Graphs

Introduction

- Graphs are a generalization of trees
  - Nodes or vertices
  - Edges or arcs
- Two kinds of graphs
  - Directed
  - Undirected

Introduction: Formal Definition

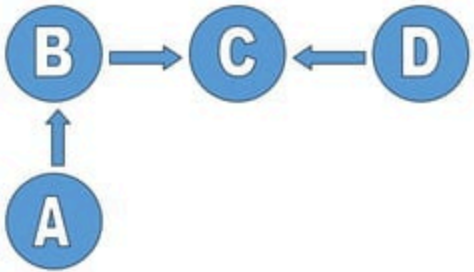
- A graph G = (V,E) consists of a finite set of vertices, V, and a finite set of edges E.
- Each edge is a pair (v,w) where v, w $\in$ V

Introduction: Formal Definition

- A **directed** graph, or **digraph**, is a graph in which the edges are ordered pairs
  - (v, w) ≠ (w, v)
- An **undirected** graph is a graph in which the edges are unordered pairs
  - (v, w) == (w, v)

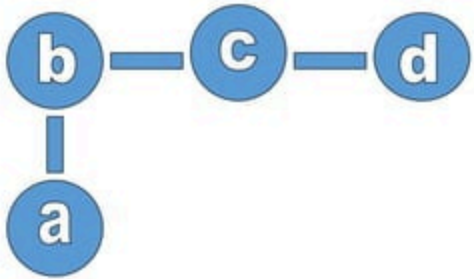
Introduction: Directed Graphs

- In a directed graph, the edges are arrows.
- Directed graphs show the flow from one node to another and not vise versa.

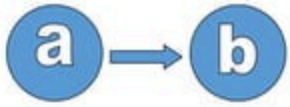


Introduction: Undirected Graphs

- In a Undirected graph, the edges are lines.
- UnDirected graphs show a relationship between two nodes.

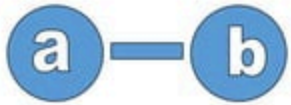


Terminology



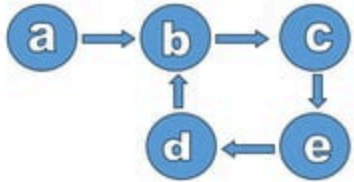
- In the directed graph above, b is **adjacent** to a because (a, b) $\in$ E. Note that a is *not* adjacent to b.
- A is a **predecessor** of node B
- B is a **successor** of node A
- The **source** of the edge is node A, the **target** is node B

Terminology



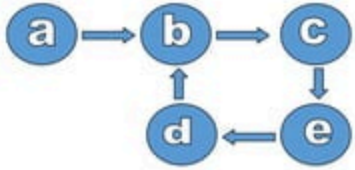
- In the undirected graph above, a and b are **adjacent** because (a,b) $\in$ E. a and b are called **neighbors.**

Terminology



- A **path** is a sequence of vertices w_1, w_2,...w_n such that $(w_i, w_{i+1}) \in E$, $1 <= i < n$, and each vertex is unique except that the path may start and end on the same vertex
- The **length** of the path is the number of edges along the path

Terminology

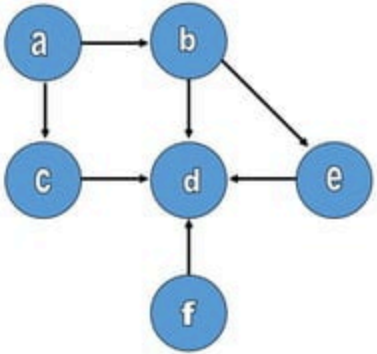


- An **acyclic path** is a path which does not follow a sequence.
- A **cyclic path** is a path such that
  - There are at least two vertices on the path
  - $w_1 = w_n$ (path starts and ends at same vertex)
  - And also maintains the sequence

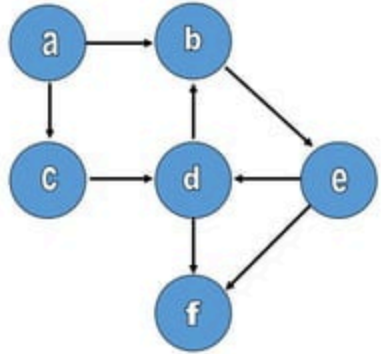
Test Your Knowledge

Cyclic or Acyclic?

1.



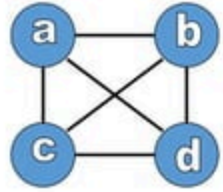
2.



Terminology

- A directed graph that has *no* cyclic paths is called a **DAG** (a Directed Acyclic Graph).
- An undirected graph that has an edge between every pair of vertices is called a **complete** graph.

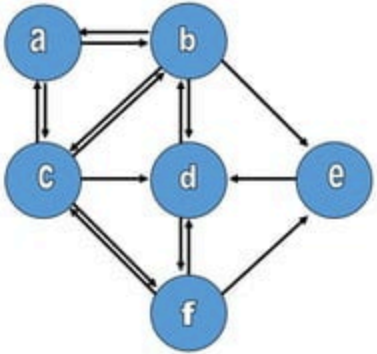
Note: A directed graph can also be a complete graph; in that case, there must be an edge from every vertex to every other vertex.



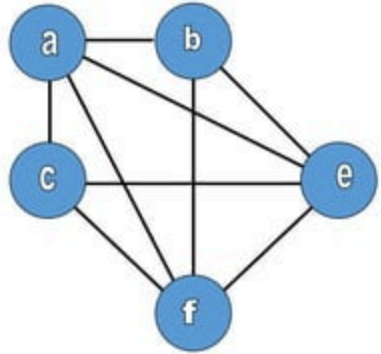
Test Your Knowledge

Complete, or "Acomplete" (Not Complete)

1.



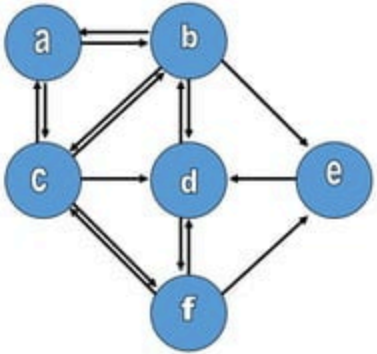
2.



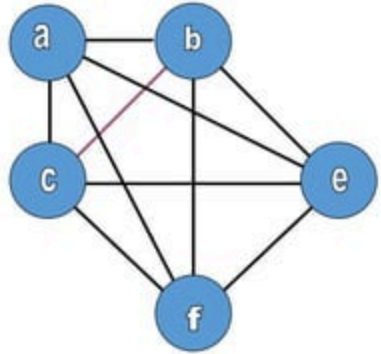
Test Your Knowledge

Complete, or "Acomplete" (Not Complete)

1.



2.

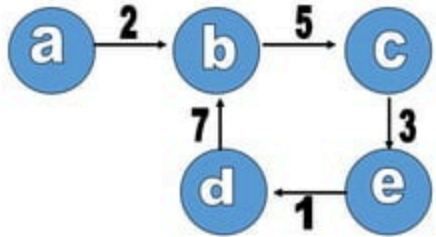


Terminology

- An undirected graph is **connected** if a path exists from every vertex to every other vertex
- A directed graph is **strongly connected** if a path exists from every vertex to every other vertex
- A directed graph is **weakly connected** if a path exists from every vertex to every other vertex, disregarding the direction of the edge

Terminology

- A graph is known as a **weighted graph** if a weight or metric is associated with each edge.



Representation of Graph

Following are the representation of a graph

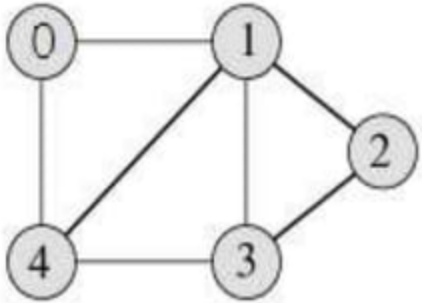
- Adjacency Matrix
- Adjacency List

There are other representations also like, Incidence Matrix and Incidence List. The choice of the graph representation is situation specific. It totally depends on the type of operations to be performed and ease of use.

Adjacency Matrix

- Adjacency Matrix is a 2D array of size V x V where V is the number of vertices in a graph. Let the 2D array be adj[][], a slot adj[i][j] = 1 indicates that there is an edge from vertex i to vertex j. Adjacency matrix for undirected graph is always symmetric. Adjacency Matrix is also used to represent weighted graphs. If adj[i][j] = w, then there is an edge from vertex i to vertex j with weight w.

Adjacency Matrix

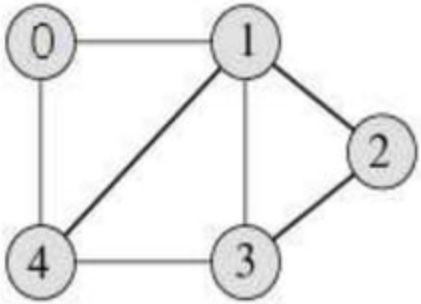


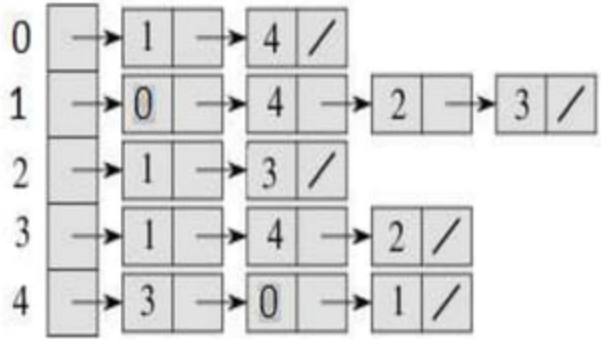
	0	1	2	3	4
0	0	1	0	0	1
1	1	0	1	1	1
2	0	1	0	1	0
3	0	1	1	0	1
4	1	1	0	1	0

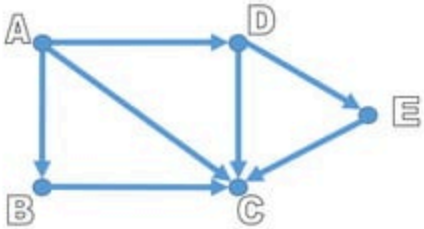
Adjacency List

- An array of linked lists is used. Size of the array is equal to number of vertices. Let the array be array[]. An entry array[i] represents the linked list of vertices adjacent to the ith vertex. This representation can also be used to represent a weighted graph. The weights of edges can be stored in nodes of linked lists. Following is adjacency list representation of the above graph.

Adjacency List



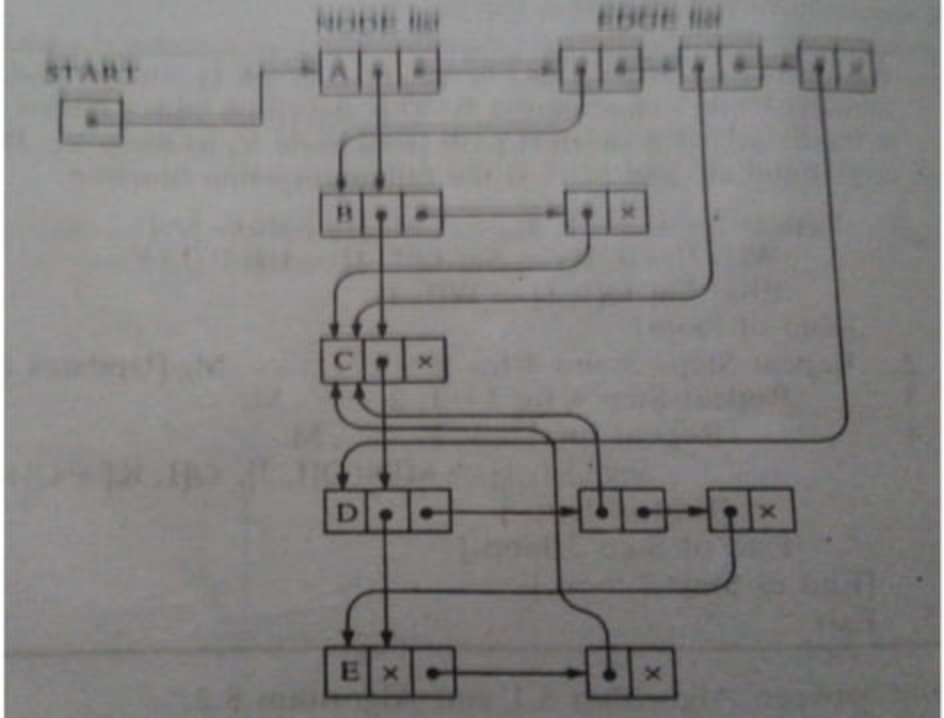




a) Graphs G

Node	Adjacency List
A	B,C,D
B	C
C	
D	C,E
E	C

b) Adjacency lists of G


START
A
B
C
D
E
×

Uses For Graphs

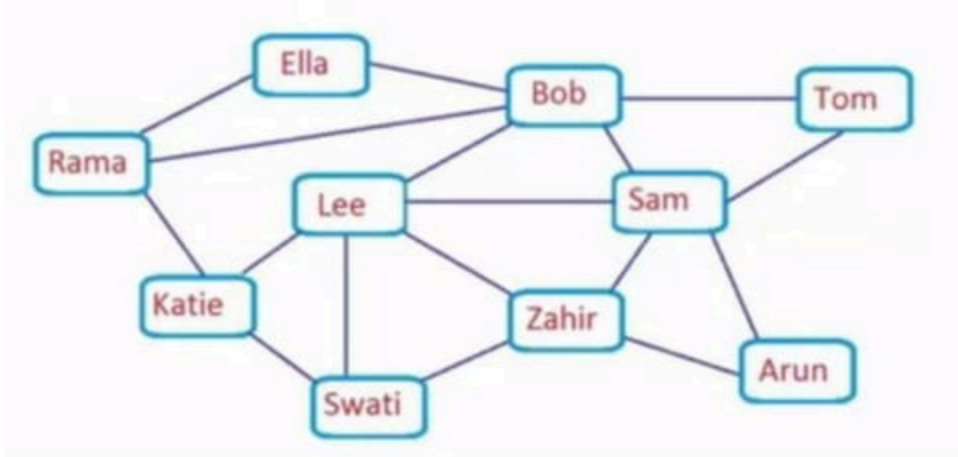
Uses for Graphs

- **Computer network:** The set of vertices V represents the set of computers in the network. There is an edge (u, v) if and only if there is a direct communication link between the computers corresponding to u and v.

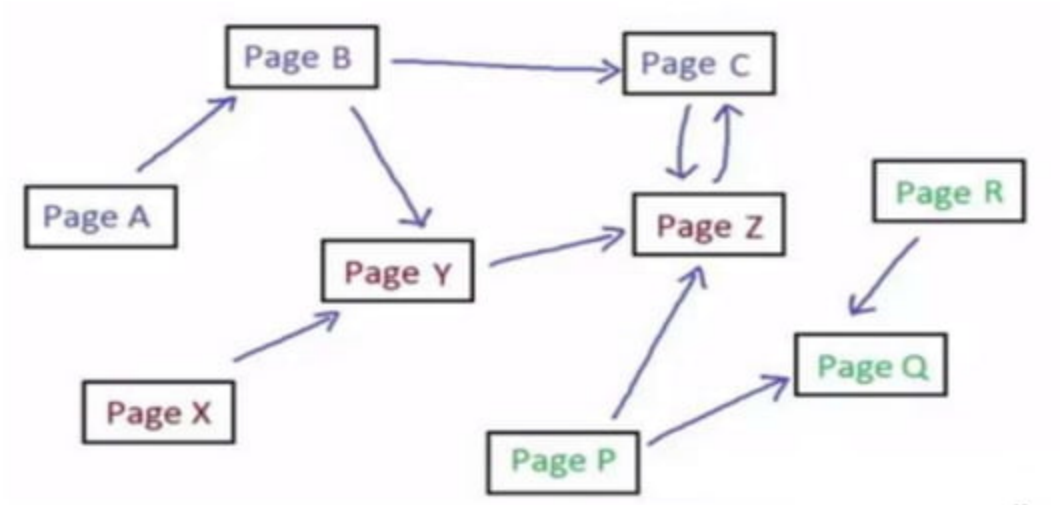
Uses for Graphs

- **Two-Player Game Tree:** All of the possibilities in a board game like chess can be represented in a graph. Each vertex stands for one possible board position. (For chess, this is a very big graph!)

Social Media (Facebook)



Websites



Intercity Road Network