GRAPH THEORY
BASIC TERMINOLOGY

CS 441
Basic Graph Definitions

• A data structure that consists of a set of nodes (*vertices*) and a set of edges that relate the nodes to each other.

• The set of edges describes relationships among the vertices.

Some Examples,
- Car navigation system
- Efficient database
- Build a bot to retrieve info off WWW
- Representing computational models
Applications

- electronic circuits
- networks (roads, flights, communications)
Formal Definition:

• A graph, $G=(V, E)$, consists of two sets:
  • a finite non empty set of vertices($V$), and
  • a finite set $(E)$ of unordered pairs of distinct vertices called edges.

• $V(G)$ and $E(G)$ represent the sets of vertices and edges of $G$, respectively.

• Vertex: In graph theory, a vertex (plural vertices) or node or points is the fundamental unit out of which graphs are formed.

• Edge or Arcs or Links: Gives the relationship between the Two vertices.
Examples for Graph

\[ V(G_1) = \{0,1,2,3\} \]
\[ V(G_2) = \{0,1,2,3,4,5,6\} \]
\[ V(G_3) = \{0,1,2\} \]

\[ E(G_1) = \{(0,1),(0,2),(0,3),(1,2),(1,3),(2,3)\} \]
\[ E(G_2) = \{(0,1),(0,2),(1,3),(1,4),(2,5),(2,6)\} \]
\[ E(G_3) = \{<0,1>,<1,0>,<1,2>\} \]
Graph Terminology

- Two vertices joined by an edge are called the end vertices or endpoints of the edge.

- If an edge is directed its first endpoint is called the origin and the other is called the destination.

- Two vertices are said to be adjacent if they are endpoints of the same edge.
Graph Terminology
Graph Terminology

Vertices A and B are endpoints of edge a
Graph Terminology

Vertex A is the origin of edge a
Vertex B is the destination of edge a
Vertices A and B are adjacent as they are endpoints of edge a.
Graph Terminology

- An edge is said to be *incident* on a vertex if the vertex is one of the edges endpoints.

- The *outgoing* edges of a vertex are the directed edges whose origin is that vertex.

- The *incoming* edges of a vertex are the directed edges whose destination is that vertex.
Graph Terminology

Edge 'a' is incident on vertex V
Edge 'h' is incident on vertex Z
Edge 'g' is incident on vertex Y
Graph Terminology

The outgoing edges of vertex W are the edges with vertex W as origin \{d, e, f\}.
The incoming edges of vertex X are the edges with vertex X as destination \{b, e, g, i\}
Adjacent, neighbors

- Two vertices are *adjacent* and are *neighbors* if they are the endpoints of an edge

- Example:
  - A and B are adjacent
  - A and D are *not* adjacent
Degree: Number of edges incident on a node

The degree of B is 2.
Degree (Directed Graphs)

- In degree: Number of edges entering a node
- Out degree: Number of edges leaving a node
- Degree = Indegree + Outdegree

The in degree of 2 is 2 and the out degree of 2 is 3.
Path

• A **path** is a sequence of vertices such that there is an edge from each vertex to its successor.
• A path is **simple** if each vertex is distinct.
• A **circuit** is a path in which the terminal vertex coincides with the initial vertex.

Simple path: [1, 2, 4, 5]
Path: [1, 2, 4, 5, 4]
Circuit: [1, 2, 4, 5, 4, 1]
Cycle

• A path from a vertex to itself is called a \textit{cycle}.
• A graph is called \textit{cyclic} if it contains a cycle;
  • otherwise it is called \textit{acyclic}
Types of Graph
Null graph, Trivial Graph

• A graph $G=(V,E)$ where $E=0$ is said to be Null or Empty graph.

• A graph with One vertex and no edge is called as a trivial graph.
Connected and Disconnected

• **Connected**: There exists at least one path between two vertices
• **Disconnected**: Otherwise

• Example:
  • $H_1$ and $H_2$ are connected
  • $H_3$ is disconnected
Undirected Graph

• In an undirected graph, there is no distinction between \((u, v)\) and \((v, u)\).
• An edge \((u, v)\) is said to be directed from \(u\) to \(v\) if the pair \((u, v)\) is ordered with \(u\) preceding \(v\).
  
  E.g. A Flight Route

• An edge \((u, v)\) is said to be undirected if the pair \((u, v)\) is not ordered
  
  E.g. Road Map
Undirected Graph

Here $(u,v)$ and $(v,u)$ both are possible.
Undirected Graph

- A graph whose definition makes reference to unordered pairs of vertices as Edges is known as undirected graph.
- Thus an undirected edge \((u, v)\) is equivalent to \((v, u)\) where \(u\) and \(v\) are distinct vertices.
- In the case of undirected edge \((u, v)\) in a graph, the vertices \(u, v\) are said to be adjacent or the edge \((u, v)\) is said to be incident on vertices \(u, v\).
Complete Graph

- **Complete Graph**: A simple graph in which every pair of vertices are adjacent.
- If no of vertices = n, then there are \( n(n-1) \) edges.
Complete Graph

• In a complete graph: Every node should be connected to all other nodes.

• The above means “Every node is adjacent to all other nodes in that graph”.

• The degree of all the vertices must be same.

• $K_n$ = Denotes a complete with n number of vertices.
Complete Undirected Graph

An undirected graph with ‘n’ number of vertices is said to be complete, iff each vertex

Number of vertices = 3
Degree of each vertex = (n-1) = (3-1) = 2
Complete Undirected Graph

- An $n$ vertex undirected graph with exactly $(n.(n-1))/2$ edges is said to be complete.

Here we have 4 number of vertices and hence 
$\frac{(4.(4-1))}{2} = \frac{(4.3)}{2} = 6$

Hence the graph has 6 number of edges and it is a Complete Undirected graph.
Directed Graph

• A directed graph is one in which every edge \((u, v)\) has a direction, so that \((u, v)\) is different from \((v, u)\)

There are two possible situations that can arise in a directed graph between vertices \(u\) and \(v\).

• i) only one of \((u, v)\) and \((v, u)\) is present.

• ii) both \((u, v)\) and \((v, u)\) are present.
Directed Graph

Here \((u,v)\) is possible where as \((v,u)\) is not possible

In a directed edge, \(u\) is said to be adjacent to \(v\) and \(v\) is said to be adjacent from \(u\).

The edge \(<u,v>\) is incident to both \(u\) and \(v\).
Directed Graph

- Directed Graphs are also called as **Digraph**.
- Directed graph or the digraph make reference to edges which are directed (i.e) edges which are Ordered pairs of vertices.
- The edge \((uv)\) is referred to as \(<u,v>\) which is distinct from \(<v,u>\) where \(u,v\) are distinct vertices.
Weighted Graph

Weighted graph is a graph for which each edge has an associated_weight, usually given by a_weight function_w: E → R.
Planar Graph

- Can be drawn on a plane such that no two edges intersect
Sub Graph

- A graph whose vertices and edges are subsets of another graph.
- A subgraph $G'=(V',E')$ of a graph $G = (V,E)$ such that $V' \subseteq V$ and $E' \subseteq E$, then $G$ is a supergraph for $G'$. 

![Diagram of subgraph and supergraph](image-url)
Spanning Subgraph

- A *spanning subgraph* is a subgraph that contains all the vertices of the original graph.
Induced-Subgraph

• Vertex-Induced Subgraph:
  • A *vertex-induced subgraph* is one that consists of some of the vertices of the original graph and all of the edges that connect them in the original.
**Induced-Subgraph**

- **Edge-Induced Subgraph:**
  - An *edge-induced subgraph* consists of some of the edges of the original graph and the vertices that are at their endpoints.
Minimum Spanning Tree
Minimum Spanning Tree

- What is MST?
- Kruskal's Algorithm
- Prim's Algorithm
A spanning tree of a graph is just a subgraph that contains all the vertices and is a tree.

A graph may have many spanning trees.
Complete Graph

All 16 of its Spanning Trees
The Minimum Spanning Tree for a given graph is the Spanning Tree of minimum cost for that graph.
GRAPH REPRESENTATION

- Adjacency Matrix
- Incidence Matrix
- Adjacency List
Adjacency, Incidence, and Degree

- Assume $e_i$ is an edge whose endpoints are $(v_j, v_k)$
- The vertices $v_j$ and $v_k$ are said to be adjacent
- The edge $e_i$ is said to be incident upon $v_j$
- *Degree* of a vertex $v_k$ is the number of edges incident upon $v_k$. It is denoted as $d(v_k)$
Adjacency Matrix

- Let $G = (V, E)$, $|V| = n$ and $|E| = m$
- The **adjacency matrix** of $G$ written $A(G)$, is the $|V| \times |V|$ matrix in which entry $a_{i,j}$ is the number of edges in $G$ with endpoints $\{v_i, v_j\}$.
Adjacency Matrix

• Let $G = (V, E)$, $|V| = n$ and $|E| = m$

• The *adjacency matrix* of $G$ written $A(G)$, is the $|V| \times |V|$ matrix in which entry $a_{i,j}$ is 1 if an edge exists otherwise it is 0.
Adjacency Matrix (Weighted Graph)

• Let $G = (V, E)$, $|V| = n$ and $|E| = m$

• The *adjacency matrix* of $G$ written $A(G)$, is the $|V| \times |V|$ matrix in which entry $a_{i,j}$ is weight of the edge if it exists otherwise it is 0
Incidence Matrix

- Let \( G = (V, E) \), \(|V| = n\) and \(|E| = m\).
- The **incidence matrix** \( M(G) \) is the \(|V| \times |E|\) matrix in which entry \( m_{i,j} \) is 1 if \( v_i \) is an endpoint of \( e_j \) and otherwise is 0.

\[
\begin{array}{cccccc}
 & a & b & c & d & e \\
 w & 1 & 1 & 0 & 0 & 0 \\
 x & 1 & 0 & 1 & 1 & 0 \\
 y & 0 & 1 & 1 & 1 & 1 \\
 z & 0 & 0 & 0 & 0 & 1 \\
\end{array}
\]
Adjacency List Representation

• Adjacency-list representation
  – an array of $|V|$ elements, one for each vertex in $V$
  – For each $u \in V$, $ADJ[u]$ points to all its adjacent vertices.
Adjacency List Representation for a Digraph
Adjacency lists

• Advantage:
  – Saves space for sparse graphs. Most graphs are sparse.
  – Traverse all the edges that start at v, in $\theta(\text{degree}(v))$

• Disadvantage:
  – Check for existence of an edge $(v, u)$ in worst case time $\theta(\text{degree}(v))$
Adjacency List

• Storage
  – For a directed graph the number of items are
    \[ \sum_{v \in V} \text{(out-degree (v))} = | E | \]
    So we need \( \Theta( V + E ) \)
  – For undirected graph the number of items are
    \[ \sum_{v \in V} \text{(degree (v))} = 2 \times | E | \]
    Also \( \Theta( V + E ) \)
• Easy to modify to handle weighted graphs. How?
Adjacency Matrix Representation

• Advantage:
  – Saves space for:
    • Dense graphs.
    • Small unweighted graphs using 1 bit per edge.
  – Check for existence of an edge in $\theta(1)$

• Disadvantage:
  – Traverse all the edges that start at v, in $\theta(|V|)$
Adjacency Matrix Representation

• Storage
  – $\Theta(|V|^2)$ (We usually just write, $\Theta(V^2)$)
  – For undirected graphs you can save storage (only $1/2(V^2)$) by noticing the adjacency matrix of an undirected graph is symmetric. How?

• Easy to handle weighted graphs. How?