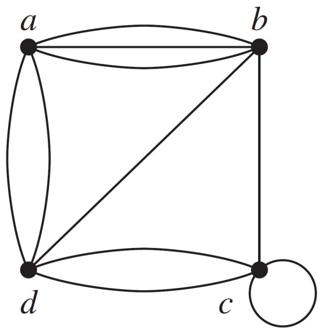
Graphs
(some material from Rosen, 7th edition)

- Maps
- Crystals
- Processor architecture
- Computer networks
- Social networks (small world)
- Collaboration networks
- Contagion networks
- Computation graphs
- …

Directed / undirected graphs

- 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.
  - *Undirected graph* (V, E): The edges are undirected. An undirected edge is associated with the unordered pair of vertices $\{u, v\}$.
  - D*irected graph* (or *digraph*) (V, E): The edges are directed. Each directed edge (or *arc*) 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.

EXAMPLE 5



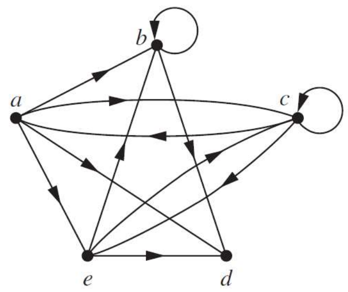


FIGURE 2 **A Directed Graph.**

Types of nodes, edges, and graphs

- Nodes:
  - Dangling node.
  - Isolated node
- Edges:
  - Parallel edges
  - Loops

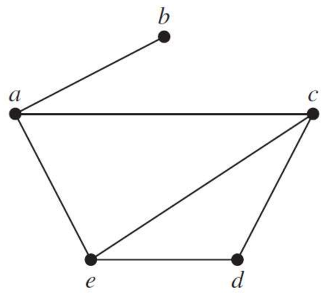


FIGURE 1 **A Simple Graph.**

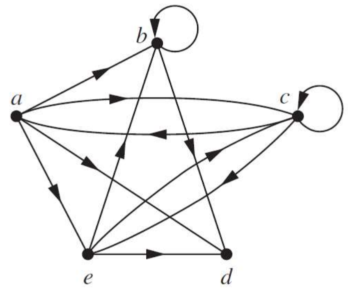
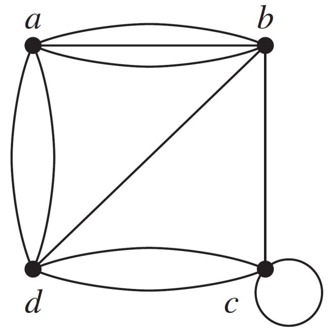


FIGURE 2 **A Directed Graph.**

EXAMPLE 5



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

Adjacency (undirected graphs)

- 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 e of G. Such an edge e is called *incident with* the vertices u and v and e is said to *connect* u and v.
- Neighborhood of a node u: Set of nodes adjacent to node
- Degree of a node (undirected graph): *deg(v)* number of edges incident on node v (loops count as 2 incidences)

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

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

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

Proof: Each Edge contributes 2 to degree count.

THEOREM 2 An undirected graph has an even number of vertices of odd degree.

Proof:

Consider the graph G = (V,E).

- Let V_1 be the set of vertices with even degree
- Let V_2 be the set of vertices with odd degree

- According to Theorem 1:

$$2m = \sum_{v \in V} \deg(v) = \sum_{v_1 \in V_1} \deg(v_1) + \sum_{v_2 \in V_2} \deg(v_2)$$

- Since $\sum_{v_1 \in V_1} \deg(v_1)$ is even, $\sum_{v_2 \in V_2} \deg(v_2)$ must be even
- For $\sum_{v_2 \in V_2} \deg(v_2)$ to be even, $|V_2|$ (the number of vertices of odd degree) has to be even.

Adjacency (directed graphs)

- 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 and terminal vertices of a loop coincide.
- Degree:
  - $\deg^-(v)$: *in-degree of vertex v*. Number of edges with v as their terminal vertex.
  - $\deg^+(v)$: *out-degree of vertex v.* Number of edges with v as their initial vertex.

THEOREM 3 Let $G = (V, E)$ be a graph with directed edges. Then

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

Proof: Each edge contributes 1 to the in-degree and 1 to the out-degree.

Special types of graphs

EXAMPLE 5 **Complete Graphs** A **complete graph on n vertices**, denoted by K_n, is a 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. A simple graph for which there is at least one pair of distinct vertex not connected by an edge is called **noncomplete**.

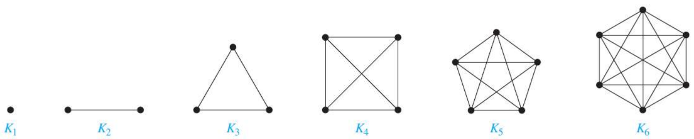


FIGURE 3 **The Graphs K_n for $1 \leq n \leq 6$.**

EXAMPLE 6 **Cycles** A **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.

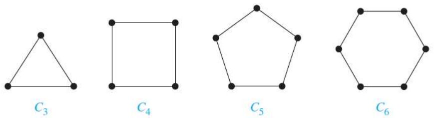


FIGURE 4 **The Cycles C_3, C_4, C_5, and C_6.**

EXAMPLE 7 **Wheels** We obtain a **wheel** W_n when we add an additional vertex to a 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.

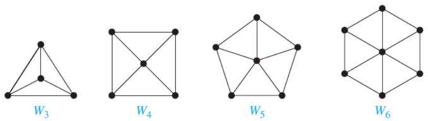


FIGURE 5 **The Wheels W_3, W_4, W_5, and W_6.**

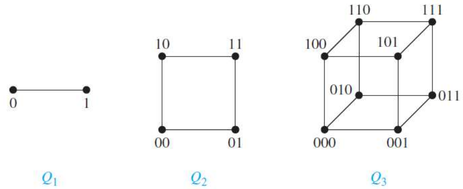


FIGURE 6 **The n-cube Q_n, $n = 1, 2, 3$.**

EXAMPLE 16 **Local Area Networks** The various computers in a building, such as minicomputers and personal computers, as well as peripheral devices such as printers and plotters, can be connected using a *local area network*. Some of these networks are based on a *star topology*, where all devices are connected to a central control device. A local area network can be represented using a complete bipartite graph $K_{1,n}$, as shown in Figure 11(a). Messages are sent from device to device through the central control device.

Links

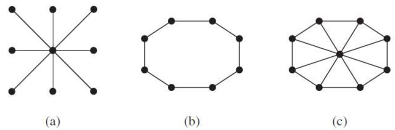


FIGURE 11 **Star, Ring, and Hybrid Topologies for Local Area Networks.**

FIGURE 12 **A Linear Array for Six Processors.**

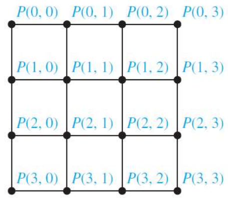
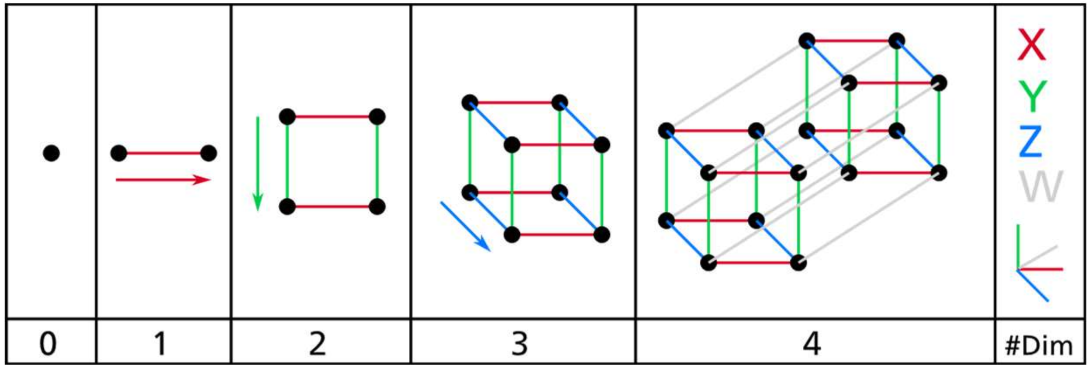
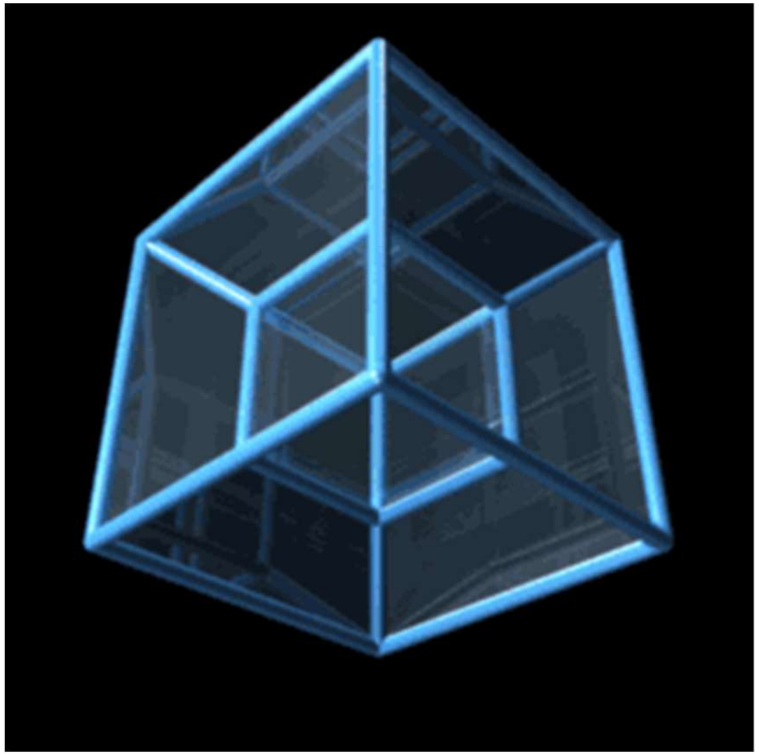


FIGURE 13 **A Mesh Network for 16 Processors.**

The tesseract: A hypercube in 4D



By NerdBoy1392 - Own work, CC BY-SA 3.0,
https://commons.wikimedia.org/w/index.php?curid=5514315



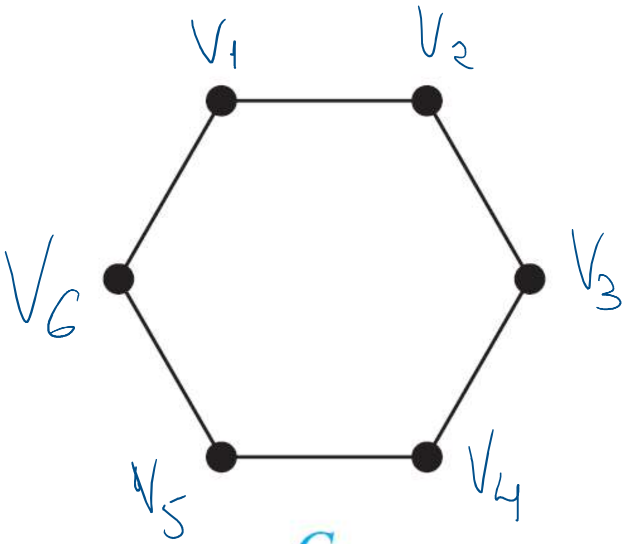
By Jason Hise –
http://www.jasonhise.com/index.php?option=com_content&view=article&id=53:4d-animations&catid=35:art&Itemid=54, CC0,
https://commons.wikimedia.org/w/index.php?curid=13772818

Exercises (EDyL 2013-2014)

- Draw, if possible, simple undirected graphs. Indicate the reason, if not possible.
  a. {1,2,3,4,5}
  b. {1,2,3,4,4}
  c. {3,4,3,4,3}
  d. {0,1,2,2,3}

Bipartite graphs

DEFINITION 6 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.



C_6

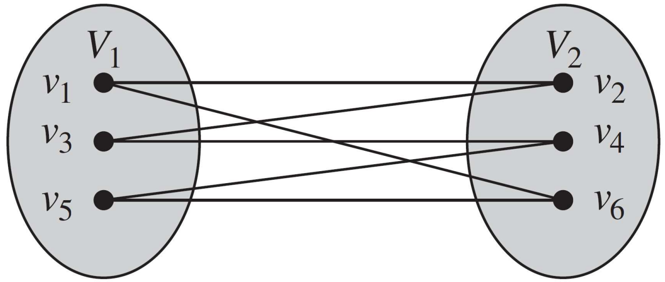


FIGURE 7 **Showing That C_6 Is Bipartite.**

Which of these graphs are bipartite?

THEOREM 4 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.

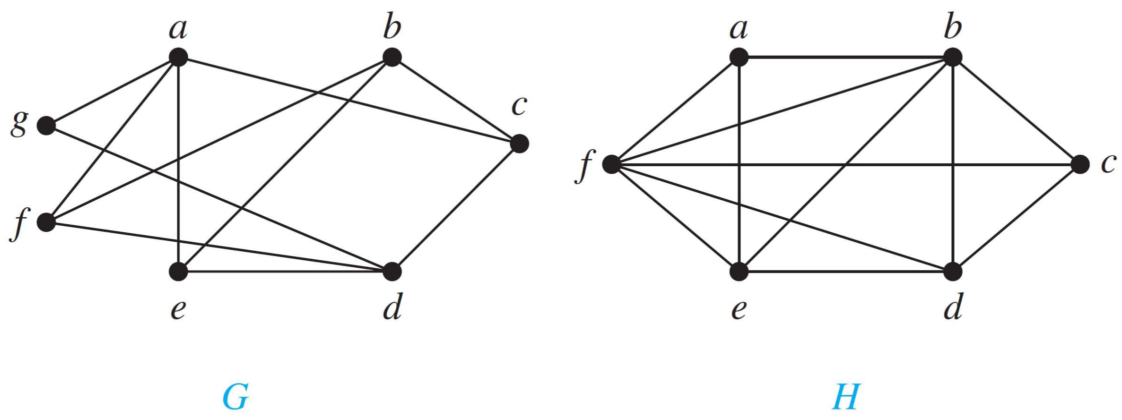


FIGURE 8 **The Undirected Graphs *G* and *H*.**

Complete bipartite graphs

EXAMPLE 13 **Complete Bipartite Graphs** A **complete bipartite graph $K_{m,n}$** is a graph that has its vertex set partitioned into two subsets of m and n vertices, respectively with an edge between two vertices if and only if one vertex is in the first subset and the other vertex is in the second subset. The complete bipartite graphs $K_{2,3}$, $K_{3,3}$, $K_{3,5}$, and $K_{2,6}$ are displayed in Figure 9. ◀

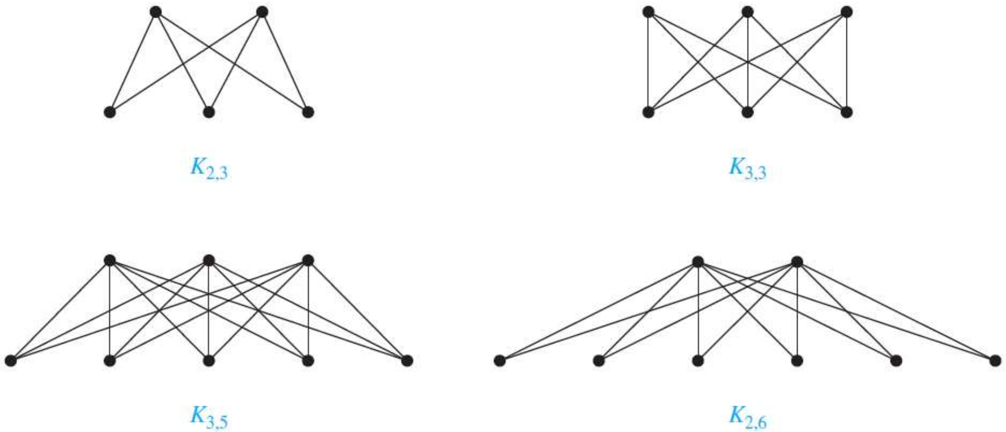
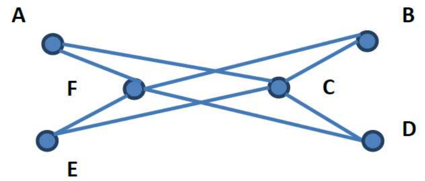


FIGURE 9 **Some Complete Bipartite Graphs.**

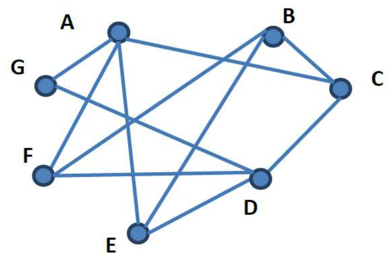
Exercises (EDyL 203-2014)

- Indicate whether the following graphs are bipartite / complete bipartite

(a)



(b)



Representing simple graphs: Adjacency list

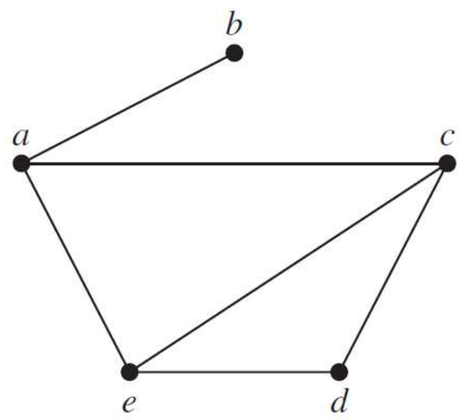


FIGURE 1 **A Simple Graph.**

Vertex	Adjacent vertices

Representing directed multigraphs: Adjacency list

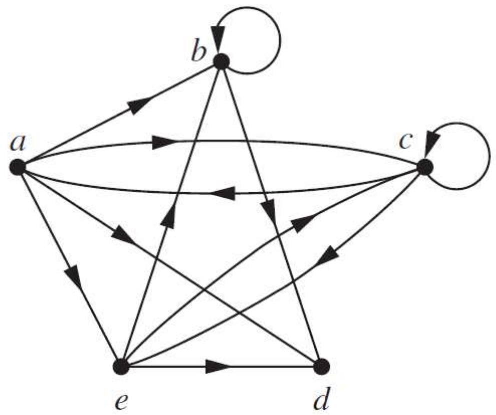


FIGURE 2 A Directed Graph.

Initial vertex	Terminal vertices

Representing simple graphs: Adjacency matrix

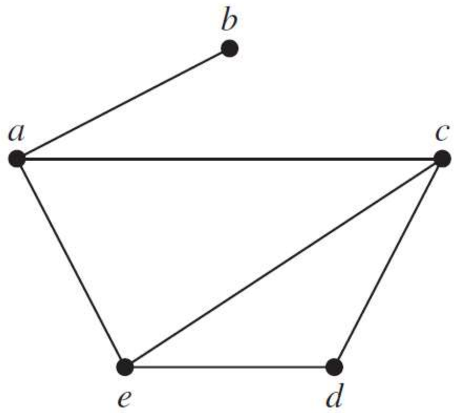
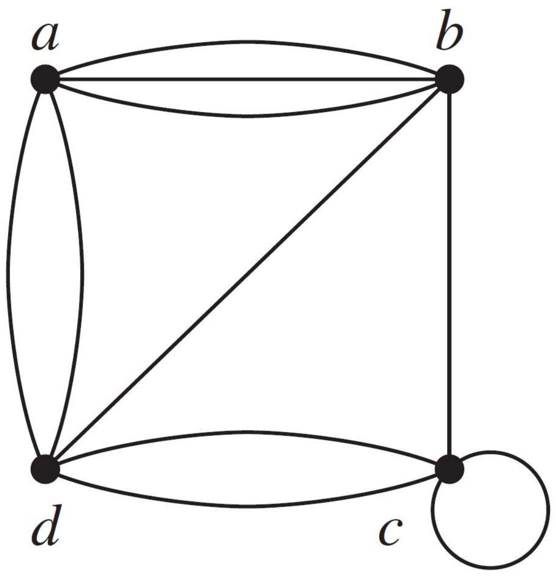


FIGURE 1 **A Simple Graph.**

	a	b	c	d	e
a					
b					
c					
d					
e					

Representing pseudographs: Adjacency matrix

EXAMPLE 5



	a	b	c	d
a				
b				
c				
d				

Representing graphs: Incidence matrix

EXAMPLE 7 Represent the pseudograph shown in Figure 7 usin

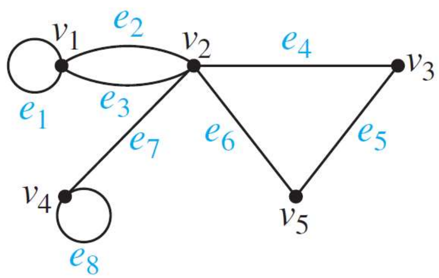


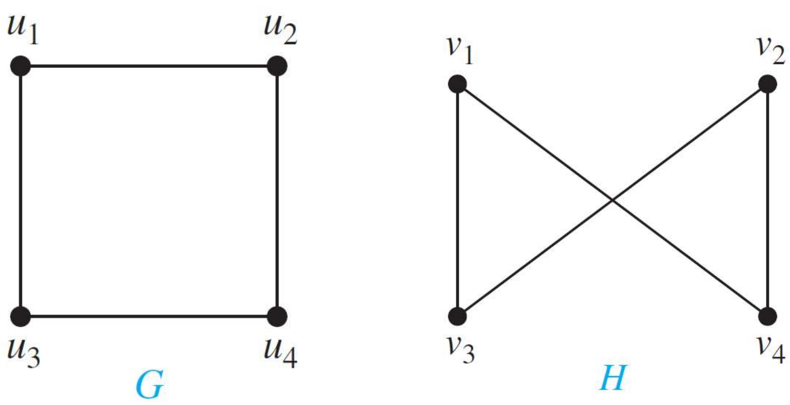
FIGURE 7
A Pseudograph.

Solution: The incidence matrix for this graph is

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

Graph isomorphism

DEFINITION 1 The simple graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are *isomorphic* if there exists a one-to-one and onto function f from V_1 to V_2 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*.* Two simple graphs that are not isomorphic are called *nonisomorphic*.



G	H
u_1	
u_2	
u_3	
u_4	

Isomorphism invariants

- Properties of a graph that are invariant under an isomorfism
  - The number of vertices
  - The number of edges
  - The number of vertices of each degree
- If any of these quantities differ in two graphs, these graphs cannot be isomorphic.
- However, when these invariants are the same, it does not necessarily mean that the two graphs are isomorphic.
- There are no useful sets of invariants currently known that can be used to determine whether simple graphs are isomorphic.

Which graphs are isomorphic?

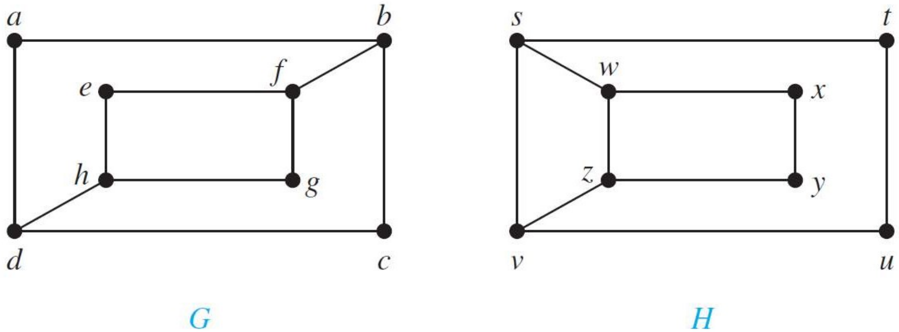


FIGURE 10 The Graphs G and H.

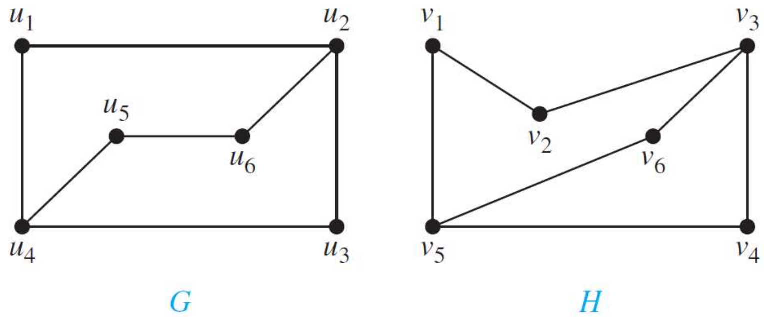


FIGURE 12 Graphs G and H.

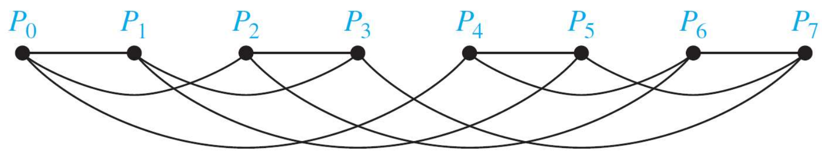
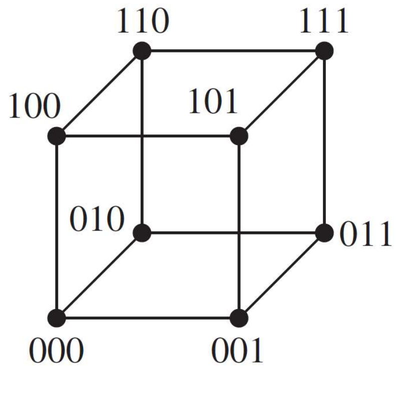
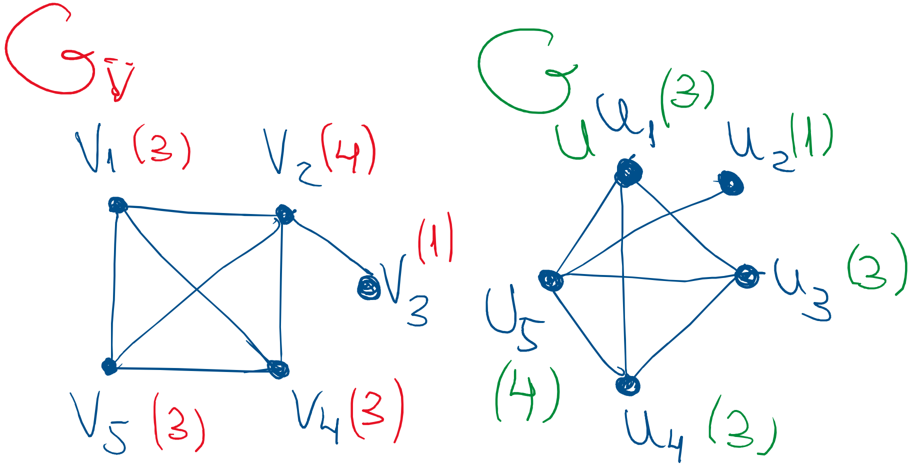


FIGURE 14 A Hypercube Network for Eight Processors.



Q_3

Isomorphism: permutation matrix



PERMUTATION MATRIX (P)

	u_1	u_2	u_3	u_4	u_5
v_1	1	0	0	0	0
v_2	0	0	0	0	1
v_3	0	1	0	0	0
v_4	0	0	0	1	0
v_5	0	0	1	0	0

TRANSPOSE OF PERMUTATION MATRIX

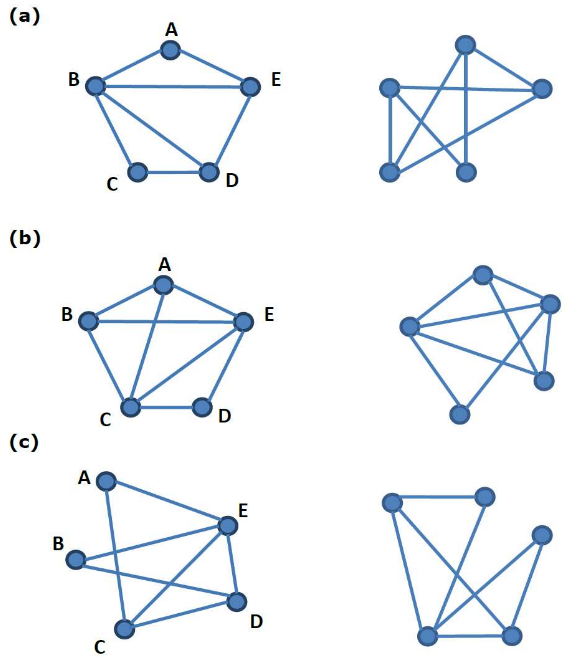
ADJACENCY MATRIX FOR G_V

$$\underline{P}^T \cdot \begin{pmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{pmatrix} \cdot \underline{P} = \begin{pmatrix} 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{pmatrix}$$

ADJACENCY MATRIX FOR G_U

Exercises (EDyL 203-2014)

- Indicate whether the following graphs are isomorphic



MapReduce: Word count example