Discrete Mathematics

5. Graphs & Trees
What are Graphs?

• General meaning in everyday math:
  *A plot or chart of numerical data using a coordinate system.*

• Technical meaning in discrete mathematics:
  *A particular class of discrete structures (to be defined) that is useful for representing relations and has a convenient webby-looking graphical representation.*
Simple Graphs

- The graph in which each edge connects two different vertices and where no two edges connect the same pair of vertices
- Correspond to symmetric binary relations $R$. 
Simple Graphs

- **Definition:**
  A simple graph \( G = (V, E) \) consists of:
  - a set \( V \) of vertices or nodes (\( V \) corresponds to the universe of the relation \( R \)), and
  - a set \( E \) of edges / arcs / links: unordered pairs of [distinct?] elements \( u, v \in V \), such that \( uRv \).
Let $V$ be the set of states in the far-southeastern U.S.:
$V=\{\text{FL, GA, AL, MS, LA, SC, TN, NC}\}$
Let $E=\{\{u,v\} \mid u \text{ adjoins } v\}$
$=$\{\{\text{FL,GA}\},\{\text{FL,AL}\},\{\text{FL,MS}\},$
\{\text{FL,LA}\},\{\text{GA,AL}\},\{\text{AL,MS}\},$
\{\text{MS,LA}\},\{\text{GA,SC}\},\{\text{GA,TN}\},$
\{\text{SC,NC}\},\{\text{NC,TN}\},\{\text{MS,TN}\},$
\{\text{MS,AL}\}\}$
Multigraphs

- Like simple graphs, but there may be *more than one* edge connecting two given nodes.

**Definition:**

A multigraph $G=(V, E, f)$ consists of a set $V$ of vertices, a set $E$ of edges (as primitive objects), and a function $f:E\rightarrow\{\{u,v\} | u,v\in V \land u\neq v\}$.

- Example:

  nodes are cities and
  edges are segments of major highways.
Pseudographs

- Like a multigraph, but edges connecting a node to itself are allowed.

**Definition:**

A pseudograph $G = (V, E, f)$ where $f: E \rightarrow \{\{u, v\} \mid u, v \in V\}$. Edge $e \in E$ is a loop if $f(e) = \{u, u\} = \{u\}$.

**Example:**

nodes are campsites in a state park and edges are hiking trails through the woods.
Directed Graphs

- Correspond to arbitrary binary relations $R$, which need not be symmetric.

- **Definition:**
  A directed graph $(V, E)$ consists of a set of vertices $V$ and a binary relation $E$ on $V$.

- **Example:**
  $V = \text{people}$, $E = \{(x, y) \mid x \text{ loves } y\}$
Walk, loop, sling, and path

- **Definition:**
  - A *walk* is a sequence $x_0, x_1, \ldots, x_n$ of the vertices of a digraph such that $x_ix_{i+1}$, $0 \leq i \leq n-1$, is an edge.
  - The *length of a walk* is the number of edges in the walk.
  - If a walk holds $x_i \neq x_j$ (i\neq j) $i,j=0, \ldots, n$, except $x_0, x_n$(i.e $x_0=x_n$), the walk is called a *cycle*.
  - A *loop* is a cycle of length one.
  - A *sling* is a cycle of length two.
  - A walk is a *path* if no edge is repeated more than once.
Directed Multigraphs

- Like directed graphs, but there may be more than one arc from a node to another.

- **Definition:**
  
  A directed multigraph $G=(V, E, f)$ consists of a set $V$ of vertices, a set $E$ of edges, and a function $f: E \to V \times V$.

- **Example:**
  - The WWW is a directed multigraph.
  - $V$=web pages, $E$=hyperlinks.
Types of Graphs: Summary

- Keep in mind this terminology is not fully standardized...

<table>
<thead>
<tr>
<th>Term</th>
<th>Edge type</th>
<th>Multiple edges ok?</th>
<th>Self-loops ok?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple graph</td>
<td>Undir.</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Multigraph</td>
<td>Undir.</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Pseudograph</td>
<td>Undir.</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Directed graph</td>
<td>Directed</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Directed multigraph</td>
<td>Directed</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Graph Terminology

- Adjacent, connects, endpoints, degree, initial, terminal, in-degree, out-degree, complete, cycles, wheels, n-cubes, bipartite, subgraph, and union.
Adjacency

Let $G$ be an undirected graph with edge set $E$. Let $e \in E$ be (or map to) the pair $\{u,v\}$. Then we say:

- $u$, $v$ are adjacent / neighbors / connected.
- Edge $e$ is incident with vertices $u$ and $v$.
- Edge $e$ connects $u$ and $v$.
- Vertices $u$ and $v$ are endpoints of edge $e$. 
Degree of a Vertex

- Let $G$ be an undirected graph, $v \in V$ a vertex.
- The degree of $v$, $\deg(v)$, is its number of incident edges. (Except that any self-loops are counted twice.)
- A vertex with degree 0 is isolated.
- A vertex of degree 1 is pendant.
Handshaking Theorem

• **Theorem:**
  - Let $G$ be an undirected (simple, multi-, or pseudo-) graph with vertex set $V$ and edge set $E$. Then
  
  \[
  \sum_{v \in V} \deg(v) = 2|E|
  \]

• **Corollary:**
  - Any undirected graph has an even number of vertices of odd degree.
Directed Adjacency

- Let $G$ be a directed (possibly multi-) graph, and let $e$ be an edge of $G$ that is (or maps to) $(u,v)$. Then we say:
  - $u$ is adjacent to $v$, $v$ is adjacent from $u$
  - $e$ comes from $u$, $e$ goes to $v$.
  - $e$ connects $u$ to $v$, $e$ goes from $u$ to $v$
  - the initial vertex of $e$ is $u$
  - the terminal vertex of $e$ is $v$
Directed Degree

• Definition:
  Let $G$ be a directed graph, $v$ a vertex of $G$.
  - The in-degree of $v$, $\deg^-(v)$, is the number of edges going to $v$.
  - The out-degree of $v$, $\deg^+(v)$, is the number of edges coming from $v$.
  - The degree of $v$, $\deg(v) \equiv \deg^-(v) + \deg^+(v)$, is the sum of $v$’s in-degree and out-degree.
Directed Handshaking Theorem

- **Theorem:**
  - Let $G$ be a directed (possibly multi-) graph with vertex set $V$ and edge set $E$. Then:
  \[
  \sum_{v \in V} \deg^-(v) = \sum_{v \in V} \deg^+(v) = \frac{1}{2} \sum_{v \in V} \deg(v) = |E|
  \]

- Note that the degree of a node is unchanged by whether we consider its edges to be directed or undirected.
Special Graph Structures

Special cases of undirected graph structures:
- Complete Graphs $K_n$
- Cycles $C_n$
- Wheels $W_n$
- $n$-Cubes $Q_n$
- Bipartite Graphs
- Complete Bipartite Graphs $K_{m,n}$
Complete Graphs

- **Definition:**
  - For any $n \in \mathbb{N}$, a *complete graph* on $n$ vertices, $K_n$, is a simple graph with $n$ nodes in which every node is adjacent to every other node: $\forall u, v \in V$: $u \neq v \leftrightarrow \{u, v\} \in E$.

- Note that $K_n$ has $\sum_{i=1}^{n-1} \frac{n(n-1)}{2}$ edges.
Cycles

- Definition:
  - For any \( n \geq 3 \), a **cycle** on \( n \) vertices, \( C_n \), is a simple graph where \( V = \{v_1, v_2, \ldots, v_n\} \) and \( E = \{\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{n-1}, v_n\}, \{v_n, v_1\}\} \).

How many edges are there in \( C_n \)?
Wheels

- **Definition:**
  - For any $n \geq 3$, a *wheel* $W_n$ is a simple graph obtained by taking the cycle $C_n$ and adding one extra vertex $v_{\text{hub}}$ and $n$ extra edges $\{v_{\text{hub}}, v_1\}, \{v_{\text{hub}}, v_2\}, \ldots, \{v_{\text{hub}}, v_n\}$.
**$n$-Cubes (hypercubes)**

- **Definition:**
  - A graph that has vertices representing the $2^n$ bit strings of length $n$
  - For any $n \in \mathbb{N}$, the **hypercube** $Q_n$ is a simple graph consisting of two copies of $Q_{n-1}$ connected together at corresponding nodes. $Q_0$ has 1 node.

  \[ Q_0 \quad Q_1 \quad Q_2 \quad Q_3 \quad Q_4 \]

  *Number of vertices: $2^n$. Number of edges: Exercise to try!*
$n$-Cubes (hypercubes)

- **Definition**: For any $n \in \mathbb{N}$, the hypercube $Q_n$ can be defined recursively as follows:
  - $Q_0=({v_0}, \emptyset)$ (one node and no edges)
  - For any $n \in \mathbb{N}$, if $Q_n=(V,E)$, where $V={v_1,\ldots,v_a}$ and $E={e_1,\ldots,e_b}$, then $Q_{n+1}=(V \cup \{{v_1}',\ldots,{v_a}'\}, E \cup \{{e_1}',\ldots,{e_b}'\} \cup \{{v_1},{v_1}'\}, \ldots,\{{v_a},{v_a}'\})$ where ${v_1}',\ldots,{v_a}'$ are new vertices, and where if $e_i=\{v_j,v_k\}$ then ${e_i}'=\{v_j',v_k'\}$. 

Discrete Mathematics, Spring 2009
Bipartite Graphs

• Definition:
  - A simple graph $G$ is called **bipartite** if its vertex set $V$ 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 $V_2$ (so that no edge in $G$ connects either two vertices in $V_1$ or two vertices in $V_2$)
Complete Bipartite Graphs

• Definition:

Let $m$, $n$ be positive integers. The complete bipartite graph $K_{m,n}$ is the graph whose vertices can be partitioned $V = V_1 \cup V_2$ such that

1. $|V_1| = m$
2. $|V_2| = n$
3. For all $x \in V_1$ and for all $y \in V_2$, there is an edge between $x$ and $y$
4. No edge has both its endpoints in $V_1$ or both its endpoints in $V_2$
Complete Bipartite Graphs (cont.)

$K_{2,3}$

$K_{3,4}$
Subgraphs

- *Definition:*
  - A subgraph of a graph $G=(V,E)$ is a graph $H=(W,F)$ where $W \subseteq V$ and $F \subseteq E$. 
Graph Unions

- **Definition:** The union $G_1 \cup G_2$ of two simple graphs $G_1=(V_1, E_1)$ and $G_2=(V_2, E_2)$ is the simple graph $(V_1 \cup V_2, E_1 \cup E_2)$. 
Graph Representations & Isomorphism

- Graph representations:
  - Adjacency lists.
  - Adjacency matrices.
  - Incidence matrices.

- Graph isomorphism:
  - Two graphs are isomorphic iff they are identical except for their node names.
Adjacency Lists

- A table with 1 row per vertex, listing its adjacent vertices.
- A way to represent a graph w/ no multiple edges

<table>
<thead>
<tr>
<th>Vertex</th>
<th>Adjacent Vertices</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b, c</td>
</tr>
<tr>
<td>b</td>
<td>a, c, e, f</td>
</tr>
<tr>
<td>c</td>
<td>a, b, f</td>
</tr>
<tr>
<td>d</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>b</td>
</tr>
<tr>
<td>f</td>
<td>c, b</td>
</tr>
</tbody>
</table>
Directed Adjacency Lists

• 1 row per node, listing the terminal nodes of each edge incident from that node.
Adjacency Matrices

- Matrix $A=[a_{ij}]$, where $a_{ij}$ is 1 if {$v_i, v_j$} is an edge of $G$, 0 otherwise.

\[
\begin{bmatrix}
1 & 0 & 0 & 1 & 1 & 0 \\
2 & 0 & 0 & 1 & 0 & 0 & 1 \\
3 & 0 & 1 & 0 & 0 & 1 & 1 \\
4 & 1 & 0 & 0 & 0 & 0 & 0 \\
5 & 1 & 0 & 1 & 0 & 0 & 0 \\
6 & 0 & 1 & 1 & 0 & 0 & 0 \\
\end{bmatrix}
\]

- Simple graph representation
Adjacency Matrices

- Matrix $M = [m_{ij}]$, where
  - $m_{ij} = 1$ when edge $e_j$ is incident with $v_i$,  
  - $m_{ij} = 0$ otherwise.
- A way of represent graphs
  - can be used to represent multiple edges and loops
Graph Isomorphism

- **Definition:**
  Simple graphs $G_1=(V_1, E_1)$ and $G_2=(V_2, E_2)$ are isomorphic iff \( \exists \) a bijection \( f: V_1 \rightarrow V_2 \) such that \( \forall a, b \in V_1 \), \( a \) and \( b \) are adjacent in \( G_1 \) iff \( f(a) \) and \( f(b) \) are adjacent in \( G_2 \).

- \( f \) is the “renaming” function that makes the two graphs identical.
- Definition can easily be extended to other types of graphs.
Graph Invariants under Isomorphism

• Graph Invariant
  - a property preserved by isomorphism of graphs
  - *Necessary but not sufficient* conditions for \( G_1 = (V_1, E_1) \) to be isomorphic to \( G_2 = (V_2, E_2) \):
    • \( |V_1| = |V_2|, |E_1| = |E_2| \).
    • The number of vertices with degree \( n \) is the same in both graphs.
    • For every proper subgraph \( g \) of one graph, there is a proper subgraph of the other graph that is isomorphic to \( g \).
Isomorphism Example

- If isomorphic, label the 2nd graph to show the isomorphism, else identify difference.
Are These Isomorphic?

- If isomorphic, label the 2nd graph to show the isomorphism, else identify difference.

* Same # of vertices
* Same # of edges
* Different # of verts of degree 2! (1 vs 3)
Connectedness

- **Definition:**
  
  A *path of length n* from $u$ to $v$ in $G$ is a sequence of $n$ edges $e_1, e_2, \ldots, e_n$ such that $e_1$ is associated with \{x_0, x_1\}, $e_2$ is associated with \{x_1, x_2\}, and so on, with $e_n$ associated with \{x_{n-1}, x_n\}, where $x_0 = u$ and $x_n = v$;

- The path is *circuit* if it begins and ends at the same vertex, that is $u = v$, and has length greater than zero.

- A path or circuit is *simple* if it does not contain the same edge more than once.
Connectedness

• **Definition:**
  – An undirected graph is *connected* iff there is a path between every pair of distinct vertices in the graph.

• **Theorem:**
  – There is a *simple* path between any pair of vertices in a connected undirected graph.
Directed Connectedness

• **Definition:**
  - A directed graph is *strongly connected* iff there is a directed path from \( a \) to \( b \) for any two vertices \( a \) and \( b \).
  - It is *weakly connected* iff the underlying *undirected* graph (i.e., with edge directions removed) is connected.

• Note *strongly* implies *weakly* but not vice-versa.
Paths & Isomorphism

• Note that connectedness, and the existence of a circuit or simple circuit of length $k$ are graph invariants with respect to isomorphism.
Euler & Hamilton Paths

• Definition:
  - 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 $G$.

• Examples:
Euler & Hamilton Paths (cont.)

- **Definition:**
  - A *Hamilton circuit* is a simple circuit that traverses each vertex in $G$ exactly once.
  - A *Hamilton path* is a simple path that traverses each vertex in $G$ exactly once.

- **Examples:**

  ![Hamilton circuit and path examples]

  - $a, b, c, d, e, a$
    - Hamilton circuit
  - $a, b, c, d$
    - Hamilton path
Trees

- **Definition:**
  - A **tree** is a connected undirected graph with no simple circuits.

- Since a tree cannot have a circuit, a tree cannot contain multiple edges or loops.

- Therefore, any tree must be a **simple graph**.

- **Theorem:**
  - An undirected graph is a tree if and only if there is a unique simple path between any two of its vertices.

- In general, we use trees to represent **hierarchical structures**.
Trees

Example: Are the following graphs trees?

Yes. Yes.
No. No.
Root & Rooted tree

• We often designate a particular vertex of a tree as the root. Since there is a unique path from the root to each vertex of the graph, we direct each edge away from the root.

• Definition:
  A rooted tree is a tree in which one vertex has been designated as the root and every edge is directed away from the root.
Tree Terminology

• Definition:
  - If $v$ is a vertex in a rooted tree other than the root, the **parent** of $v$ is the unique vertex $u$ such that there is a directed edge from $u$ to $v$.
  - When $u$ is the parent of $v$, $v$ is called the **child** of $u$.
  - Vertices with the same parent are called **siblings**.
  - The **ancestors** of a vertex other than the root are the vertices in the path from the root to this vertex, excluding the vertex itself and including the root.
Tree Terminology (cont.)

- **Definition:**
  - The **descendants** of a vertex $v$ are those vertices that have $v$ as an ancestor.
  - A vertex of a tree is called a **leaf** if it has no children.
  - Vertices that have children are called **internal vertices**.
  - If $a$ is a vertex in a tree, then the **subtree** with $a$ as its root is the subgraph of the tree consisting of $a$ and its descendants and all edges incident to these descendants.
Tree Terminology

• *Definition:*
  - The **level** of a vertex \( v \) in a rooted tree is the length of the unique path from the root to this vertex.
  - The level of the root is defined to be zero.
  - The **height** of a rooted tree is the maximum of the levels of vertices.
Trees

Example 1: Family tree

James

Christine

Bob

Frank

Joyce

Petra

Discrete Mathematics, Spring 2009
Trees (cont.)

Example 2: File system

```
/  
  /  
usr bin temp
  |  |
bin spool ls
```
Trees (cont.)

Example 3: Arithmetic expressions

\[ (y + z) \cdot (x - y) \]

• This tree represents the expression \((y + z) \cdot (x - y)\).
**$m$-ary tree**

- **Definition:**
  - A rooted tree is called an $m$-ary tree if every internal vertex has no more than $m$ children.
  - The tree is called a **full $m$-ary tree** if every internal vertex has exactly $m$ children.
  - An $m$-ary tree with $m = 2$ is called a **binary tree**.

- **Theorem:**
  - A tree with $n$ vertices has $(n – 1)$ edges.
  - A full $m$-ary tree with $i$ internal vertices contains $n = m \cdot i + 1$ vertices.
Ordered Rooted Tree

- Definition:
  - An ordered rooted tree is a rooted tree where the children of each internal vertex are ordered.

- For example, in an ordered binary tree (just called binary tree), if an internal vertex has two children,
  - the first child is called the left child and the second is called the right child.
  - the tree rooted at the left child is called the left subtree, and at the right child, the right subtree.

- Ordered rooted trees can be defined recursively.
Ordered Rooted Tree (binary tree)

- Left child of the Vertex a
- Left subtree of the Vertex b
- Right child of the Vertex a
- Right subtree of the Vertex b
Tree Traversal

- Procedures for systematically visiting every vertex of an ordered rooted tree are called traversal algorithms.
Preorder traversal

- **Definition:**
  Let $T$ be an ordered rooted tree with root $r$. If $T$ consists only of $r$, then $r$ is the *preorder traversal* of $T$. Otherwise, suppose that $T_1, T_2, \ldots, T_n$ are the subtrees at $r$ from left to right in $T$. The *preorder traversal* begins by visiting $r$. It continues by traversing $T_1$ in preorder, then $T_2$ in preorder, and so on, until $T_n$ is traversed in preorder.
Inorder traversal

- **Definition:**
  Let $T$ be an ordered rooted tree with root $r$.
  If $T$ consists only of $r$, then $r$ is the *inorder traversal* of $T$.
  Otherwise, suppose that $T_1, T_2, \ldots, T_n$ are the subtrees at $r$ from left to right. The *inorder traversal* begins by traversing visiting $T_1$ in inorder, then visiting $r$. It continues by traversing $T_2$ in inorder, then $T_3$ in inorder, ..., and finally $T_n$ in inorder.
Postorder traversal

- **Definition:**
  Let $T$ be an ordered rooted tree with root $r$.
  If $T$ consists only of $r$, then $r$ is the **postorder traversal** of $T$.
  Otherwise, suppose that $T_1, T_2, \ldots, T_n$ are the subtrees at $r$ from left to right. The **postorder traversal** begins by traversing $T_1$ in postorder, then $T_2$ in postorder, ..., then $T_n$ in postorder, and ends by visiting $r$. 

Discrete Mathematics, Spring 2009
**Traversal Example**

- **Preorder**: a, b, e, j, k, n, o, p, f, c, d, g, l, m, h, I
- **Inorder**: j, e, n, k, o, p, b, f, a, c, l, g, m, d, h, i
- **Postorder**: j, n, o, p, k, e, f, b, c, l, m, g, h, i, d, a