Graphs

Chapter 9
Objectives

› Getting familiar with the graph model
› Understand the basic terminology of a graph
› Recognize the different types of graph
› Understand the graph ADT
› Understand the two common graph representations
Flashback (Trees)

US

CA

Riverside County

San Bernardino County

AZ

MN

Riverside

Palm Springs
Applications of Graphs

- Networks
  - Social networks
  - Business network
  - Computer networks (even wireless networks)
  - Road networks
- Many-to-many relationships
  - Students and courses
  - Students and departments
Example: Social Network
Example: Airport Network
Graph Model

- A Graph \((G)\) consists of a set of Vertices \((V)\) and Edges \((E)\). \(G = (V, E)\)
- \(V = \{v_1, v_2, ..., v_{|V|}\}\)
- \(E = \{e_1, e_2, ..., e_{|E|}\}\)
- \(e = (v, w), e \in E, v \in V, w \in V\)
Graph Terminology

- A
- B
- C
- D
- E
- F
- G

Vertex or Node

Edges or Links or Arcs
Adjacency

Two vertices with an edge connecting them are called **adjacent** vertices.

B and F are **adjacent** vertices.

All adjacent vertices of a vertex are called **neighbors**.
Path

\( A, B, F, G \) is a **path** on the graph

A and G are said to be **connected**
A graph is **connected** if every pair of vertices are connected.
Unconnected Graphs

A graph is **unconnected** if there is at least one pair of vertices that are not connected.
Cycles

$A, B, F, G, E, C, A$ is a **cycle**

A cycle is a path where the first and last vertices are the same.
Weighted Graphs

A vertex and/or edge might have an associated **weight** or **cost**.
Directed Graphs

\[ e = (v, w) \] is an ordered pair

We call \( v \) the source and \( w \) the destination
Complete Graph

In a complete graph, there is a direct edge between every pair of vertices.
Graph Representation

- Adjacency matrix
- Adjacency list
## Adjacency Matrix

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A</strong></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>B</strong></td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td><strong>C</strong></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>D</strong></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>E</strong></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>F</strong></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td><strong>G</strong></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Adjacency List
For undirected graphs, we usually store an undirected edge $e = (v, w)$ as two directed edges $e_1 = (v, w)$ and $e_2 = (w, v)$.