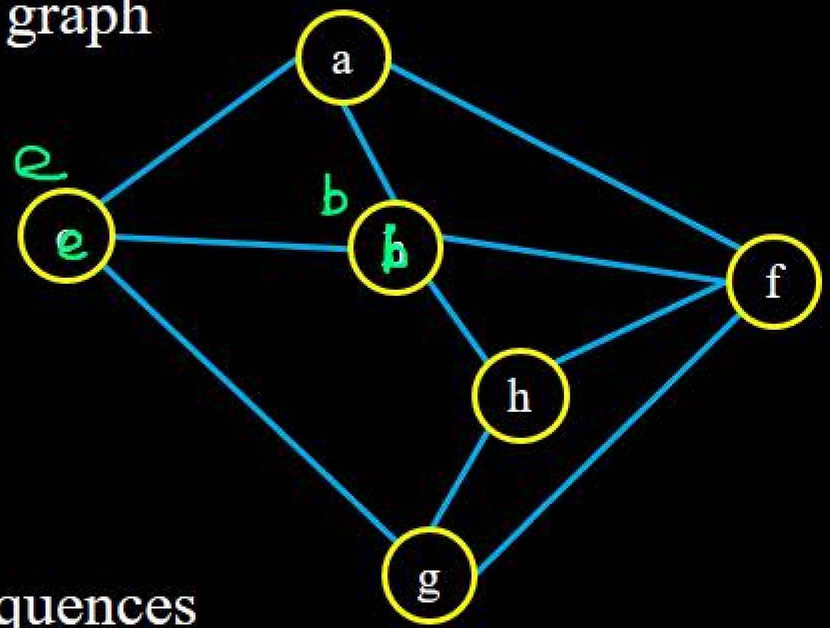
(b) $d(r,u) > d(r,v)$

(c) $d(r,u) < d(r,v)$

(d) None of the above

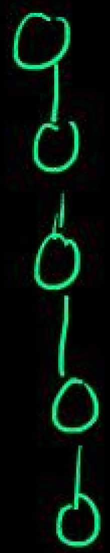
GATE 2003

Q. Consider the following graph



a b

(II) Not a DFS



Among the following sequences

I a b e g h f   II a b f e h g   III a b f h g e   IV a f g h b e

Which are depth first traversals of the above graph?

(A) I, II and IV only

(b) I and IV only

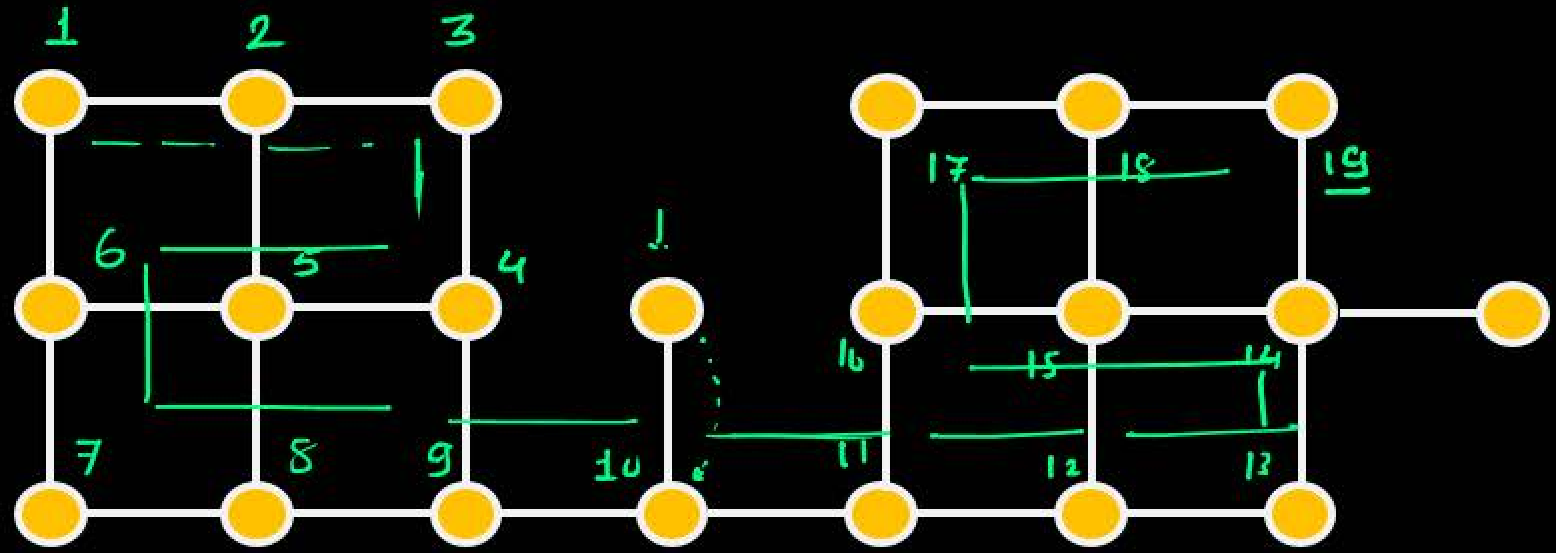
(C) II, III and IV only

(D) I, III and IV only

GATE 2014

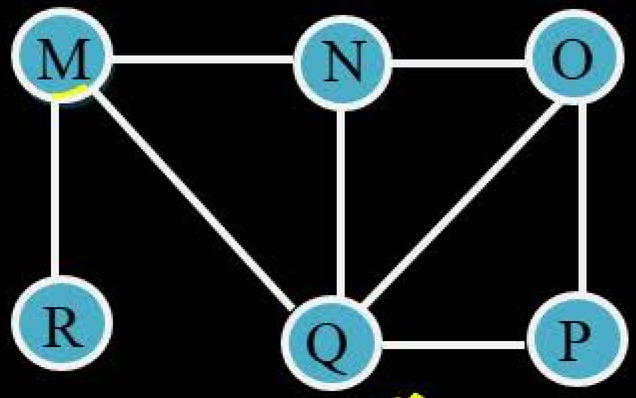
Suppose depth first search is executed on the graph below starting at some unknown vertex. Assume that a recursive call to visit a vertex is made only after first checking that the vertex has not been visited earlier. Then the maximum possible recursion depth (including the initial call) is ___19___.

Maximum depth of Recursion –



GATE 2017 Set-II

The Breadth First Search (BFS) algorithm has been implemented using the queue data structure. Which one of the following is a possible order of visiting the nodes in the graph below?



(A) MNOPQR (B) NQMPOR

(C) QMNROP (D) POQNMR

1

M – N

Sequential

one after another

(B) – NQMPOR

(C) QMNROP

POQNMR

PQO

GATE 2000

Q. Let G be an undirected graph. Consider a depth-first traversal of G and let T be the resulting depth-first search tree. Let u be a vertex in G and let ν be the first new (unvisited) vertex visited after visiting u in the traversal. Which of the following statements is always TRUE?

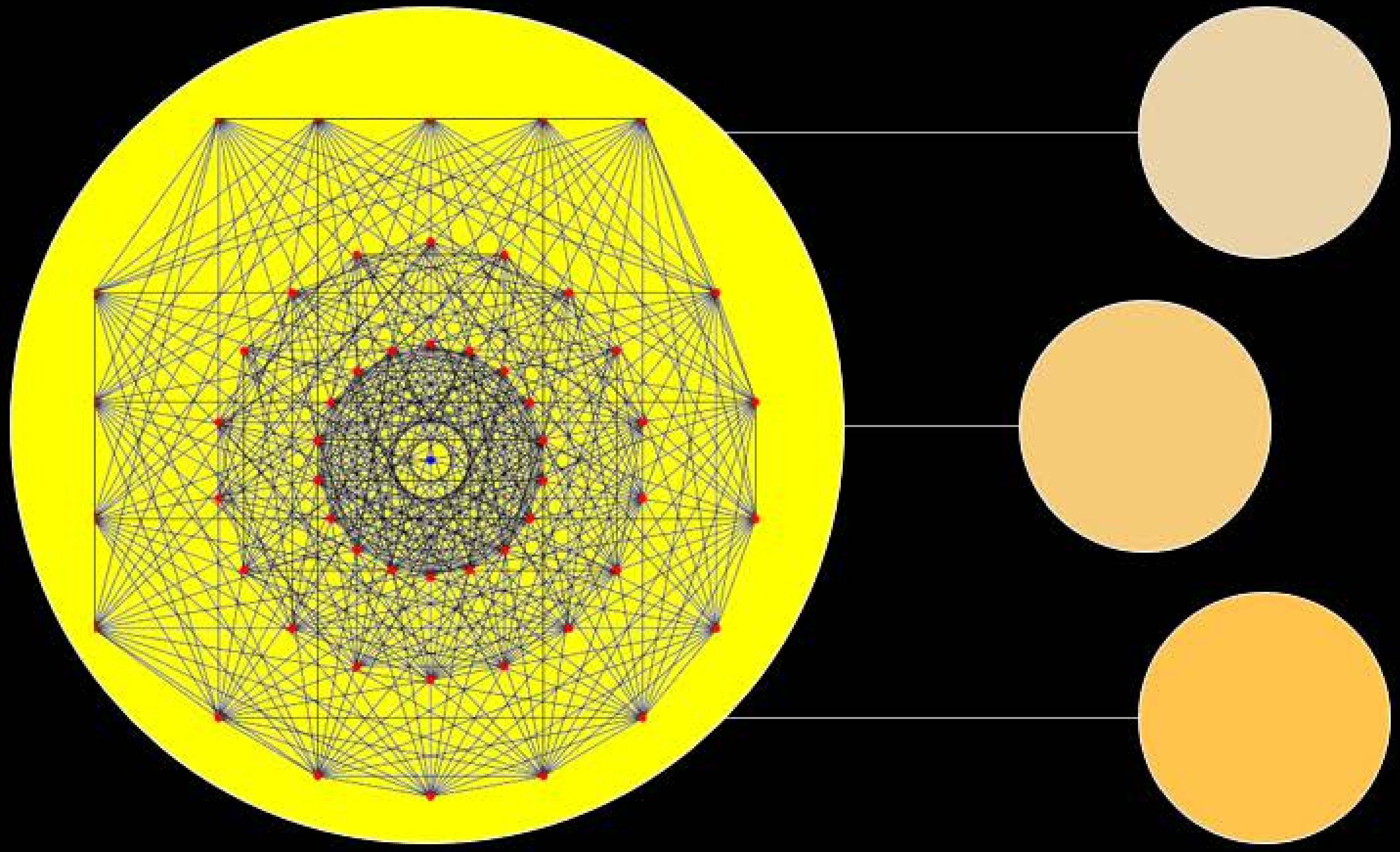
GATE 2000

Q. (a) {u,v} must be an edge in G, and u is a descendant of v in T

(b) {u,v} must be an edge in G, and v is a descendant of u in T

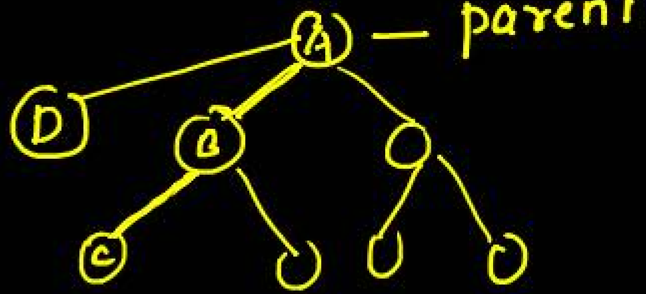
(c) If {u,v} is not an edge in G then u is a leaf in T

(d) If {u,v} is not an edge in G then u and v must have the same parent in T



ancestor

descendant



Siblings. children of same parent.

C is descendant of

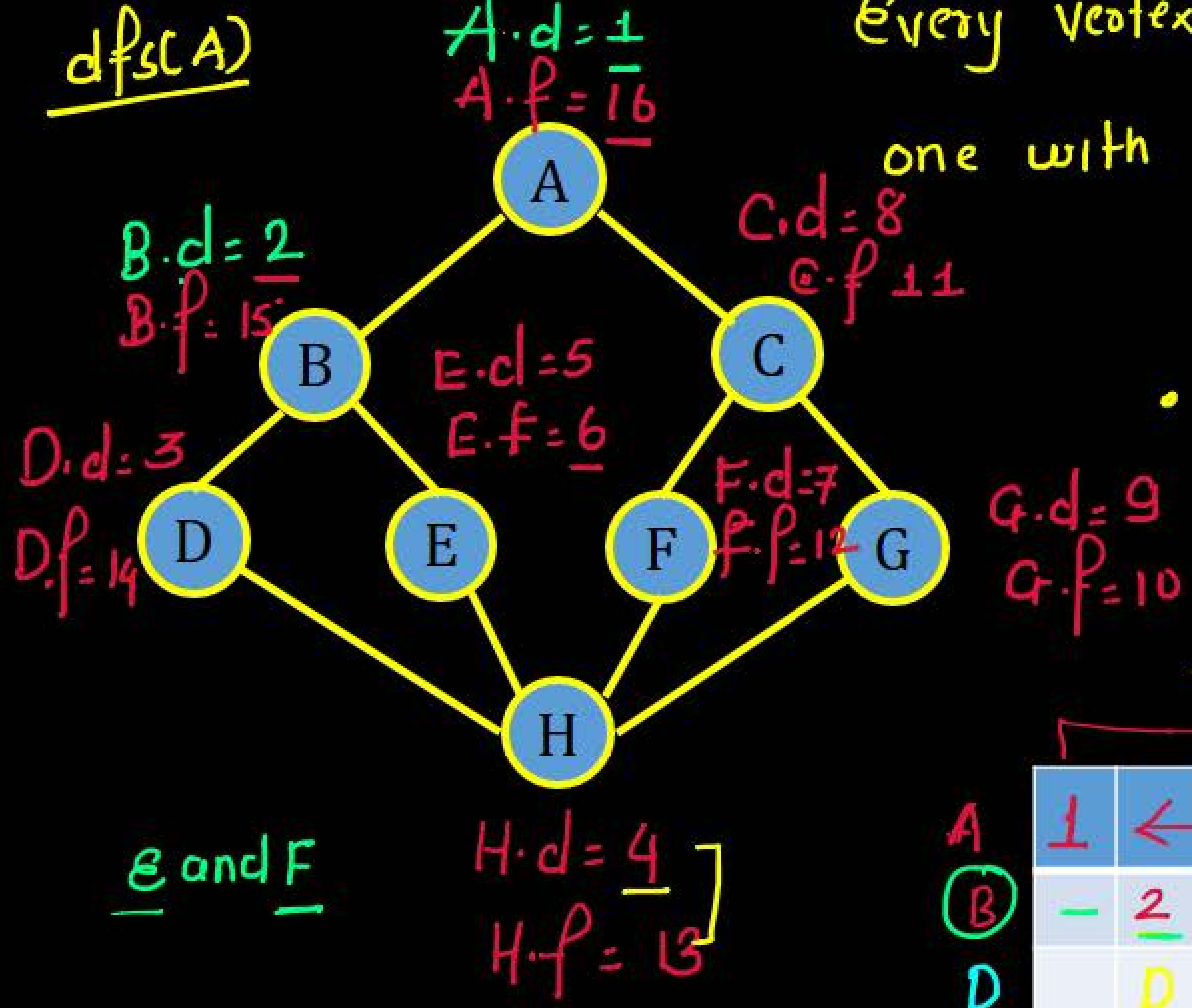
A

A is ancestor of C

Depth First Search (Directed Graph) With Time Stamp

Timestamps:

dfs(A)



E and F

Every vertex u two time stamp will be assigned

one with u.d – discovery time for u

when u mark as visited (DFS call starts)

- u.f when DFS call finishes

Each Next time stamp will one greater than previous time stamp.

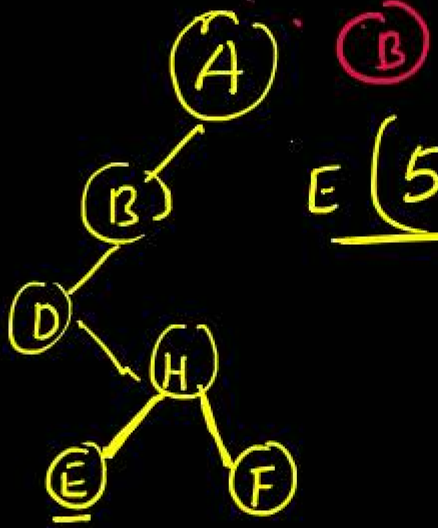
A	1															16
B	–	2													15	
D		D	3											14		
H				4									13			
				E	5	6	7					12				

A — d:6 — 30:f

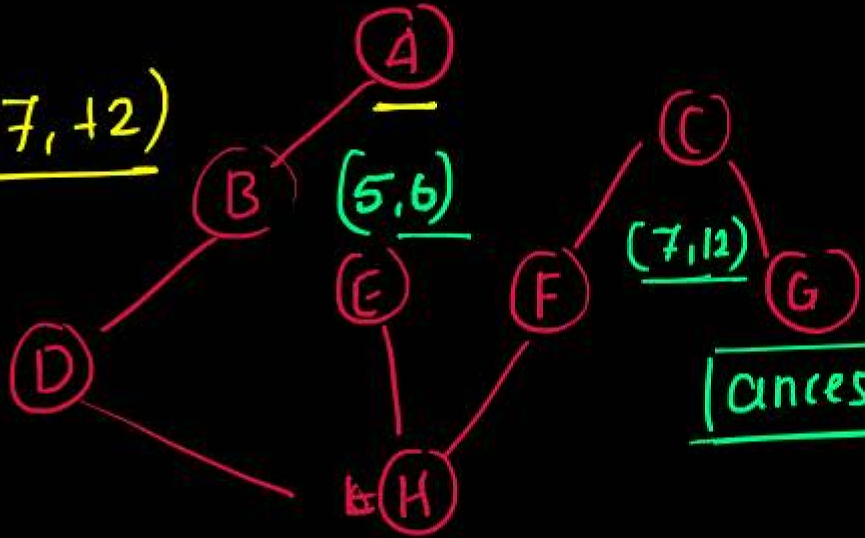
B — 8 - 10
d f

A is ancestor of B in DFS tree

Press Esc to show floating meeting controls



E (5-6) — F (7,12)



$\boxed{\text{ancestor}}$

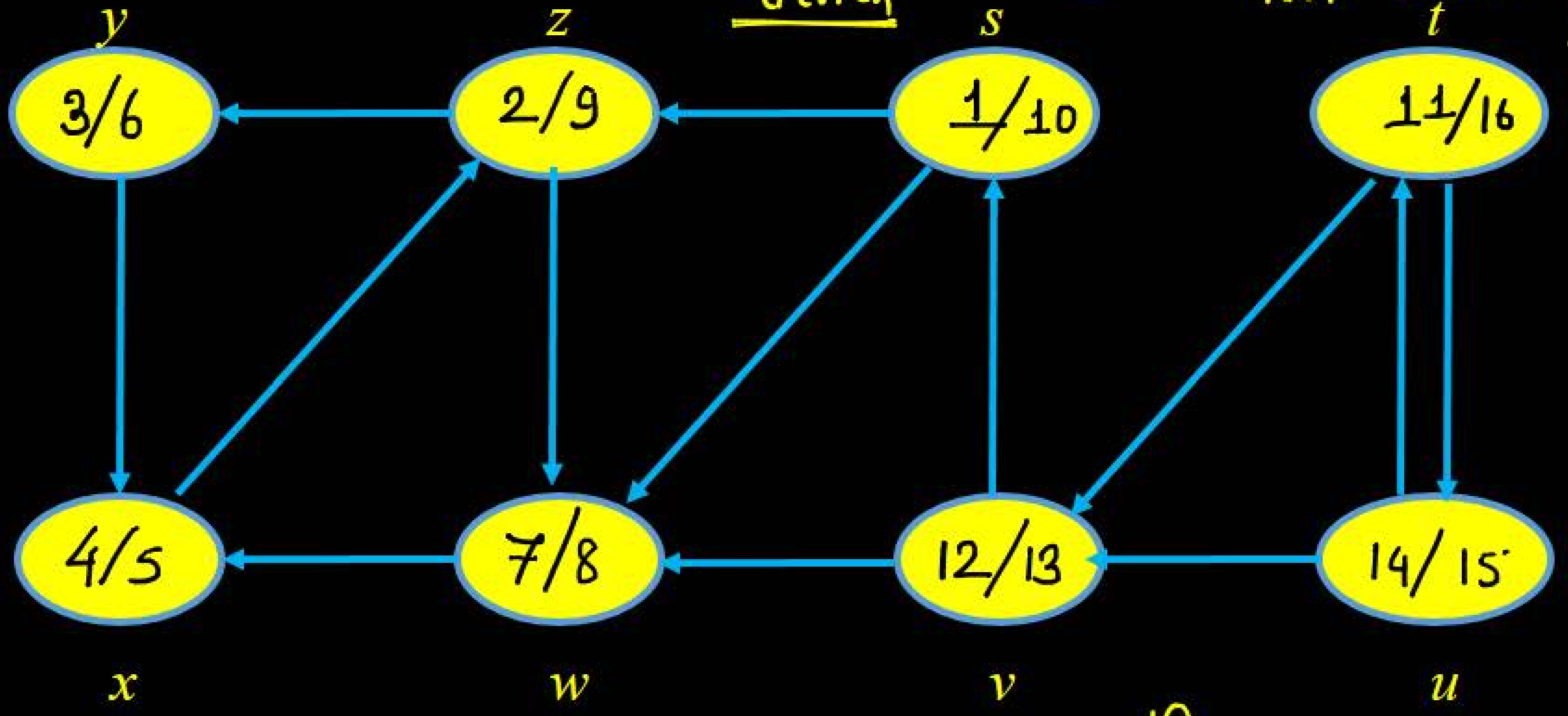
A is ancestor of all other vertex in DFS tree

Parenthesis Theorem (Directed Graph)

Not covered every vertex

One dfs

Not covered all vertices



~~f/d~~

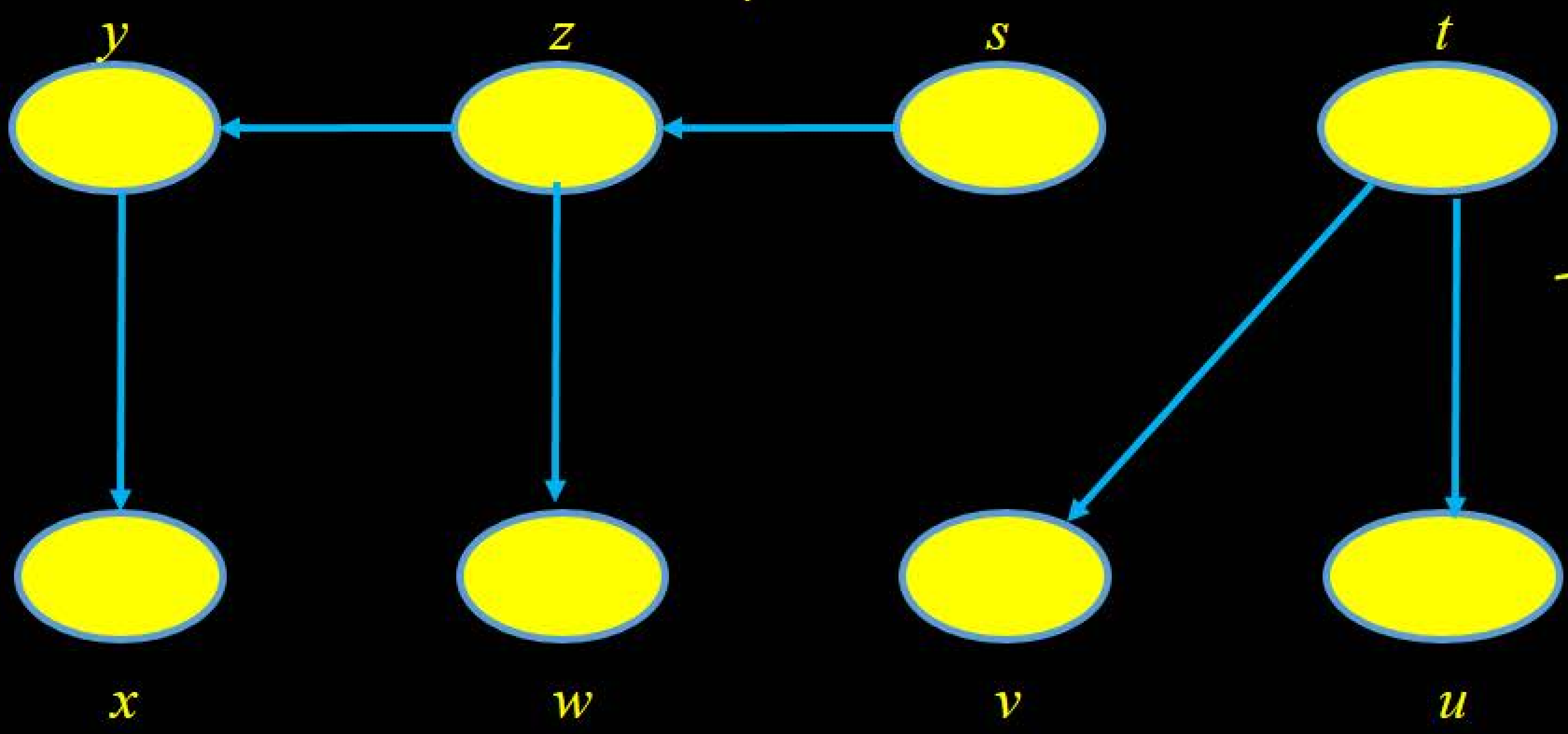
d/f

discovery/finish

one possible discovery/finish assignments

Parenthesis Theorem (Directed Graph)

Depth First tree



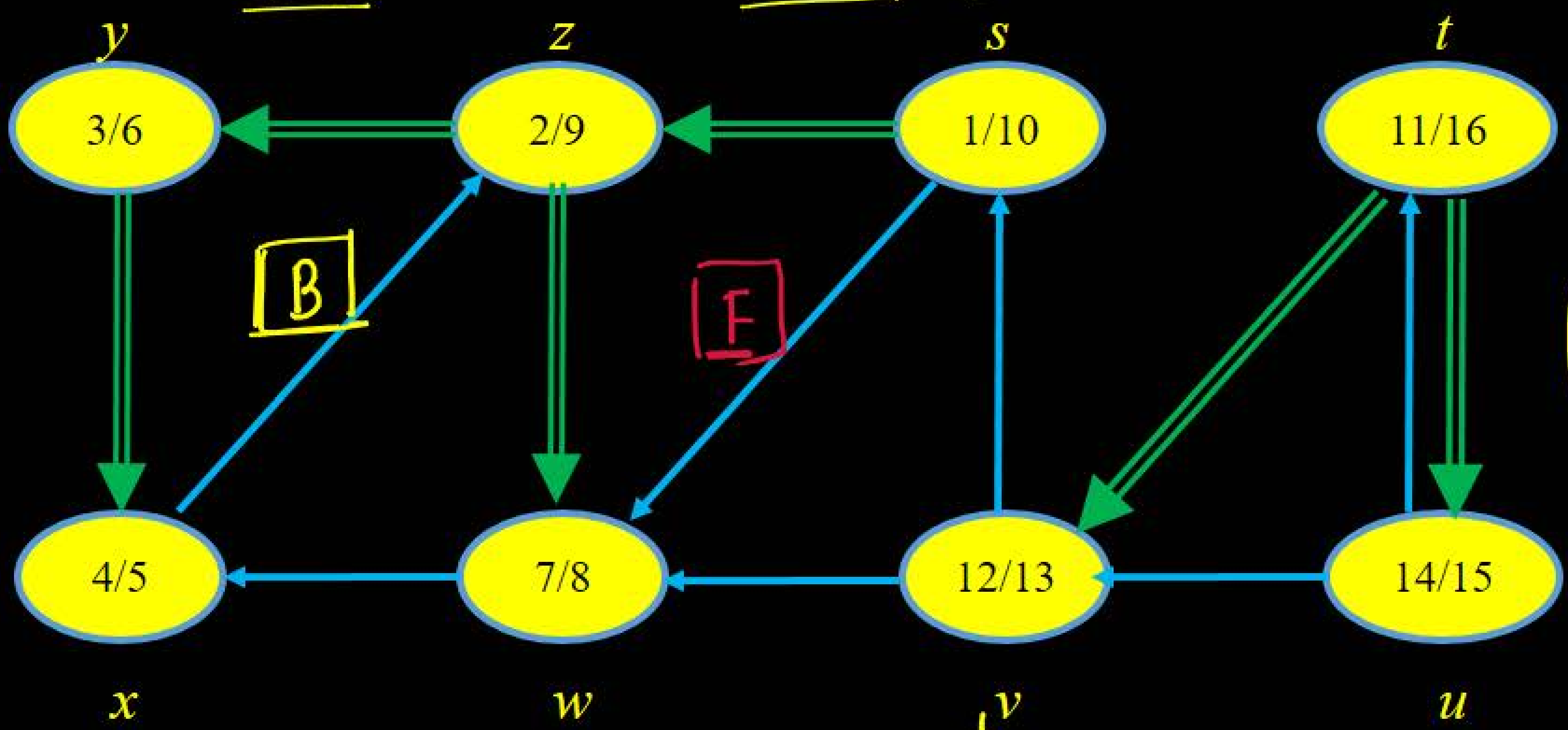
· tree edges

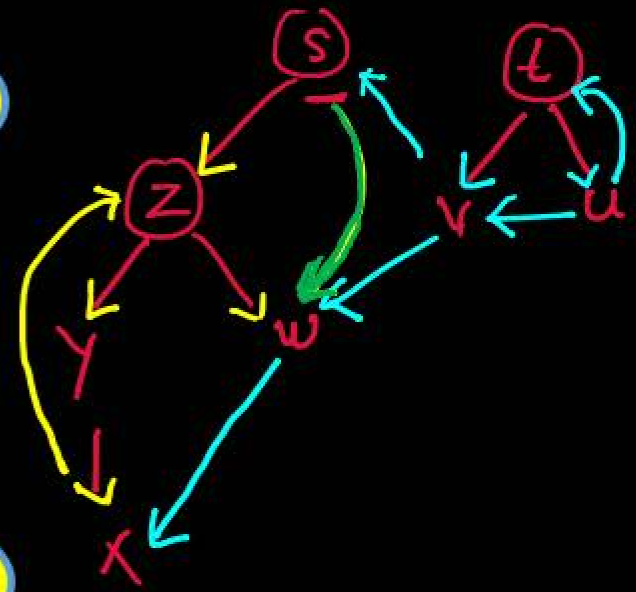
– the edges which is part of DFS tree called tree edge.

Parenthesis Theorem (Directed Graph)

NAI the

Some practice : Cross edge





* Backword edge immediate parent can be considered (u,T) backward

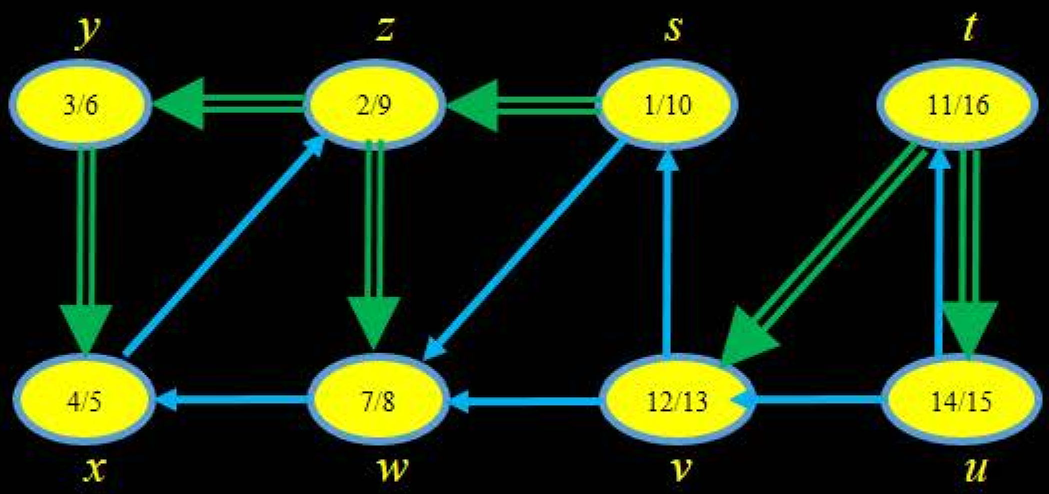
Parenthesis Theorem for Directed Graph

In any depth-first search of a (directed or undirected) graph $G = (V, E)$, for any two vertices u and v exactly one of the following three conditions holds:

DFS with directed graph

- if u.d and u.f are discovery and finish time for vertex u and v.d and v.f are discovery and finish time for v then
  - u.d and u.f entirely contained within v.d and v.f then v is ancestor of u in DFS tree.
  - u.d and u.f are entirely disjoint to v.d and v.f then neither u nor v is ancestor of each other

Parenthesis Theorem (Directed Graph)



GATE 2006-IT | 2-Marks, Question | Category-MCQ

Consider the depth-first-search of an undirected graph with 3 vertices P, Q, and R. Let discovery time $d(u)$ represent the time instant when the vertex u is first visited, and finish time $f(u)$ represent the time instant when the vertex u is last visited. Given that

~~diso~~ ~~diso~~ disconnected graph

P P

(5d) — (12P)

(6d) (10P)

$$d(P) = 5\ units \qquad f(P) = 12\ units$$
$$d(Q) = 6\ units \qquad f(Q) = 10\ units$$
$$d(R) = 14\ units \qquad f(R) = 18\ units$$

connected.

— the No. of DFS call required to reach all vertex?

P — Q

R 14 – 18

Which one of the following statements is TRUE about the graph

(A) There is only one connected component ✗

(B) There are two connected components, and P and R are connected ✗

(C) There are two connected components, and Q and R are connected ✗

(D) There are two connected components, and P and Q are connected ✓ (D)

GATE 2006-IT | 2-Marks, Question | Category-MCQ

Which one of the following statements is TRUE about the graph

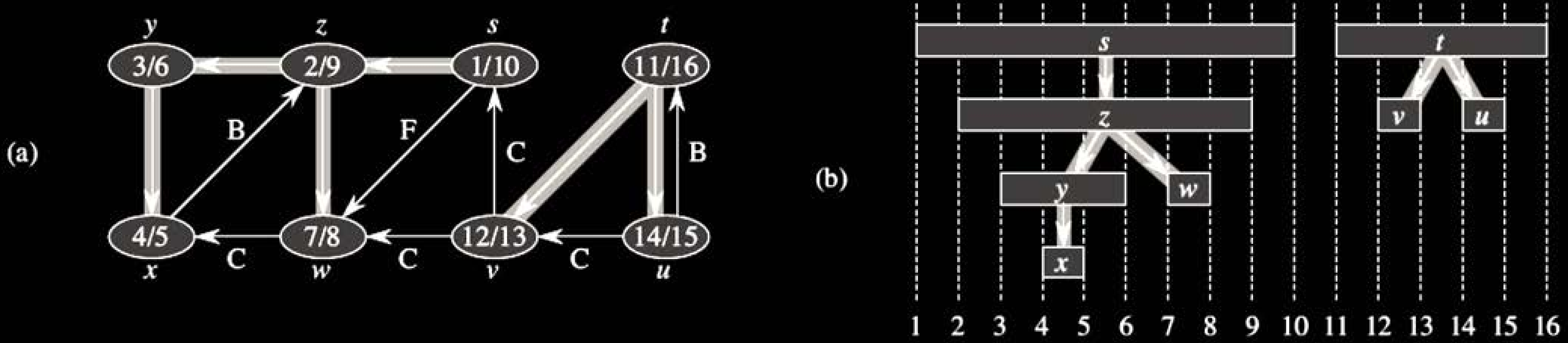
(A) There is only one connected component

(B) There are two connected components, and P and R are connected

(C) There are two connected components, and Q and R are connected

(D) There are two connected components, and P and Q are connected

- the intervals [*u.d, u.f*] and [*v.d, v.f*] arc entirely disjoint, and neither *u* nor *v* is a descendant of the other in the depth-first forest,
- the interval [*u.d, u.f*] is contained entirely within the interval [*v.d, v.f*], and *a* is a descendant of *v* in a depth-first tree, or
- the interval [*v.d. v.f*] is contained entirely within the interval [*u.d, u.f*], and *v* is a descendant of *u* in a depth-first tree.



GATE 2006

A depth-first search is performed on a directed acyclic graph. Let d[u] denote the time at which vertex u is visited for the first time, and f[u]the time at which the DFS call to the vertex u terminates. Which of the following statements is always true for all edges (u, v) in the graph?

(A) $d[u] < d[v]$ (B) $d[u] < f[v]$

(C) $f[u] < f[v]$ (D) $f[u] > f[v]$

Classification of edges:

Tree Edge:

Forward Edge:

Back edge:

Cross Edge:

Classification of edges:

Tree Edge: It is an edge which is present in the tree obtained after applying DFS on the graph. All the Green edges are tree edges.

Forward Edge: It is an edge (u, v) such that v is descendant but not part of the DFS tree. Edge from 1 to 8 is a forward edge.

Back edge: It is an edge (u, v) such that v is ancestor of node u but not part of DFS tree. Edge from 6 to 2 is a back edge. Presence of back edge indicates a cycle in directed graph.

Cross Edge: It is a edge which connects two node such that they do not have any ancestor and a descendant relationship between them. Edge from node 5 to 4 is cross edge.

Classification of edges:

Another interesting property of depth-first search is that the search can be used to classify the edges of the input graph $G = (V, E)$. The type of each edge can provide important information about a graph.

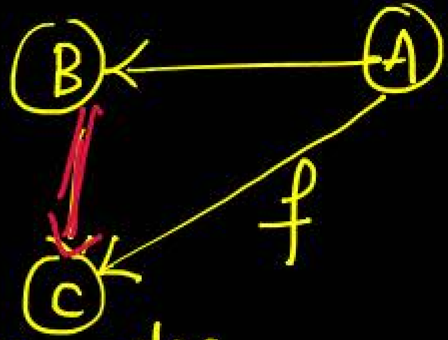
Classification of edges:

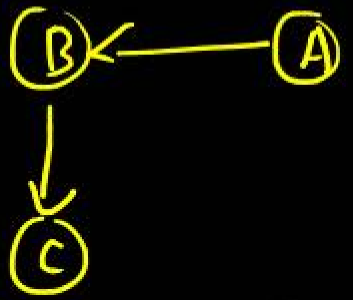
We can define four edge types in terms of the depth-first forest G_n produced by a depth-first search on G:

1. ***Tree edges*** are edges in the depth-first forest G_n. Edge (u, v) is a tree edge if v was first discovered by exploring edge $(u$, v).

depends upon in which order ~~ver~~ the graph has been traversed

2. Forward edge : Forward edge an edge that a ~~connects~~ is Non-tree edge and it connects a vertex u to its descendant v. (Immediate children is not consider as descendant)





3. Backward edge : Backward edge is Non-tree edge that ~~com~~ connects a vertex u to its ancestor v.

Immediate parent is ~~not~~ ancestor.