

CSE 21—Mathematics for Algorithm and System Analysis

Spring, 2006

June 1, 2006

Lecture 18

Graph Theory

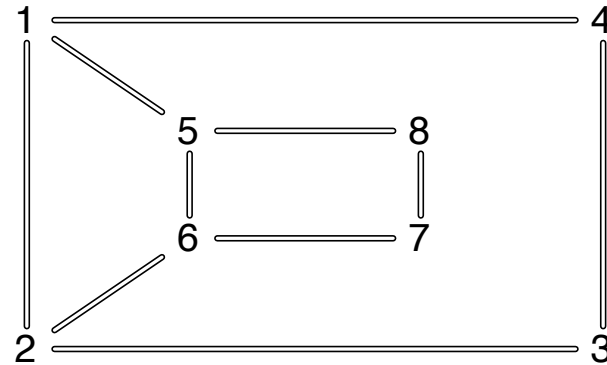
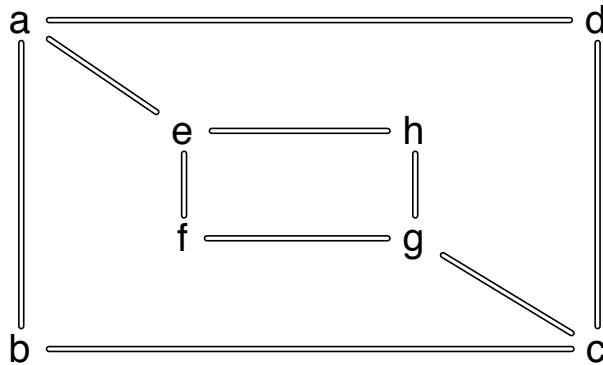
Instructor: Neil Rhodes

Isomorphism

Must match (invariant for isomorphism)

- Number of vertices
- Number of edges
- Has a vertex of degree k
- Has m vertices of degree k
- Has a circuit of length k
- is connected

Example:



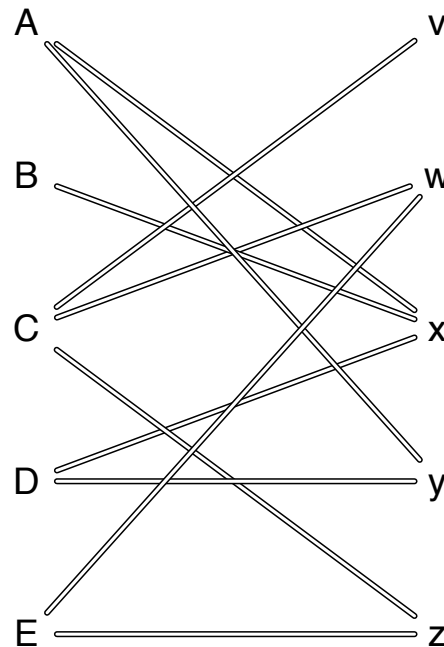
Bipartite graph

A graph whose vertices can be partitioned into two sets: V_1 and V_2 where:

- The only edges are between vertices in V_1 and V_2

Theorem:

- G is bipartite iff every circuit in G has even length

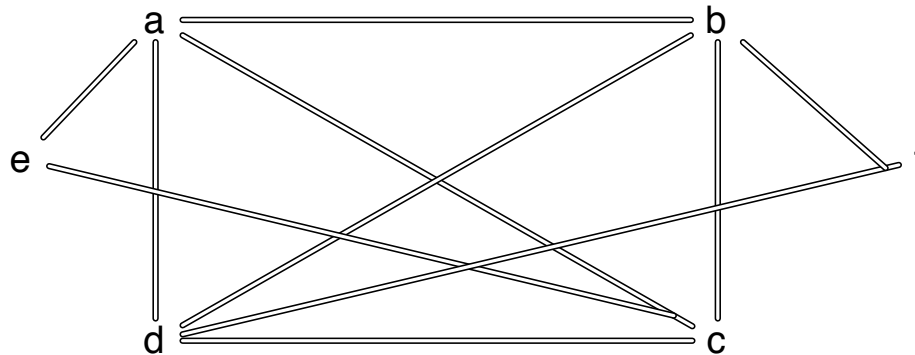


Planar graphs

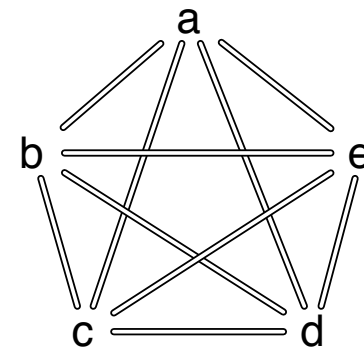
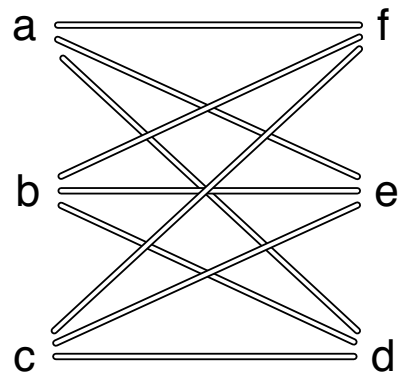
Graphs that can be drawn in the plane with no crossing edges

Plane graph

- A planar depiction of a planar graph



Some non-planar graphs



Theorem

A graph is planar iff it does not contain a subgraph that is a K_5 or $K_{3,3}$ configuration

Definition

- A $K_{3,3}$ (K_5) configuration is a graph that can be obtained from a $K_{3,3}$ (K_5) by adding vertices in the middle of some edges

Definitions

Multigraph

- We allow multiple edges between pairs of vertices (and allow loop edges between a single vertex)

Trail

- Like a path, sequence of vertices: $V_1-V_2-\dots-V_n$, except:
 - Vertices can be repeated
 - Edges cannot can be repeated

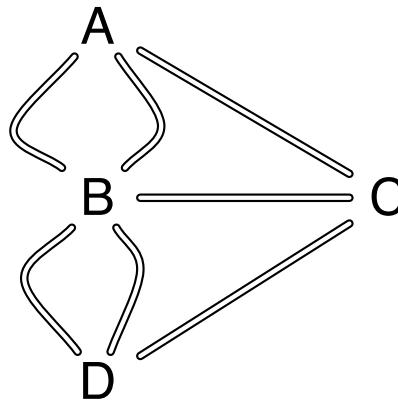
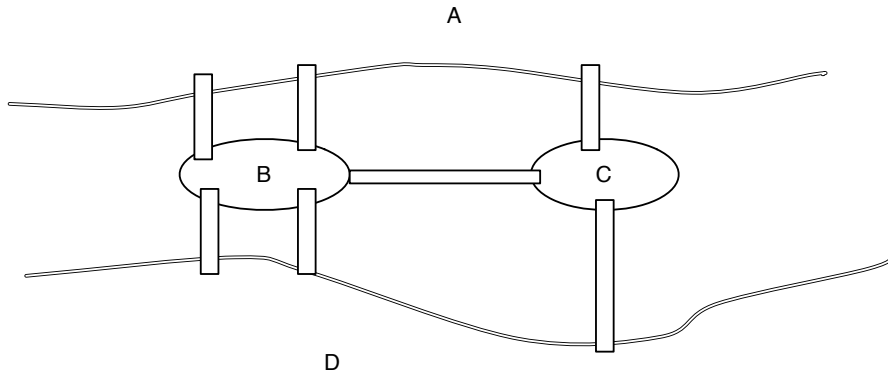
Cycle

- Like a trail, but with an edge (V_n, V_1)

Konigsberg bridges

Euler cycle: A cycle that

- Uses all vertices
- Contains all edges in the multigraph



First theorem of graph theory

A multigraph contains an Euler cycle iff:

- The graph is connected
- Each vertex has even degree

Corollary: A multigraph has an Euler trail iff:

- it is connected
- Exactly two vertices have odd degree

Summary

	Repeated edge?	Repeated vertex?	Starts and ends at same
Walk	allowed	allowed	allowed
Path	allowed	no	no
Circuit	allowed	no	yes
Trail	no	allowed	no
Cycle	no	allowed	yes

Hamiltonian Circuit

Similar to Euler cycle except:

- Must visit all vertices exactly once (rather than all edges)

Example

Trees

Definition

- If G is a connected graph without any cycles, then G is a *tree*

Equivalent definitions

- If G is a connected graph then the following are equivalent:
 - G is a tree
 - G has no cycles
 - For every pair of distinct vertices u, v in G , there is exactly one path from u to v
 - Removing any edge from G gives a graph which is not connected

Forest: a graph whose connected components are all trees

Rooted Tree

Definition

- A rooted tree, T , is a pair (G, r) where G is a tree and r is a vertex of V . r is the *root* of the tree. Rooted trees are normally directed graphs.
- For every vertex, w , other than r , given a unique path from r to w : $\langle r, \dots v_k, w \rangle$ v_k is the *parent* of w , and w is a *child* of v_k .
- Vertices that have no children are called *leaves*
- Vertices that have children are called *internal* vertices
- Vertices with the same parent are called *siblings*

Binary tree

- Each vertex has at most 2 children

M-ary tree

- Each vertex has at most m children

Properties of Trees

A tree with n vertices has $n-1$ edges.

A binary tree with n internal vertices has $n+1$ leaves

An m -ary tree of height h has at most m^h leaves.

Breadth/Depth-First Search

Breadth-First Search (BFS)

- Add root to queue
while queue is not empty
 retrieve vertex from head of queue
 print vertex
 add children (in order) to tail of queue

Depth-First Search (DFS)

- Visit(root)
- procedure Visit(vertex) *preorder traversal*
 print vertex
 foreach child of vertex
 visit(child)

Spanning Tree

Definition

- A *spanning tree* of a simple graph $G=(V,E)$ is a subgraph $T=(V,E')$ which is a tree

Minimal Spanning Tree

- Given a connected weighted graph $G=(V,E,W)$ (where W is a function with domain E and codomain $\mathbb{R}$)
- A minimal spanning tree $T=(V,E',W')$ of G is a spanning tree whose sum of weights is no more than that of any other spanning tree of G

Minimal spanning Tree

Generating a minimal spanning tree for a simple graph $G=(V,E,W)$ (Prim's)

- Start with $E'=\{\}$
- Start with $V'=\{v\}$ for some v in V .
- While $|V'| < |V|$
 - Find the edge e from E with exactly one edge in V' of minimum weight
 - Add e to E'
 - Add the other vertex of e to V'

Alternative (Kruskal's)

- Start with $E'=\{\}$
- While $T=(V,E')$ is not connected
 - Find the cheapest edge from E that doesn't create a cycle in (V,E')
 - Add e to E'