

Discrete Mathematics Class Discussion Notes

Chapter 9

[Section 9.1 Introduction to Graphs](#)

[Section 9.2 Graph Terminology](#)

[Section 9.3 Representing Graphs and Graph Isomorphism](#)

[Section 9.4: Connectivity](#)

[Section 9.5 Euler and Hamilton Paths](#)

Section 9.1 Introduction to Graphs

- Graphs consist of vertices and edges that connect them. Used to model many types of problems
- Types of graphs:
 - **Simple graph** $G = (V,E)$ consists of V , a nonempty set of **vertices**, and E , a set of *unordered* pairs of distinct elements of V called **edges**.
 - **Example:** Figure 1. Computer Network between cities -- no duplicate edges.
 - **Multigraph** $G = (V,E)$ consists of V , a nonempty set of vertices, and E , a collection of *unordered* pairs of distinct elements of V called edges. The edges may be duplicated.
 - **Example:** Figure 2. Multiple lines allowed.
 - **Pseudograph** $G(V,E)$ consists of V , a nonempty set of vertices, and E , a collection of *unordered* pairs of elements of V called edges, which are not necessarily distinct, and where the edges pairs may be of the form (v,v) . An edge of the form (v,v) is called a **loop**.
 - **Example: figure 3.** Like the multigraph in figure 2, except loops are allowed.
 - **Directed graph** $G = (V,E)$ consists of V , a nonempty set of vertices, and E , a set of *ordered* pairs of distinct elements of V called edges.
 - **Example: figure 4.** Edges are represented with arrows indicating ordering of pairs.
 - **Other examples of applications of graphs:**
 - **Niche overlap graph:** (Edge between two species indicates they compete for the same food sources.) Simple graph
 - **Predator-Prey relationships.** A directed edge exists from species A to species B if A preys on B. Directed graph.
 - **Influence graphs.** An edge exists from Brian to Linda if Brian can influence Linda's thinking.
 - **Round-Robin Tournaments.** Directed edge from Team1 to Team 6 indicates that Team 1 beat Team 6.
 - **Precedence Graph** for a computer program. An edge exists from statement S to statement T if S must be executed before T for the program to work correctly. Such graphs can be used to determine which statements can be executed in parallel using parallel processing algorithms.

Section 8.2 Graph Terminology

- **Undirected graph (simple, multi, or pseudo) terms and properties:**
 - **Adjacent vertices:** Two vertices u and v are adjacent if $\{u,v\}$ is an edge of G .
 - **Degree of a vertex:** number of edges incident with it. When an edge that is a loop, it is counted twice.
 - **Isolated vertex:** degree 0.
 - **Pendant vertex:** degree 1.
 - **Theorem 1 Handshaking Theorem:** The sum of all of the degrees of vertices is equal to twice the number of edges.
 - **Theorem 2:** An undirected graph has an even number of vertices of odd degree.
- **Directed graph terms and properties:**
 - Let (u,v) be an edge in E . Then u is **adjacent to** v , v is **adjacent from** u , u is the **initial vertex** and v is the **terminal vertex**.
 - The **in-degree** of a vertex v is the number of edges adjacent to v .
 - The **out-degree** of a vertex v is the number of edges to which v is adjacent.
 - **Theorem 3** In a directed graph, the sum of the out degrees over all vertices must equal the sum of the in-degrees over all vertices.
- **Special simple graphs:**
 - **Complete graph with n vertices, K_n .** Exactly one edge between every pair of vertices.
 - **Cycle C_n , $n \geq 3$**
 - **Wheel W_n , $n \geq 3$.**
 - **n -Cube, Q_n .** Number each vertex with distinct bit-strings of length n . Two vertices are adjacent if and only if their bit strings differ by at most one bit. Some parallel processors are set up this way.
- **Bipartite Graphs:**
 - **Definition:** A simple graph G is called **bipartite** if its vertex set V can be partitioned into two disjoint nonempty 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 .
 - **Complete bipartite graph $K_{m,n}$.** In the two subsets, every vertex in the first set is connected to every vertex in the second set.
- **New graphs from old**
 - **Subgraph** of a graph $G = (V,E)$. A graph $H = (W,F)$ where W is a subset of V and F is a subset of E .
 - **Union of two graphs** $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ is the graph $G = (V_1 \cup V_2, E_1 \cup E_2)$

Section 8.3 Representing Graphs and Graph Isomorphism

Representing graphs

- Adjacency lists: For each vertex, list the vertices adjacent to it.
- Adjacency matrices: Number the n vertices 1 through n . Create an $n \times n$ zero-one matrix A , where the entry in the i 'th row, j th column, $a_{ij} = 1$ if vertex i is adjacent to vertex j , 0 otherwise.
- The adjacency matrix of a simple graph is symmetric (i.e. $a_{ij} = a_{ji}$).
- For graphs with loops and multiple edges, a_{ij} stores the number of edges from vertex i to vertex j .

- See examples 1 -- 7

Isomorphism of Graphs

- Informally two graphs are isomorphic if they can be drawn in the same way.
- **Definition:** Two simple graphs $G_1 = \{V_1, E_1\}$ and $G_2 = \{V_2, E_2\}$ are **isomorphic** if there exists a one-to-one and onto function f from V_1 to V_2 such that for every pair of vertices a and b , a and b are adjacent in G_1 if and only if $f(a)$ and $f(b)$ are adjacent in G_2 .
- To show two graphs are isomorphic, you must construct the function f and verify that it satisfies the isomorphic requirements.
- To show two graphs are not isomorphic, you must show such a function cannot be constructed.
- Invariants which must hold for graphs to be isomorphic:
 - They must have the same number of edges
 - They must have the same number of vertices.
 - The degrees of the vertices must agree.
- If any of these fail, you can conclude that the graphs are not isomorphic. (If they hold, the graphs may or may not be isomorphic).
- See examples 9 -- 12

Section 9.4: Connectivity

Paths

- **Definition: Path of length n from u to v** where n is a positive integer is a set of edges $e_1 = \{u, w_1\}$, $e_2 = \{w_1, w_2\}$, ... $e_n = \{w_n, v\}$. The definition applies to directed graphs as well as undirected graphs. (In directed graphs the edges would be the ordered pairs (u, w_1) , etc.).
- **Definition:** A **circuit** is a path from u to u .
- **Definition:** A path is **simple** if it does not contain the same edge more than once.

Connectedness in Undirected Graphs

- **Definition:** An undirected graph is **connected** if there is a path between every pair of distinct vertices of the graph.
- If the graph is not connected, then it can be expressed as the union of disjoint connected subgraphs, called its **connected components**.
- See example 3 (figure 3), page 464.
- **Cut vertices (articulation points)** -- A vertex that produces new connected components when removed.
- **Cut edge (bridge)** -- An edge that produces new connected components when removed.
- See example 4.

Connectedness in Directed Graphs

- A directed graph is **strongly connected** if there is a path from a to b and from b to a whenever a and b are vertices in the graph.
- A directed graph is **weakly connected** if there is a path between any two vertices if the directions of the edges are ignored.

Using Paths to Help Determine Isomorphism

- Isomorphic graphs must have circuits of the same length (See example 6)

- Paths can be used to help define potential isomorphisms -- See example 7. Find paths that move through each graph where the vertices match in degrees -- Make f map corresponding vertices along the paths.

Section 8.5 Euler and Hamilton Paths

- **Definition 1:** An **Euler circuit** in a graph G is a *simple circuit* containing *every edge* of G . An **Euler path** in G is a *simple path* containing *every edge* of G .
- **Example 1** shows that G_1 has an Euler circuit, G_2 , G_3 do not. G_3 has an Euler path, but G_2 does not.
- **Example 2** (Shows directed graphs (digraphs). H_1 does not have an Euler path. H_2 has an Euler circuit. H_3 has an Euler path, but no Euler Circuit.
- **The Konigsberg Bridge Problem** Can one cross each bridge exactly once, returning to the starting position? -- i.e is there an Euler path?

No, by the following result:

- **Theorem 1** A connected multigraph has an **Euler circuit** if and only if each of its vertices **has an even degree**.
- **Theorem 2** A connected multigraph has an **Euler path, but no Euler circuit** if and only if it has exactly **two vertices of odd degree**. (See examples).

Hamilton Paths and Circuits

- A **Hamilton path** is a path that visits every vertex exactly once.
- A **Hamilton circuit** is a circuit that visits every vertex exactly once, and then returns to the starting vertex.
- The existence of a Hamilton circuit for a given graph is also called the **Traveling Salesman Problem**.
 - There are no known simple necessary and sufficient conditions for the existence of Hamilton circuits.
 - The finding of Hamilton circuits in a graph is a computationally complex problem (The best known algorithms require exponential computing time as a function of the number of vertices and edges in the graph.)