Graph Theory

PROF. NAVNEET GOYAL (CSIS)

Most of the figures and examples have been taken/adopted from the Book on Discrete Structure and its Applications, 7e, by Kenneth Rosen, McGraw Hill

Types of Graphs

TABLE 1 Graph Terminology.

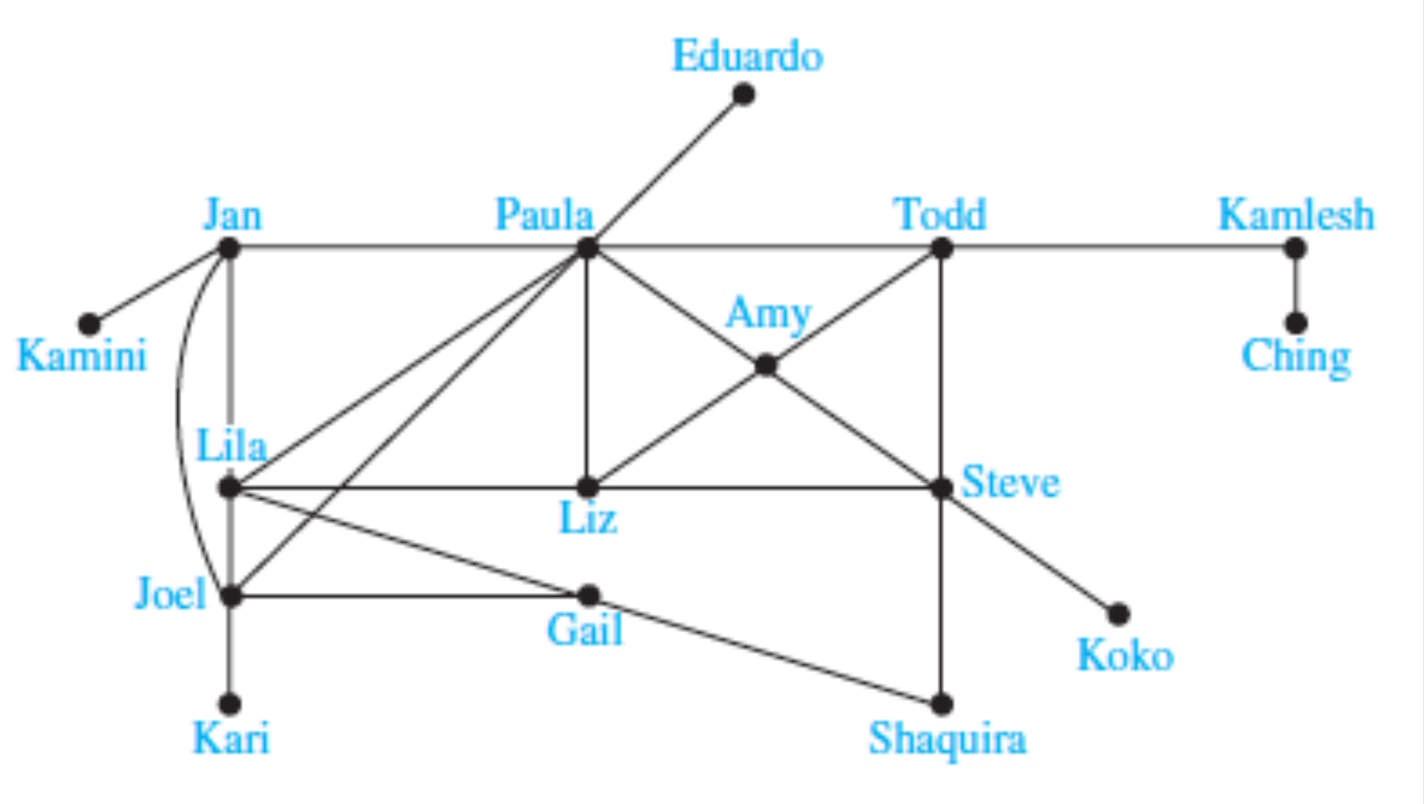
Type	*Edges*	*Multiple Edges Allowed?*	*Loops Allowed?*
Simple graph	Undirected	No	No
Multigraph	Undirected	Yes	No
Pseudograph	Undirected	Yes	Yes
Simple directed graph	Directed	No	No
Directed multigraph	Directed	Yes	Yes
Mixed graph	Directed and undirected	Yes	Yes

Why Study Graphs & Trees?

- Graphs

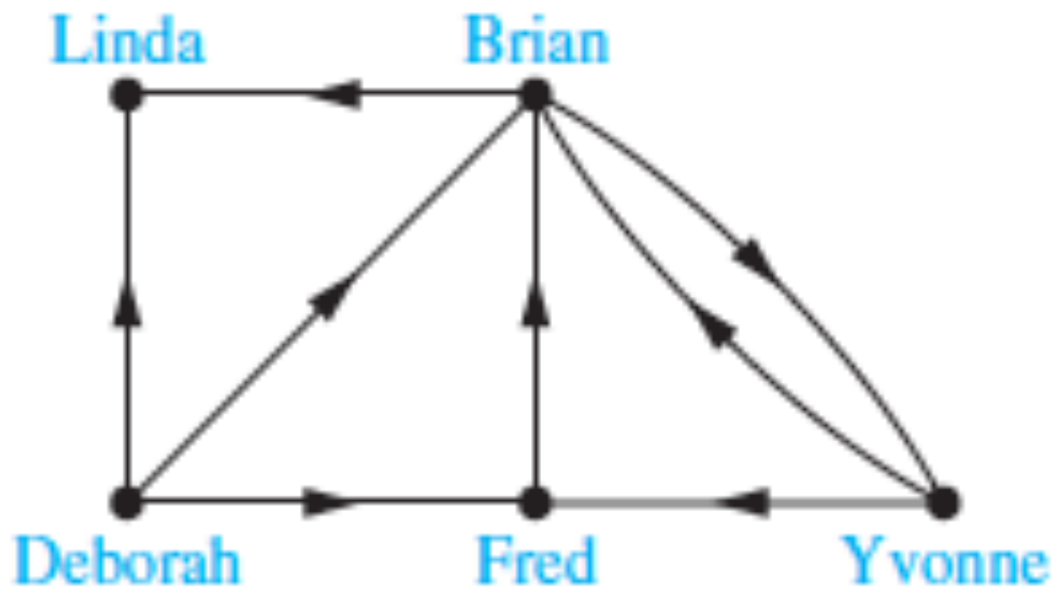
  G=(V,E), where E is a set of edges (ordered or unordered)
  - Mainly used for modeling
    - Website
    - File system
    - Social Networks
- Trees
  - Simplified graphs
  - Mainly used for modeling
    - Website
    - File system
    - Indexing structures

Graphs and Social Networks



- Acquaintanceship Graph

Graphs and Social Networks

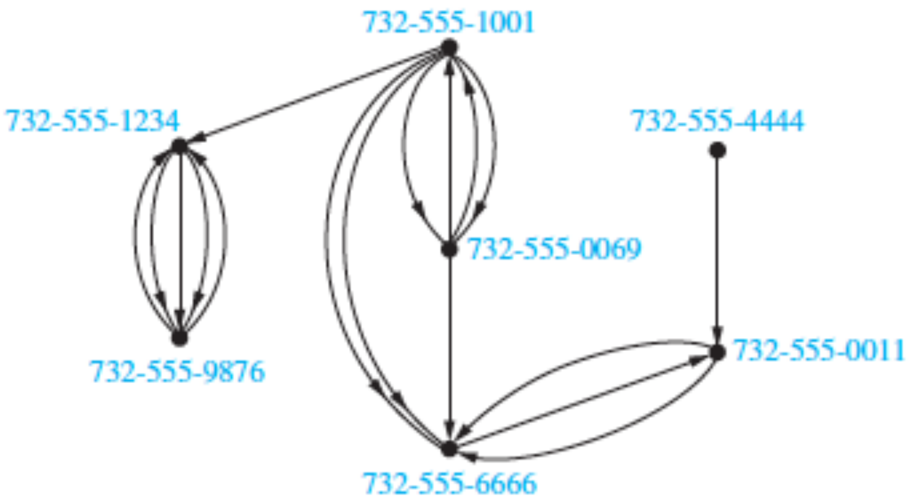


- Influence Graph

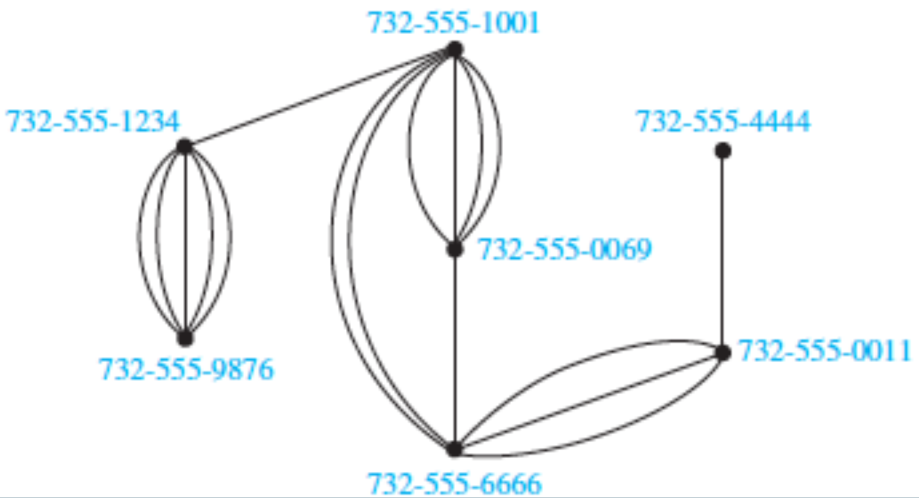
Some Interesting Graphs

- Collaboration Graphs
  - Hollywood Collaboration Graph
  - Academic Collaboration Graph
- Call Graphs
- Citation Graphs
- ??

Call Graphs



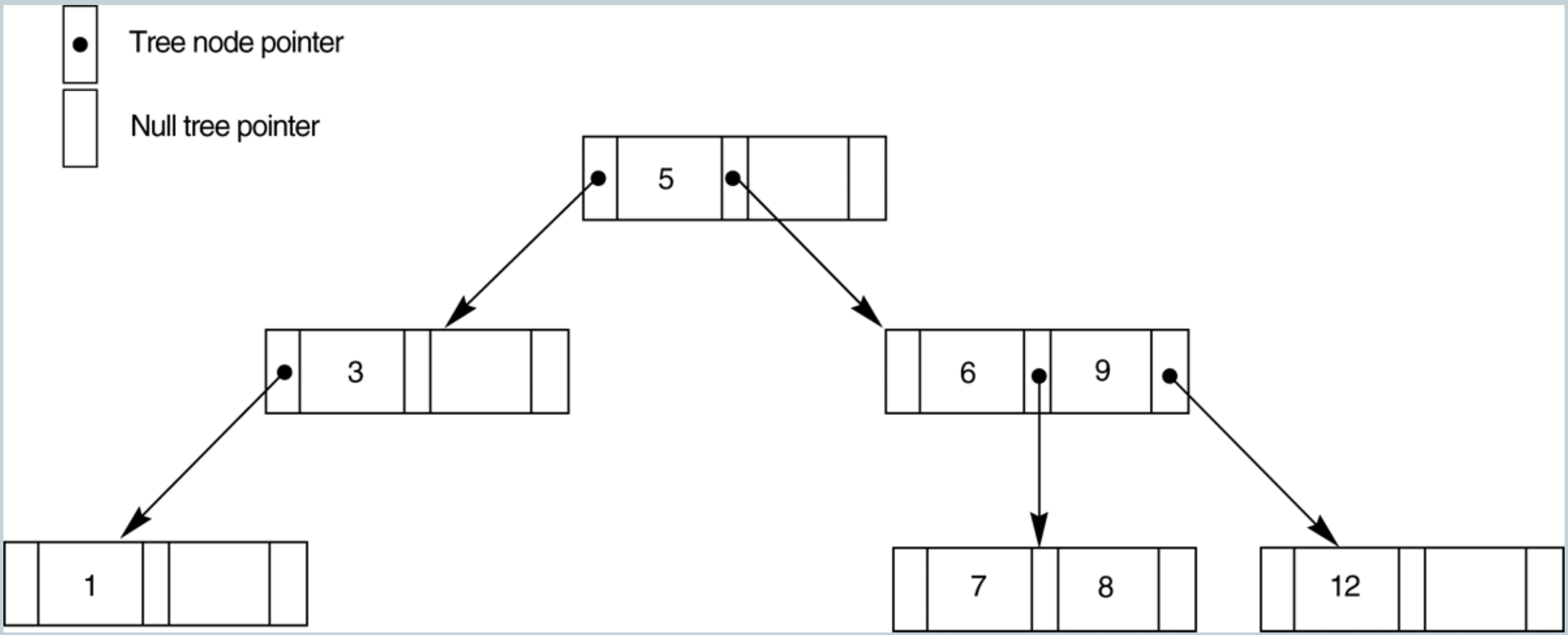
732-555-1001
732-555-1234
732-555-4444
732-555-0069
732-555-0011
732-555-9876
732-555-6666



732-555-1001
732-555-1234
732-555-4444
732-555-0069
732-555-0011
732-555-9876
732-555-6666

Why Study Graphs & Trees?

- Search tree of order 3



Definitions

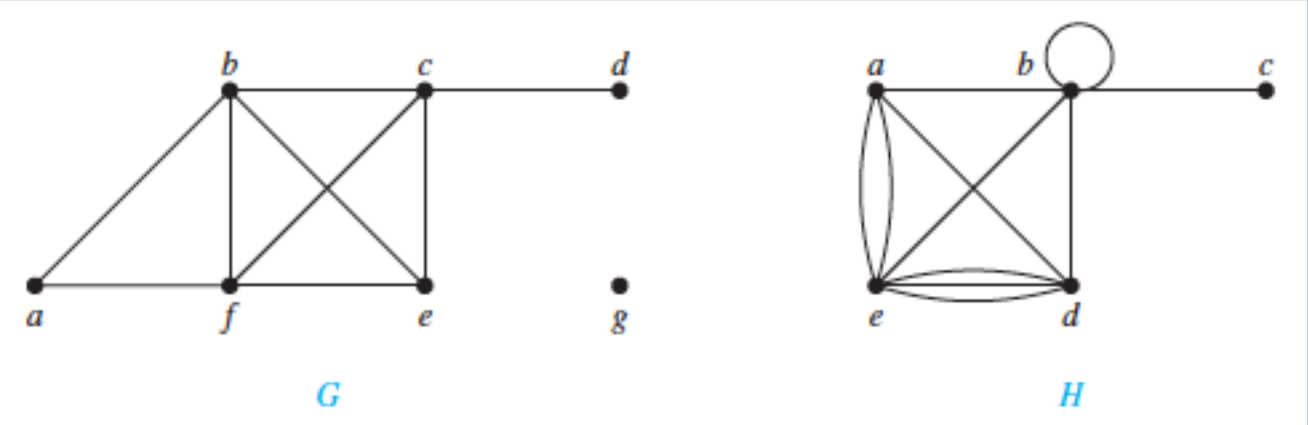
- Graph is a pair of sets G(V,E), where V is the set of vertices & E is the set of edges
- Digraph – elements of E are ordered pairs of vertices (u,v) u➔v
- Non-directed graph - elements of E are unordered pairs of vertices (u,v) u-v
- Loop: v-v
- Simple graph – graph without loops
- V(G) & E(G) – sets of vertices & sets of edges in G
- |V(G)| - # of vertices (order of G)
- |E(G)| - # of edges (size of G)

Definitions

- Multi-graphs: more than 1 edge to join 2 vertices
- In-degree(+) & out-degree (-) of vertices
- In unordered graph – loop counted twice & every other edge once
- Min degree – δ(G)
- Max degree – Δ(G)
- If δ(G)= Δ(G)=k, graph is k-regular (3-regular: cubic)
- Degree sequence of G – order vertices so that the degree sequence $\{d_1, d_2, \ldots d_n\}$ is monotonically increasing from δ(G) to Δ(G).

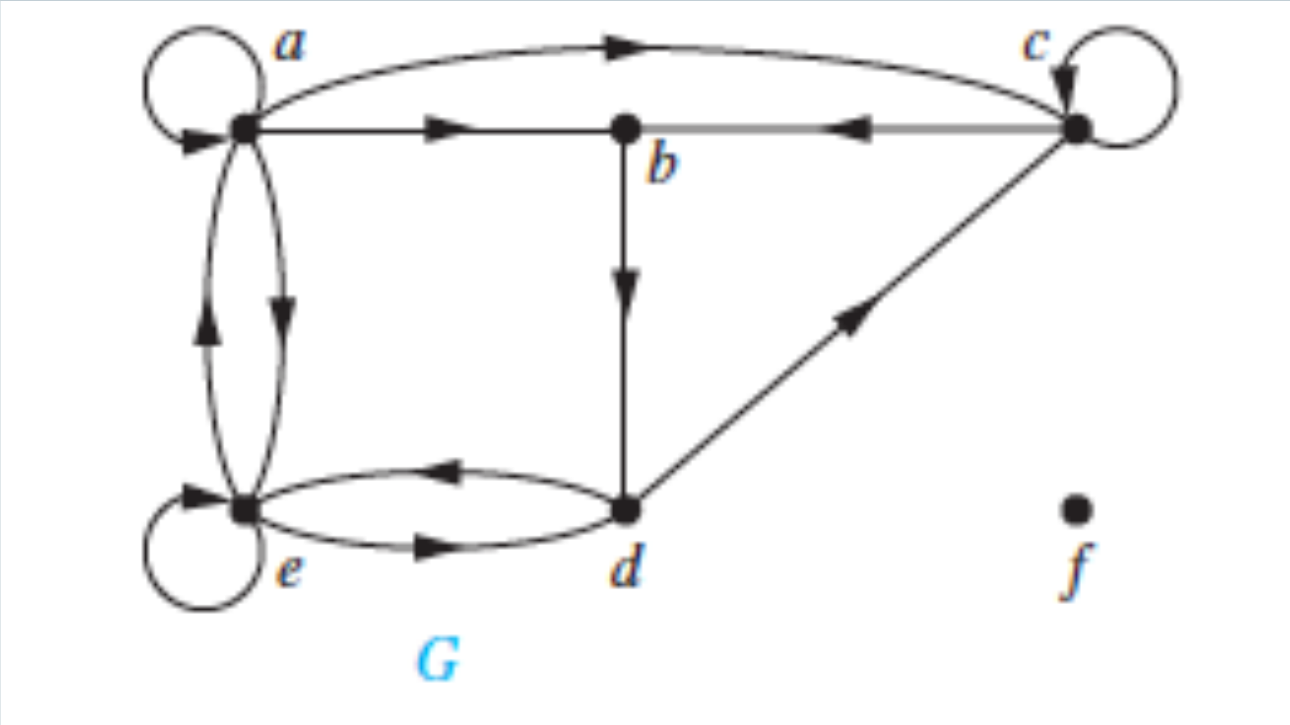
Definitions

- Adjacent or neighboring vertices in undirected graphs
- Degree of a vertex in undirected graphs – loop counted twice
- Isolated vertex
- Pendant vertex



Definitions

- In-degree & Out-degree



First Results

- Theorem 5.1.1 – If V $= \{v_1, v_2, \ldots v_n\}$ is the vertex set of a non-directed graph G, then

$$\sum_{i=1}^{n} \deg(v_i) = 2|E|$$

called the Handshaking Theorem.

If G is a directed graph then,

$$\sum_{i=1}^{n} \deg^{+}(v_i) = \sum_{i=1}^{n} \deg^{-}(v_i) = |E|$$

Proof: when degrees are summed, each edge contributes a count of 1 to the degree of each of the 2 vertices on which the edge is incident

First Results

- How many edges are there in a graph with 10 vertices, each of degree 6?
- Handshaking theorem says that the sum of the number of vertices in an undirected graph is even
- Many consequences of this simple fact:

First Results

- Corollary 5.1.1 – In any non-directed graph, there is an even no. of vertices with odd degree.
- Proof?
- Corollary 5.1.2 – if k=δ(G), in a non-directed graph G, then:

$$k\,|V| \leq \sum_{v \in V(G)} \deg(v) = 2\,|E|$$

- Proof?
- What if G is a k-regular graph?

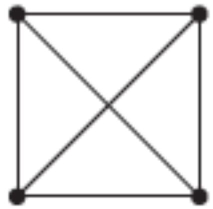
Special Graphs

- Complete Graphs K_n

K_1

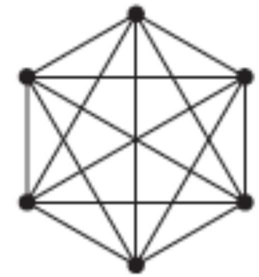
K_2

K_3



K_4

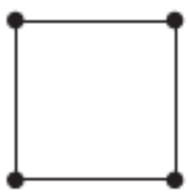
K_5



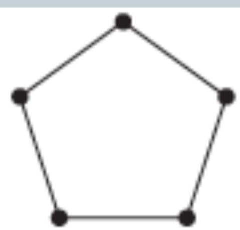
K_6

- Cycles C_n $(n>=3)$

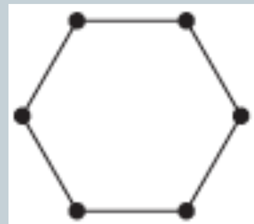
C_3



C_4



C_5

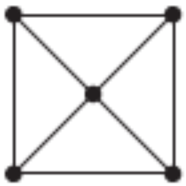


C_6

Special Graphs

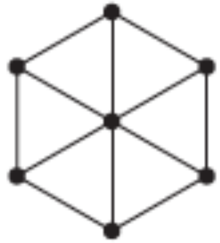
- Wheels W_n *(Add an additional vertex to a cycle)*

W_3



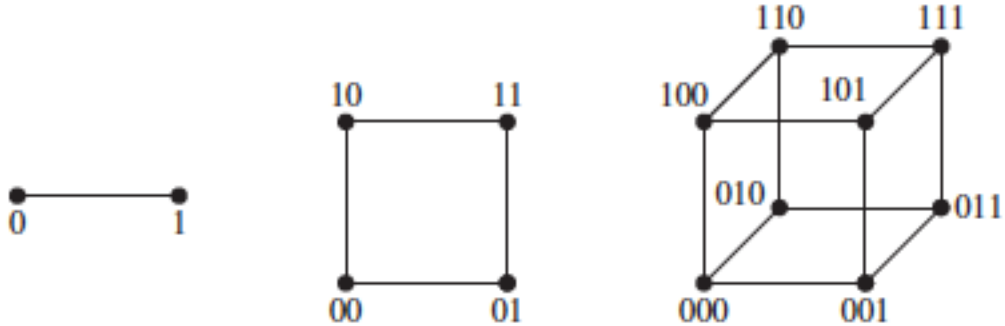
W_4

W_5



W_6

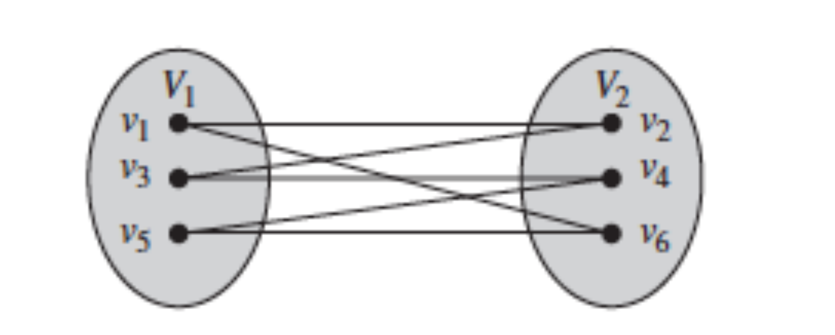
- *N-Cubes*

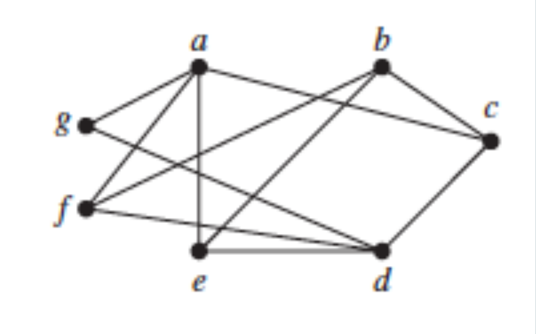


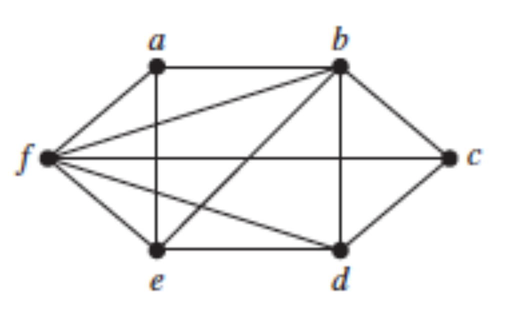
Q_1 Q_2 Q_3

Special Graphs

- Bipartite Graphs – vertex set can be divided into 2 disjoint subsets such that each edge connects a vertex in one of these subsets to a vertex in other subset







Special Graphs

- Bipartite Graphs
  - marriages between men & women?
  - Is C_6 bipartite?
  - Is K_3 bipartite?
  - Is K_n bipartite?

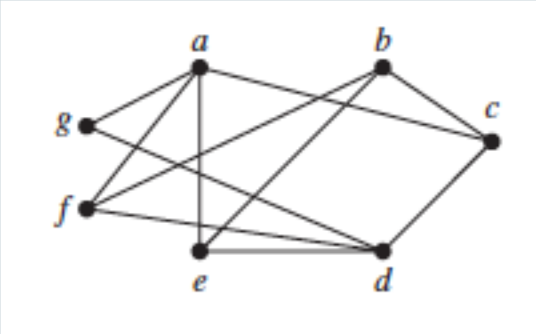
How can we determine that a given graph is bipartite or not?

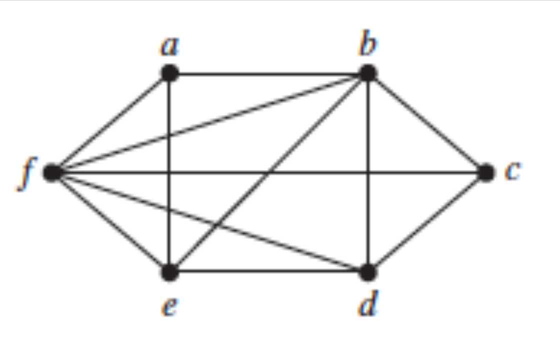
Bipartite Graphs

- *Theorem:* A simple graph is bipartite iff it is possible to assign one of two different colors to each vertex so that no two adjacent vertices are assigned the same color.
- Proof?

Bipartite Graphs

- *Theorem:* A simple graph is bipartite iff it is possible to assign one of two different colors to each vertex so that no two adjacent vertices are assigned the same color.
- Use the theorem to determine whether the graphs below are bipartite





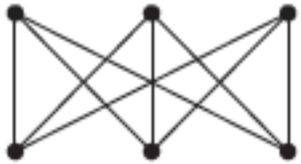
- We have just introduced graph coloring!!!

Bipartite Graphs

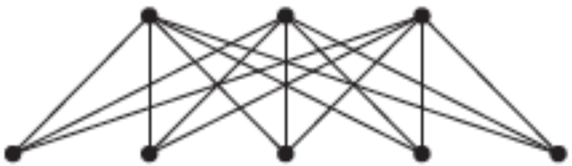
- Another criterion for determining if a graph is bipartite or not is based on the notion of a path.
- A graph is bipartite iff it is not possible to start at a vertex and return to this vertex by traversing an odd no. of distinct edges.
- Coming soon!!

Complete Bipartite Graphs: $K_{m:n}$

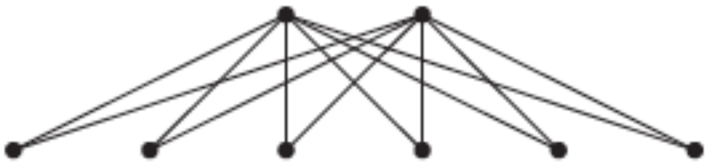
$K_{2,3}$



$K_{3,3}$



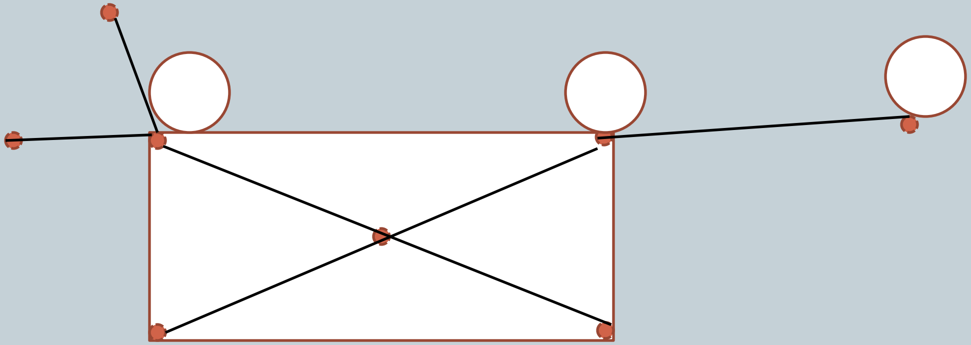
$K_{3,5}$



$K_{2,6}$

First Results

- Is there a graph with degree sequence {1,3,3,3,5,6,6}?
- Is there a simple graph with degree sequence {1,1,3,3,3,4,6,7}?
- Is there a non-simple graph with degree sequence {1,1,3,3,3,4,6,7}



Some more Definitions

- Isolated vertex – a vertex with degree zero
- Adjacent vertices or neighbors – if there is an edge (u,v) then u & v are neighbors
- Path – In a NDG, G, a sequence P of zero or more edges v_o-v_1-…-v_n is called a path from v_o to v_n.
- Open & closed paths – vertices & edges may be repeated in a path. If $v_o = v_n$, then P is a closed path and if $v_o \,!= v_n$, then P is an open path.
- $V(P) = \{v_o, v_1, \ldots v_n\}$ and E(P) are subsets of V(G) & E(G) resp.
- Simple path – if all the edges & vertices on the path are distinct, except possibly the end point

Some more Definitions

- Length of a path – no. of edges in P
- P has no edges – length =0, P is called a trivial path. V(P) is a singleton set $\{v_o\}$
- Circuit – path of length ≥ 1 with no repeated edges and whose endpoints are equal is called a circuit. It may have repeated vertices other than the endpoints
- Cycle – a circuit with no repeated vertices except its endpoints
- A loop is a cycle of length 1!

Graph Representations

- Adjacency Lists
- Adjacency Matrices

Graph Representations

- Adjacency Lists

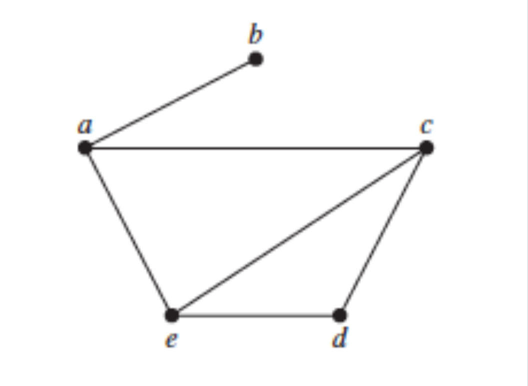


TABLE 1 An Adjacency List for a Simple Graph.

Vertex	*Adjacent Vertices*
a	b, c, e
b	a
c	a, d, e
d	c, e
e	a, c, d

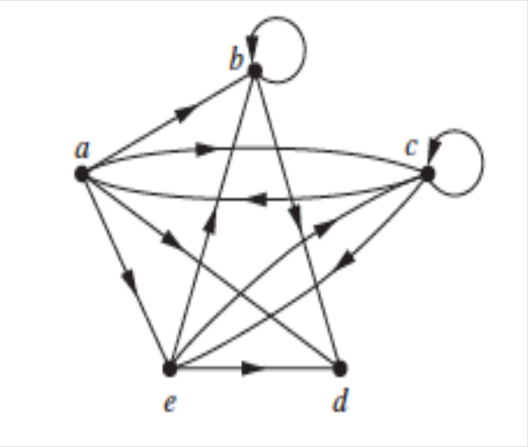
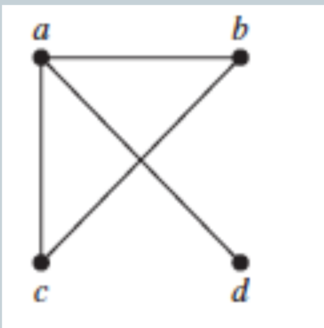


TABLE 2 An Adjacency List for a Directed Graph.

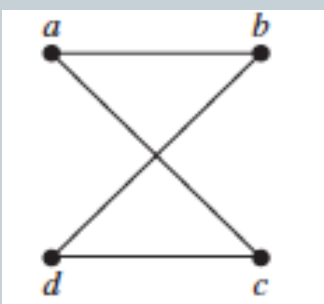
Initial Vertex	*Terminal Vertices*
a	b, c, d, e
b	b, d
c	a, c, e
d	
e	b, c, d

Graph Representations

- Adjacency Matrices:



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



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

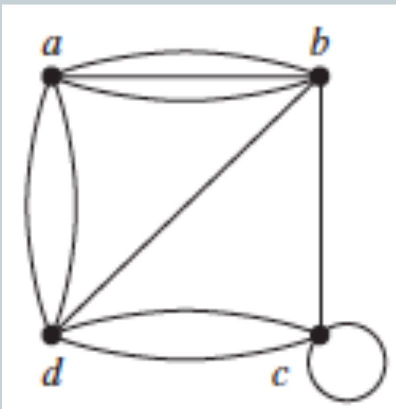
Graph Representations

Adjacency Matrices:

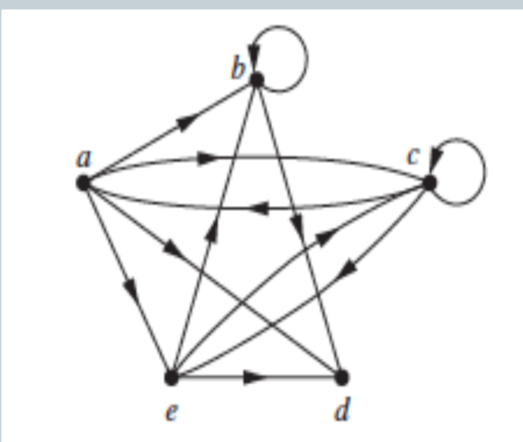
- Ordering of vertices?
- How many adjacency matrices for a graph with n-vertices?
- Adjacency matrix of a simple graph is ________ ?
- Can adjacency matrices be used to represent undirected graphs with loops and with multiple edges?
- Adjacency matrices for representing directed graphs?
- Adjacency matrices for representing directed multi-graphs?

Graph Representations

Adjacency Matrices:



$$\begin{bmatrix} 0 & 3 & 0 & 2 \\ 3 & 0 & 1 & 1 \\ 0 & 1 & 1 & 2 \\ 2 & 1 & 2 & 0 \end{bmatrix}$$



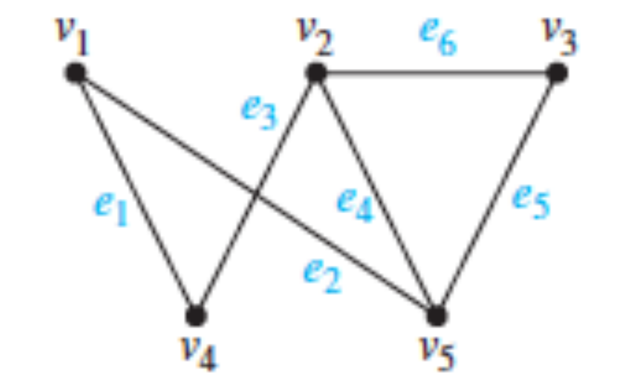
Graph Representations

Adjacency Lists vs. Adjacency Matrices

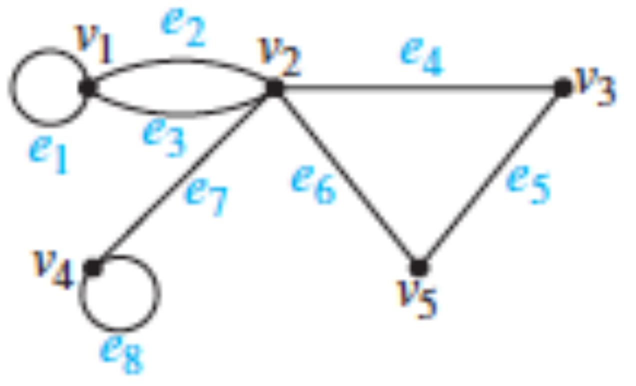
- Ordering of vertices?
- How many adjacency matrices for a graph with n-vertices?
- Adjacency matrix of a simple graph is ________ ?
- Can adjacency matrices be used to represent undirected graphs with loops and with multiple edges?
- Adjacency matrices for representing directed graphs?
- Adjacency matrices for representing directed multi-graphs?

Graph Representations

Incidence Matrices



$$
\begin{array}{c}
\\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5
\end{array}
\begin{array}{c}
\begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array} \\
\begin{bmatrix}
1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 \\
1 & 0 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 1 & 0
\end{bmatrix}.
\end{array}
$$



$$
\begin{array}{c}
\\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5
\end{array}
\begin{array}{c}
\begin{array}{cccccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 \end{array} \\
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0
\end{bmatrix}.
\end{array}
$$

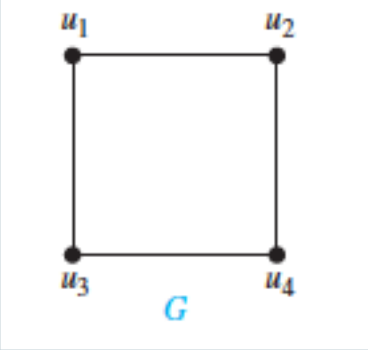
Graph Isomorphism

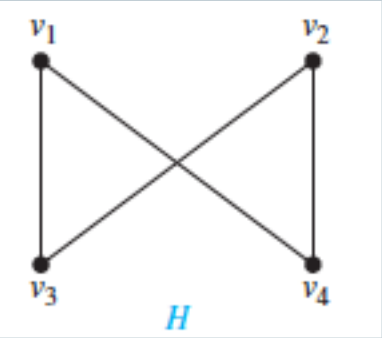
- Is it possible to draw two graphs in the same way?
- Do two graphs have the same structure when we ignore the identities of the vertices?

Two simple graphs $G_1(V_1,E_1)$ & $G_2(V_2,E_2)$ are said to be isomorphic if there exists a 1-1 onto function from V_1 to V_2 with the property that a & b are adjacent in G_1 iff *f(a)* and *f(b)* are adjacent in G_2, for all a & b in V_1.

- When two graphs are isomorphic, there is a 1-1 correspondence between the vertices of the two graphs such that adjacency is preserved.
- Isomorphism between simple graphs is an equivalence relation!! (show it)

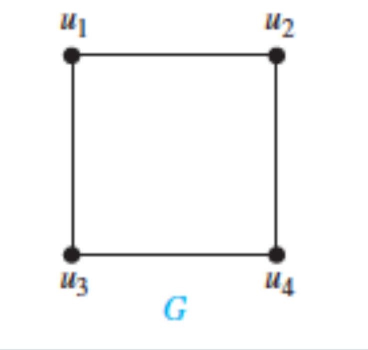
Graph Isomorphism

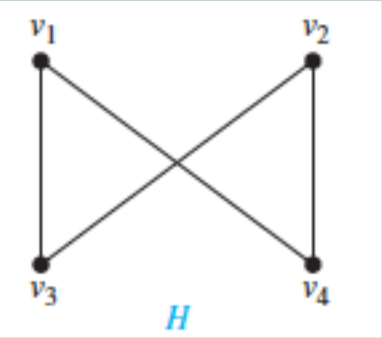




- Why it is often difficult to determine if two simple graphs are isomorphic?
- n! possible 1-1 correspondences between the vertex sets of 2 graphs, each having n vertices!!
- Testing for adjacency preservation for each correspondence is impractical for large n!!
- Sometimes it is not too hard to show that two graphs are not isomorphic

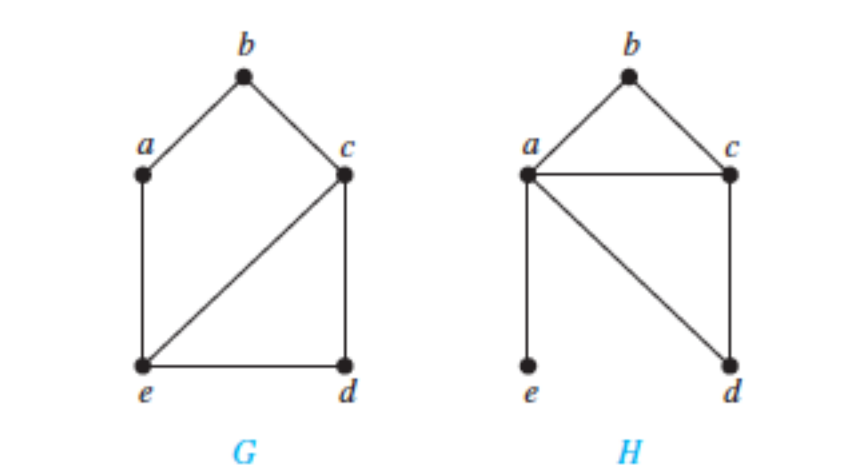
Graph Isomorphism

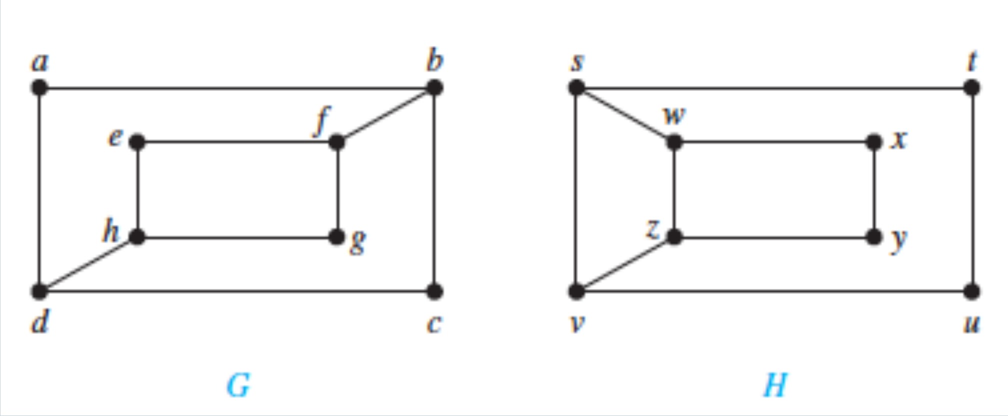




- Same number of vertices
- Same number of edges
- Degrees of corresponding vertices must be same
- Degree sequences must be same?

Graph Isomorphism





Graph Isomorphism

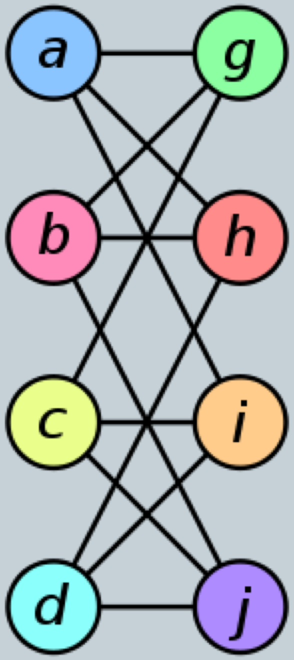
- Two graphs, G & G’ are isomorphic, if there is a function f: V(G) ➔ V(G’) such that
  1. f is 1-1
  2. f is onto
  3. For each pair of vertices u & v of G, {u,v} ∈ E(G) iff {f(u), f(v)} ∈ E(G’) [adjacency preservation property]

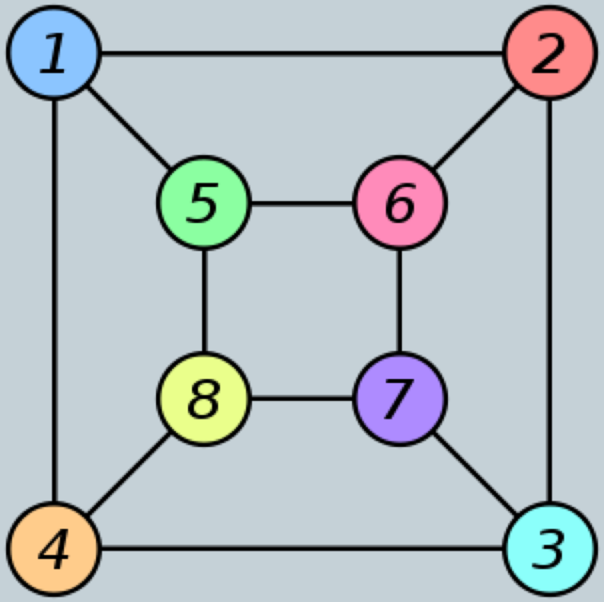
f is called isomorphism from G to G’.

Graph Isomorphism

- Graph Isomorphism

$f(a)=1$
$f(b)=6$
$f(c) = 8$
$f(d) = 3$
$f(g) = 5$
$f(h) = 2$
$f(i) = 4$
$f(j) = 7$





G & G' – isomorphic if there exists a fn f: V(G) ➔ V(G') if f is 1-1 onto and for each pair of vertices u & v of G belonging to E(G) iff f(u), f(v) belong to E(G')

Graph Isomorphism

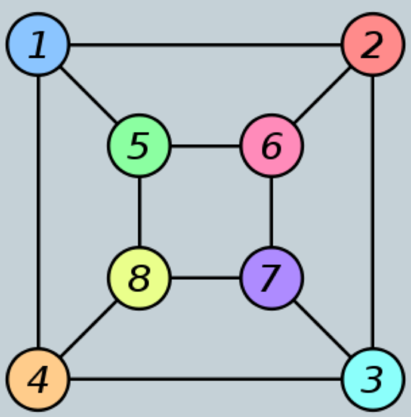
- Change the names of vertices of *G'* from *f(v)* to *v* for each $v \in V(G)$, then *G'* with the newly named vertices would be identical to *G*
- If *G* & *G'* are isomorphic, then the isomorphism f need not be unique.
- If f is an isomorphism between *G* & *G'*, then
  - $|V(G)| = |V(G')|$
  - $|E(G)| = |E(G')|$
  - If $v \in V(G)$, then $deg_G(v) = deg_{G'}(f(v))$ & thus the degree sequences of *G* & *G'* are the same
  - Loops & cycles are preserved under *f*

Discovering Isomorphism

- Isomorphism problem – determining whether or not two graphs are isomorphic
- Algorithms that guarantee a correct answer would require approx. 2^n operations, where n is the number of vertices
- Application: organic chemistry:
  - determining if two molecules are identical

Discovering Isomorphism

- So what's an Adjacency matrix?
- If $\{v_1, v_2, \ldots v_n\}$ are the vertices of *G*, then the adjacency matrix for G is an nxn matrix *A*, where the ijth entry *A(i,j)* of *A* is 1 iff the edge (v_i, v_j) is an edge of *G*, otherwise *A(i,j)*=0.
- Symmetric matrix

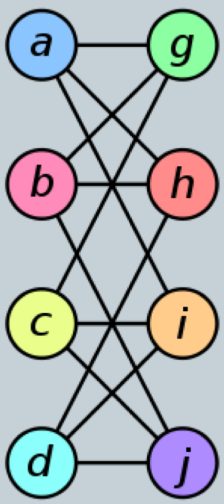


0	1	0	1	1	0	0	0
1	0	1	0	0	1	0	0
0	1	0	1	0	0	1	0
1	0	1	0	0	0	0	1
1	0	0	0	0	1	0	1
0	1	0	0	1	0	1	0
0	0	1	0	0	1	0	1
0	0	0	1	1	0	1	0

Discovering Isomorphism

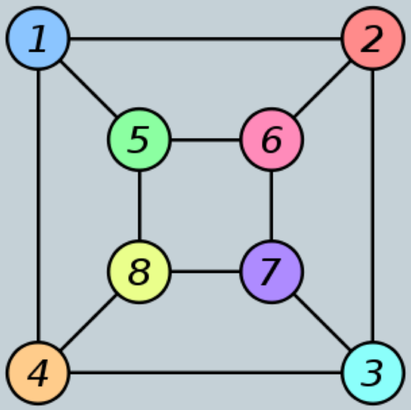
- Result: if we have a 1-1 & onto mapping between G & G'. Let A be the adj. matrix of G with vertex ordering $\{v_1, v_2, \ldots v_n\}$
  Let A' be the adj. matrix of G' with vertex ordering $\{f(v_1), f(v_2), \ldots f(v_n)\}$
  then f is isomorphic from G to G' iff A=A'

$f(a)=1$
$f(b)=6$
$f(c) = 8$
$f(d) = 3$
$f(g) = 5$
$f(h) = 2$
$f(i) = 4$
$f(j) = 7$

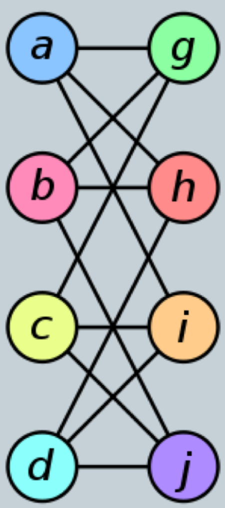


0	1	0	1	1	0	0	0
1	0	1	0	0	1	0	0
0	1	0	1	0	0	1	0
1	0	1	0	0	0	0	1
1	0	0	0	0	1	0	1
0	1	0	0	1	0	1	0
0	0	1	0	0	1	0	1
0	0	0	1	1	0	1	0

Discovering Isomorphism



0	1	0	1	1	0	0	0
1	0	1	0	0	1	0	0
0	1	0	1	0	0	1	0
1	0	1	0	0	0	0	1
1	0	0	0	0	1	0	1
0	1	0	0	1	0	1	0
0	0	1	0	0	1	0	1
0	0	0	1	1	0	1	0



0	1	0	1	1	0	0	0
1	0	1	0	0	1	0	0
0	1	0	1	0	0	1	0
1	0	1	0	0	0	0	1
1	0	0	0	0	1	0	1
0	1	0	0	1	0	1	0
0	0	1	0	0	1	0	1
0	0	0	1	1	0	1	0

Discovering Isomorphism

- What if A≠A'?
- Then can we conclude that G & G' are not isomorphic??
- Or it is just that f is not isomorphic?
- G & G' may still be isomorphic under some other f?
- Can we do any better than having to look at all the possible 1-1 & onto mappings between G & G'?

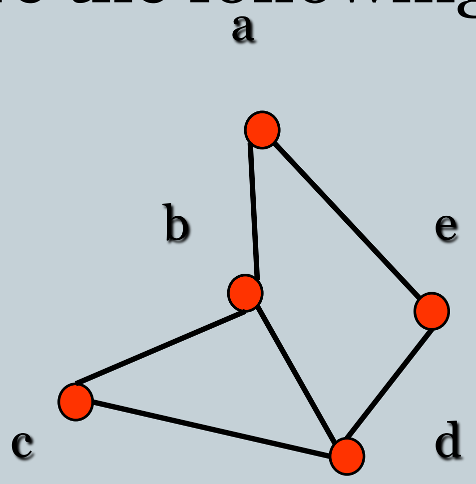
  (brute force method)
- Can we consider a smaller set of mappings?
- Observation: an isomorphism maps a vertex of G of degree d to a vertex of G' of degree d (degree is preserved)
- In many situations, this will shorten the search for isomorphisms considerably!

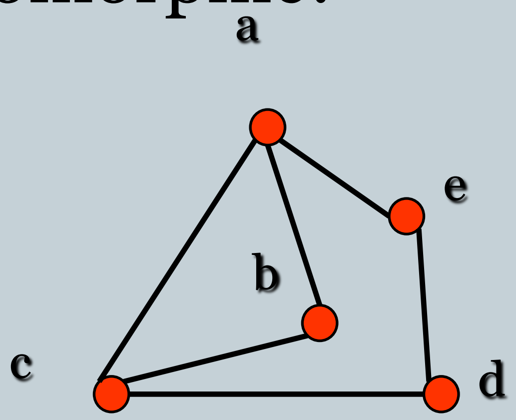
Discovering Isomorphism

- Example 5.2.1 page: 450 (Text book)
- Note that vertices b,d, & e of G must be mapped to vertices b',d', & e' of G' resp. since these are unique vertices of degree 2, 5, & 1.
- Instead of 5! Maps, we have only 2 maps to consider!!
- Both these maps are isomorphic
- a c'
- b b'
- c a'
- d d'
- e e'
- Degree sequence of a graph can be used to shorten the search for an isomorphism!

Isomorphism of Graphs

Are the following two graphs isomorphic?

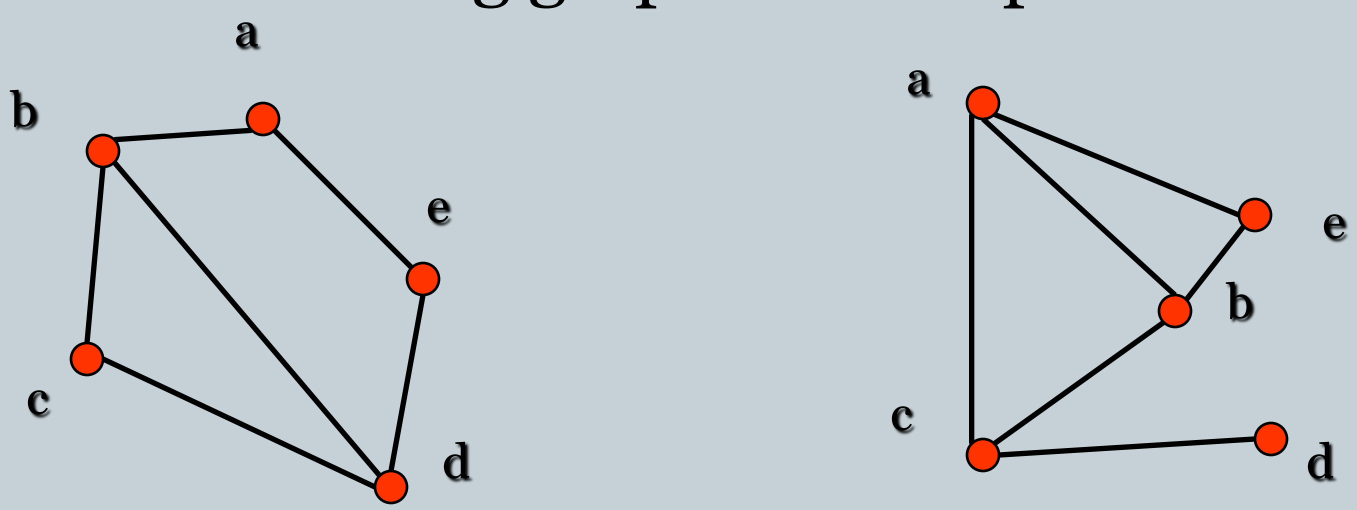




Yes, they are isomorphic, because they can be arranged to look identical. You can see this if in the right graph you move vertex b to the left of the edge {a, c}. Then the isomorphism f from the left to the right graph is: f(a) = e, f(b) = a,
f(c) = b, f(d) = c, f(e) = d.

Isomorphism of Graphs

Are the following graphs isomorphic?



No, they are not isomorphic, because they differ in the degrees of their vertices.

Vertex d in right graph is of degree one, but there is no such vertex in the left graph.

Discovering Isomorphism

- Check **invariants**, that is, properties that two isomorphic graphs must both have.
- For example, they must have
  - the same number of vertices,
  - the same number of edges, and
  - the same degrees of vertices.
- Showing that two graphs are **not** isomorphic can be easy
- Note that two graphs that **differ** in any of these invariants are not isomorphic, but two graphs that **match** in all of them are not necessarily isomorphic.

Discovering Isomorphism

- Different no. of vertices – not isomorphic
- Different no. of edges – not isomorphic
- Different degree sequence – not isomorphic
- Even if they have same degree sequence (i.e. having same no. of vertices & edges), they need not be isomorphic
- No set procedure to establish isomorphism!
- Let us see if we can work out an approach that works often!

Discovering Isomorphism

- Classify vertices according to some property preserved by isomorphism
- Classify (put together in a group) all vertices of degree 2 in one class
- Classes are graphs in themselves! (sub-graphs)
- Sub-graphs – If G & H are graphs then H is a sub-graph of G iff $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$
- Spanning sub-graph – H is called a spanning sub-graph of G iff $V(H) = V(G)$
- if $W \subseteq V(G)$, then the sub-graph induced by W is the sub-graph H obtained by taking $V(H)=W$ and $E(H)$ to be those edges of G that connect vertices in W

Discovering Isomorphism

- Example 5.2.2 pp 452 (Text book)
- Example 5.2.3 pp 453 (Text book)
- G – e
- G – v (along with all edges incident on v)
- K_n – complete graph with n vertices - A simple NDG with n mutually adjacent vertices.
- ${}^{n}C_{2}$ =n(n-1)/2 edges!
- each vertex has degree (n-1)

Discovering Isomorphism

- Result: Any graph may be viewed as made up of building blocks which are complete sub-graphs
- How this result will help us in solving the isomorphism problem?
- Ex. 5.2.3 pp 453 (Text book)
- If the 2 graphs were isomorphic, then the respective SGs induced by the vertices of degree 2 would be isomorphic
- In the first graph, no pair of vertices of degree 2 are adjacent, whereas g' & h' are vertices of degree 2 that are adjacent in the second
- Difference in str. of the SGs induced by vertices of degree 3 also.
- In first graph, vertex with degree 4 is adj to 2 vertices of deg 3, and in the 2^{nd}, it is adj to 2 vertices of degree 2.

Complement of Graphs

- Example 5.2.4 pp 454 (Text book)
- Example 5.2.5 pp 454 (Text book)
- Concept of graph complements !
- Role of graph complements in discovering isomorphisms?

Complement of Graphs

- Complement of a Graph: If H is a SG of G, then the complement of H in G ($\overline{H}$) is the SG G-E(H)
- If H is a simple graph with n vertices, the complement $\overline{H}$ of H is the complement of H in K
- $V(\overline{H}) = V(H)$
- Any two vertices that are adj in $\overline{H}$ iff they are not adj in H
- Deg of a vertex in $\overline{H}$ + degree in H = n-1, where n=|V(H)|
- Example 5.2.6 pp 456 (Text book)
- 2 simple graphs are isomorphic iff their complements are isomorphic (form defns of isomorphisms & complement)

Intersection/Union of Graphs

- Let G & G' be two graphs. G$\cap$G' is the graph whose vertex set is V(G) $\cap$ V(G') and whose edge set is E(G) $\cap$ E(G').

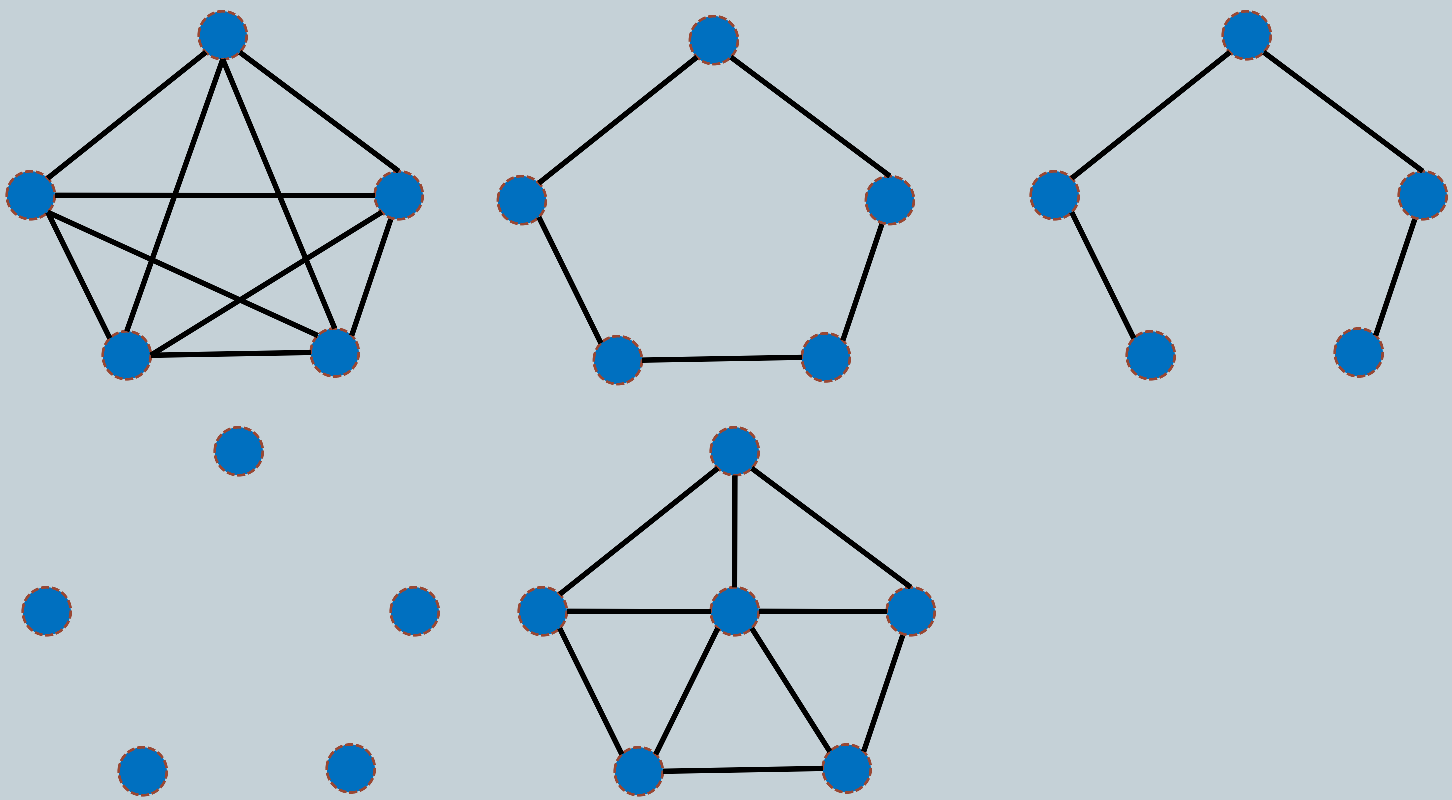
- Let G & G' be two graphs. G$\cup$G' is the graph whose vertex set is V(G) $\cup$ V(G') and whose edge set is E(G) $\cup$ E(G').

- What is G$\cup\overline{G}$? Where G is a simple graph with n vertices.

Special Graphs

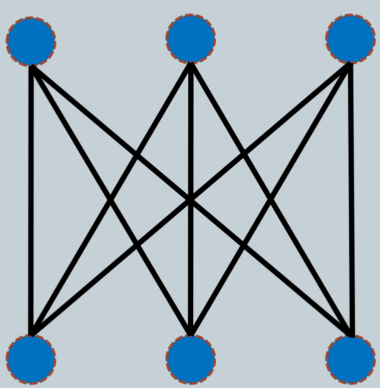
- Every complete graph with n vertices is isomorphic to every other complete graph with n vertices
- Some other special classes of graphs which form a class of isomorphic graphs
- Cycle graph
- Path graph
- Null graph
- Bipartite graph
- Star graph

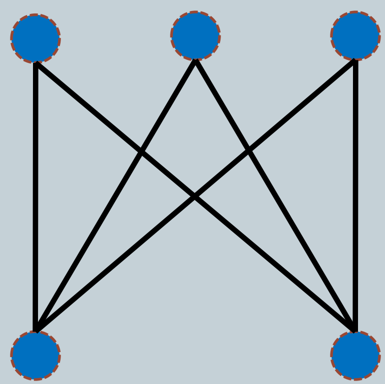
Special Graphs

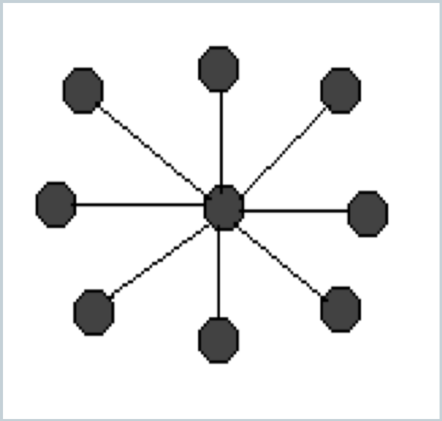


Bipartite Graphs

- A NDG whose set of vertices can be partitioned into 2 sets M & N such that each edge joins a vertex in M to a vertex in N
- Complete Bipartite: A bipartite graph in which every vertex of M is adjacent to every vertex in N. Denoted by $K_{m,n}$, where $|M|=m$ & $|N| = n$ (generally $m<=n$)
- Star Graph: $K_{1,n}$







Tests for Isomorphism: Summary

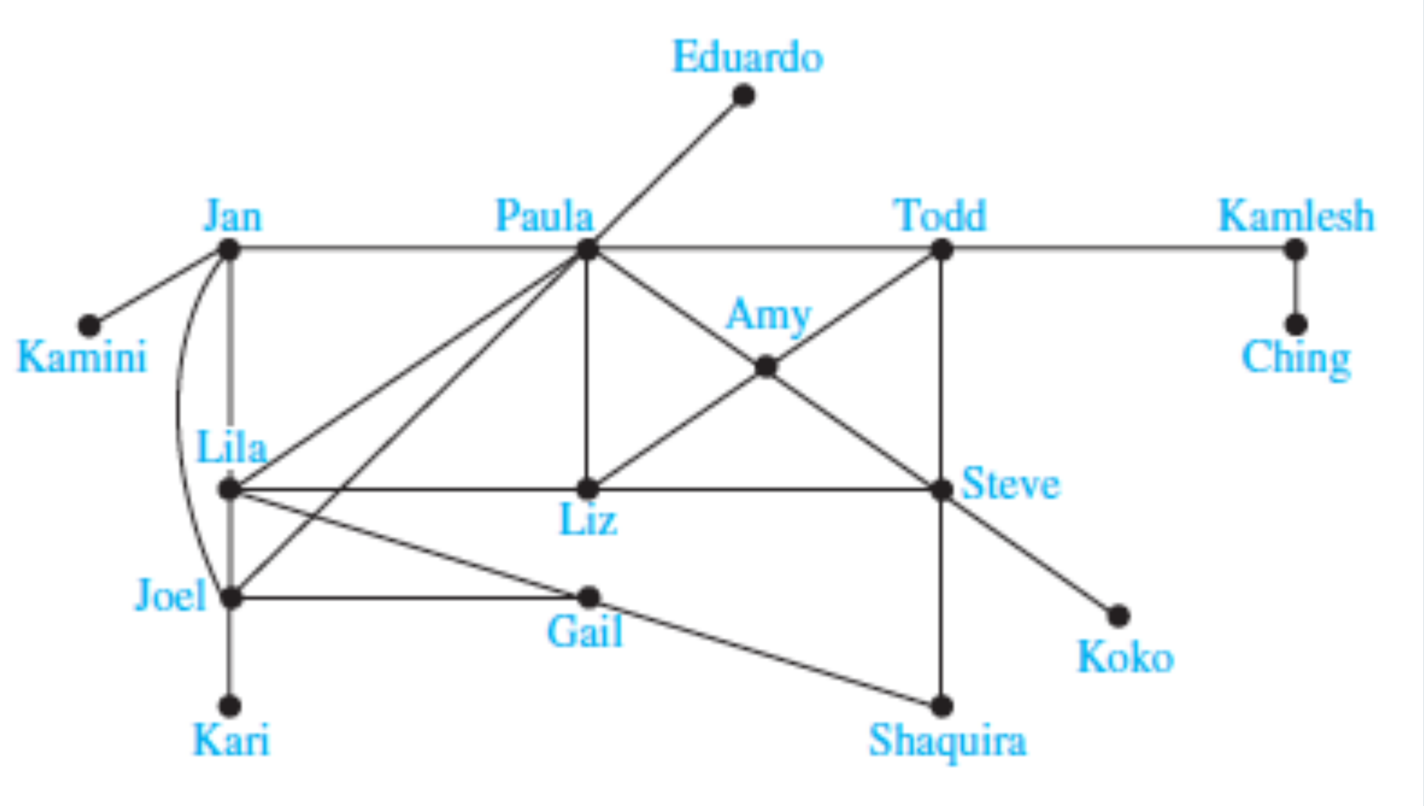
- Talk about all (n!) 1-1 & onto mappings between G & G'. Use the concept of adjacency matrices
- Reduce the no. of mappings by using degree sequence
- Use sub-graphs & induced sub-graphs
  - Graphs induced by vertices of a particular degree would be isomorphic
- Complete graphs: two complete graphs with same no. of vertices are isomorphic
- Use complements of graphs! Two simple graphs are isomorphic iff their complements are isomorphic

Connectivity

- Paths
- Cycles/circuits

Let n be a nonnegative integer and G an undirected graph. A *path* of *length* n from u to v in G is a sequence of n edges $e_1, \ldots, e_n$ of G for which there exists a sequence $x_0 = u, x_1, \ldots, x_{n-1}, x_n = v$ of vertices such that e_i has, for $i = 1, \ldots, n$, the endpoints x_{i-1} and x_i. When the graph is simple, we denote this path by its vertex sequence $x_0, x_1, \ldots, x_n$ (because listing these vertices uniquely determines the path). The path is a *circuit* if it begins and ends at the same vertex, that is, if $u = v$, and has length greater than zero. The path or circuit is said to *pass through* the vertices $x_1, x_2, \ldots, x_{n-1}$ or *traverse* the edges $e_1, e_2, \ldots, e_n$. A path or circuit is *simple* if it does not contain the same edge more than once.

Paths in Social Networks



- Acquaintanceship Graph

Paths in Collaboration Graphs

TABLE 1 The Number of Mathematicians with a Given Erdős Number (as of early 2006).

Erdős Number	*Number of People*
0	1
1	504
2	6,593
3	33,605
4	83,642
5	87,760
6	40,014
7	11,591
8	3,146
9	819
10	244
11	68
12	23
13	5

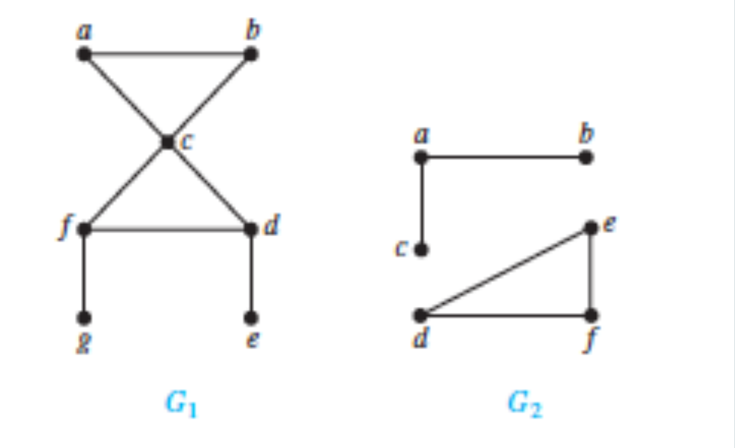
Paths in Hollywood Graph

TABLE 2 The Number of Actors with a Given Bacon Number (as of early 2011).

Bacon Number	*Number of People*
0	1
1	2,367
2	242,407
3	785,389
4	200,602
5	14,048
6	1,277
7	114
8	16

Connectedness in Undirected Graphs

- An undirected graph is called connected if there is a path between every pair of distinct vertices
  - A graph that is not connected is called disconnected graph
- Any two computers in a network can communicate iff the graph of this network is connected



Connectedness in Undirected Graphs

An undirected graph is called connected if there is a path between every pair of distinct vertices

Theorem: There is a simple path between any two vertices in a connected graph.

Connected Components

- A connected component of a graph G is a connected subgraph of G that is not proper subgraph of another connected subgraph of G
- A connected component of a graph G is a maximally connected subgraph of G
- A disconnected graph G has two or more connected components that are disjoint and have G as their union

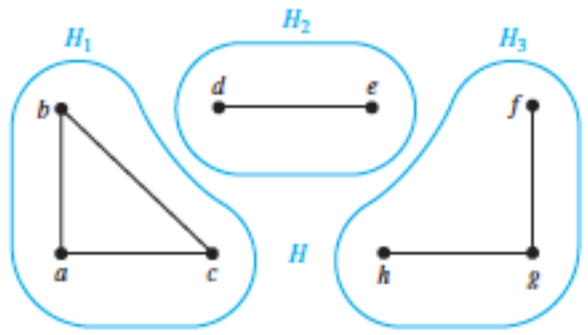


FIGURE 3 The Graph H and Its Connected Components H_1, H_2, and H_3.

Connected Components of a Call Graph

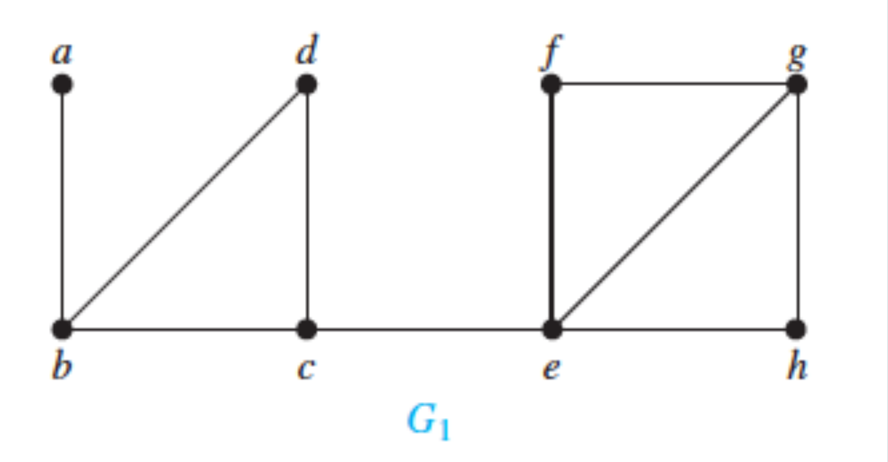
- Two vertices x & y are in the same component of a telephone call graph when there is a sequence of telephone call beginning at x and ending at y
- A day's data at AT&T:
  - 53 million vertices
  - 170 million edges
  - 3,7 million connected components
    - 3/4th contained only 2 vertices
    - One huge connected component – 44 million vertices (80% of total). Every vertex can be linked to any other vertex in this component by a chain of not more than 20 calls

How Connected is a Graph?

- Cut Vertices
  - removal of a vertex and all incident edges produces a graph with more connected components. Such vertices are called cut vertices
  - removal of a cut vertex from a connected graph produces a subgraph that is not connected
- Cut edge
  - removal of an edge produces a graph with more connected components. Such edges are called cut edges
  - removal of a cut vertex from a connected graph produces a subgraph that is not connected
- In a graph representing a computer network, a cut vertex and a cut edge represent an essential router and a communication link that can not fail for all computers to be able to communicate

How Connected is a Graph?

- Find cut vertices & cut edges



Cut vertices are b,c, & e
Removal of any one of them disconnects G1

Cut edges are (a,b) & (c,e)
Removal of any one of them disconnects G1

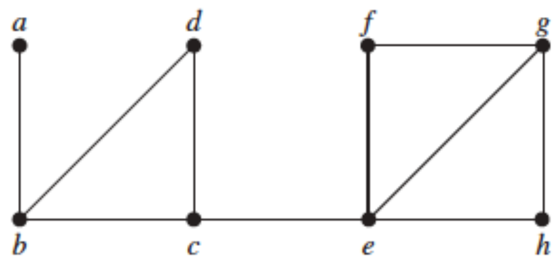
How Connected is a Graph?

- Vertex Connectivity
  - Not all graphs have cut vertices, for example K_n ($n>=3$) have no cut vertices
  - Connected graphs with no cut vertices are called non-separable graphs
    - More connected than one with a cut vertex
- Can we define a more granular measure of graph connectivity based on minimum number of vertices that needs to be removed to disconnect a graph
- Vertex cut – a subset V' of V of G=(V,E) is a vertex cut if G-V' is disconnected.
  - For G1 – {b,c,e} is a vertex cut with 3 vertices
- Every connected graph, except complete graph, has a vertex cut

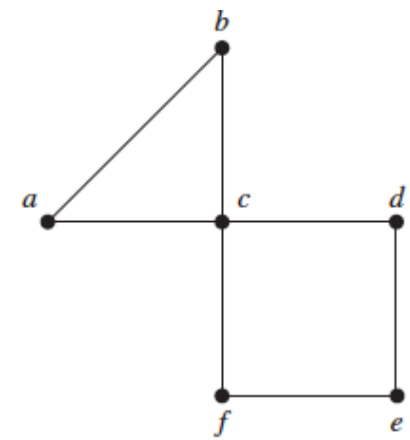
How Connected is a Graph?

- Vertex Connectivity of a non-complete graph G is denoted by κ(G), is the minimum number of vertices in a vertex cut
- A graph G is k-connected if κ(G) >=k

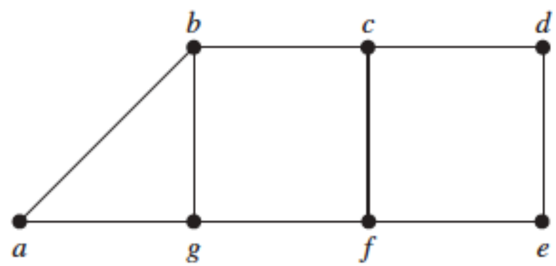
How Connected is a Graph?



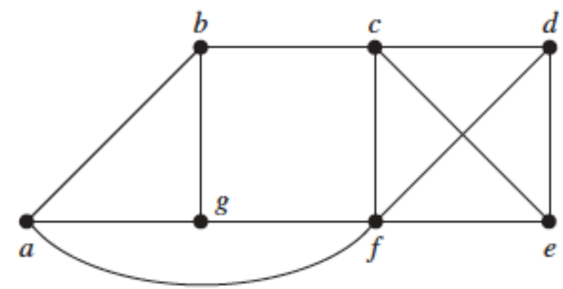
G_1



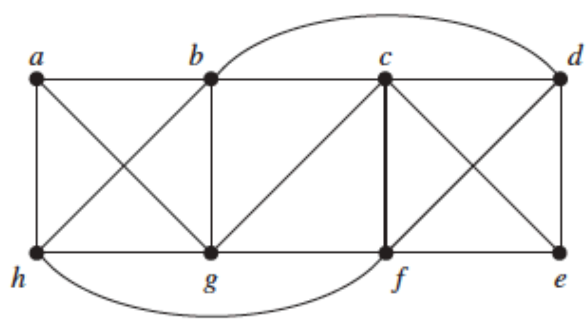
G_2



G_3



G_4



G_5

How Connected is a Graph?

- Edge connectivity, $\lambda(G)$
- Analogous to vertex connectivity!!

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.

- For a directed graph to be strongly connected, there must be a sequence of directed edges from any vertex in the graph to any other vertex
- A directed graph can fail to be strongly connected, but still be in "one piece"

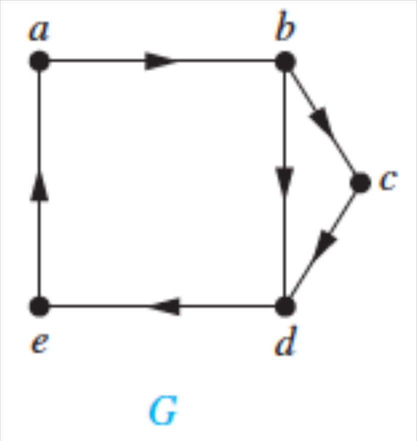
In 2010, the Web Graph had 55 bn vertices and 1 tr edges – 40 TB of space needed to store its adjacency matrix

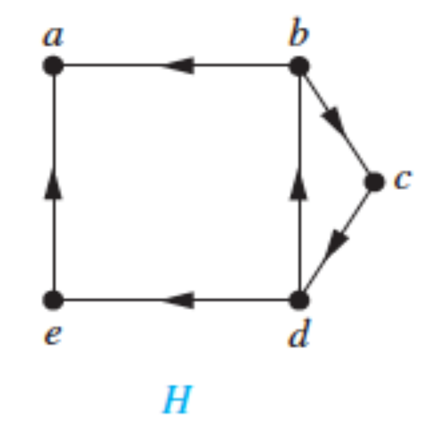
Connectedness in Directed Graphs

A directed graph is *weakly connected* if there is a path between every two vertices in the underlying undirected graph.

- A directed graph is weakly connected iff there is always a path between two vertices when the direction of the edges is disregarded.
- Any strongly connected graph is always weakly connected.

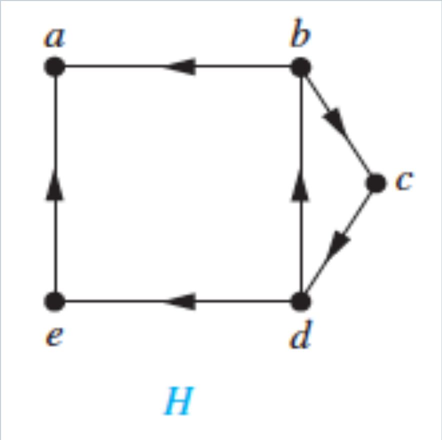
Connectedness in Directed Graphs





- G is strongly connected
- G is also weakly connected
- H is not strongly connected
- Although, H is weakly connected

Connectedness in Directed Graphs



- Strongly connected components of a Directed Graph
  - The subgraphs of G that are strongly connected, but not contained in larger strongly connected subgraphs, ie, maximal strongly connected subgraphs, are called Strongly connected components of a Directed Graph
  - Identify Strongly connected components of H

Counting Paths between Vertices

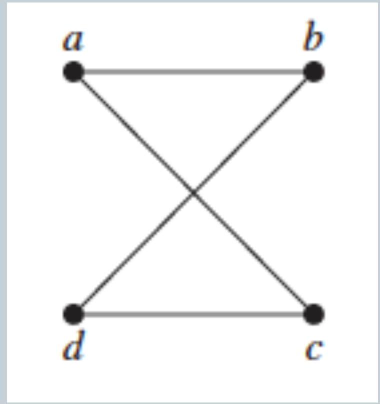
- The number of paths between two vertices in graph can be determined using its adjacency matrix
- Theorem:

Let G be a graph with adjacency matrix $\mathbf{A}$ with respect to the ordering $v_1, v_2, \ldots, v_n$ of the vertices of the graph (with directed or undirected edges, with multiple edges and loops allowed). The number of different paths of length r from v_i to v_j, where r is a positive integer, equals the (i, j)th entry of $\mathbf{A}^r$.

- Proof by Mathematical Induction!!

Counting Paths between Vertices

- How many paths of length 4 from a to d in the simple graph G?



- Adjacency matrix of G with vertex ordering (a,b,c,d) is:

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}.$$

$$\mathbf{A}^4 = \begin{bmatrix} 8 & 0 & 0 & 8 \\ 0 & 8 & 8 & 0 \\ 0 & 8 & 8 & 0 \\ 8 & 0 & 0 & 8 \end{bmatrix}.$$

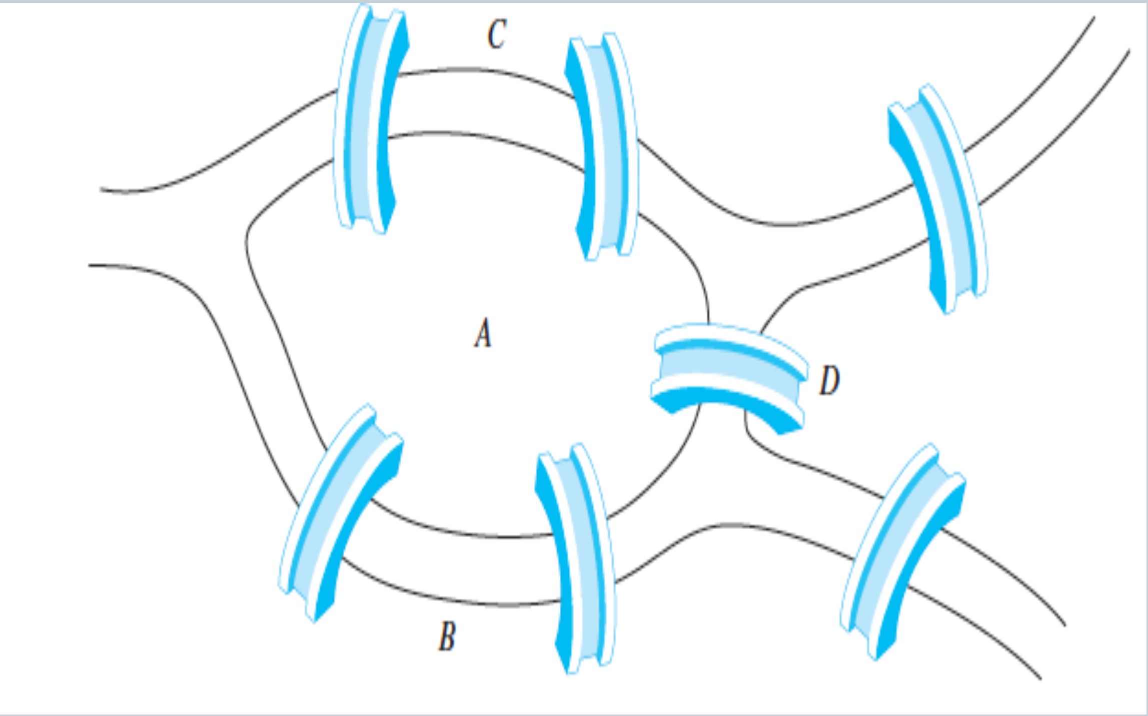
Counting Paths between Vertices

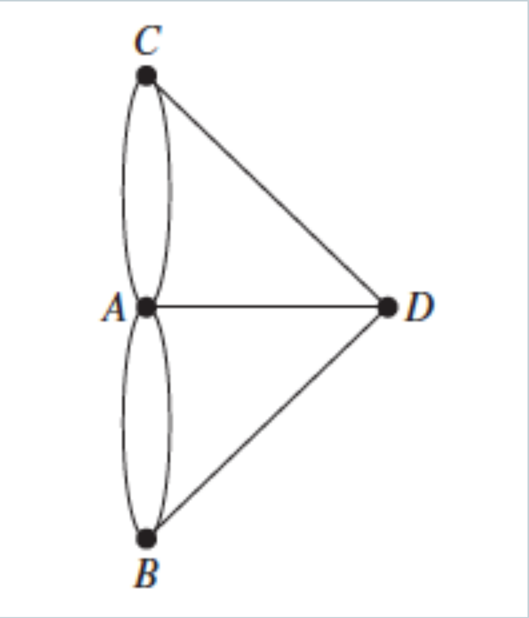
- Theorem:

Let G be a graph with adjacency matrix $\mathbf{A}$ with respect to the ordering $v_1, v_2, \ldots, v_n$ of the vertices of the graph (with directed or undirected edges, with multiple edges and loops allowed). The number of different paths of length r from v_i to v_j, where r is a positive integer, equals the (i, j)th entry of $\mathbf{A}^r$.

- Possible applications of the theorem?
  - Shortest path between two vertices of a graph
  - Checking whether a graph s connected or not

Euler & Hamiltonian Paths



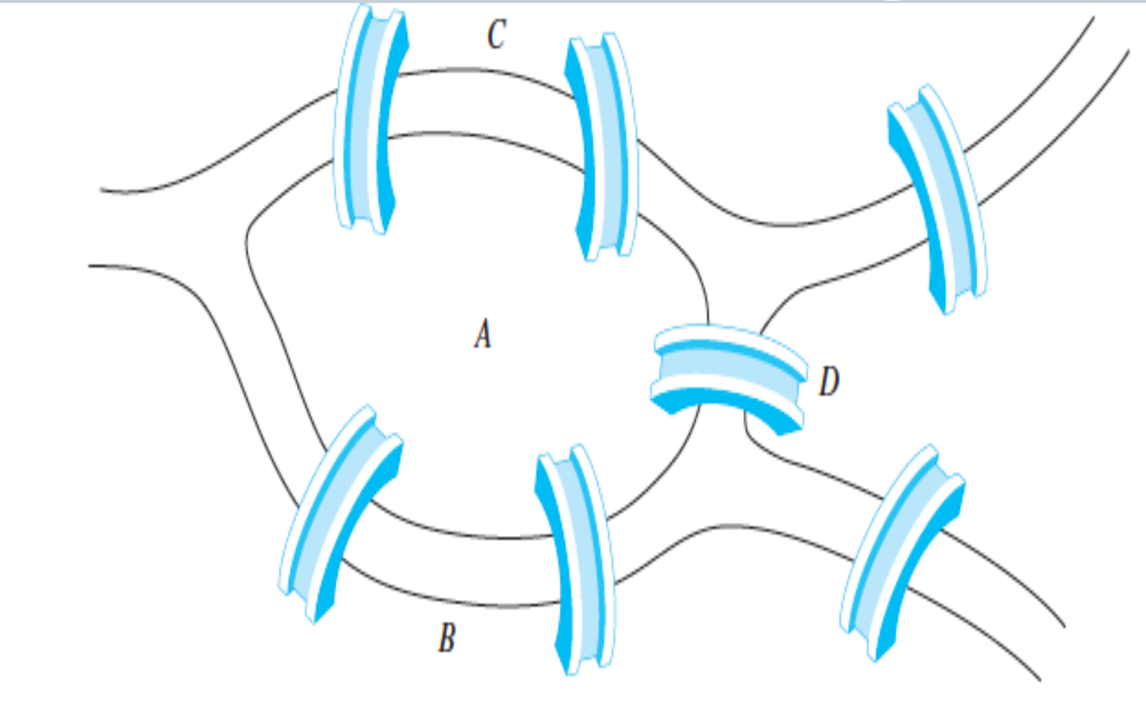


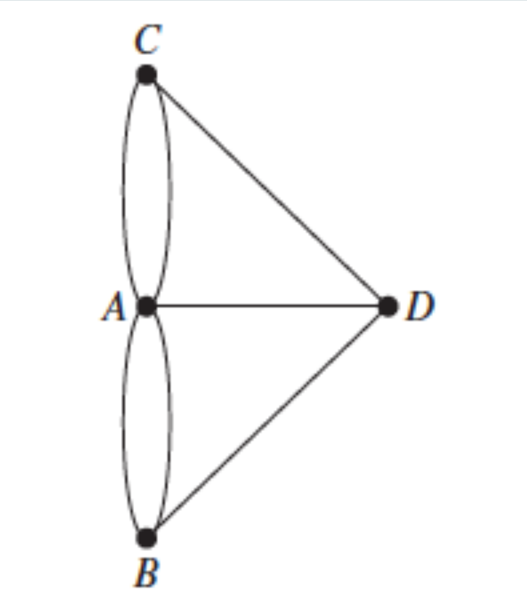
The Seven Bridges of Königsberg.

Euler & Hamiltonian Paths

- Can we travel along the edges of a graph starting at a vertex and return to it after traversing each edge of the graph exactly once?
  - Euler circuit
- Similarly, can we travel along the edges of a graph starting at a vertex and return to it after traversing each vertex of the graph exactly once?
  - Hamiltonian circuit

Euler & Hamiltonian Paths

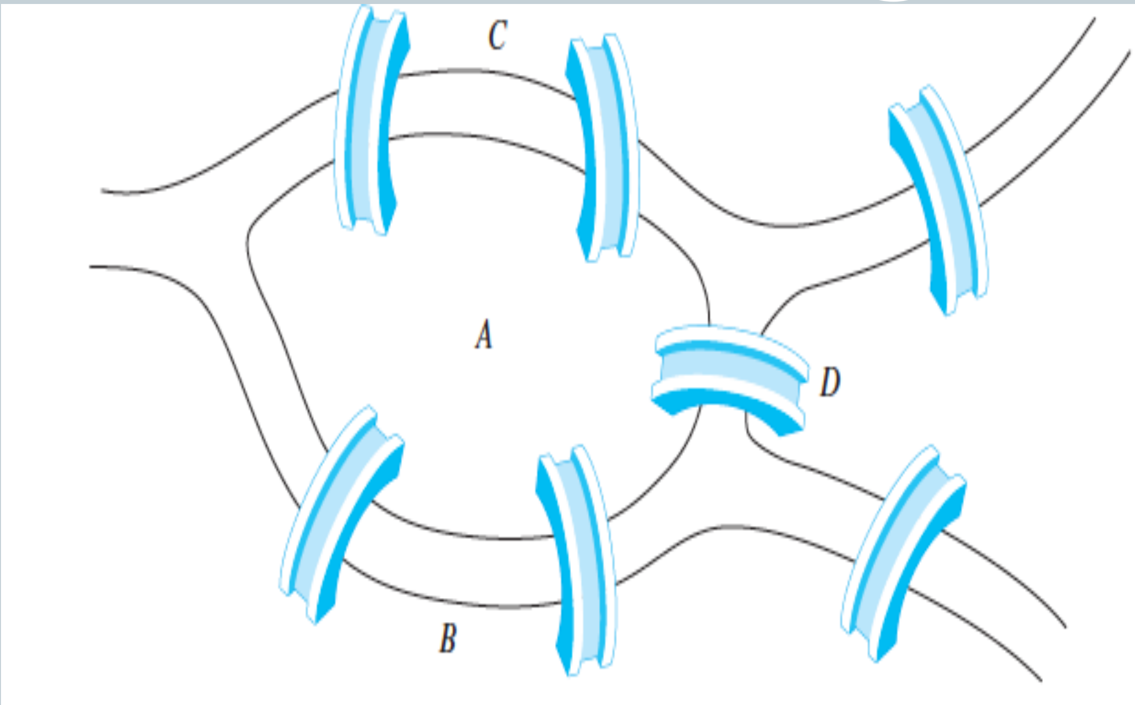


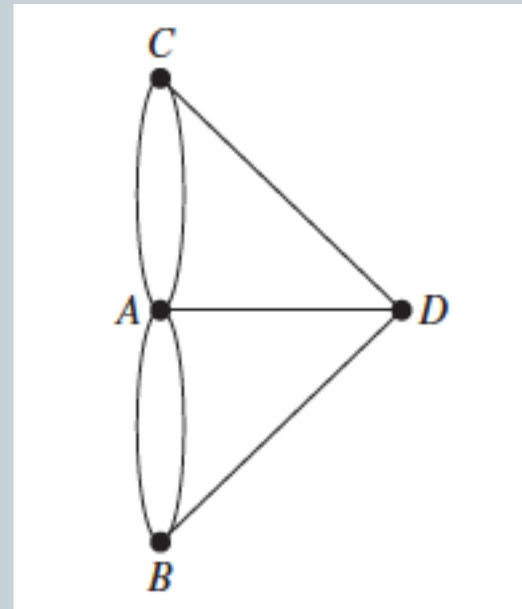


Can we start at any point in the town and return to the same location by crossing each bridge just once?

Is there a simple circuit in this multigraph that contains every edge?

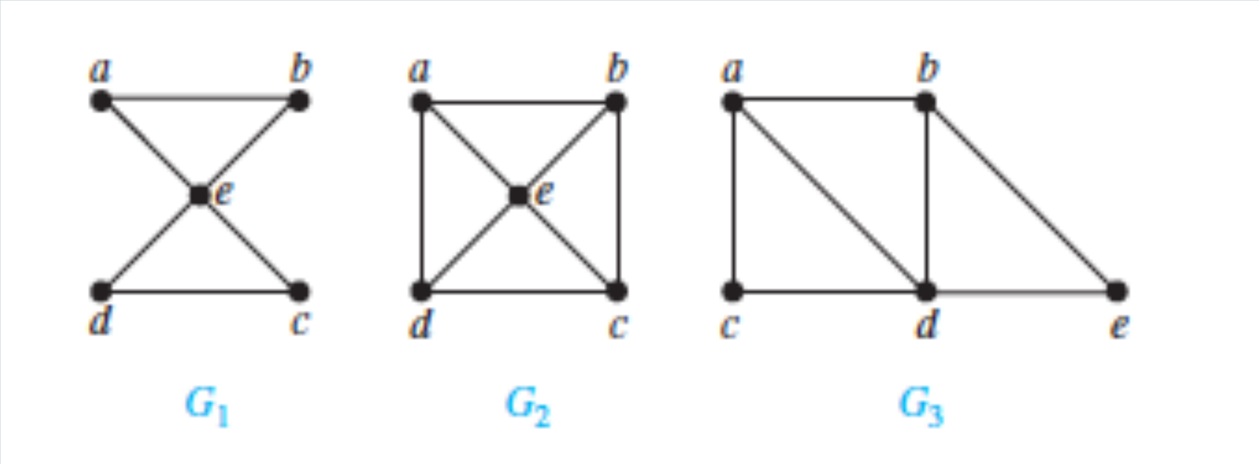
Euler & Hamiltonian Paths





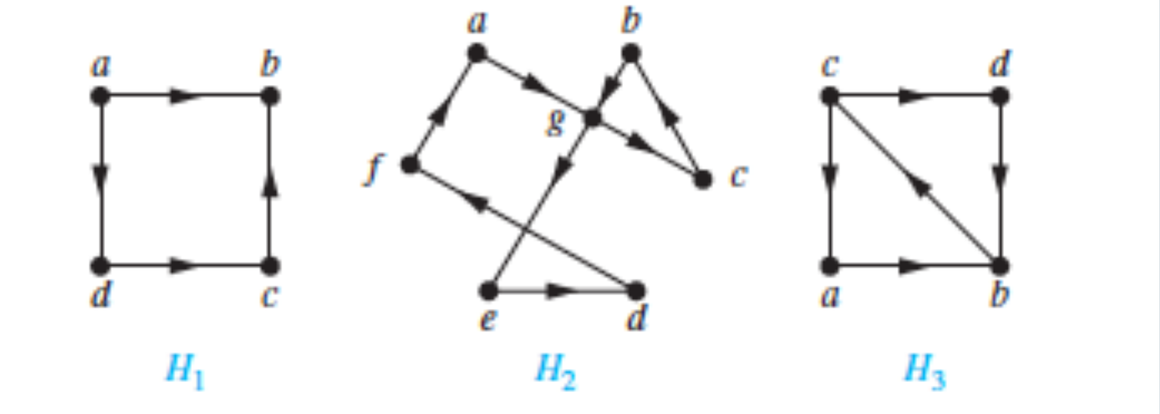
An *Euler circuit* in a graph G is a simple circuit containing every edge of G. An *Euler path* in G is a simple path containing every edge of G.

Euler & Hamiltonian Paths



- G1 has Euler circuit – a e c d e b a
- G2 has no Euler circuit & No Euler path
- G3 has no Euler circuit, but has an Euler path – a c d e b d a b

Euler & Hamiltonian Paths



- H1 has neither
- H2 has an Euler circuit – a g c b d e f a
- H3 has no Euler circuit, but an Euler path – c a b c d b

Euler & Hamiltonian Paths

Necessary & Sufficient conditions for Euler Circuits and Paths

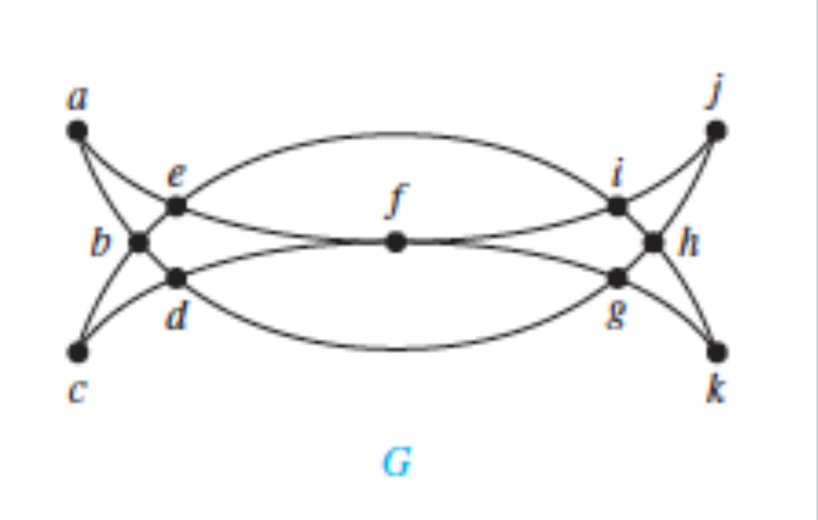
- If a connected graph has an Euler circuit, then every vertex must have an even degree
- Is this necessary condition also sufficient? That is, if in a connected multigraph, all vertices are of even degree, then an Euler circuit exists!
- Theorem:

A connected multigraph with at least two vertices has an Euler circuit if and only if each of its vertices has even degree.

Euler & Hamiltonian Paths

Puzzle Solving

- Mohammad's Scimitars
- Can we draw a picture in a continuous motion without lifting a pencil so that no part of the picture is retraced?

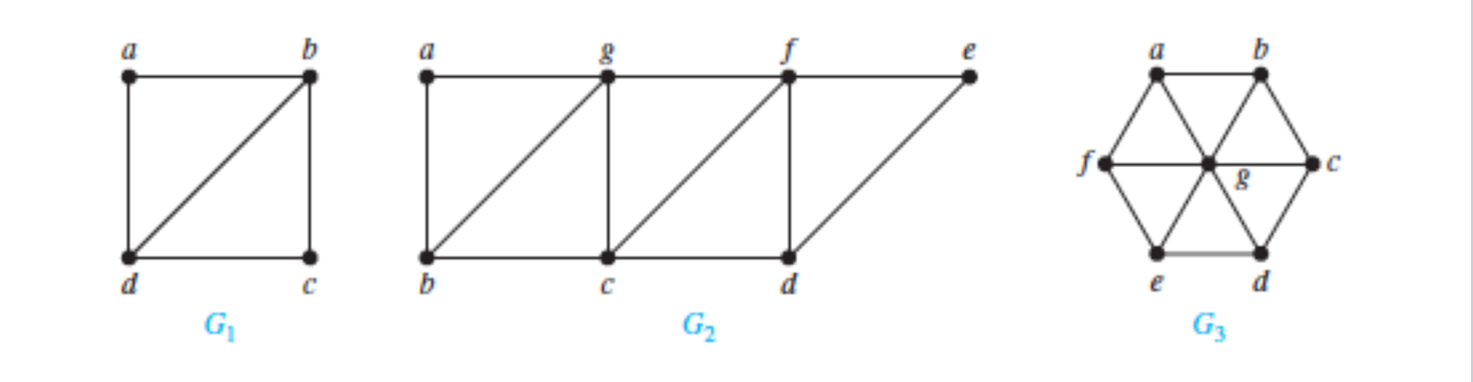


Euler & Hamiltonian Paths

Euler Path

- Theorem:

A connected multigraph has an Euler path but not an Euler circuit if and only if it has exactly two vertices of odd degree.



Applications of Euler Paths & Circuits

Many applications require that we:

- Traverse each street in a nbd.
- Traverse each road in a network
- Traverse each connection in a utility grid
- Traverse each link in a communication network

Exactly once!!

For eg., if a postman can find an Euler Path in a graph representing streets and houses

In Molecular Biology, Eps are used for sequencing of DNAs

Layout of circuits

Hamiltonian Paths & Circuits

A simple path in a graph G that passes through every vertex exactly once is called a *Hamilton path*, and a simple circuit in a graph G that passes through every vertex exactly once is called a *Hamilton circuit*. That is, the simple path $x_0, x_1, \ldots, x_{n-1}, x_n$ in the graph $G = (V, E)$ is a Hamilton path if $V = \{x_0, x_1, \ldots, x_{n-1}, x_n\}$ and $x_i \neq x_j$ for $0 \leq i < j \leq n$, and the simple circuit $x_0, x_1, \ldots, x_{n-1}, x_n, x_0$ (with $n > 0$) is a Hamilton circuit if $x_0, x_1, \ldots, x_{n-1}, x_n$ is a Hamilton path.

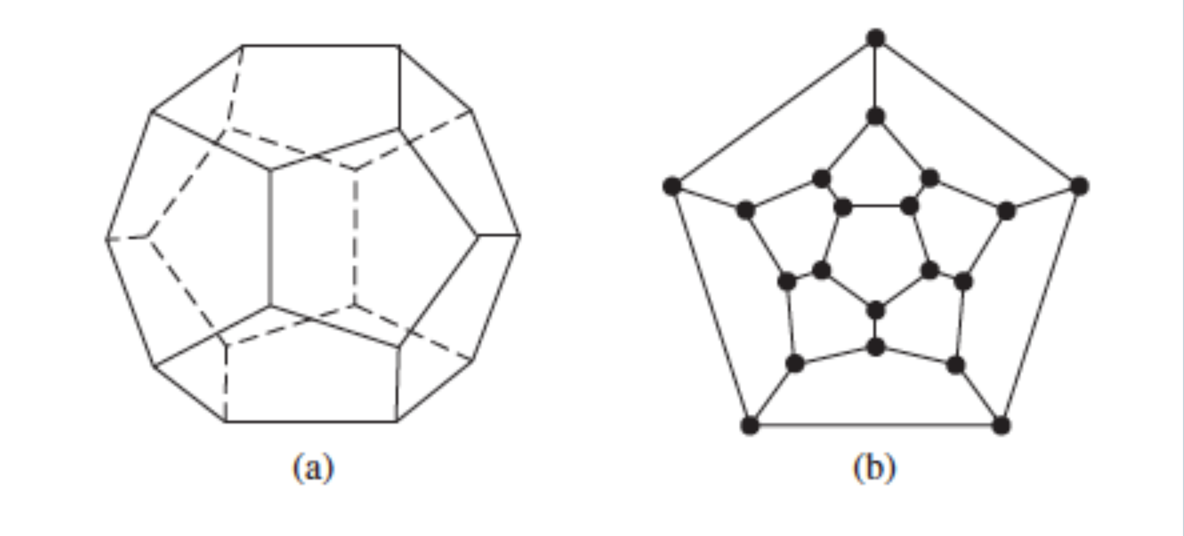


FIGURE 8 **Hamilton's "A Voyage Round the World" Puzzle.**

Hamiltonian Paths & Circuits

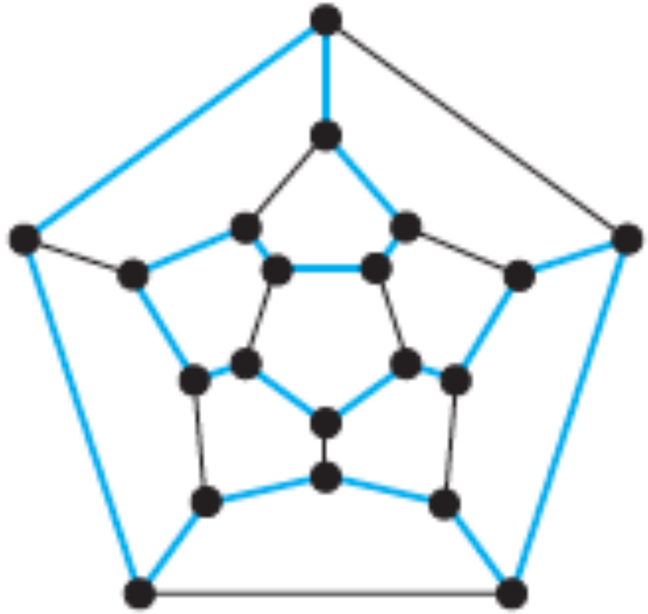
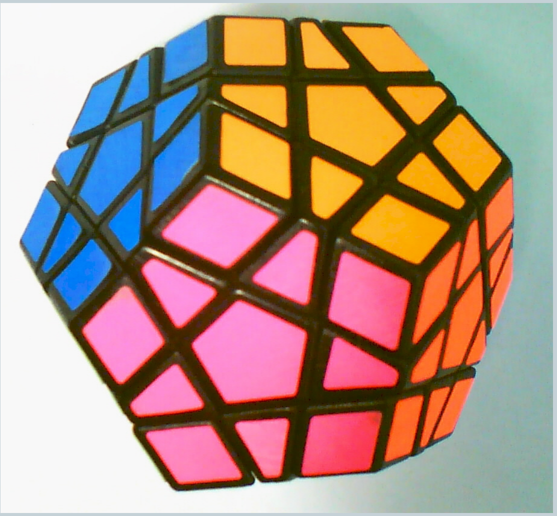


FIGURE 9 **A Solution to the "A Voyage Round the World" Puzzle.**

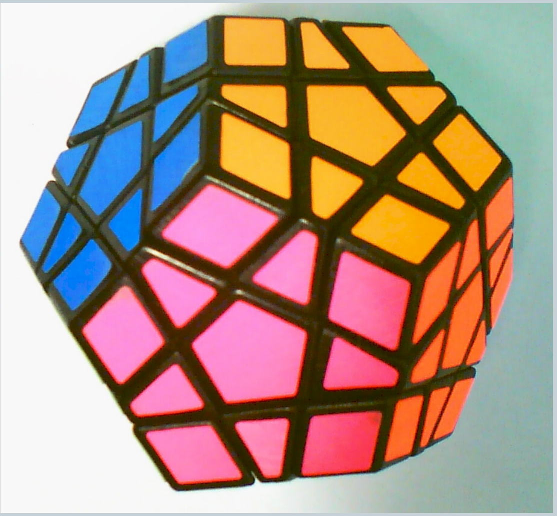
Hamiltonian Paths & Circuits

- Roots in Icosian Puzzle invented in 1857 by Irish Mathematician Hamilton!
  - Wooden dodecahedron (a polyhedron with 12 regular pentagons as faces)
  - A peg at each vertex (20 cities)

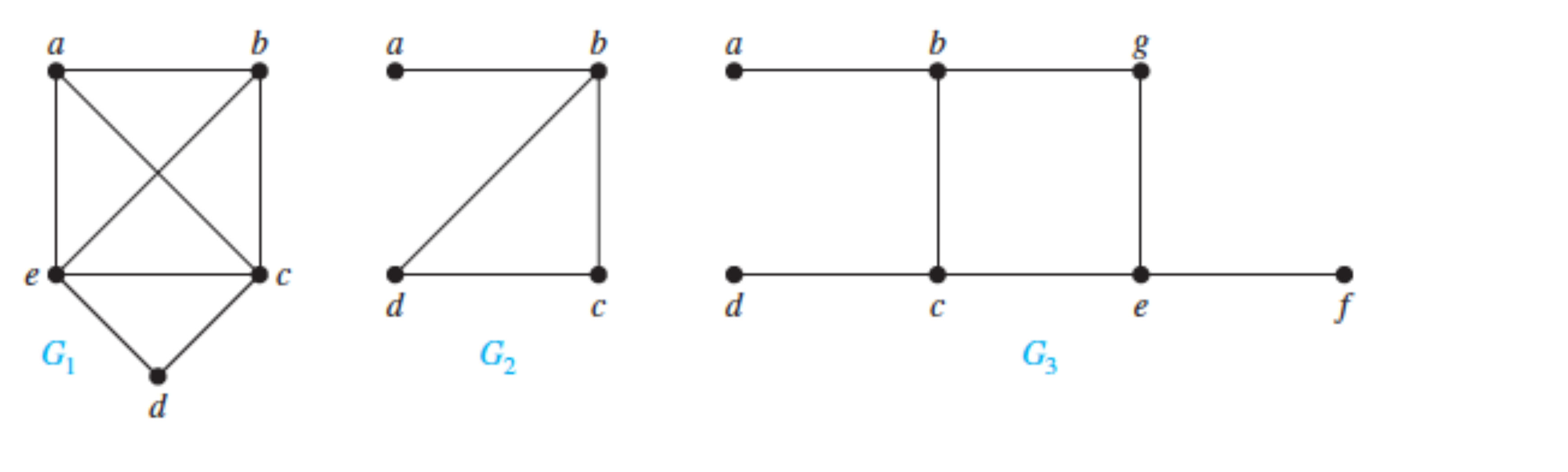


Hamiltonian Paths & Circuits

- Roots in Icosian Puzzle invented in 1857 by Irish Mathematician Hamilton!
  - Wooden dodecahedron (a polyhedron with 12 regular pentagons as faces)
  - A peg at each vertex (20 cities)



Hamiltonian Paths & Circuits

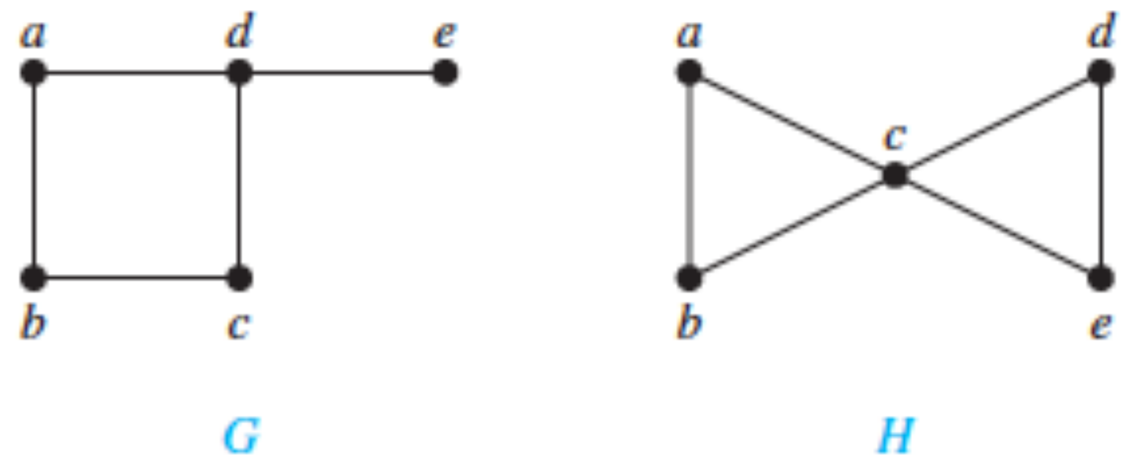


- G1 has a Hamiltonian circuit - a b c d e a
- No Hamiltonian circuit in G2, but has a Hamiltonian path – a b c d
- G3 has neither

Conditions for Existence of Hamiltonian Circuits

- No simple N & S conditions, unlike Euler circuits, to determine if there are Hamiltonian circuits or not
- Many theorems give sufficient conditions for the existence of Hamiltonian circuits
- In addition, there are certain properties that can be used to show that a graph has no Hamiltonian circuits
  - What about a graph with a vertex of degree 1
- A Hamiltonian circuit is minima

Hamiltonian Paths & Circuits



- Neither has a Hamiltonian circuit
- Show that Kn (n>=3) has a Hamiltonian circuit

Hamiltonian Paths & Circuits

DIRAC'S THEOREM If G is a simple graph with n vertices with $n \geq 3$ such that the degree of every vertex in G is at least $n/2$, then G has a Hamilton circuit.

ORE'S THEOREM If G is a simple graph with n vertices with $n \geq 3$ such that $\deg(u) + \deg(v) \geq n$ for every pair of nonadjacent vertices u and v in G, then G has a Hamilton circuit.

- Both theorems provide Sufficient conditions
- None of them provide Necessary conditions
- C5 has a Hamiltonian circuit, but does not satisfy either theorem
- Algorithms have exponential worst case complexity

Applications of Hamiltonian Circuits

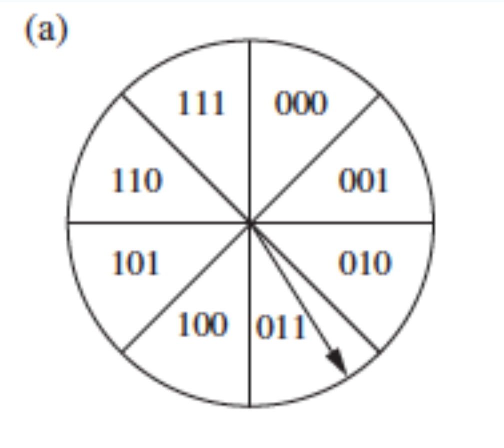
- Gray Codes
- Shortest Path Problem

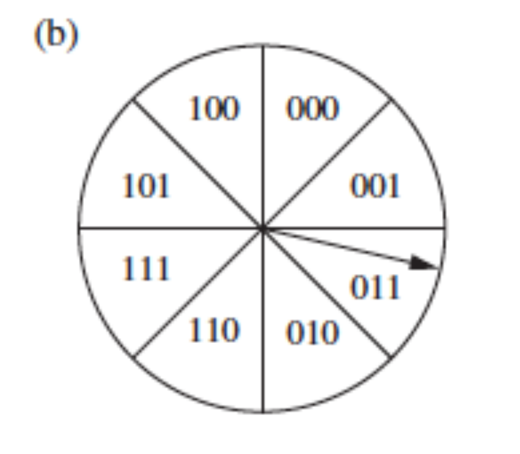
Applications of Hamiltonian Circuits

- Gray Codes
  - Position of a rotating pointer can be represented in digital form
  - Split is circle in 2n arcs of equal length and assign a bit string of length n to each arc
  - When the pointer is near the boundary of an arc, a mistake may be made in reading its position

Applications of Hamiltonian Circuits

- Gray Codes
  - In Fig. (a): 100 & 011 – all three bits are incorrect
  - In Fig. (b): 010 & 011 – only 1 bit is incorrect
  - (b) represents a Gray code!

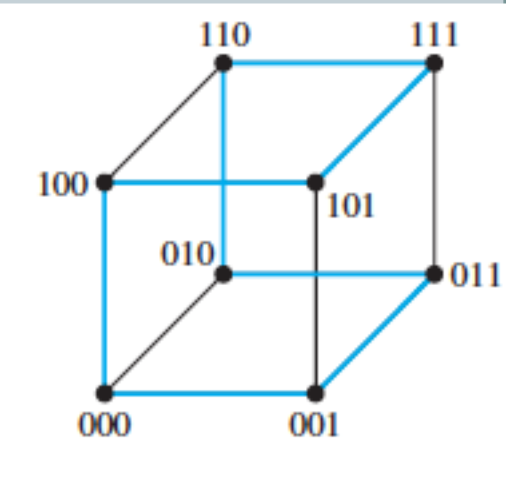




Applications of Hamiltonian Circuits

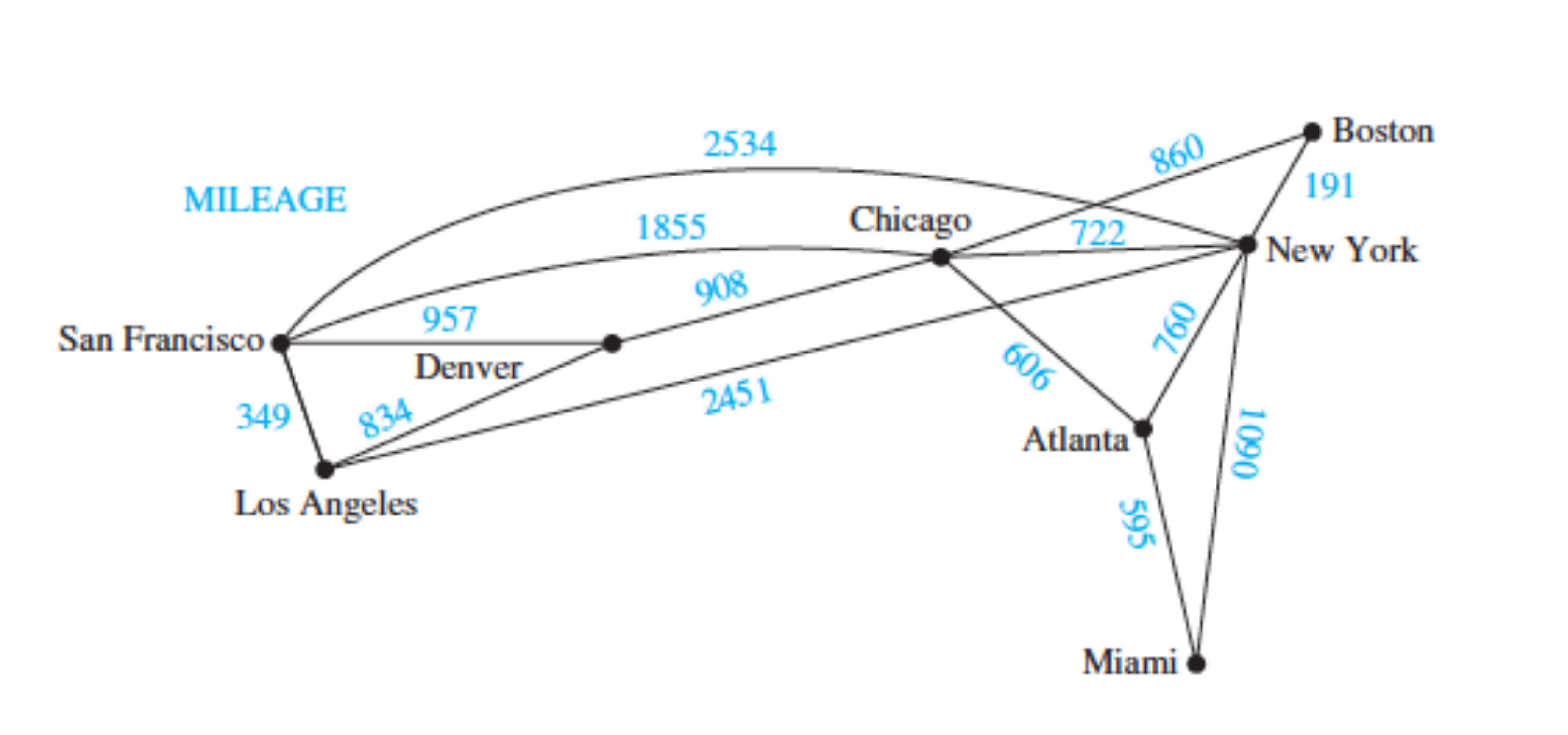
- Gray Codes - Modeling
  - We can model this problem using n-cube Q_n
  - Find a Hamiltonian circuit in Q_n
  - The sequence of bit strings differing in exactly one bit produced by this Hamiltonian circuit is 000,001, 011, 010, 110, 111, 101, 100
  - Gray codes are used to minimize errors in

  Transmitting digital signal



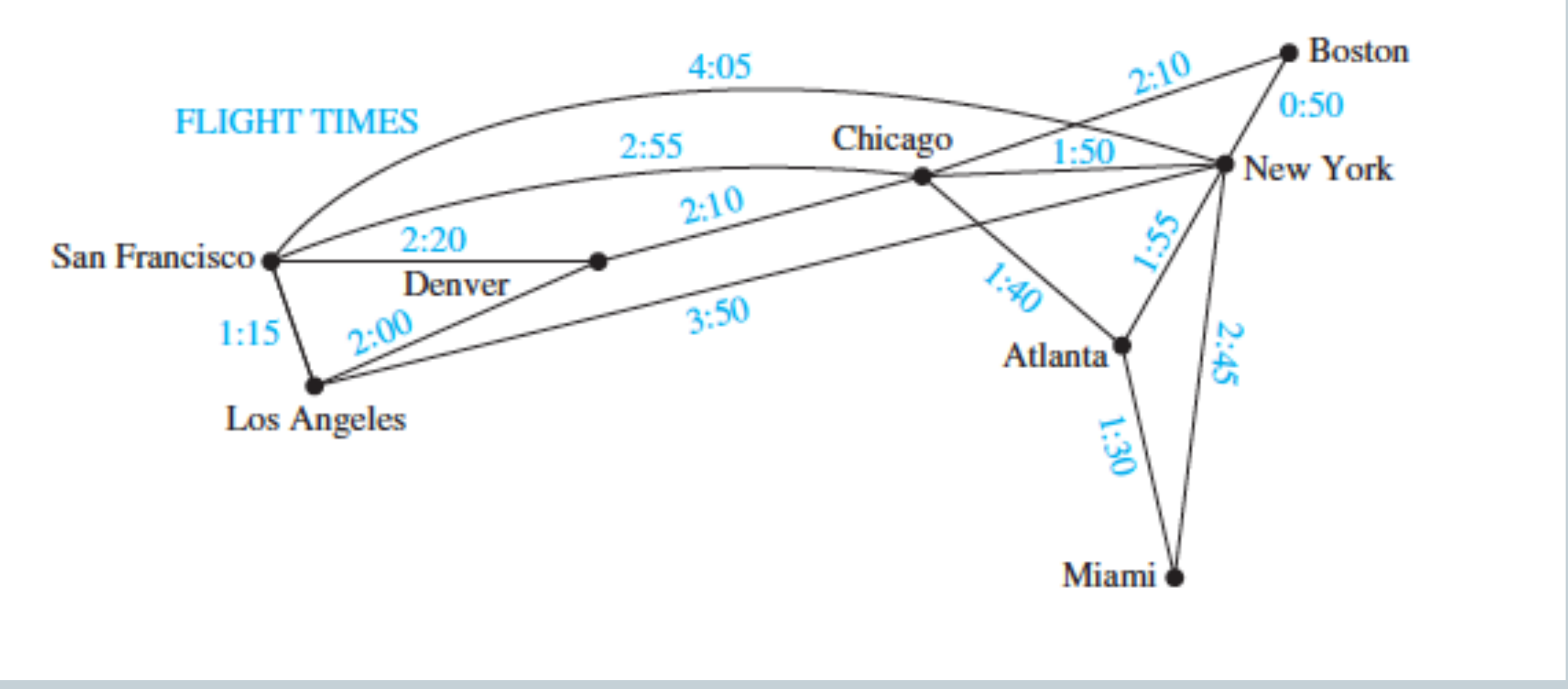
Shortest Path Problems

- Weighted Graphs



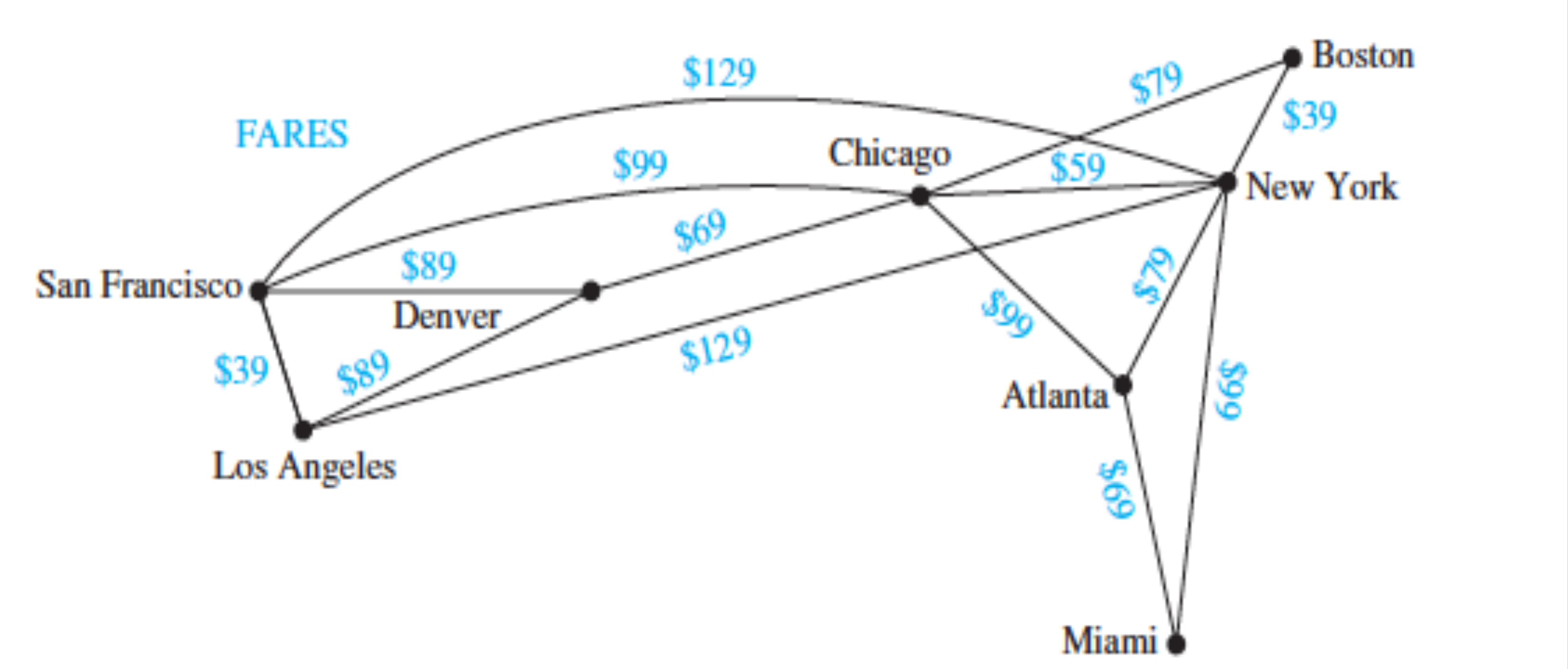
Shortest Path Problems

- Weighted Graphs



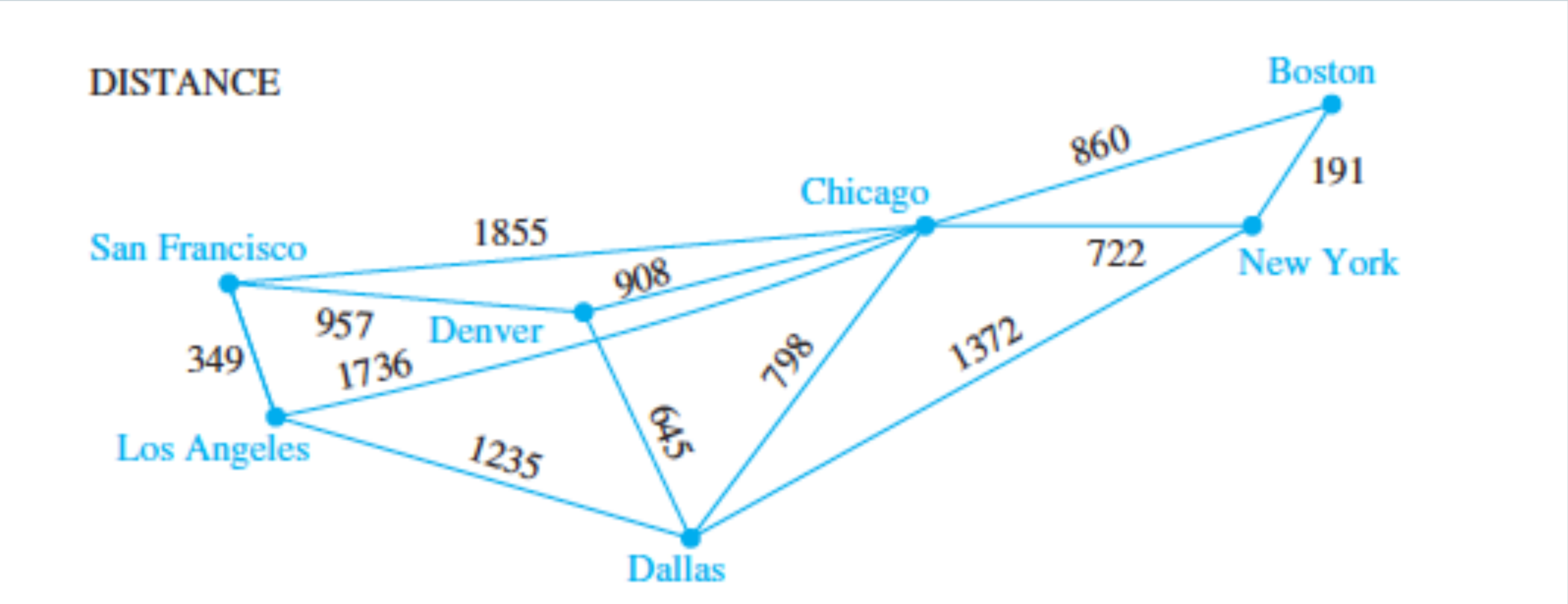
Shortest Path Problems

- Weighted Graphs



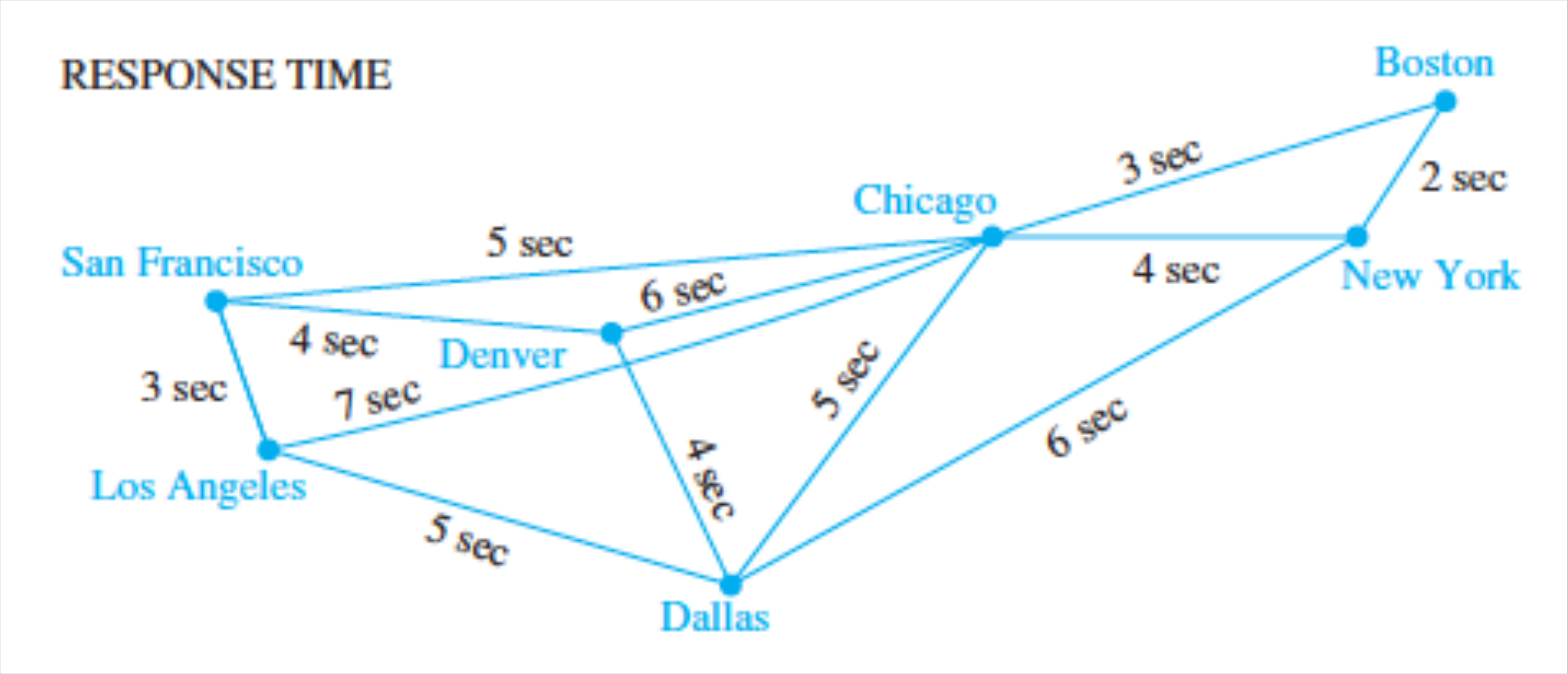
Shortest Path Problems

- Weighted Graphs – Computer Networks



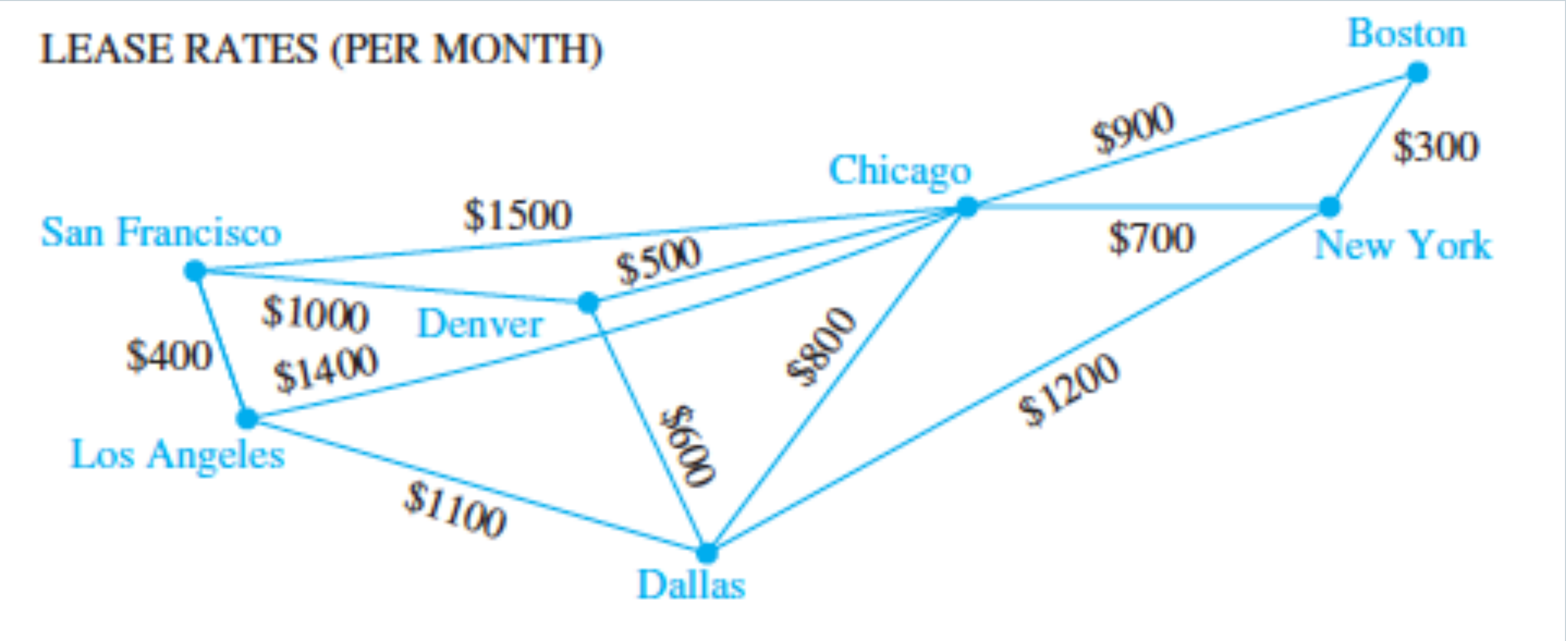
Shortest Path Problems

- Weighted Graphs – Computer Networks



Shortest Path Problems

- Weighted Graphs – Computer Networks



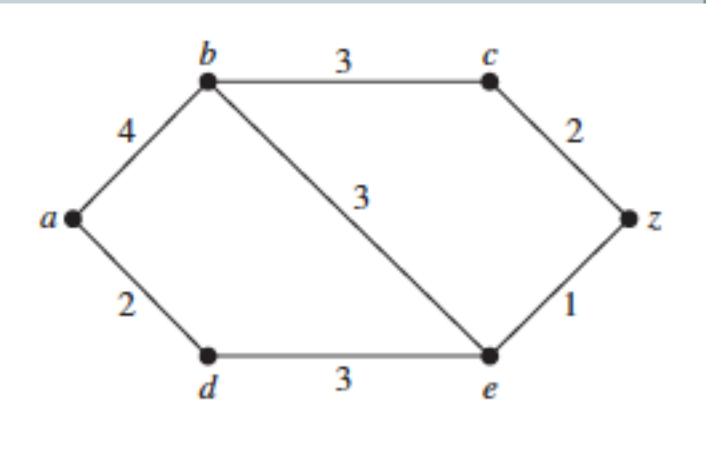
Shortest Path Problems

- Weighted Graphs
- Find a circuit of shortest total length that visits every vertex of a complete graph exactly once – TSP
- Shortest Path Algorithms
  - Greedy Algorithms
  - Dijkstra Algorithm

Shortest Path Problems

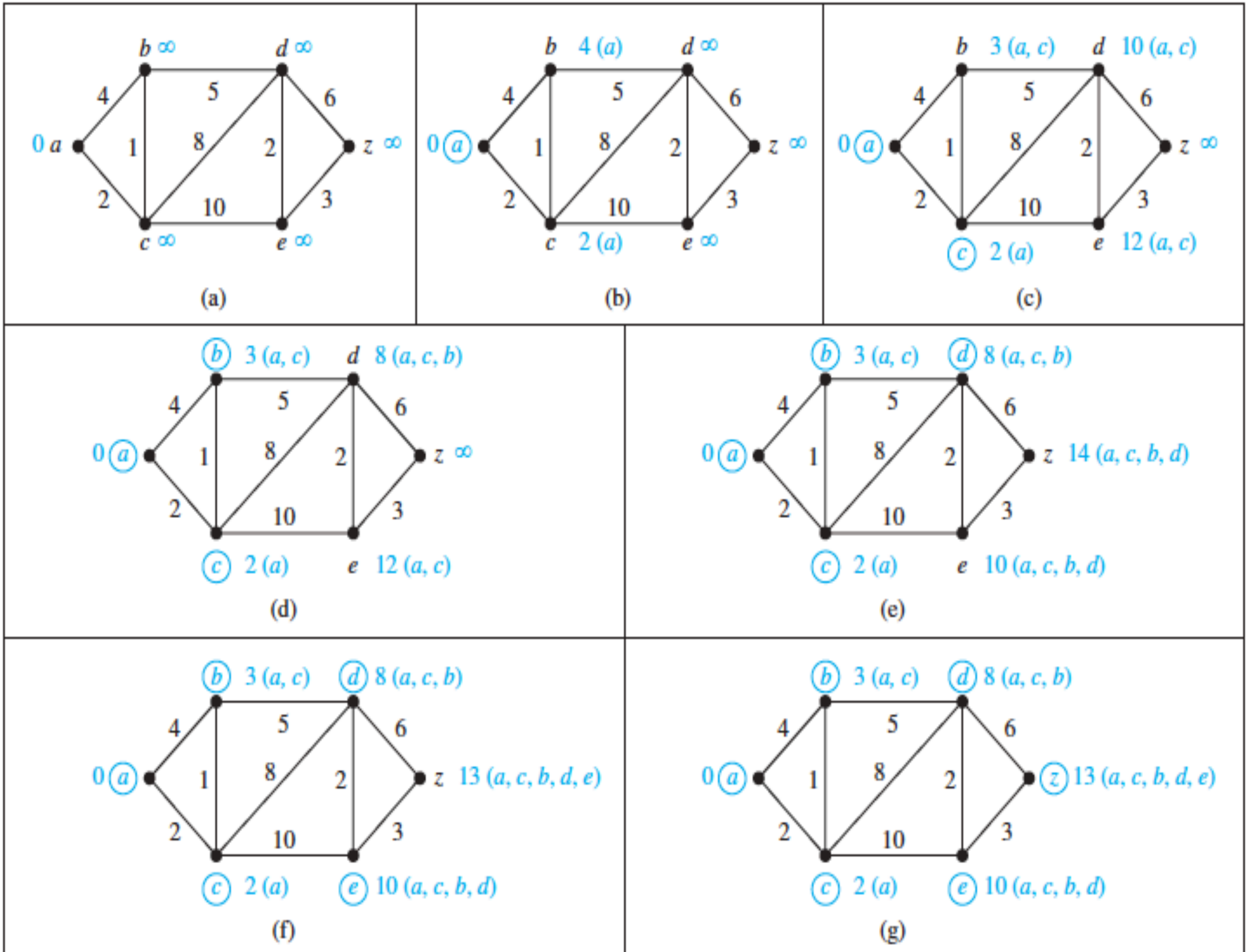
- Shortest Path Algorithms
  - Greedy Algorithms
  - Dijkstra Algorithm
  - All weights are positive and graph

  Is undirected



Dijkstra's Algorithm

procedure *Dijkstra*(G: weighted connected simple graph, with
    all weights positive)
{G has vertices $a = v_0, v_1, \ldots, v_n = z$ and lengths $w(v_i, v_j)$
    where $w(v_i, v_j) = \infty$ if $\{v_i, v_j\}$ is not an edge in G}
for $i := 1$ **to** n
    $L(v_i) := \infty$
$L(a) := 0$
$S := \emptyset$
{the labels are now initialized so that the label of a is 0 and all
    other labels are ∞, and S is the empty set}
while $z \notin S$
    $u :=$ a vertex not in S with $L(u)$ minimal
    $S := S \cup \{u\}$
    for all vertices v not in S
        if $L(u) + w(u, v) < L(v)$ **then** $L(v) := L(u) + w(u, v)$
        {this adds a vertex to S with minimal label and updates the
        labels of vertices not in S}
return $L(z)$ {$L(z) =$ length of a shortest path from a to z}



0 a
b ∞
d ∞
z ∞
c ∞
e ∞
4 5 6 1 8 2 2 10 3
(a)
0 a
b 4 (a)
d ∞
z ∞
c 2 (a)
e ∞
(b)
0 a
b 3 (a, c)
d 10 (a, c)
z ∞
c 2 (a)
e 12 (a, c)
(c)
0 a
b 3 (a, c)
d 8 (a, c, b)
z ∞
c 2 (a)
e 12 (a, c)
(d)
0 a
b 3 (a, c)
d 8 (a, c, b)
z 14 (a, c, b, d)
c 2 (a)
e 10 (a, c, b, d)
(e)
0 a
b 3 (a, c)
d 8 (a, c, b)
z 13 (a, c, b, d, e)
c 2 (a)
e 10 (a, c, b, d)
(f)
0 a
b 3 (a, c)
d 8 (a, c, b)
z 13 (a, c, b, d, e)
c 2 (a)
e 10 (a, c, b, d)
(g)

Dijkstra's Algorithm

Dijkstra's algorithm finds the length of a shortest path between two vertices in a connected simple undirected weighted graph.

Dijkstra's algorithm uses $O(n^2)$ operations (additions and comparisons) to find the length of a shortest path between two vertices in a connected simple undirected weighted graph with n vertices.

Planar Graphs

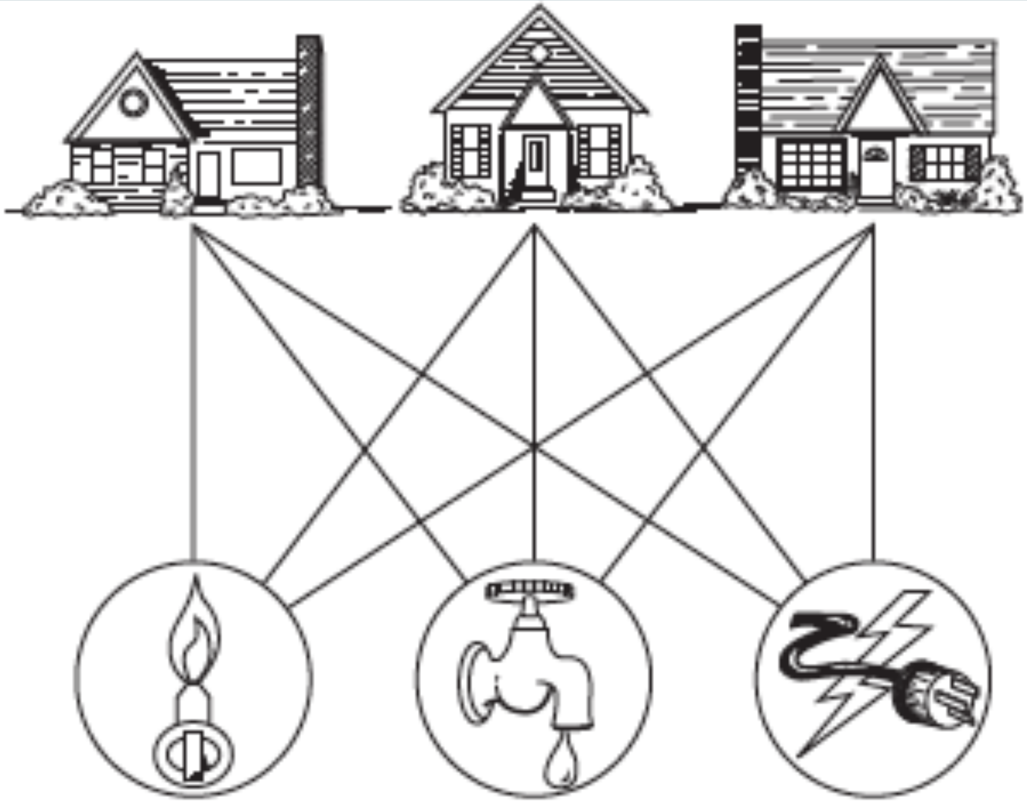


FIGURE 1 Three Houses and Three Utilities.

Whether a graph can be drawn in plane without edges crossing

Planar Graphs

A graph is called *planar* if it can be drawn in the plane without any edges crossing (where a crossing of edges is the intersection of the lines or arcs representing them at a point other than their common endpoint). Such a drawing is called a *planar representation* of the graph.

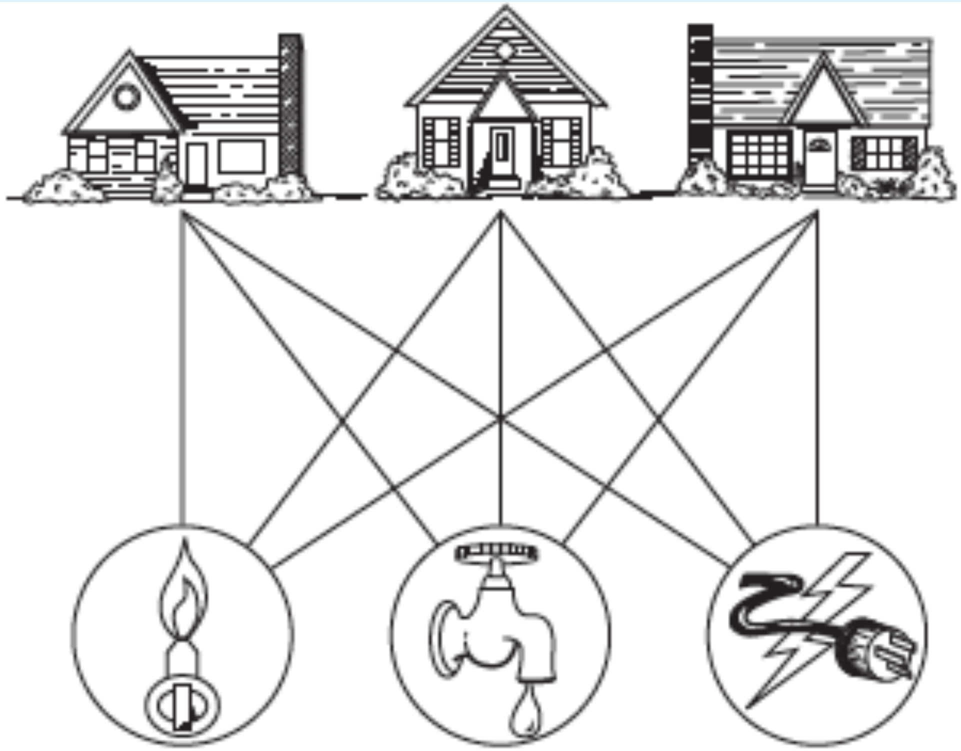


FIGURE 1 **Three Houses and Three Utilities.**

Planar Graphs

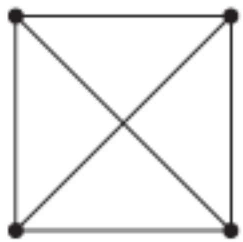


FIGURE 2 The Graph K_4.

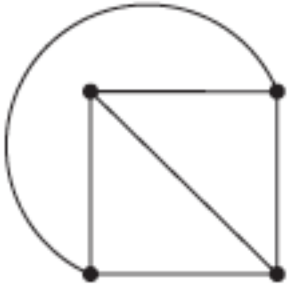


FIGURE 3 K_4 Drawn with No Crossings.

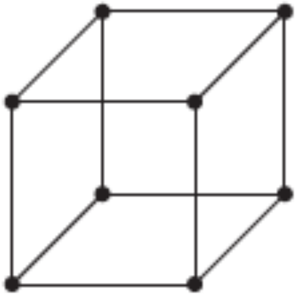


FIGURE 4 The Graph Q_3.

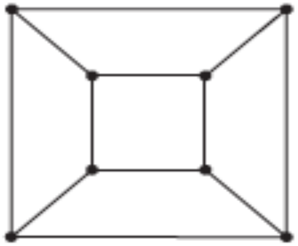


FIGURE 5 A Planar Representation of Q_3.

Planar Graphs

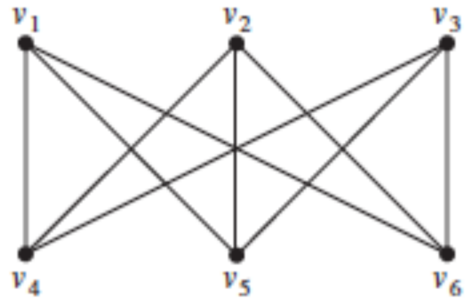


FIGURE 6 **The Graph $K_{3,3}$.**

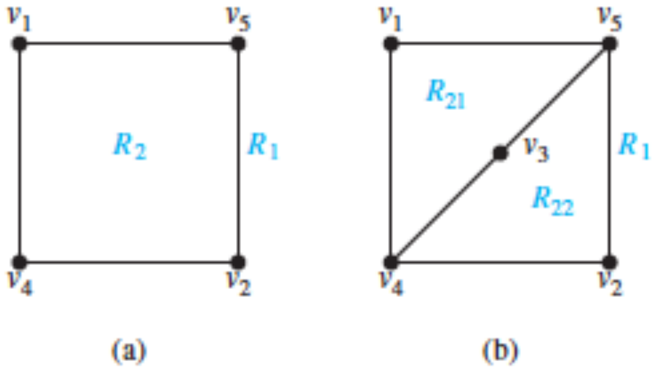


FIGURE 7 **Showing that $K_{3,3}$ Is Nonplanar.**

Application of Graph Planarity

- Design of electronic circuits
- Design of road networks

Graph Coloring

A *coloring* of a simple graph is the assignment of a color to each vertex of the graph so that no two adjacent vertices are assigned the same color.

The *chromatic number* of a graph is the least number of colors needed for a coloring of this graph. The chromatic number of a graph G is denoted by $\chi(G)$. (Here χ is the Greek letter *chi*.)

THE FOUR COLOR THEOREM The chromatic number of a planar graph is no greater than four.