Graphs and Graphs Models

In Book : (Chapter 10, section 10.1)
A Graph, $G = (V, E)$, is a data structure which consists of a nonempty set of vertices (or nodes), $V$, and a set of edges, $E$, that connect (some of) them.

Each edge has either one or two vertices associated with it called its endpoints.

The set of vertices $V$ of a graph $G$ may be infinite.

A graph with infinite vertex set or an infinite number of edges is called an infinite graph.
Example of Graphs Applications:

- Study of the structure of the World Wide Web
- Shortest path between 2 cities in a transportation network
- Electrical circuits
- Molecular chemistry
• **Simple Graph:**

A graph in which each edge connects two different vertices, and no two edges connect the same pair of vertices.

• **Multi Edge:**

More than one edge connect the same pair of vertices.

• **Multigraphs:**

Graphs that may have multi edges.

• **Loops:**

Edges that connect a vertex to itself.
**Graph** - Graph Terminologies

- **Undirected Graph:**
  Graph that does not have direction for its edges which are called *undirected* edges.

- **Directed Graph (Digraph):**
  Graph that consists of a nonempty set of vertices V and a set of *directed* edges (or arcs) E. Each *directed edge* is associated with an *ordered pair* of vertices. The directed edge associated with the ordered pair (u,v) is said to *start* at u and *end* at v.

- A directed graph may contain loops, multiple edges: **Directed Multigraph**
  or may **not** have any loop and any multiple directed edge: **Simple directed**
# Graph - Graph Terminologies

<table>
<thead>
<tr>
<th>Type</th>
<th>Edges</th>
<th>Multiple Edges Allowed?</th>
<th>Loops Allowed?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple graph</td>
<td>Undirected</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Multigraph</td>
<td>Undirected</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Simple directed graph</td>
<td>Directed</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Directed multigraph</td>
<td>Directed</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Graphs Terminology and special Types of Graphs

In Book : (Chapter 10, section 10.2)
Graph - Graph Terminologies

• Two vertices \( u \) and \( v \) in an undirected graph \( G \) are called adjacent (or neighbors) in \( G \) if \( u \) and \( v \) are endpoints of an edge \( e \) of \( G \). Such an edge \( e \) is called incident with the vertices \( u \) and \( v \) and \( e \) is said to connect \( u \) and \( v \).

• The set of all neighbors of a vertex \( v \) of \( G = (U, V) \), denoted by \( N(v) \), is called the neighborhood of \( v \).

• If \( A \) is a subset of \( V \), we denote by \( N(A) \) the set of all vertices in \( G \) that are adjacent to at least one vertex in \( A \).

• Example:

In the Graph, \( V = \{ a, b, c, d \} \)

let \( A = \{ a, c \} \), \( N(A) = \{ b, c, a, d \} \)
Graph - Graph Terminologies

• The degree of a vertex $\text{deg}(v)$ in an undirected graph is the number of edges incident with it, except that a loop at a vertex contributes twice to the degree of that vertex.

• Example: what are the degrees and what are the neighborhoods of the vertices in the graph $G$ and $H$?
In $G$:
\[
\begin{align*}
\text{deg}(a) &= 2 \\
\text{deg}(b) &= \text{deg}(c) = \text{deg}(f) = 4 \\
\text{deg}(d) &= 1 \\
\text{deg}(e) &= 3 \\
\text{deg}(g) &= 0
\end{align*}
\]

In $H$:
\[
\begin{align*}
\text{deg}(a) &= 4 \\
\text{deg}(b) &= \text{deg}(e) = 6 \\
\text{deg}(c) &= 1 \\
\text{deg}(d) &= 5
\end{align*}
\]

<table>
<thead>
<tr>
<th>vertex</th>
<th>$N(G)$</th>
<th>$N(H)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>{b, f}</td>
<td>{b, d, e}</td>
</tr>
<tr>
<td>b</td>
<td>{a, c, e, f}</td>
<td>{a, b, c, d, e}</td>
</tr>
<tr>
<td>c</td>
<td>{b, d, e, f}</td>
<td>{b}</td>
</tr>
<tr>
<td>d</td>
<td>{c}</td>
<td>{a, b, e}</td>
</tr>
<tr>
<td>e</td>
<td>{b, c, f}</td>
<td>{a, b, d}</td>
</tr>
<tr>
<td>f</td>
<td>{a, b, c, e}</td>
<td></td>
</tr>
<tr>
<td>g</td>
<td>\varnothing</td>
<td></td>
</tr>
</tbody>
</table>
A vertex of degree zero is called isolated. It follows that an isolated vertex is not adjacent to any vertex.

Example:
vertex g in graph G.

A vertex is pendent if and only if it has degree one. It follows that a pendent vertex is adjacent to exactly one vertex.

Example:
vertex d in graph G.
The Handshaking Theorem:
Let $G = (V, E)$ be an **undirected graph** with $m$ edges. Then
$$2m = \sum_{v \in V} \text{deg}(v)$$
(Note that this applies even if **multiple edges** and **loops** are present.)

Example:
How many edges are there in a graph with 10 vertices each of degree six?
Because the sum of the degrees of the vertices is $6 \times 10 = 60$,
it follows that $2m = 60$. Therefore $m = 30$
**Example:**

How many edges are there in a graph $H$ with following degrees of vertices?

\[
\begin{align*}
\text{In } H & \quad \begin{cases} 
\text{deg}(a) = 4 \\
\text{deg}(b) = \text{deg}(e) = 6 \\
\text{deg}(c) = 1 \\
\text{deg}(d) = 5
\end{cases} \\
2m & = \sum_{v \in V} \text{deg} \ (v) \\
2m & = 4 + 6 + 6 + 1 + 5 \\
2m & = 22 \\
m & = 11 \text{ edges}
\end{align*}
\]
In a graph with directed edges:

- **The in-degree of a vertex** $v$, denoted $\text{deg}^{-}(v)$, is the number of edges with $v$ as their **terminal vertex**.

- **The out-degree of vertex** $v$, denoted by $\text{deg}^{+}(v)$, is the number of edges with $v$ as their **initial vertex**.

(Note that a **loop** at a vertex **contributes 1 to both** the in-degree and the out-degree of this vertex)

**Theorem:**

- For a digraph $G=(V, E)$ be a graph with directed edges, then:

$$\sum \text{deg}^{-}(v) = \sum \text{deg}^{+}(v) = |E|, \forall \ v \in V$$
Example 4:
Find the in-degree and the out-degree of each vertex in the graph G

The in-degree of G are:
\[
\begin{align*}
\deg^-(a) &= 2, \quad \deg^-(b) = 2, \quad \deg^-(c) = 3, \\
\deg^-(d) &= 2, \quad \deg^-(e) = 3, \quad \text{and} \quad \deg^-(f) = 0.
\end{align*}
\]

The out-degree of G are:
\[
\begin{align*}
\deg^+(a) &= 4, \quad \deg^+(b) = 1, \quad \deg^+(c) = 2, \\
\deg^+(d) &= 2, \quad \deg^+(e) = 3, \quad \text{and} \quad \deg^+(f) = 0.
\end{align*}
\]
The **complete graph** on $n$ vertices, $K_n$, is the simple graph that contains exactly one edge between each pair of distinct nodes.

The **cycle** $C_n$, $n \geq 3$, consists of $n$ vertices $v_1, v_2, \ldots, v_n$, and edges $(v_1, v_2), (v_2, v_3), \ldots, (v_{n-1}, v_n)$, and $(v_n, v_1)$. That is, $C_n$ consists of $n$ vertices and $n$ edges.
A simple graph $G$ is called **bipartite** if its vertex set can be **partitioned** into two **disjoint** sets $V_1$ and $V_2$ such that every edge in the graph connects a vertex in $V_1$ and a vertex in $V_2$. When this condition holds, we call the pair $(V_1, V_2)$ a **bipartition** of the vertex set $V$ of $G$.

**Example:**

- $C_6$ is bipartite, because its vertex set can be partitioned into two disjoint sets $V_1=\{v_1,v_3,v_5\}$ and $V_2=\{v_2,v_4,v_6\}$, and every edge of $C_6$ connects a vertex in $V_1$ and a vertex in $V_2$. 

\[ C_6 \]
**Graph - Some Special Simple Graphs**

- **Example:**
  
  $K_3$ is not bipartite, because if we divide the vertex set into two disjoint sets, one set must contain two vertices. If the graph were bipartite, these two vertices could not be connected by an edge, but in $K_3$ each vertex is connected to every other vertex by an edge.

- **Theorem:**

  A simple graph is **bipartite** if and only if it is possible to assigns one of two different colors to each vertex of the graph so that **no two adjacent vertices are assigned the same color**.
Are the graphs $G$ and $H$ displayed in Figure bipartite?

Graph $G$ is bipartite because its vertex set is the union of two disjoint sets, $\{a, b, d\}$ and $\{c, e, j, g\}$ and each edge connects a vertex in one of these subsets to a vertex in the other subset.

Graph $H$ is not bipartite because its vertex set cannot be partitioned into two subsets so that edges do not connect two vertices from the same subset.
A **Subgraph** of graph \( G=(V,E) \) is a graph \( H=(U,F) \) such that \( U \subseteq V \) and \( F \subseteq E \).

A subgraph \( H \) is a proper subgraph of \( G \) if \( H \neq G \).

The **union** of two simple graphs \( G_1=(V_1, E_1) \) and \( G_2=(V_2, E_2) \) is the simple graph with vertex set \( (V_1 \cup V_2) \) and edge set \( (E_1 \cup E_2) \). The union of \( G_1 \) and \( G_2 \) is denoted by \( G_1 \cup G_2 \).

For example:
Representation of Graphs and Graph Isomorphism

In Book : (Chapter 10, section 10.3)
Graph - Representation of Graph

- Graph could be represented in three methods:
  1. Adjacency list
  2. Adjacency matrix
  3. Incidence Matrix

1. **Adjacency list:**

   specifies the vertices that are adjacent to each vertex of the graph.

- **Examples:** use adjacency list to represent the following simple graph:

<table>
<thead>
<tr>
<th>Vertex</th>
<th>Adjacent Vertices</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b, c, e</td>
</tr>
<tr>
<td>b</td>
<td>a</td>
</tr>
<tr>
<td>c</td>
<td>a, d, e</td>
</tr>
<tr>
<td>d</td>
<td>c, e</td>
</tr>
<tr>
<td>e</td>
<td>a, c, d</td>
</tr>
</tbody>
</table>
**Examples**: use adjacency list to represent the following directed graph:

![Graph Diagram](image)

<table>
<thead>
<tr>
<th>Initial Vertex</th>
<th>Terminal Vertices</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b, c, d, e</td>
</tr>
<tr>
<td>b</td>
<td>b, d</td>
</tr>
<tr>
<td>c</td>
<td>a, c, e</td>
</tr>
<tr>
<td>d</td>
<td>b, c, d</td>
</tr>
<tr>
<td>e</td>
<td>b, c, d</td>
</tr>
</tbody>
</table>
Graph - Representation of Graph

2. **Adjacency Matrix**:

The Adjacency Matrix $A=(a_{ij})$ of a graph $G=(V,E)$ with $n$ nodes is an $n \times n$ zero-one matrix. Elements of $A$ can be defined as follows:

$$a_{ij} = \begin{cases} 
1 & \text{if } (v_i, v_j) \in E \\
0 & \text{otherwise}
\end{cases}$$

- **Example**: Find the adjacency matrices of the following graphs

$$\begin{bmatrix}
0 & 1 & 1 & 1 \\
1 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 \\
1 & 0 & 0 & 0
\end{bmatrix}$$

- **Example**: Draw a graph from the adjacency matrix

$$\begin{bmatrix}
0 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 \\
1 & 0 & 0 & 1 \\
0 & 1 & 1 & 0
\end{bmatrix}$$
Remark:

- An adjacency matrix of a graph is based on the ordering chosen for the vertices. Hence adjacency matrix for a given graph is not unique.
- Adjacency matrices can be used to represent undirected graph with loops and with multiple edges.
- A loop at the vertex $v_i$ is represented as 1 at the $(i, i)$.
- A multiple edge connecting the same pair of vertices $v_i$ and $v_j$ is represented in the adjacency matrix at the entry $(i, j)$ as the number of edges that are associated to $\{v_i, v_j\}$.
3. **Incidence Matrix:**

Let $G=(V,E)$ be a graph with $v_1, v_2, ..., v_n$ are vertices and $e_1, e_2, ..., e_m$ are edges. Then the incidence matrix with respect to this ordering of $V$ and $E$ is the $n \times m$ matrix $M=[m_{ij}]$, where

$$m_{ij} = \begin{cases} 
1 & \text{when edge } e_j \text{ is incidence on } v_i \\
0 & \text{otherwise} 
\end{cases}$$

**Example 9:** Find the incidence matrices of this graph

\[
\begin{bmatrix}
1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 \\
1 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 1 & 0 
\end{bmatrix}
\]
Graph - Isomorphism of Graph

- The simple graphs $G_1=(V_1, E_1)$ and $G_2=(V_2, E_2)$ are isomorphic if there is one-to-one and onto function $f$ from $V_1$ to $V_2$ with the property that $a$ and $b$ are adjacent in $G_1$ if and only if $f(a)$ and $f(b)$ are adjacent in $G_2$, for all $a, b$ in $V_1$. Such a function $f$ is called an \textbf{isomorphism}.

- Isomorphism comes from the Greek roots \textquotedblleft isos\textquotedblright{} for \textquotedblleft equal\textquotedblright{} and \textquotedblleft morphe\textquotedblright{} for \textquotedblleft form\textquotedblright{}.

- **Example:**

  Show that the graphs $G=(V, E)$ and $H=(U, F)$ are isomorphic.
Graph - Isomorphism of Graph

- The function $f$ with $f(u_1)=v_1$, $f(u_2)=v_4$, $f(u_3)=v_3$, and $f(u_4)=v_2$ is a one to one correspondence (bijection: one to one and onto) between $V$ and $W$.

- To see that this correspondence preserves adjacency, note that

  \[
  \begin{align*}
  \text{adjacent vertices in } G & \quad \text{adjacent vertices in } H \\
  u_1 \text{ and } u_2 & \quad f(u_1)=v_1 \text{ and } f(u_2)=v_4 \\
  u_1 \text{ and } u_3 & \quad f(u_1)=v_1 \text{ and } f(u_3)=v_3 \\
  u_2 \text{ and } u_4 & \quad f(u_2)=v_4 \text{ and } f(u_4)=v_2 \\
  u_3 \text{ and } u_4 & \quad f(u_3)=v_3 \text{ and } f(u_4)=v_2
  \end{align*}
  \]

Read examples 9, 10 and 11 in the book, pages 648 – 649.
Connectivity, Euler and Hamilton Paths

In Book: (Chapter 10, section 10.4 – section 10.5)
**Path:**

is a sequence of edges that begins at a vertex of a graph and travels from vertex to vertex along edges of the graph.

**Simple Path:**

Is a path where all the vertices are distinct, i.e. (it does not contain the same edge more than once)

**Example:**

In the graph, 1, 4, 5, 3 is a simple path,

but 1, 4, 5, 4 is not simple path.
**Graph - Connectivity**

**Length of a path** is the number of edges considered in the path.

e.g. Length of the path 1, 4, 5, 3 is 3

**A circuit**: a path whose first and last vertices are the same.

**Example:**

In the graph, Path 3, 2, 1, 4, 5, 3 is a circuit

Path 4, 5, 3, 2, 1, 4 is a circuit
Graph - Hamilton Way and Euler Path

**Cycle:**
is a circuit where all the vertices are distinct except for the first (and the last) vertex.

**Example:**
In the graph, 1,4,5,3,1 is a cycle, but 1,4,5,4,1 is not a cycle, it is just a circuit.

**Hamiltonian Circuit:**
Is a simple circuit that passes through every vertex exactly once.

**Example:**
In the graph, 1,4,5,3,2,1 is a Hamiltonian Circuit.
Connected Graph: A graph is said to be connected if there is at least one path from every vertex to every other vertex in the graph.

Euler circuit: Is a circuit that uses every edge of a graph exactly once.

Euler path: Is a path that uses every edge of a graph exactly once.

- An Euler path starts and ends at different vertices.
- An Euler circuit starts and ends at the same vertex

Example:

G has an Euler circuit,
but H does not have
Tree:
A connected undirected graph that contains no cycles.

Forest:
A graph that does not contain a cycle.

Spanning Tree
of a Graph G is a subgraph of G that is a tree and contains all the vertices of G.
End of Chapter 6