GRAPH

Types of Graph

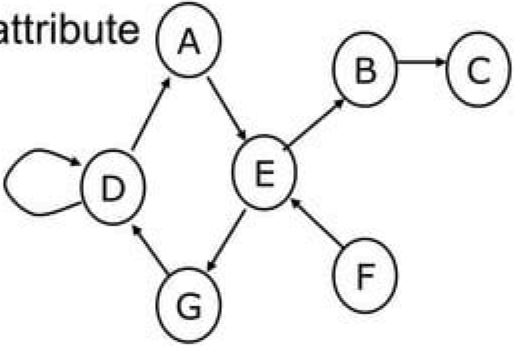
Terminology

Storage Structure

Graph

A graph is a collection of nodes (or vertices, singular is vertex) and edges (or arcs)

- Each node contains an element
- Each edge connects two nodes together (or possibly the same node to itself) and may contain an edge attribute



Formal definition of graph

A graph *G* is defined as follows:

$$G=(V,E)$$

V(G): a finite, nonempty set of vertices

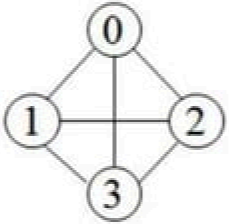
E(G): a set of edges (pairs of vertices)

Types of graph

- Directed graph (digraph)
- Undirected graph (graph)

Undirected graph

- An undirected graph is one in which the edges do not have a direction
- 'graph' denotes undirected graph.
- Undirected graph:
  - (v1, v2) in E is un-ordered.
  - (v1,v2) and (v2, v1) represent the same edge.



G1

G1 = (4, 6)

V(G1) = { 0, 1, 2 , 3 }

E(G1) = { (0,1), (0,2), (0,3

(1,2), (1,3), (2,3) }

Directed graph

- Directed graph is one in which the edges have a direction
- Also called as 'digraph'
- <u>Directed graph:</u>
  - < v1, v2 > in E is ordered.
  - <V1,v2> and <v2, v1> represent the two different edges.

G_3

G3 = (3, 3)

V(G3) = { 0,1,2 }

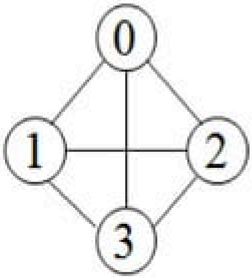
E(G3) = { <0,1> , <1,0> , <1,2>

Complete graph

A complete graph is a graph that has the maximum number of edges .

- for undirected graph with n vertices, the maximum number of edges is $n(n-1)/2$
- for directed graph with n vertices, the maximum number of edges is $n(n-1)$

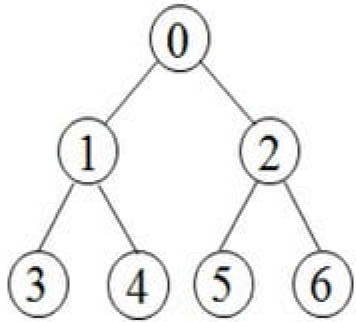
Complete graph



G_1

complete graph

No.of edges = n(n-1)/2
= 4*3/2
= 12/2
= 6



G_2

incomplete graph

No.of edges = n(n-1)/2
= 7*6/2
= 42/2
= 21

G_3

No.of edges = n(n-1
= 3*2
= 6

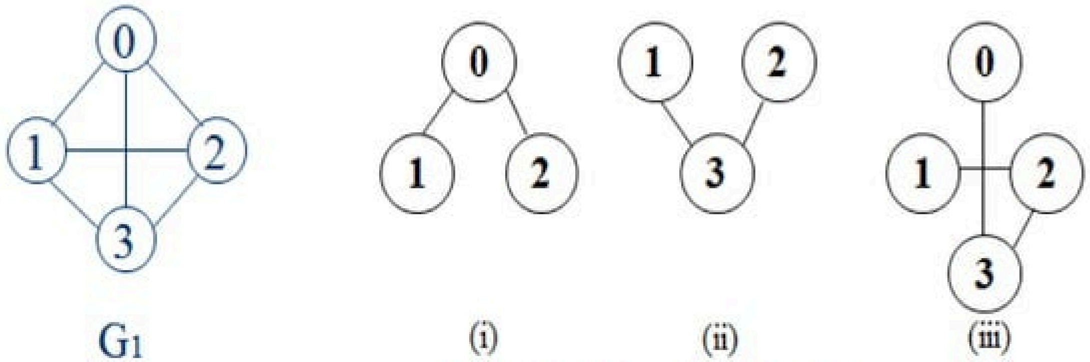
Adjacent and Incident

- If (v0, v1) is an edge in an undirected graph,
  - v0 and v1 are adjacent
  - The edge (v0, v1) is incident on vertices v0 and v1
- If <v0, v1> is an edge in a directed graph
  - v0 is adjacent to v1, and v1 is adjacent from v0
  - The edge <v0, v1> is incident on vertices v0 and v1

Sub- graph

A sub-graph of G is a graph G' such that

- V(G') is a subset of V(G)
- E(G') is a subset of E(G)



(a) Some of the subgraph of G_1

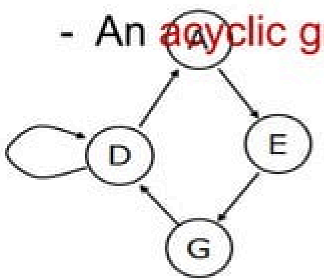
Path

- A path is a list of edges such that each node is the predecessor of the next node in the list
- A path from vertex vp to vertex vq in a graph G, is a sequence of vertices, vp, vi1, vi2, ..., vin, vq, such that (vp, vi1), (vi1, vi2), ..., (vin, vq) are edges in an undirected graph
- The length of a path is the number of edges on it
- A simple path is a path in which all vertices, except possibly the first and the last, are distinct

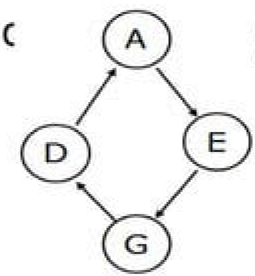
Cycle

A cycle is a path whose first and last nodes are the same

- A cyclic graph contains at least one cycle
- An acyclic graph does not c ycles



cyclic graph



acyclic graph

Connected component

- In an undirected graph G, two vertices, v0 and v1, are connected if there is a path in G from v0 to v1
- An undirected graph is connected if, for every pair of distinct vertices vi, vj, there is a path from vi to vj
- A connected component of an undirected graph is a maximal connected sub-graph.

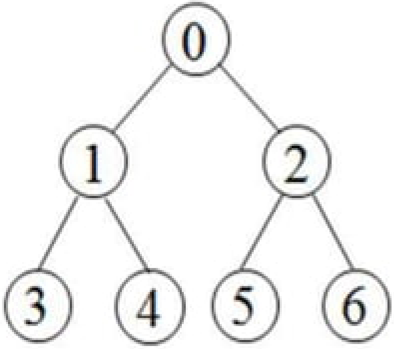
Strongly connected

- A directed graph is strongly connected if there is a directed path from vi to vj and also from vj to vi.
- A strongly connected component is a maximal sub-graph that is strongly connected

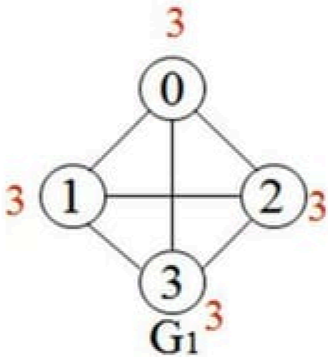
Tree

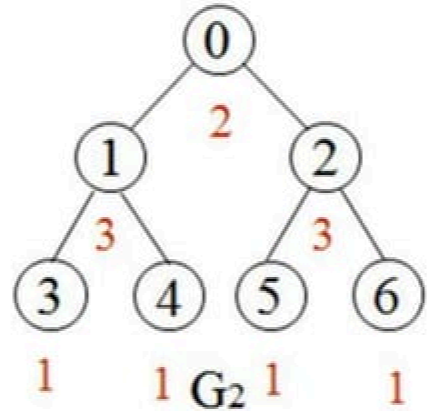
A tree is a graph that is

- connected
- acyclic.

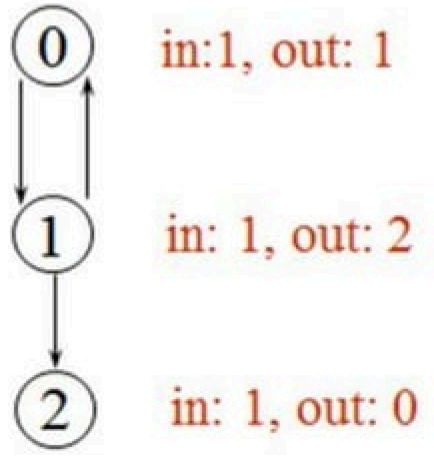


Degree - graph





Degree - digraph



G_3

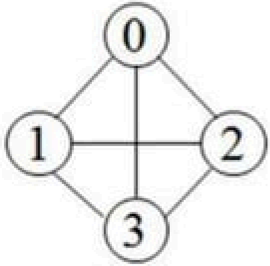
Graph Representation

- Adjacency Matrix
- Adjacency Lists

Adjacency matrix

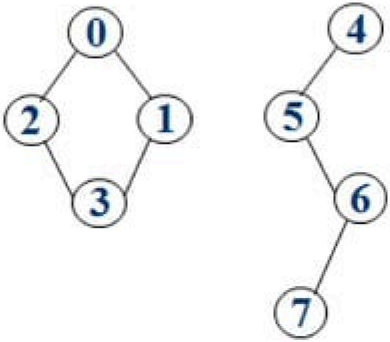
- Let G=(V,E) be a graph with n vertices.
- The adjacency matrix of G is a two-dimensional n by n array, say adj_mat
  - If the edge (vi, vj) is in E(G), adj_mat[i][j]=1
  - If there is no edge in E(G), adj_mat[i][j]=0
- The adjacency matrix for an undirected graph is symmetric
- the adjacency matrix for a digraph need not be symmetric

Adjacency matrix - graph



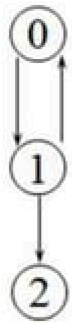
G1

	0	1	2	3
0	0	1	1	1
1	1	0	1	1
2	1	1	0	1
3	1	1	1	0



	0	1	2	3	4	5	6	7
0	0	1	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0
2	1	0	0	1	0	0	0	0
3	0	1	1	0	0	0	0	0
4	0	0	0	0	0	1	0	0
5	0	0	0	0	1	0	1	0
6	0	0	0	0	0	1	0	1
7	0	0	0	0	0	0	1	0

Adjacency matrix - digraph



G_3

$$
\begin{array}{c|ccc}
 & 0 & 1 & 2 \\
\hline
0 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 \\
2 & 0 & 0 & 0
\end{array}
$$

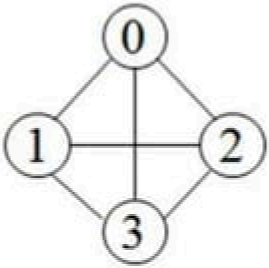
Adjacency list

- To overcome the problem arise in the adjacency matrix, linked list can be used
- The adjacency list contains two lists

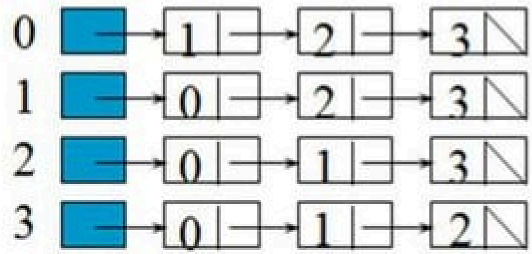
    1. node list

    2. edge list

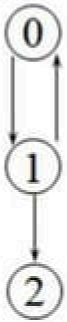
Adjacency list – graph



G1

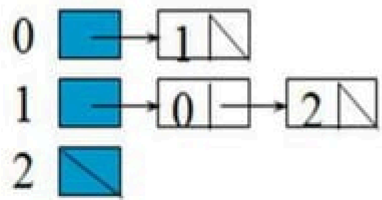


Adjacency list - digraph



G3

Adjacency list



Inverse Adjacency list