Chapter 9
GRAPHS

Objectives

- 9.1- Graphs and Graph Models
- 9.2- Graph Terminology and Special Types of Graphs
- 9.3- Representing Graphs and Graph Isomorphism
- 9.4- Connectivity
- 9.5- Euler and Hamilton Paths
- 9.6- Shortest Path Problems

9.1- Graphs and Graph Models

DEFINITION 1

A *graph* $G = (V, E)$ consists of V, a nonempty set of *vertices* (or *nodes*) and E, a set of *edges*. Each edge has either one or two vertices associated with it, called its *endpoints*. An edge is said to *connect* its endpoints.

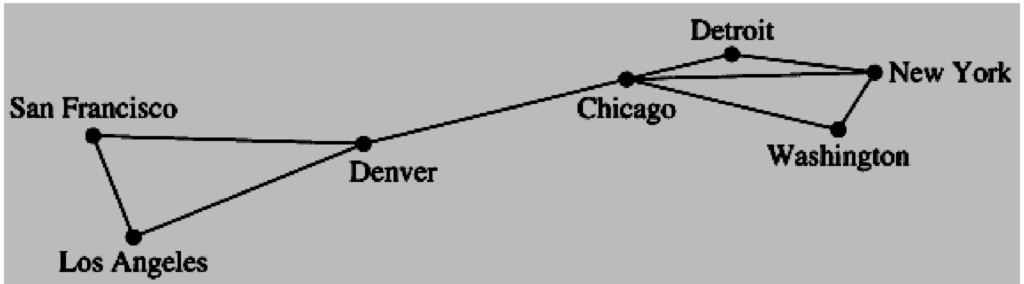


FIGURE 1 A Computer Network.

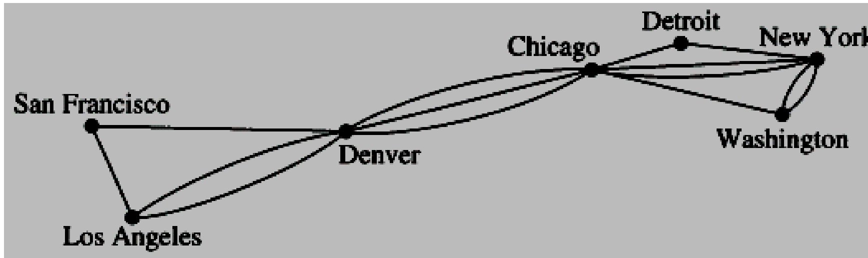


FIGURE 2 A Computer Network with Multiple Links between Data Centers.

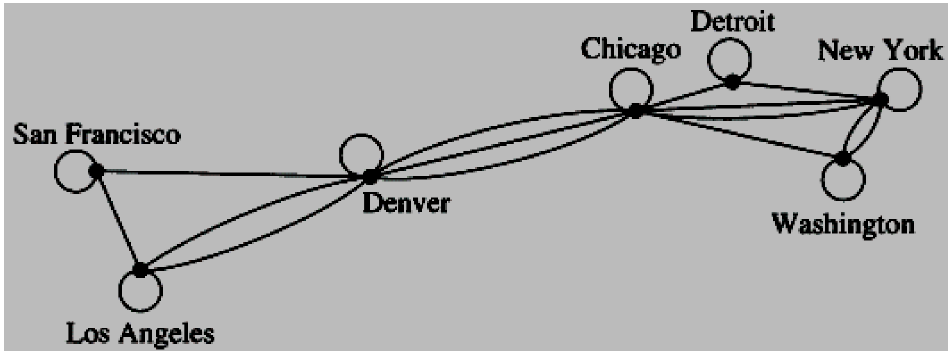


FIGURE 3 A Computer Network with Diagnostic Links.

Simple graph: No two edges connect the same pair of vertices

Multigraph: Multiple edges connect the same pair of vertices

Pseudograph: Multigraph may have loops

Undirected graph: Each edge has no direction

Directed graph: Each edge has a determined direction

Graphs and Graph Models….

DEFINITION 2

A *directed graph* (or *digraph*) (V, E) consists of a nonempty set of vertices V and a set of *directed edges* (or *arcs*) E. Each directed edge is associated with an ordered pair of vertices. The directed edge associated with the ordered pair (u, v) is said to *start* at u and *end* at v.

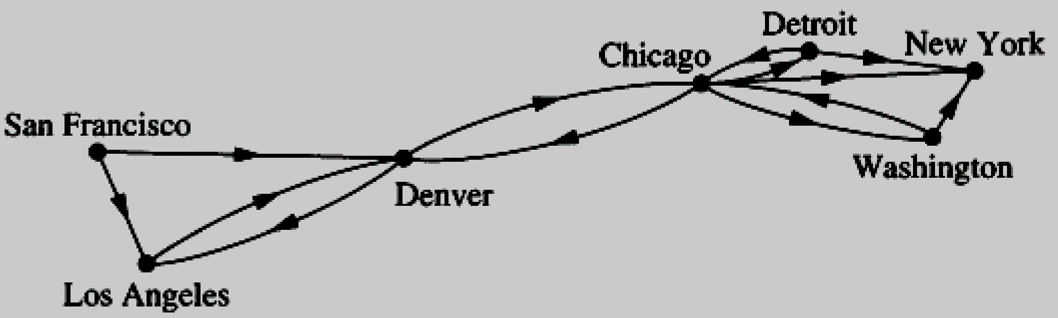


FIGURE 4 **A Communications Network with One-Way Communications Links.**

When there are m directed edges, each associated to an ordered pair of vertices (u,v), we say that (u,v) is an edge of multiplicity m

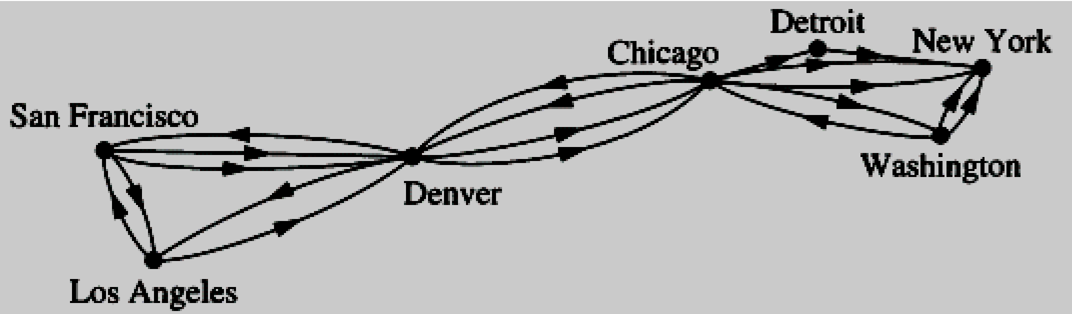


FIGURE 5 **A Computer Network with Multiple One-Way Links.**

FPT UNIVERSITY
DREAM OF INNOVATION

Graphs and Graph Models….

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

Graphs and Graph Models….

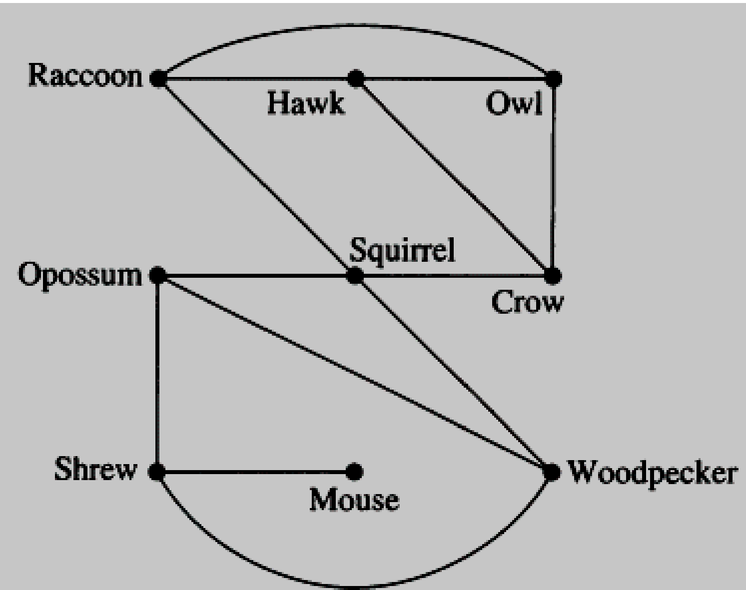


FIGURE 6 A Niche Overlap Graph.

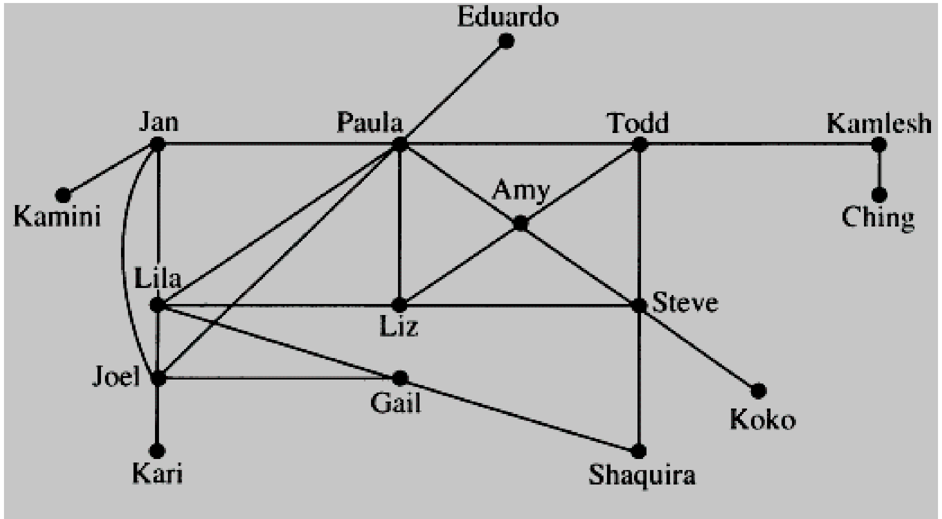
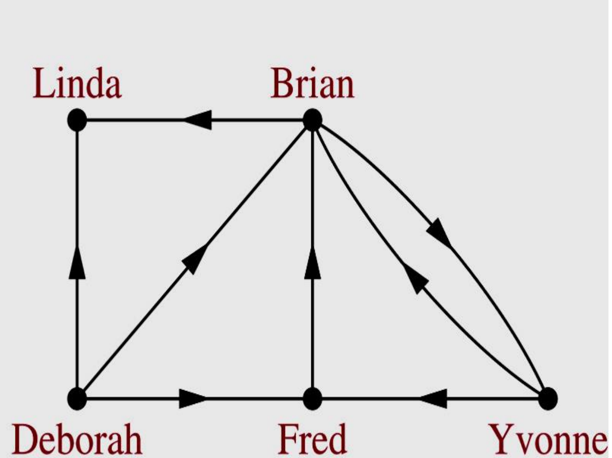


FIGURE 7 An Acquaintanceship Graph.

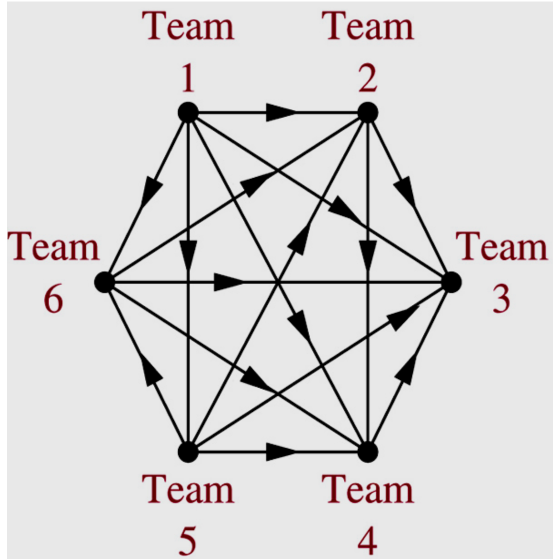
Niche Overlap Graph in Ecology
(sinh thái học) – Đồ thị lấn tổ

Acquaintanceship Graph
Đồ thị cho mô hình quan hệ giữa người

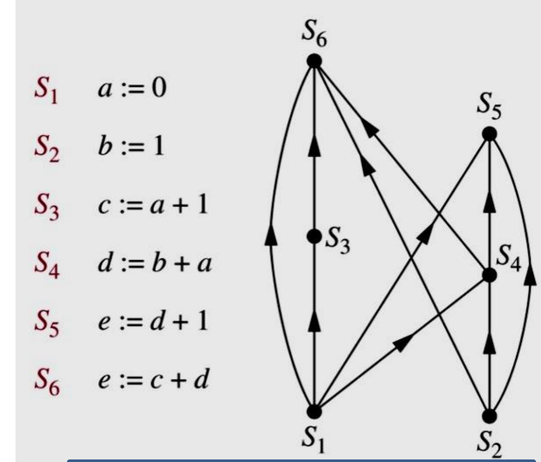
Graphs and Graph Models….



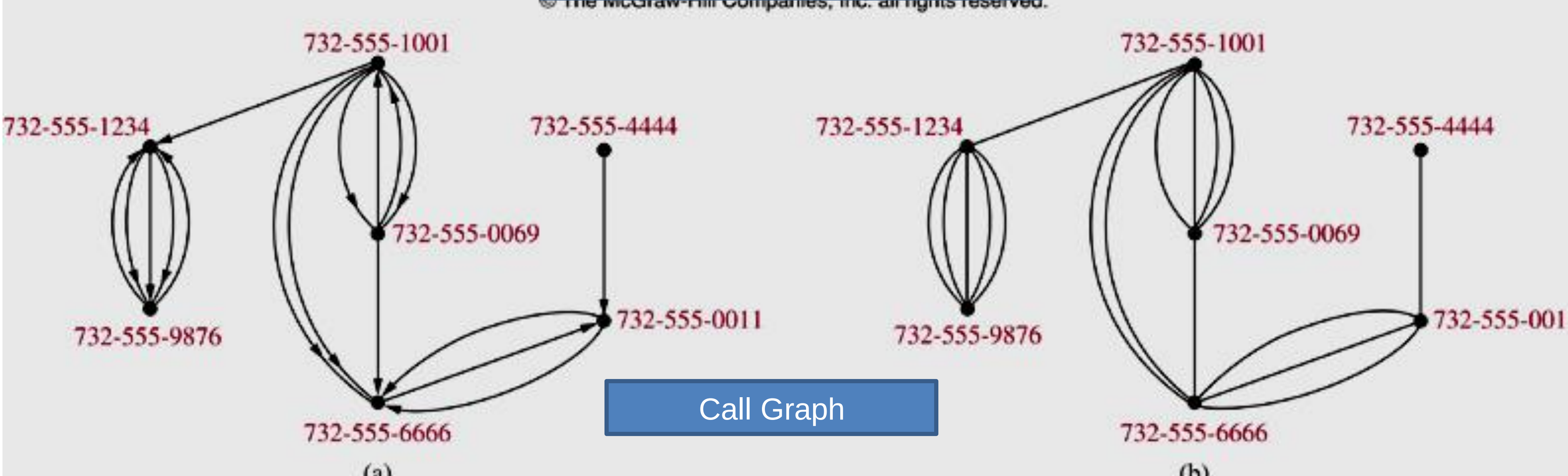
An Influence Graph



A Graph Model of a Round-Robin Tournament



An Precedence Graph



Call Graph

9.2- Graph Terminology and Special Types of Graphs

- Basic Terminology
- Some Special Simple Graphs
- Bipartite Graphs
- Some Applications of Special Types of Graphs
- New Graphs From Old

Basic Terminology

DEFINITION 1

Two vertices u and v in an undirected graph G are called *adjacent* (or *neighbors*) in G if u and v are endpoints of an edge of G. If e is associated with $\{u, v\}$, the edge e is called *incident with* the vertices u and v. The edge e is also said to *connect* u and v. The vertices u and v are called *endpoints* of an edge associated with $\{u, v\}$.

DEFINITION 2

The *degree of a vertex in an undirected graph* is the number of edges incident with it, except that a loop at a vertex contributes twice to the degree of that vertex. The degree of the vertex v is denoted by $\deg(v)$.

Determine degrees of vertices in the following graphs:

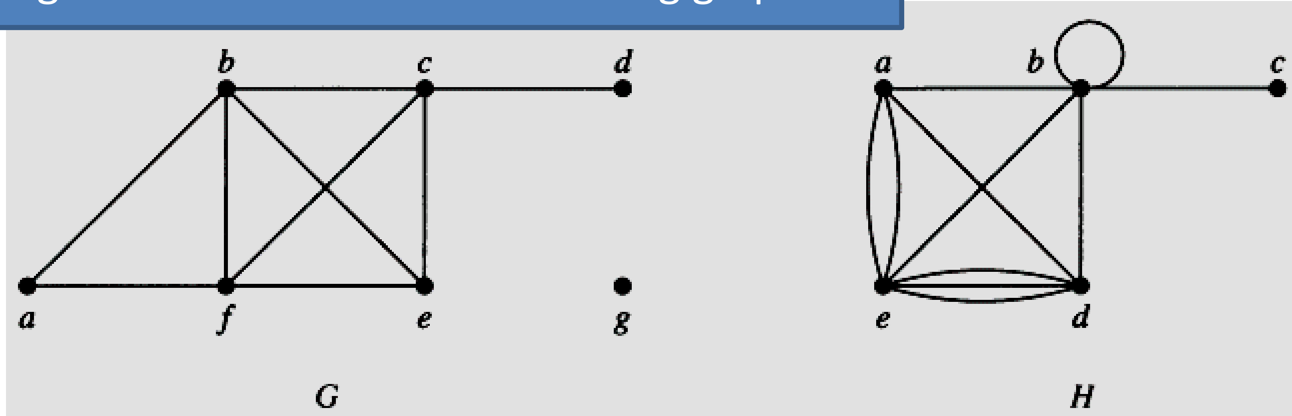


FIGURE 1 **The Undirected Graphs G and H.**

Basic Terminology …

A vertex of degree zero is called **isolated.** It follows that an isolated vertex is not adjacent to any vertex. Vertex g in graph G in Example 1 is isolated. A vertex is **pendant** if and only if it has degree one. Consequently, a pendant vertex is adjacent to exactly one other vertex. Vertex d in graph G in Example 1 is pendant.

THEOREM 1 THE HANDSHAKING THEOREM Let $G = (V, E)$ be an undirected graph with e edges. Then

$$2e = \sum_{v \in V} \deg(v).$$

(Note that this applies even if multiple edges and loops are present.)

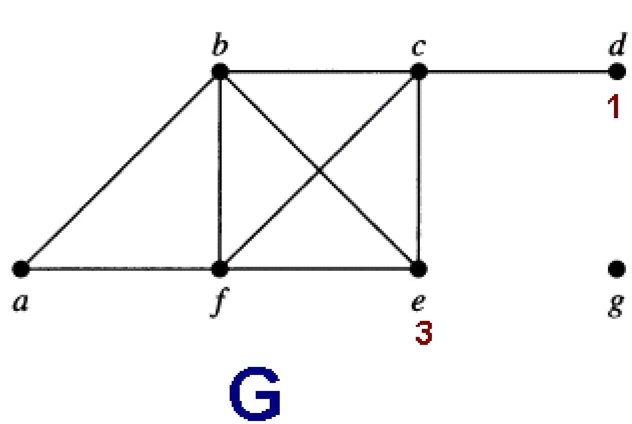
Basic Terminology….

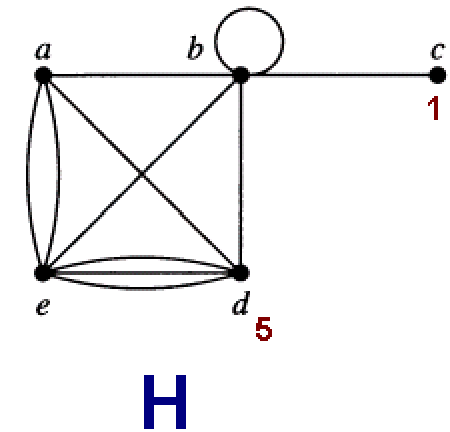
THEOREM 2

An undirected graph has an even number of vertices of odd degree.

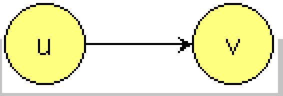
Proof: page 599

Check yourself:





Basic Terminology….



DEFINITION 3

When (u, v) is an edge of the graph G with directed edges, u is said to be *adjacent to* v and v is said to be *adjacent from* u. The vertex u is called the *initial vertex* of (u, v), and v is called the *terminal* or *end vertex* of (u, v). The initial vertex and terminal vertex of a loop are the same.

DEFINITION 4

In a graph with directed edges the *in-degree of a vertex* v, denoted by $\deg^{-}(v)$, is the number of edges with v as their terminal vertex. The *out-degree of* v, denoted by $\deg^{+}(v)$, is the number of edges with v as their initial vertex. (Note that a loop at a vertex contributes 1 to both the in-degree and the out-degree of this vertex.)

In-degree $\deg^{-}(c)=3$
Out-degree $\deg^{+}(c)=2$

Basic Terminology….

THEOREM 3

Let $G = (V, E)$ be a graph with directed edges. Then

$$\sum_{v \in V} \deg^-(v) = \sum_{v \in V} \deg^+(v) = |E|.$$

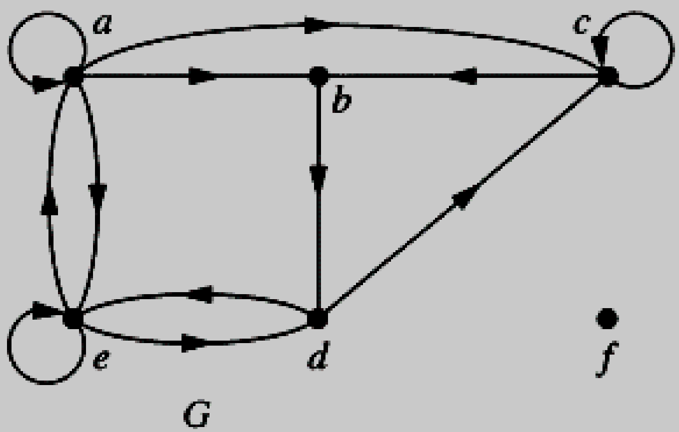


FIGURE 2 The Directed Graph G.

Vertex	In-degree	Out-degree
a	2	4
b	2	1
c	3	2
d	2	2
e	3	3
f	0	0
Sum	12	12

Some Special Simple Graphs

Complete Graphs The **complete graph on n vertices**, denoted by K_n, is the simple graph that contains exactly one edge between each pair of distinct vertices. The graphs K_n, for $n = 1, 2, 3, 4, 5, 6$, are displayed in Figure 3.

Đồ thị đầy đủ

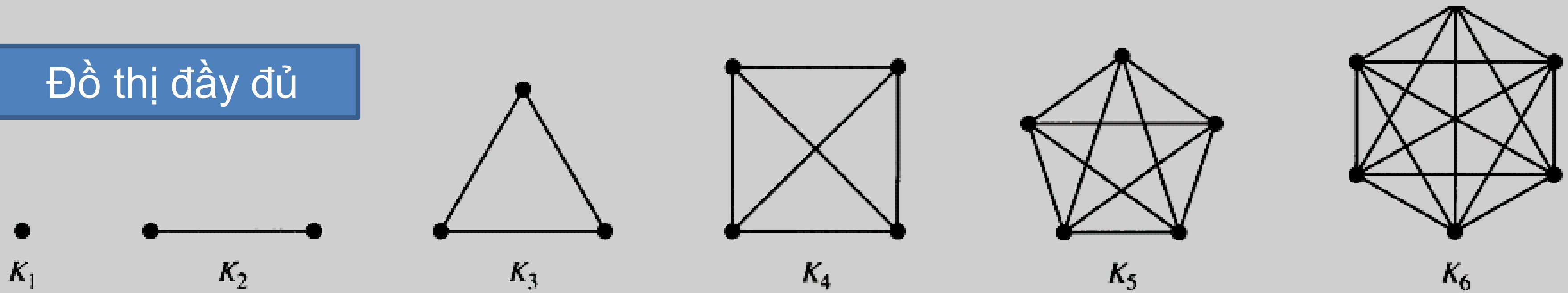
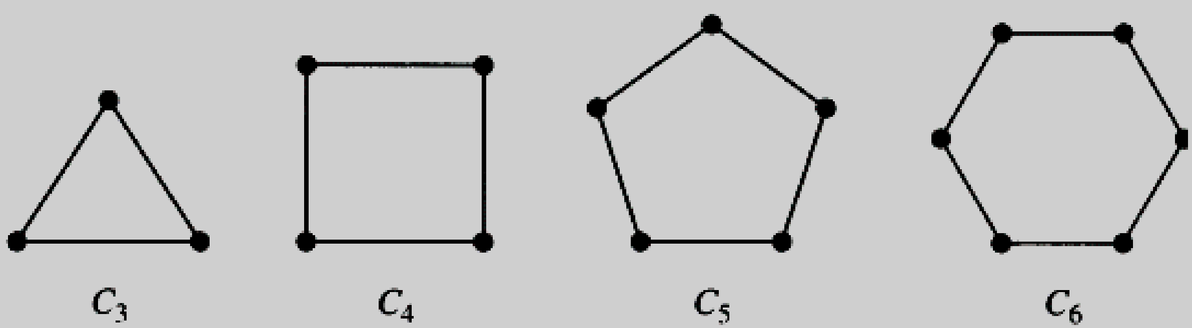


FIGURE 3 The Graphs K_n for $1 \leq n \leq 6$.

Cycles The **cycle** C_n, $n \geq 3$, consists of n vertices $v_1, v_2, \ldots, v_n$ and edges $\{v_1, v_2\}$, $\{v_2, v_3\}, \ldots, \{v_{n-1}, v_n\}$, and $\{v_n, v_1\}$. The cycles C_3, C_4, C_5, and C_6 are displayed in Figure 4.

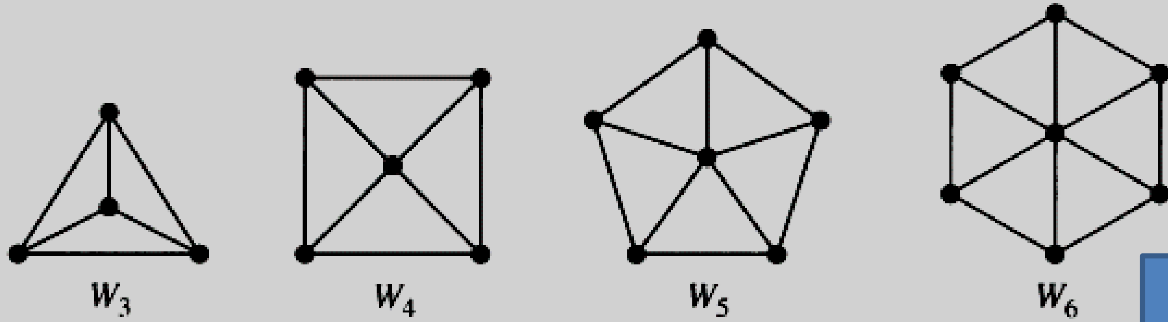


Đồ thị vòng

FIGURE 4 The Cycles C_3, C_4, C_5, and C_6.

Some Special Simple Graphs

Wheels We obtain the **wheel** W_n when we add an additional vertex to the cycle C_n, for $n \geq 3$, and connect this new vertex to each of the n vertices in C_n, by new edges. The wheels W_3, W_4, W_5, and W_6 are displayed in Figure 5.



Đồ thị bánh xe

FIGURE 5 **The Wheels W_3, W_4, W_5, and W_6.**

***n*-Cubes** The ***n*-dimensional hypercube,** or ***n*-cube,** denoted by Q_n, is the graph that has vertices representing the 2^n bit strings of length n. Two vertices are adjacent if and only if the bit strings that they represent differ in exactly one bit position.

Đồ thị khối

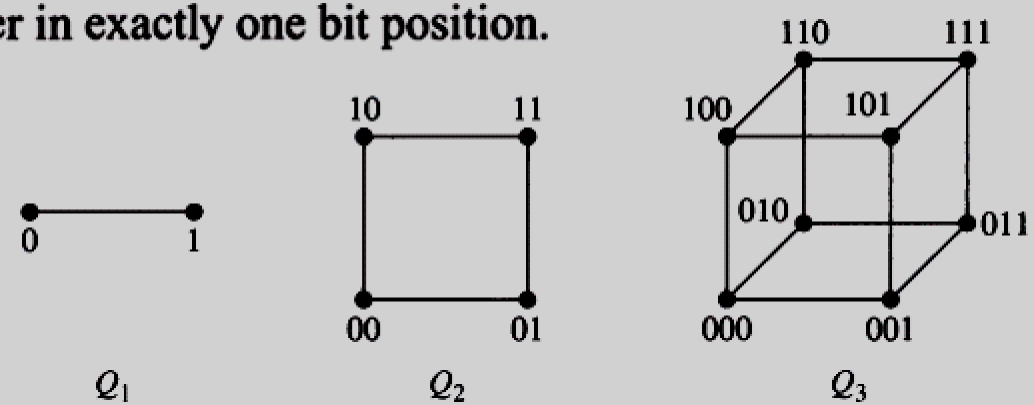
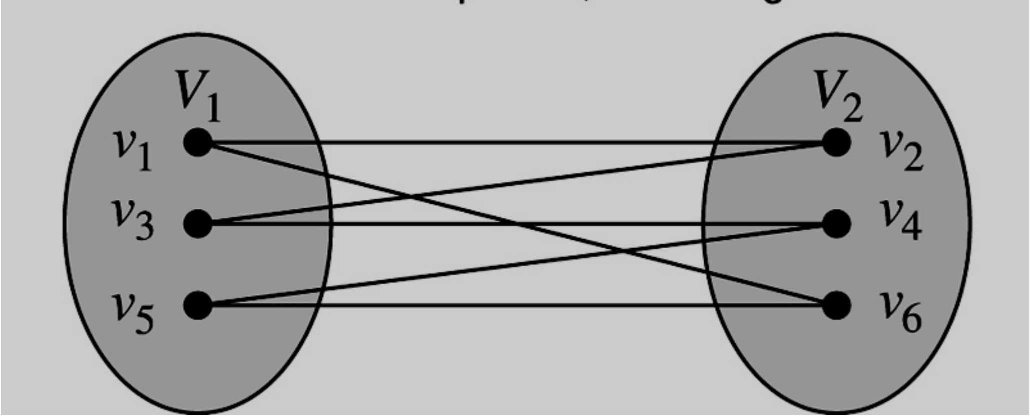


FIGURE 6 **The *n*-cube Q_n for $n = 1, 2,$ and 3.**

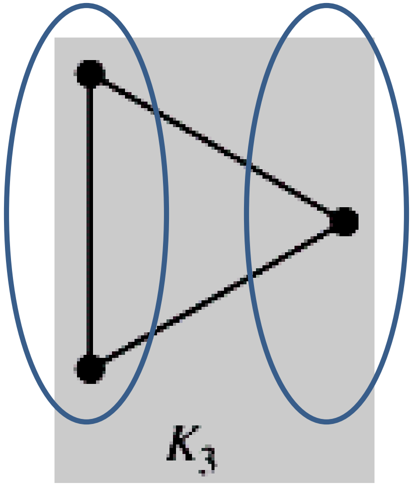
Bipartite Graphs – Đồ thị lưỡng phân

DEFINITION 5

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). When this condition holds, we call the pair (V_1, V_2) a *bipartition* of the vertex set V of G.

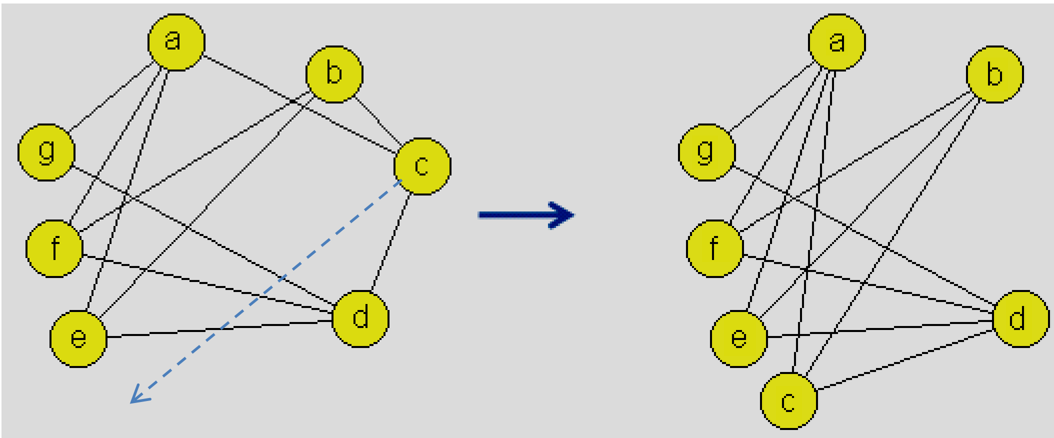


bipartite

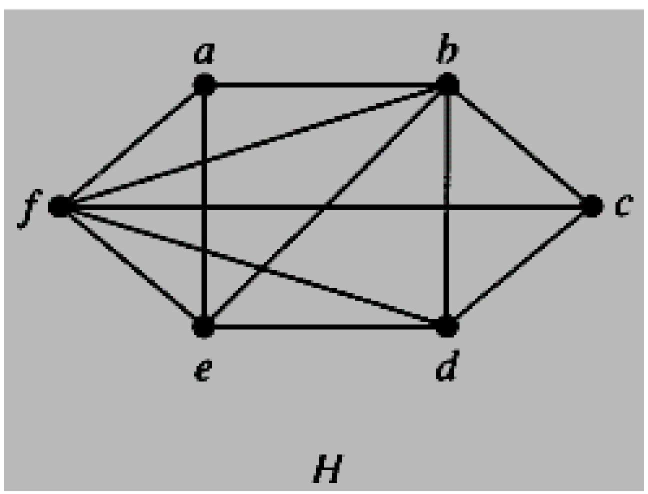


Non-bipartite

Bipartite Graphs…



bipartite



Edge considered	Set 1	Set 2
ab	a	b
ae	a	b, e
af	a	b, e, f
bc	ac	
be	a, c, e	
➔ **Non- bipartite**		

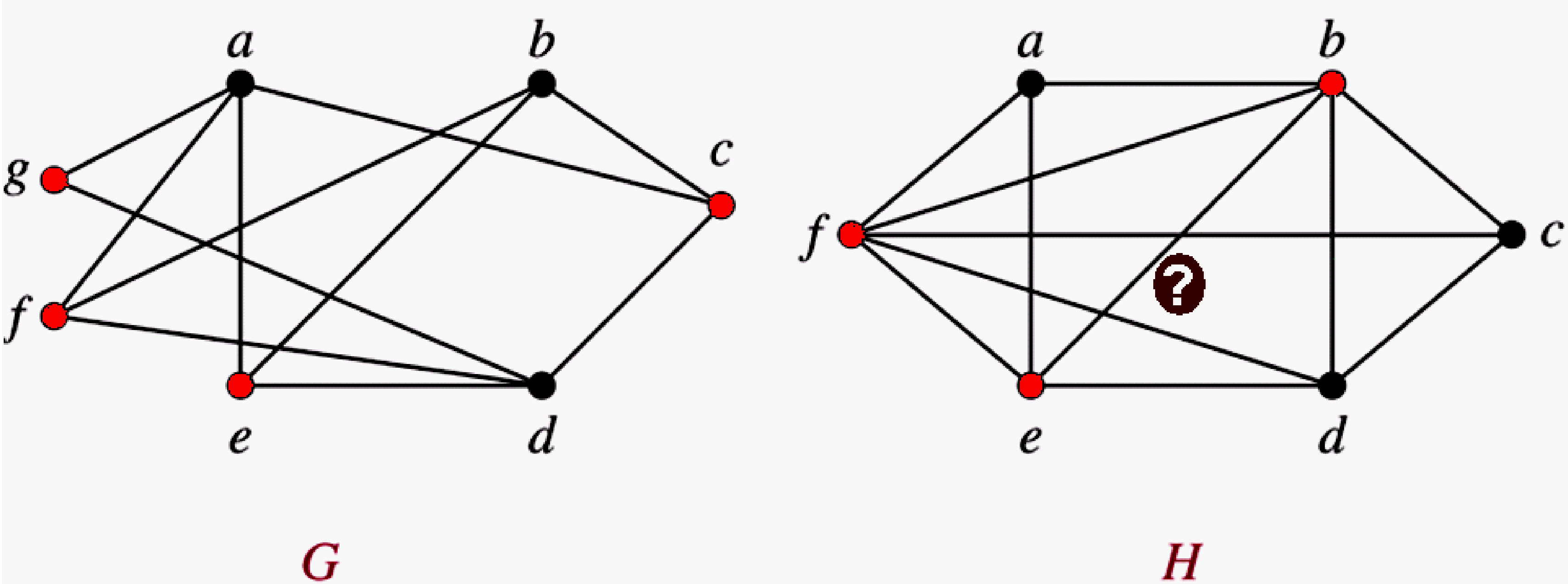
Conflict

Bipartite Graphs…

THEOREM 4

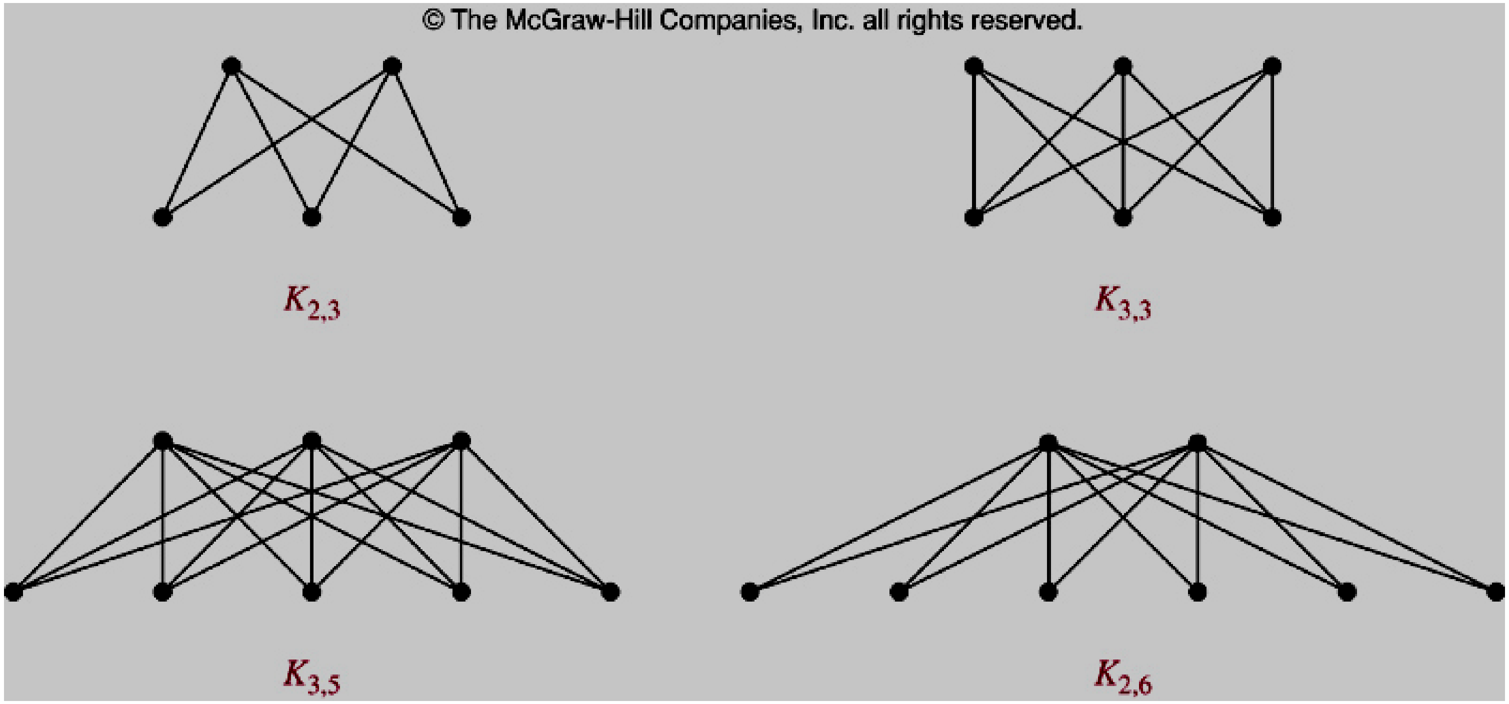
Proof: page 603

A simple graph is bipartite if and only if it is possible to assign one of two different colors to each vertex of the graph so that no two adjacent vertices are assigned the same color.



Bipartite Graphs…

Some Complete Bipartite Graphs



Complete Bipartite Graphs $K_{m,n}$

Some Applications of Special Types of Graphs

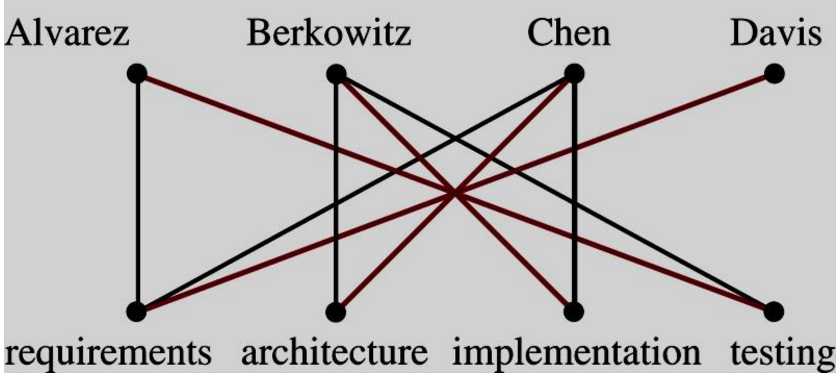


Figure 10: Modeling the jobs for which employees have been trained

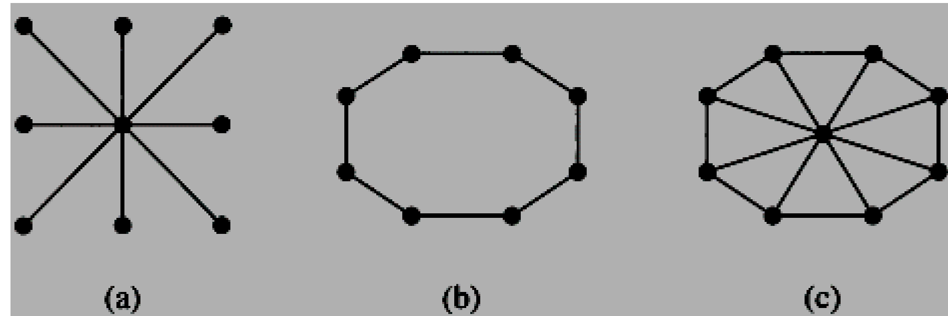


FIGURE 11 Star, Ring, and Hybrid Topologies for Local Area Networks.

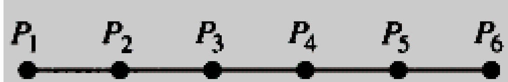


FIGURE 12 A Linear Array for Six Processors.

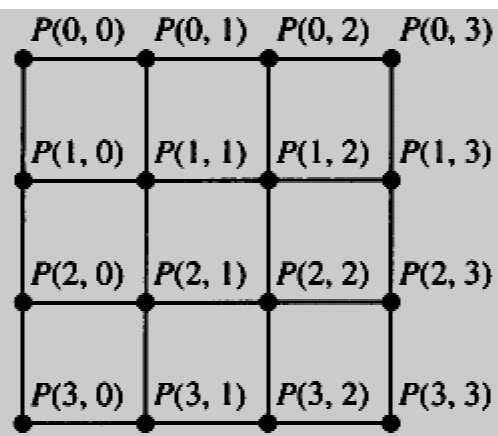


FIGURE 13 A Mesh Network for 16 Processors.

New Graphs From Old

DEFINITION 6

A *subgraph of a graph* $G = (V, E)$ is a graph $H = (W, F)$, where $W \subseteq V$ and $F \subseteq E$. A subgraph H of G is a *proper subgraph* of G if $H \neq G$.

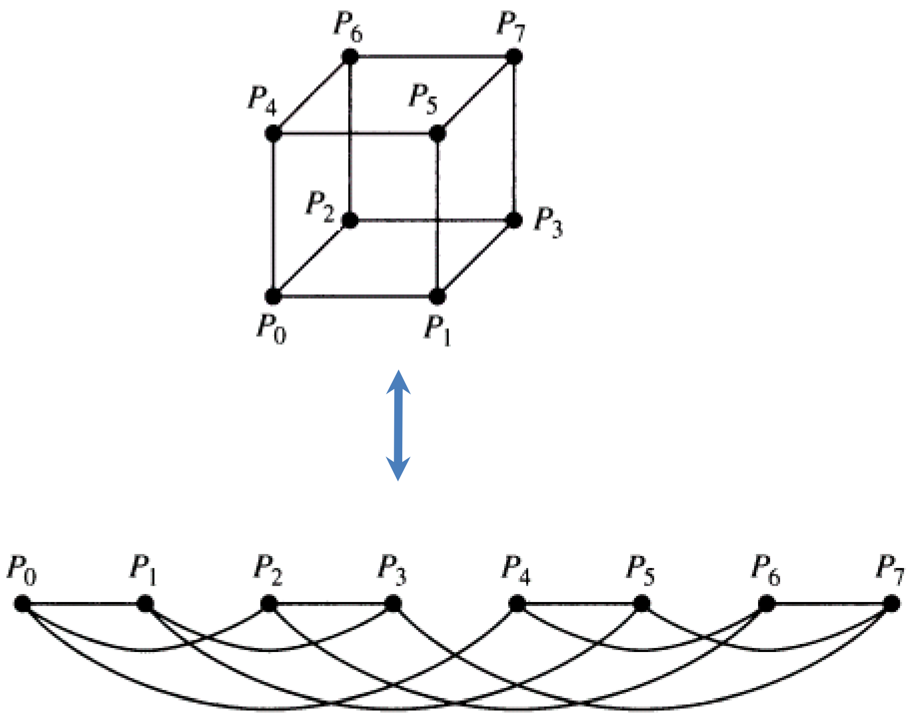


FIGURE 14 **A Hypercube Network for Eight Processors.**

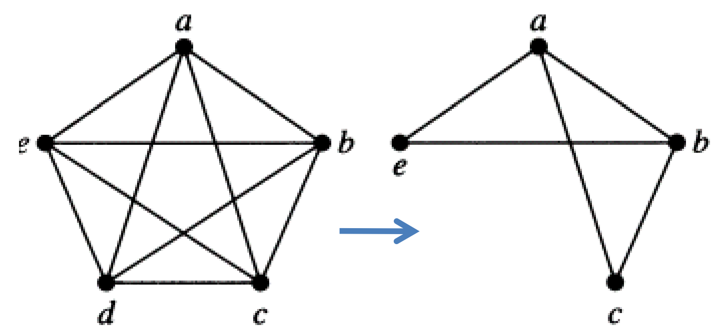


FIGURE 15 **A Subgraph of K_5.**

New Graphs From Old

DEFINITION 7

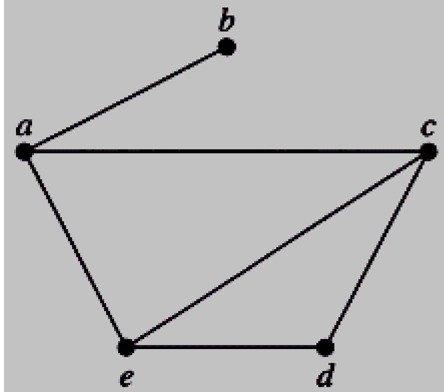
The *union* of two simple graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ is the simple graph with vertex set $V_1 \cup V_2$ and edge set $E_1 \cup E_2$. The union of G_1 and G_2 is denoted by $G_1 \cup G_2$.

9.3- Representing Graphs and Graph Isomorphism (sự đẳng cấu của đồ thị)

- Representing Graphs
- Adjacent Matrices – Ma trận kề
- Incidence Matrices – Ma trận cạnh nối
- Isomorphism of Graphs : Tính đồng dạng, đẳng cấu của đồ thị

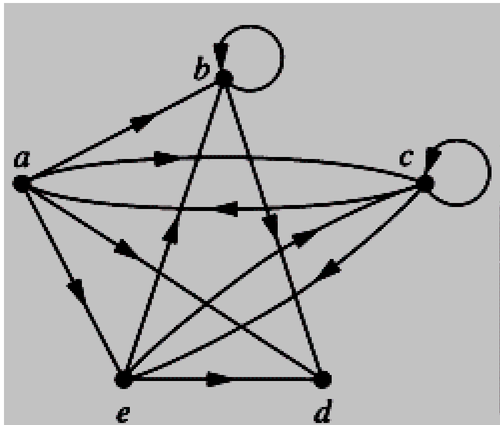
Representing Graphs

- Using **Adjacency list** : For each vertex u in the graph, there is a list of vertex v which there is an edge between u and v.



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

FIGURE 1 A Simple Graph.

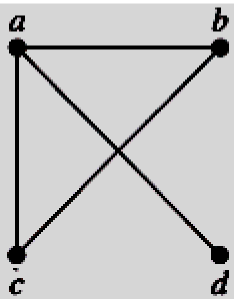


Initial Vertex	*Terminal Vertices*
a	*b, c, d, e*
b	*b, d*
c	*a, c, e*
d	
e	*b, c, d*

FIGURE 2 A Directed Graph.

Adjacent Matrices

$$\mathbf{A} = [a_{ij}],\ a_{ij} = \begin{cases} 1 & \text{if } \{v_i, v_j\} \text{ is an edge of } G, \\ 0 & \text{otherwise.} \end{cases}$$



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

FIGURE 3 Simple Graph.

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

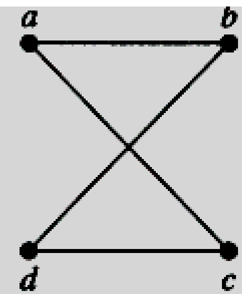
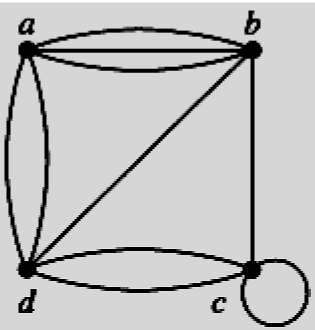


FIGURE 4 A Graph with the Given Adjacency Matrix.



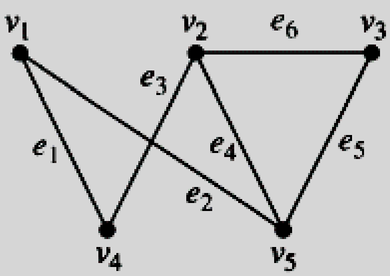
$$\begin{bmatrix} 0 & 3 & 0 & 2 \\ 3 & 0 & 1 & 1 \\ 0 & 1 & 1 & 2 \\ 2 & 1 & 2 & 0 \end{bmatrix}$$

FIGURE 5 A Pseudograph.

We need a trade-off between adjacency list and adjacency matrices because:
-Dense matrices : Memory is used efficient.
-sparse matrices: there is a waste in memory using.

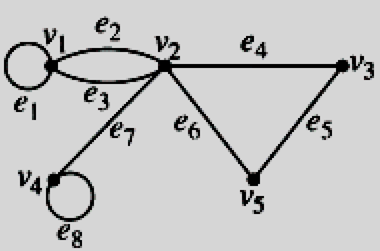
Incidence Matrices

$$\mathbf{M} = [m_{ij}],\ m_{ij} = \begin{cases} 1 & \text{when edge } e_j \text{ is incident with } v_i, \\ 0 & \text{otherwise.} \end{cases}$$



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

FIGURE 6 An Undirected Graph.



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

FIGURE 7 A Pseudograph.

Isomorphism of Graphs

- We may draw two graphs in the same way?
- In chemistry, different compounds can have the same molecular formula but can differ in structure. So, they can not be drawn the same way.

DEFINITION 1

The simple graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are *isomorphic* if there is a one-to-one and onto function f from V_1 to V_2 with the property that a and b are adjacent in G_1 if and only if $f(a)$ and $f(b)$ are adjacent in G_2, for all a and b in V_1. Such a function f is called an *isomorphism*.

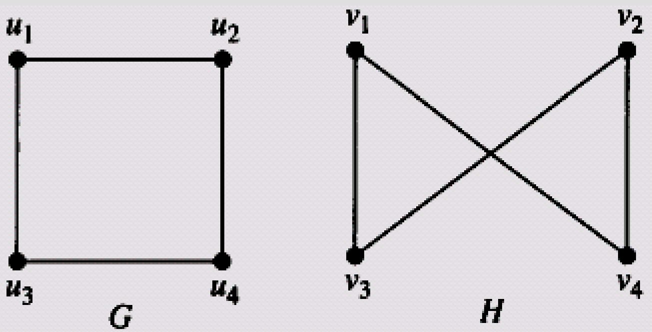


Figure 8: G and H are isomorphic

A permutation of vertices in H is similar with vertices in G ➔ Complexity: O (n!) ➔ It is often difficult to determine whether two graphs are isomorphic if number of vertices is large.

Isomorphism of Graphs

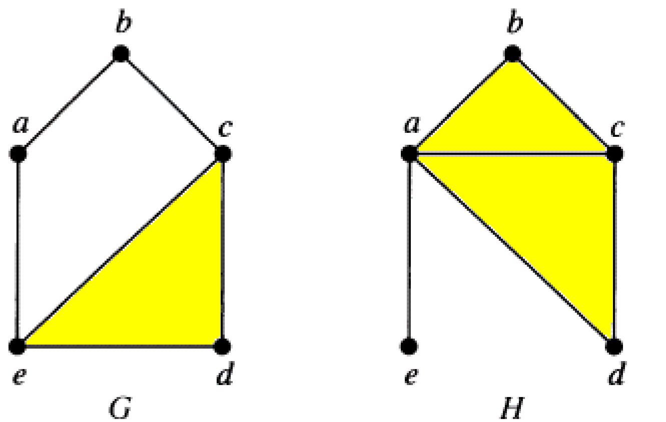


Figure 9: G and H are not isomorphic

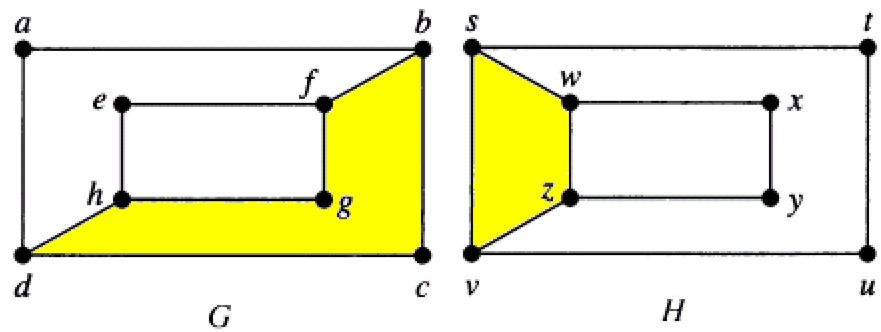


Figure 10: G and H are not isomorphic

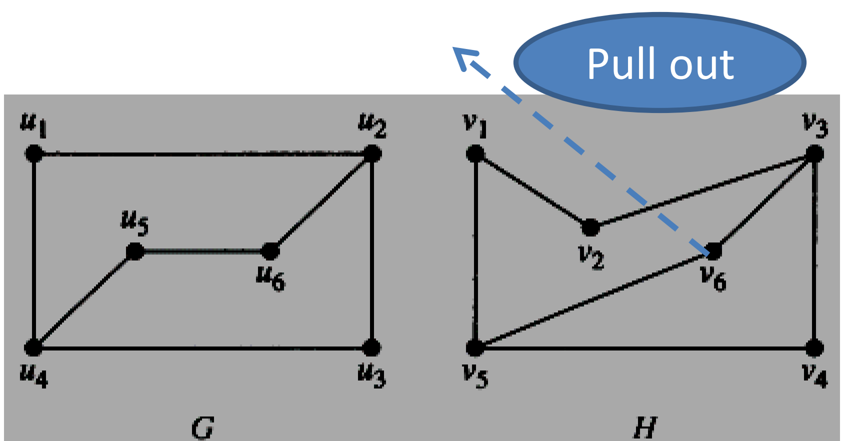


Figure 12: G and H are isomorphic

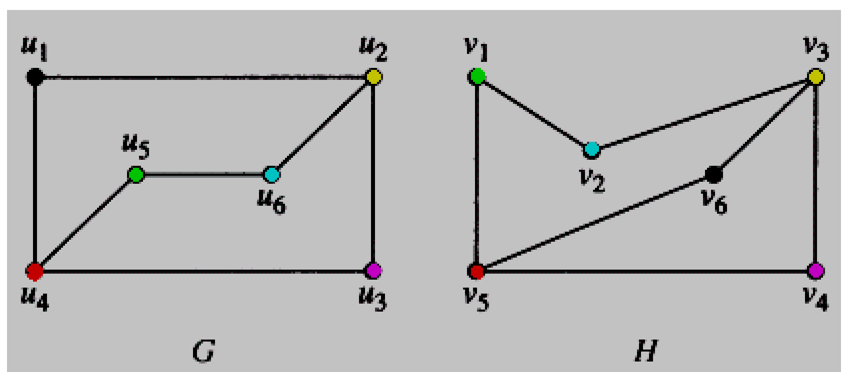


Figure 12: Function from G to H

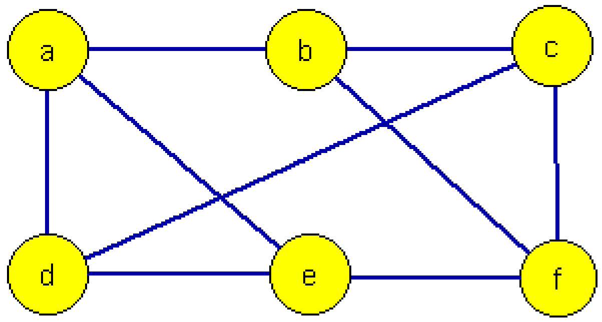
9.4- Connectivity (tính liên thông)

- Path
- Connectedness In Undirected Graphs
- Connectedness In Directed Graphs
- Paths and Isomorphism
- Counting Paths Between Vertices

Path

DEFINITION 1

- 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 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 $x_n = v$.
- 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.



Determine the following ordered set of vertices whether it is a path and if it is a path, what is it's length?, is it a circuit?, is it simple?

{a,b,c,f,e}

{c,b,d,e,f}

{a,b,c,d,e,a,b,f}

FPT UNIVERSITY
DREAM OF INNOVATION

Path

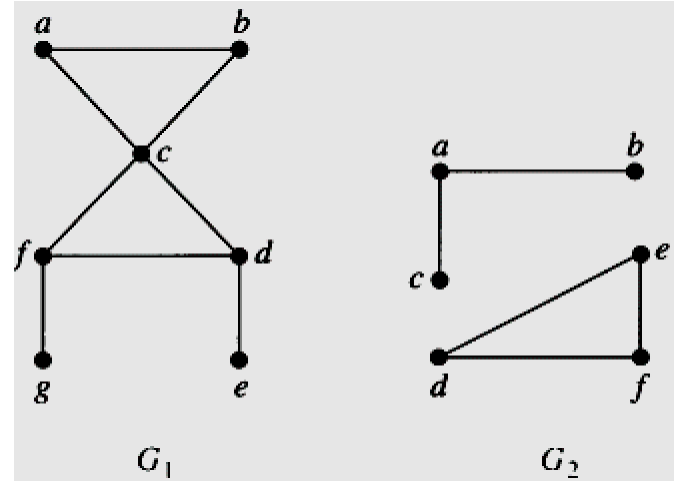
DEFINITION 2

- Let n be a nonnegative integer and G a directed graph.
- A *path* of length n from u to v in G is a sequence of edges $e_1, e_2, \ldots, e_n$ of G 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 $x_n = v$.
- When there are no multiple edges in the directed graph, this path is denoted by its vertex sequence $x_0, x_1, x_2, \ldots, x_n$.
- A path of length greater than zero that begins and ends at the same vertex is called a *circuit* or *cycle*.
- A path or circuit is called *simple* if it does not contain the same edge more than once.

Connectedness In Undirected Graphs

DEFINITION 3

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



Connected **Disconnected**

THEOREM 1

Proof: page 625

There is a simple path between every pair of distinct vertices of a connected undirected graph.

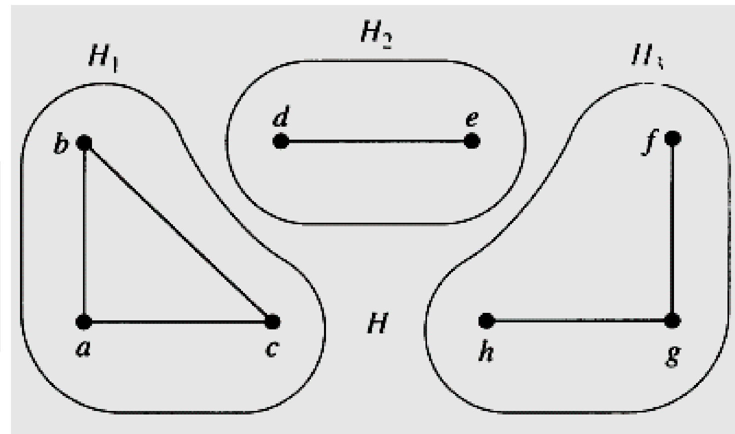
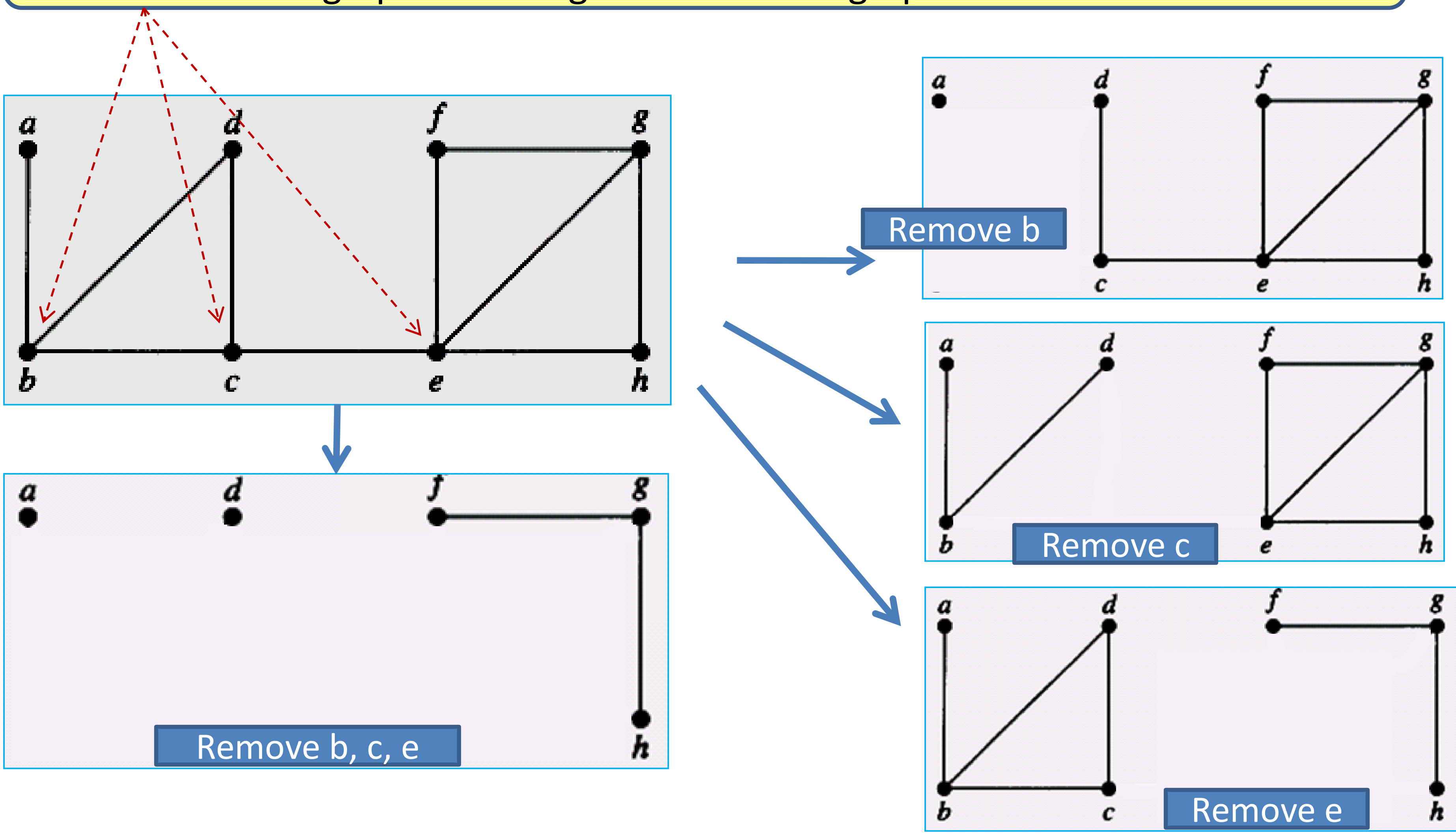


FIGURE 3 The Graph *H* and Its Connected Components H_1, H_2, and H_3.

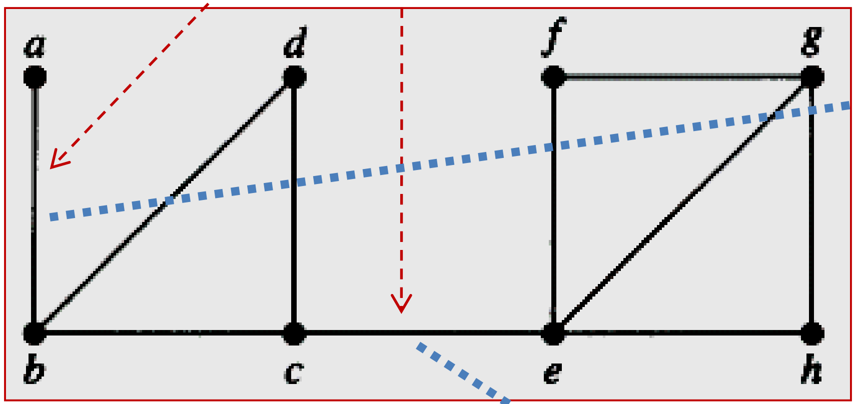
Connectedness In Undirected Graphs

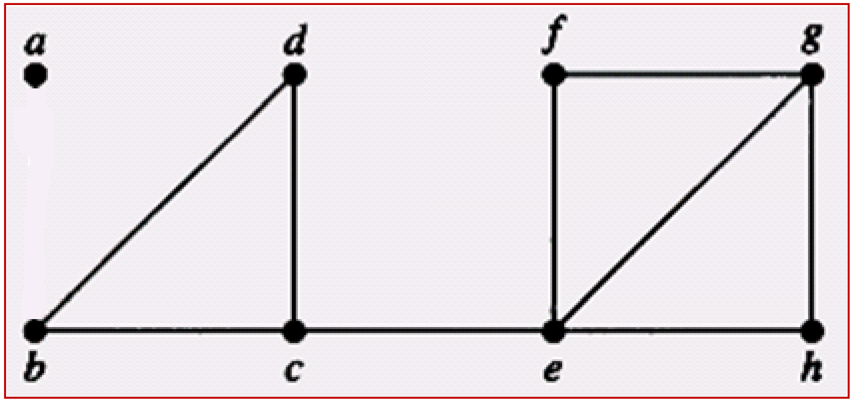
Cut vertex (articulation point – điểm khớp): It’s removal will produce disconnected subgraph from original connected graph.

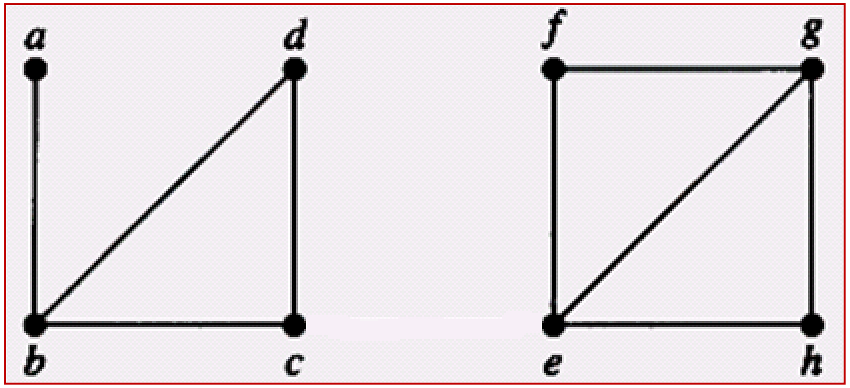


Connectedness In Undirected Graphs

Cut edge (bridge): It’s removal will produce subgraphs which are more connected components (thành phần liên thông) than in the original graph







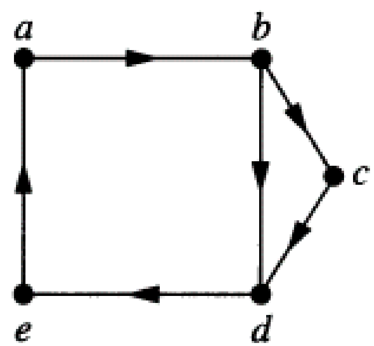
Connectedness In Directed Graphs

DEFINITION 4

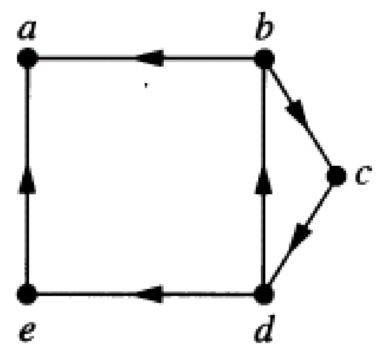
A directed graph is *strongly connected* if there is a path from a to b and from b to a whenever a and b are vertices in the graph.

DEFINITION 5

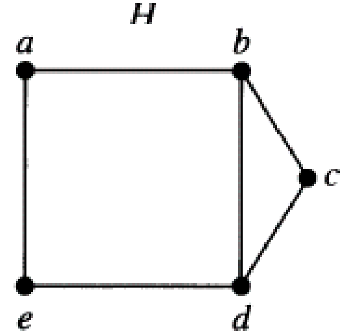
A directed graph is *weakly connected* if there is a path between every two vertices in the underlying undirected graph.



G



H



underlying undirected graph of H

-G is strongly connected → G is weakly also.
- H is not strongly connected and it is weakly connected.
-(you can verify these)

Path and Isomorphism

- Using path to determine whether two graphs are isomorphic:

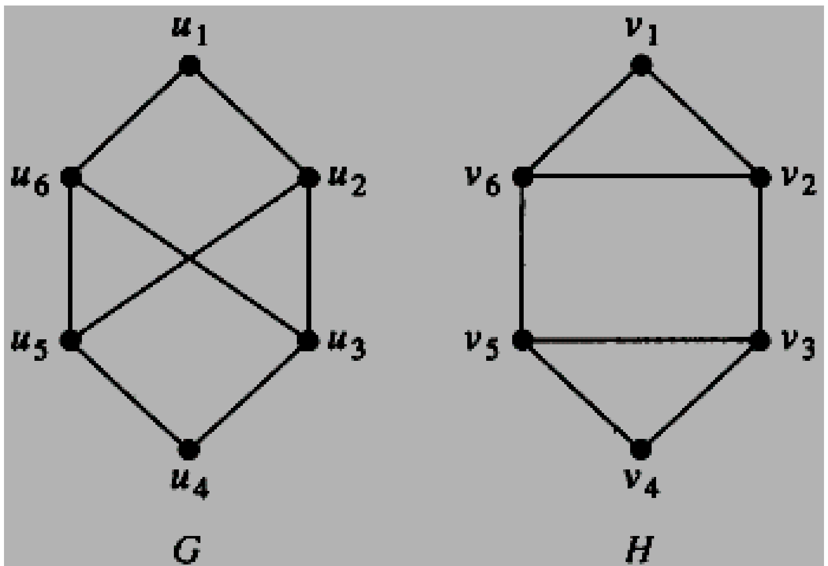


FIGURE 6 The Graphs *G* and *H*.

G and H have the same:
-Number of vertices
-Number of edges
- 2 vertices degree 2
- 2 vertices degree 3
But:
H has a loop with minimum length 3
and
G has a loop with minimum length 4
➔ They are not isomorphic.

Path and Isomorphism

- Using path to determine whether two graphs are isomorphic:

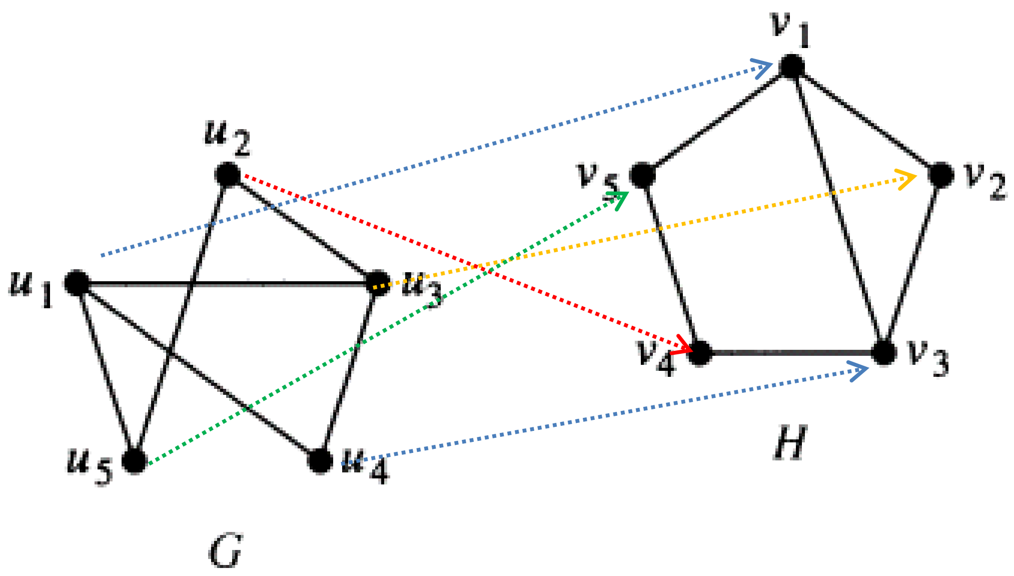


FIGURE 7 The Graphs G and H.

They are isomorphic.

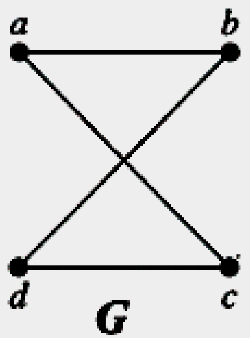
Counting Paths Between Vertices

THEOREM 2 Proof: page 628

Let G be a graph with adjacency matrix $\mathbf{A}$ with respect to the ordering $v_1, v_2, \ldots, v_n$ (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 $\mathbf{A}^r$.

EXAMPLE 14

How many paths of length four are there from a to d in the simple graph G?



Solution: The adjacency matrix of G (ordering the vertices as a, b, c, d) is

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \qquad \mathbf{A}^{\boxed{4}} = \begin{bmatrix} 8 & 0 & 0 & 8 \\ 0 & 8 & 8 & 0 \\ 0 & 8 & 8 & 0 \\ 8 & 0 & 0 & 8 \end{bmatrix}$$

Result= 8

a, b, a, b, d
a, b, a, c, d
a, b, d, b, d
a, b, d, c, d
a, c, a, b, d
a, c, a, c, d
a, c, d, b, d
a, c, d, c, d

9.5- Euler and Hamilton Path

- Euler Paths and Circuit
  - Paths and circuits contains all Edges of a graph.
- Hamilton Paths and Circuit
  - Paths and circuits contains all vertices of a graph.

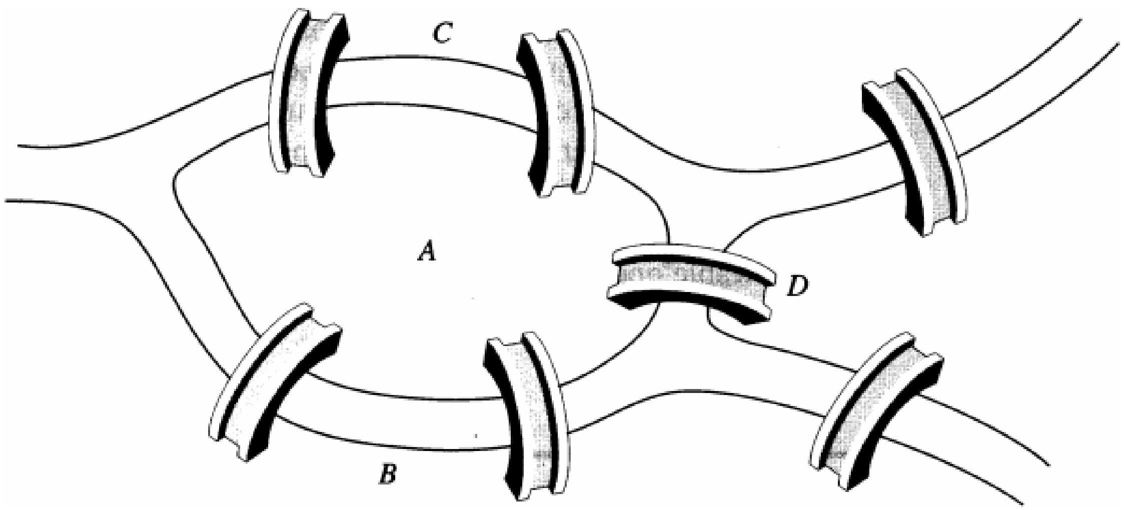


FIGURE 1 **The Seven Bridges of Königsberg.**

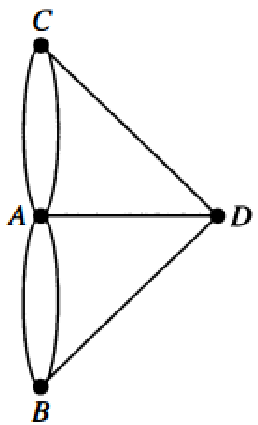


FIGURE 2 **Multigraph Model of the Town of Königsberg.**

Euler Paths and Circuits

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 Which of the undirected graphs in Figure 3 have an Euler circuit? Of those that do not, which have an Euler path?

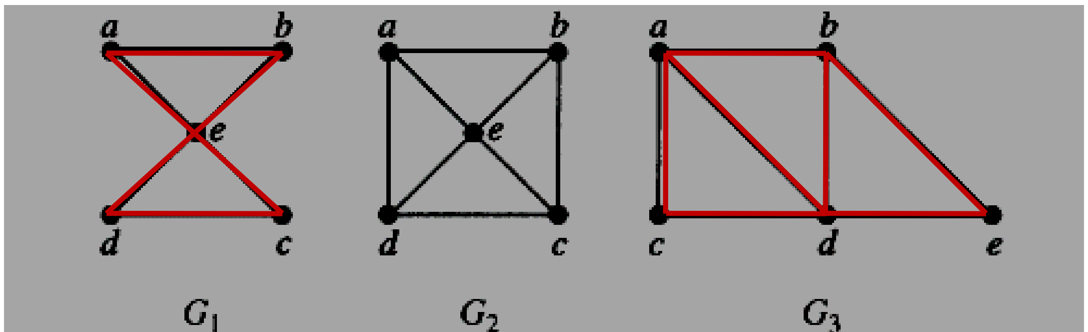


FIGURE 3 The Undirected Graphs G_1, G_2, and G_3.

G_1: has an Euler circuit:
a,e,c,d,e,b,a
G3: has no Euler circuit but it has Euler path a,c,d,e,b,d,a,b.
G2 has no Euler circuit and also has no Euler path.

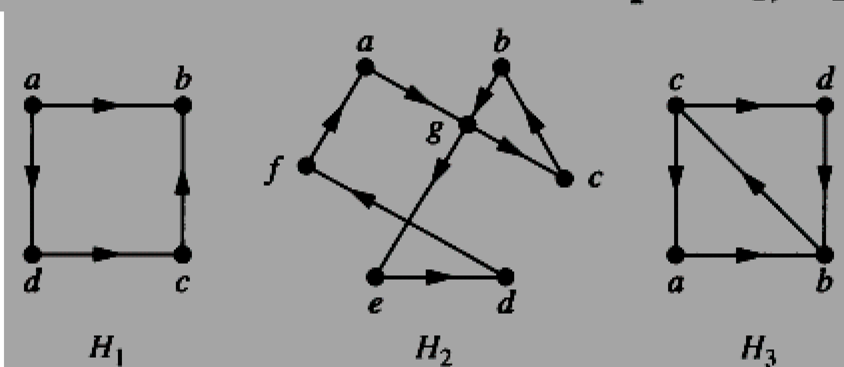


Figure 4: Directed Graphs

EXAMPLE 2
Which of the directed graphs in Figure 4 have an Euler circuit? Of those that do not, which have an Euler path?

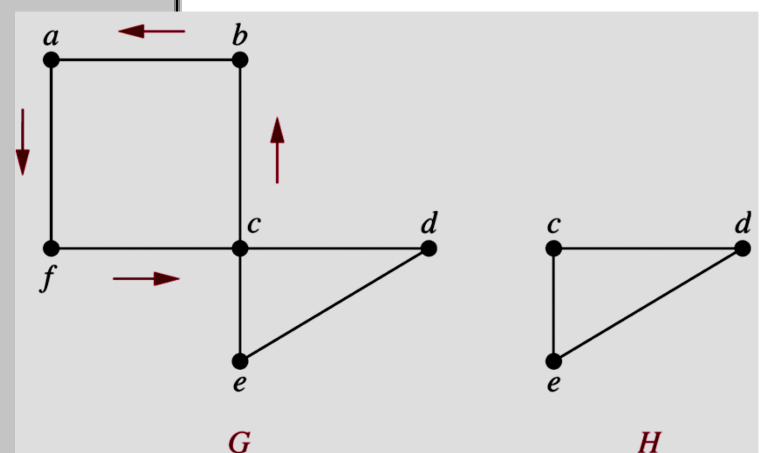
Euler Paths and Circuits

THEOREM 1

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

ALGORITHM 1 Constructing Euler Circuits.

procedure *Euler*(G: connected multigraph with all vertices of even degree)
circuit := a circuit in G beginning at an arbitrarily chosen
  vertex with edges successively added to form a path that
  returns to this vertex
H := G with the edges of this circuit removed
while H has edges
begin
  subcircuit := a circuit in H beginning at a vertex in H that
    also is an endpoint of an edge of *circuit*
  H := H with edges of *subcircuit* and all isolated vertices
    removed
  circuit := *circuit* with *subcircuit* inserted at the appropriate
    vertex
end {*circuit* is an Euler circuit}



Euler Paths and Circuits

THEOREM 2

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

Example 4

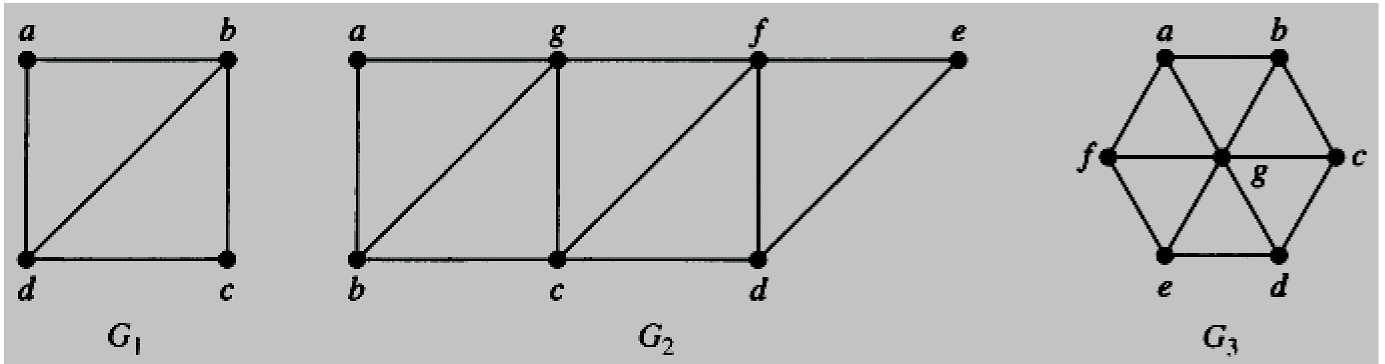


FIGURE 7 Three Undirected Graphs.

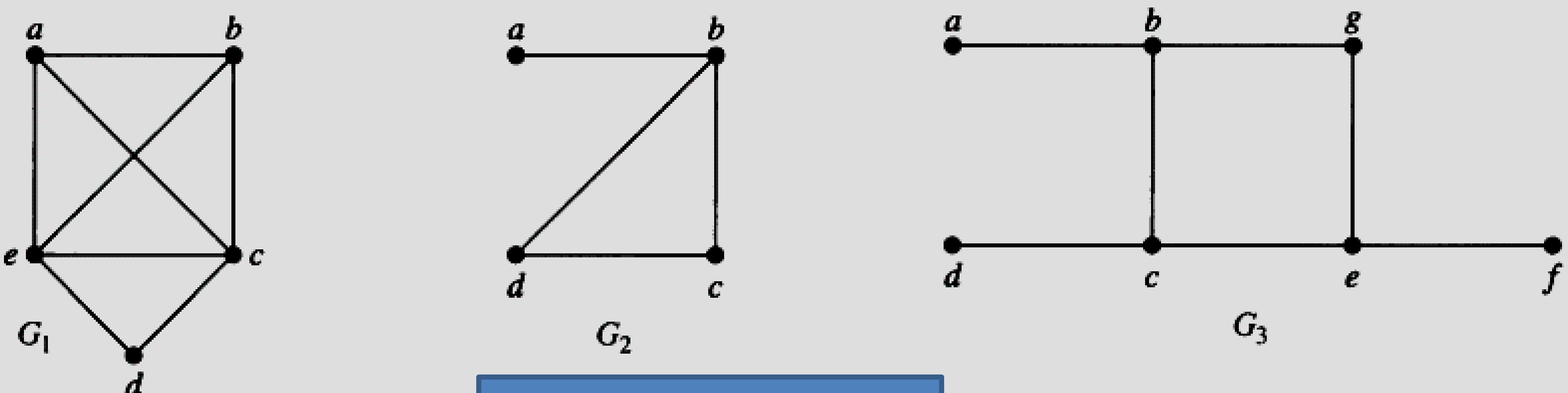
- G_1 contains exactly two vertices of odd degree, namely b and d. So, they are endpoints of Euler path: b,d,c,b,a,d or b,c,d,b,a,d, . . .
- Do similarly on G_2
- G_3 have six vertices of add degree. So, G_3 has no Euler path

Hamilton Paths and Circuits

DEFINITION 2

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 10 Three Simple Graphs.



Hamilton circuit:
(a,b,c,d,e,a)
(a,b,e,d,c,a)
. . .

Hamilton circuit: none
Hamilton Paths:
(a,b,c,d)
(a,b,d,c)
. . .

Hamilton circuit: none
Hamilton Path: none

Hamilton Paths and Circuits

There are no known simple necessary and sufficient criteria for the existence of Hamilton circuits. However, many theorems are known that give sufficient conditions for the existence of Hamilton circuits.

Also, certain properties can be used to show that a graph has no Hamilton circuit. For instance, a graph with a vertex of degree one cannot have a Hamilton circuit, because in a Hamilton circuit, each vertex is incident with two edges in the circuit. Moreover, if a vertex in the graph has degree two, then both edges that are incident with this vertex must be part of any Hamilton circuit. Also, note that when a Hamilton circuit is being constructed and this circuit has passed through a vertex, then all remaining edges incident with this vertex, other than the two used in the circuit, can be removed from consideration. Furthermore, a Hamilton circuit cannot contain a smaller circuit within it.

THEOREM 3

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.

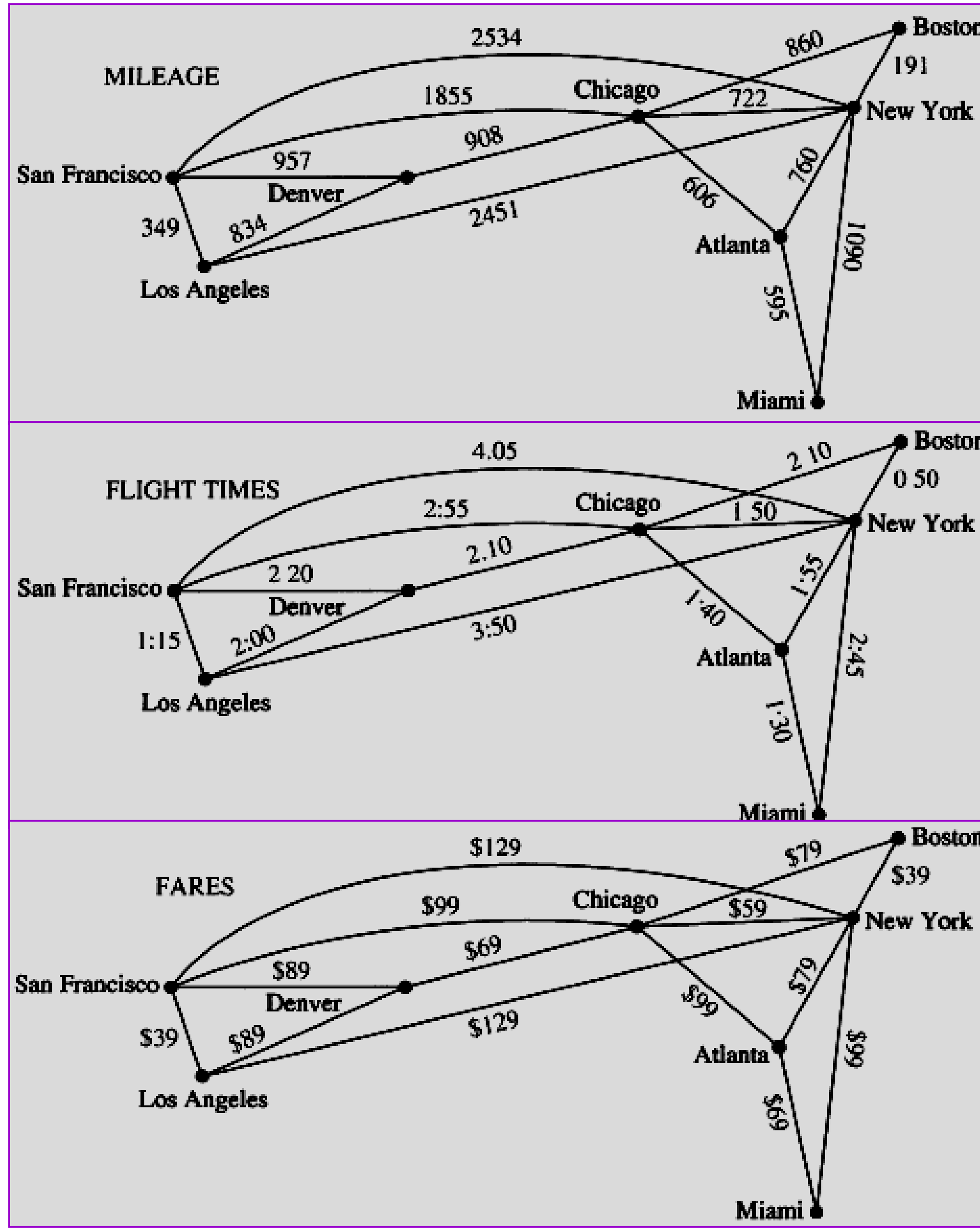
THEOREM 4

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.

9.6- Shortest Path Problems

- Introduction
- A Shortest Path Algorithm
- The Traveling Salesman Problem

Introduction



Introduction

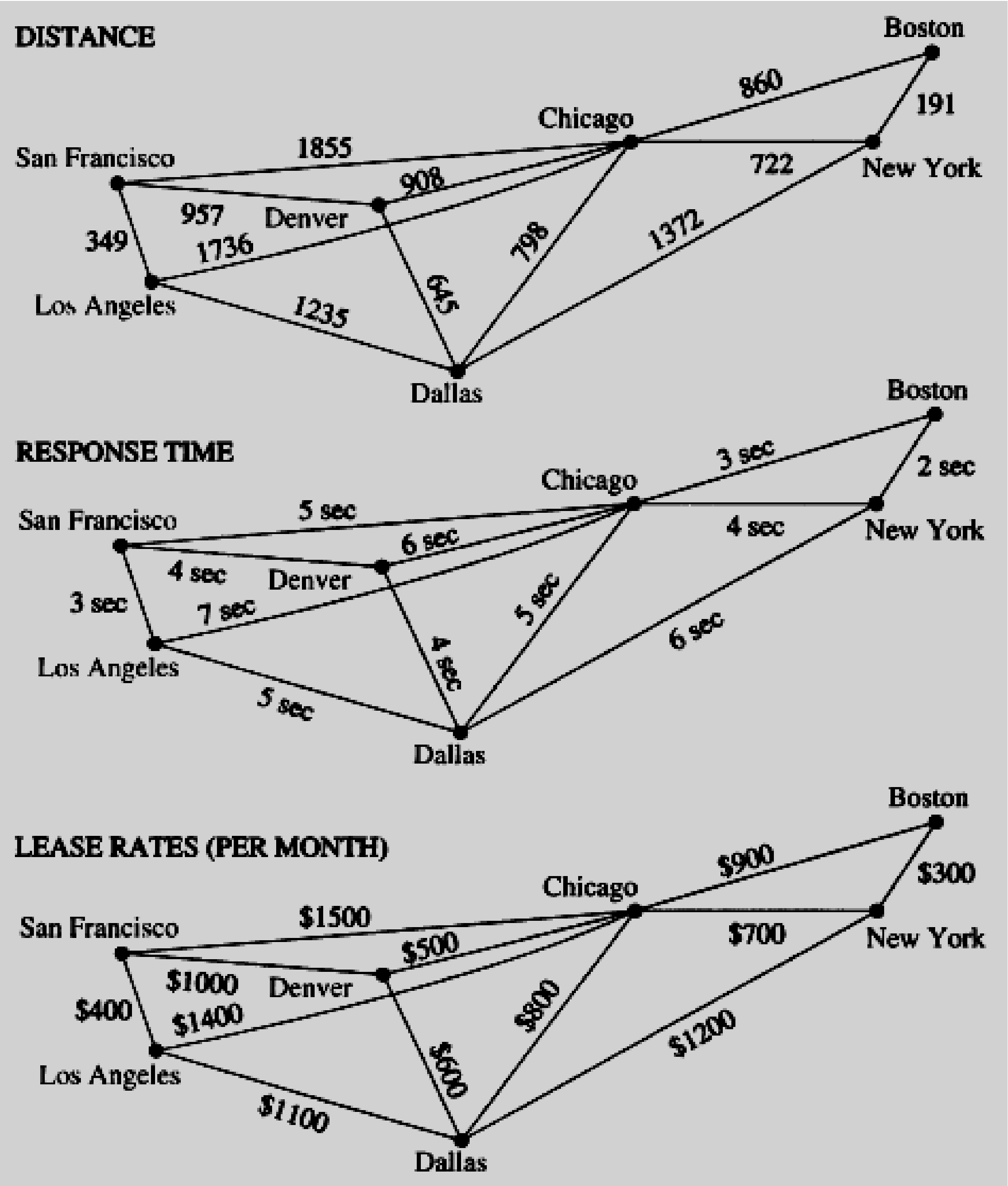


FIGURE 2 Weighted Graphs Modeling a Computer Network.

Shortest Path Problems…
Dijkstra Algorithm

ALGORITHM 1 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 weights $w(v_i, v_j)$ 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 $z \notin S$
begin
  $u :=$ 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}
end {$L(z) =$ length of a shortest path from a to z}

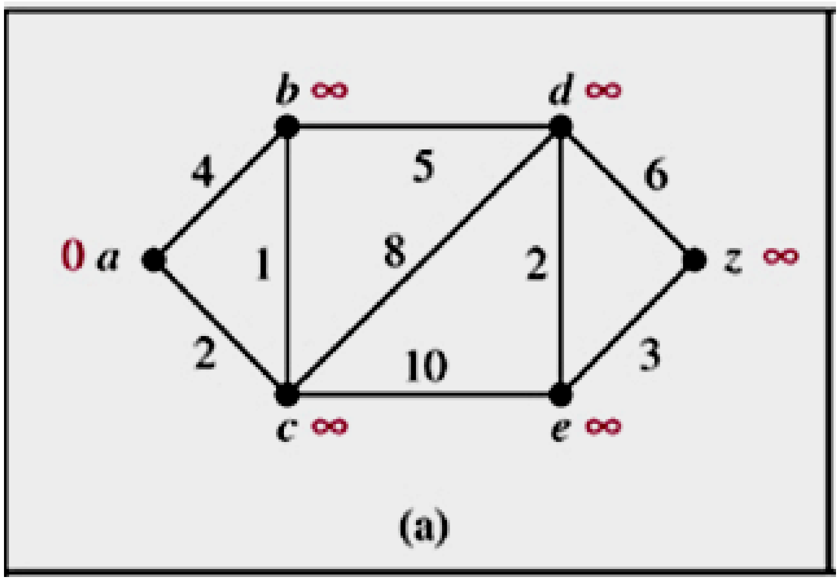
THEOREM 1

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

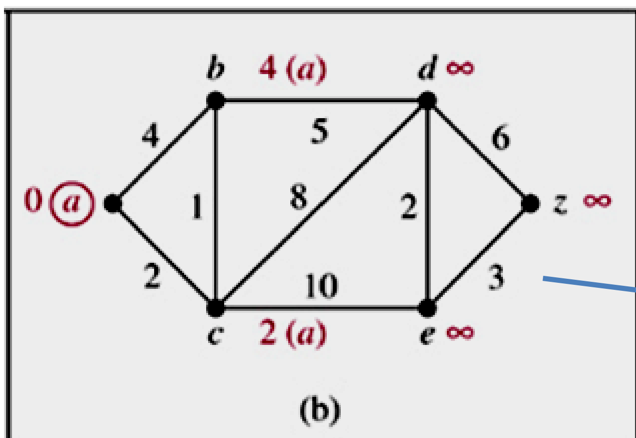
THEOREM 2

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.

Shortest Path Problems…



(a)



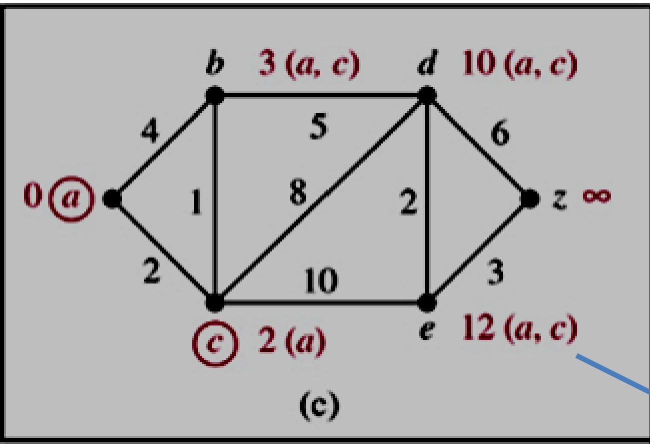
(b)

while (z ∉S)

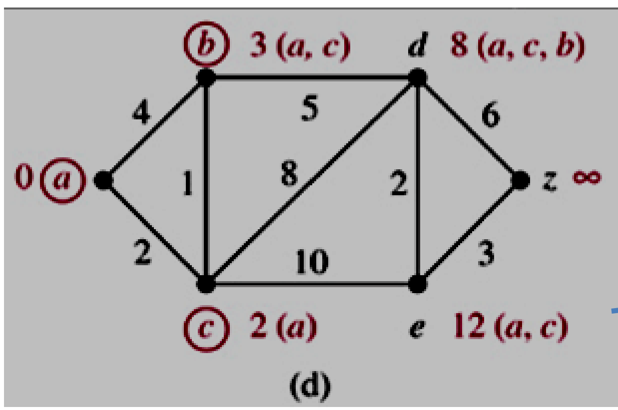
L(a)	L(b)	L(c)	L(d)	L(e)	L(z)	S
0	∞	∞	∞	∞	∞	{a}
[redacted]	∞	∞	∞	∞	∞	{a}
[redacted]	4	2	∞	∞	∞	

Examining all vertex connected to a but not in S

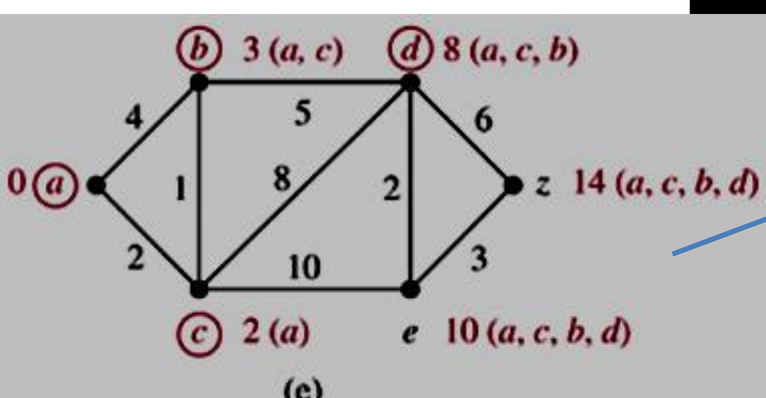
Shortest Path Problems…



(c)



(d)



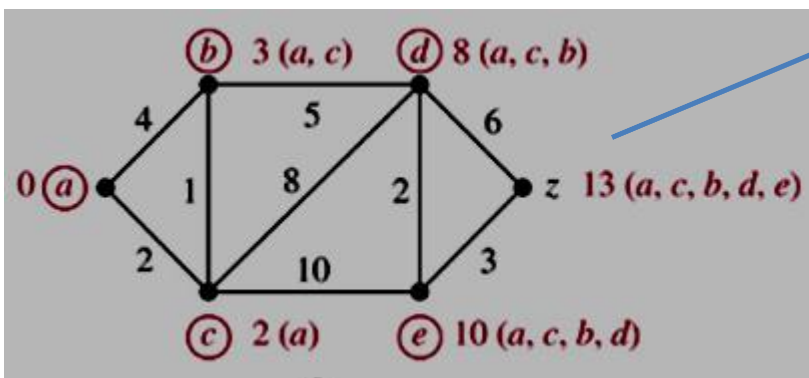
(e)

L(a)	L(b)	L(c)	L(d)	L(e)	L(z)	S
0	∞	∞	∞	∞	∞	{a}
	∞	∞	∞	∞	∞	{a}
	4	2	∞	∞	∞	{a,c}
	(cb) 2+1 = 3		(cd) 2+8 =10	(ce) 2+10=12	∞	
			10	12	∞	{a,c,b}
			(bd) 3+5 =8	12	∞	{a,c,b,d}
				(de) 8+2=10	(dz) 8+6= 14	{a,c,b,d,e}

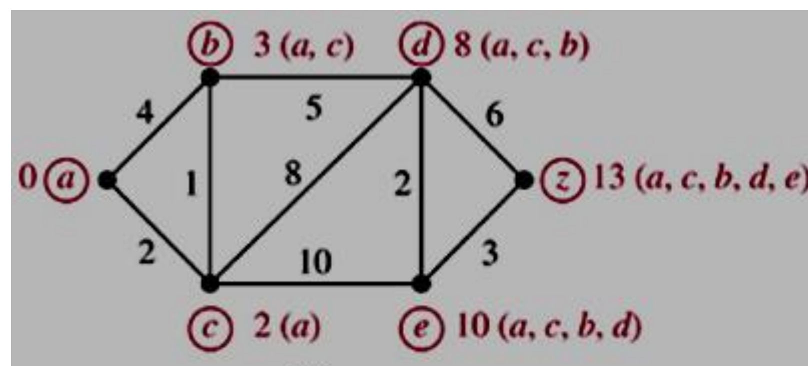
Shortest Path Problems…

L(a)	L(b)	L(c)	L(d)	L(e)	L(z)	S
[redacted]	[redacted]	[redacted]	[redacted]	(de) 8+2=**10**	(dz) 8+6= 14	{**a,c,b,d,e**}
[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	(ez) 10+3 =**13**	{**a,c,b,d,e,z**}

Stop



(f)



(g)

The Traveling Salesman Problem

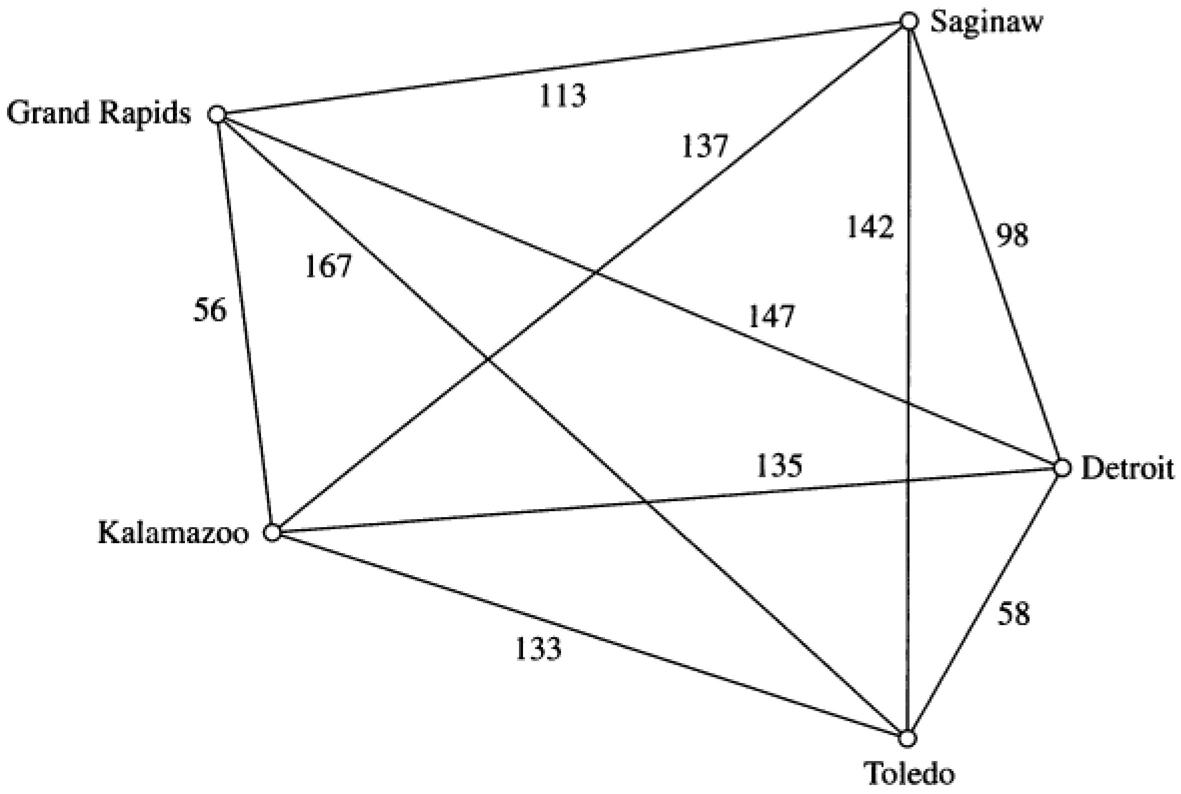


FIGURE 5 The Graph Showing the Distances between Five Cities.

Salesman starts in one city (ex. Detroit). He wants to visit n cities exactly once and return to his starting point (Detroit). In which order should he visit theses cities to travel the minimum total distance ?

The Traveling Salesman Problem

Route	Total Distance (miles)
Detroit–Toledo–Grand Rapids–Saginaw–Kalamazoo–Detroit	610
Detroit–Toledo–Grand Rapids–Kalamazoo–Saginaw–Detroit	516
Detroit–Toledo–Kalamazoo–Saginaw–Grand Rapids–Detroit	588
Detroit–Toledo–Kalamazoo–Grand Rapids–Saginaw–Detroit	458
Detroit–Toledo–Saginaw–Kalamazoo–Grand Rapids–Detroit	540
Detroit–Toledo–Saginaw–Grand Rapids–Kalamazoo–Detroit	504
Detroit–Saginaw–Toledo–Grand Rapids–Kalamazoo–Detroit	598
Detroit–Saginaw–Toledo–Kalamazoo–Grand Rapids–Detroit	576
Detroit–Saginaw–Kalamazoo–Toledo–Grand Rapids–Detroit	682
Detroit–Saginaw–Grand Rapids–Toledo–Kalamazoo–Detroit	646
Detroit–Grand Rapids–Saginaw–Toledo–Kalamazoo–Detroit	670
Detroit–Grand Rapids–Toledo–Saginaw–Kalamazoo–Detroit	728

-The problem is equivalent to asking for a Hamilton circuit with minimum total weight .

- How many way do we have to examine to solve the problem if there are n vertex in the graph? ➔ **Exhaustive search technique**

- (n-1) (n-2) (n-2) … 3.2.1 = (n-1)! ➔ Complexity

-Approximation algorithm: W <= W' <= cW (Part III—Graph Theory/Apps_Ch15.pdf)

Summary

- 9.1- Graphs and Graph Models
- 9.2- Graph Terminology and Special Types of Graphs
- 9.3- Representing Graphs and Graph Isomorphism
- 9.4- Connectivity
- 9.5- Euler and Hamilton Paths
- 9.6- Shortest Path Problems

DREAM OF INNOVATION

Thanks