南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

DISCRETE MATHEMATICS FOR COMPUTER SCIENCE

Dr. Qi Wang

Department of Computer Science and Engineering
Office: Room903, Nanshan iPark A7 Building
Email: wangqi@sustc.edu.cn

Properties of Relations

- **Reflexive Relation**: A relation R on a set A is called *reflexive* if $(a, a) \in R$ for every element $a \in A$.

  Irreflexive Relation: A relation R on a set A is called *irreflexive* if $(a, a) \notin R$ for every element $a \in A$.

  Symmetric Relation: A relation R on a set A is called *symmetric* if $(b, a) \in R$ whenever $(a, b) \in R$ for all $a, b \in A$.

  Antisymmetric Relation: A relation R on a set A is called *antisymmetric* if $(b, a) \in R$ and $(a, b) \in R$ implies $a = b$ for all $a, b \in A$.

  Transitive Relation: A relation R on a set A is called *reflexive* if $(a, b) \in R$ and $(b, c) \in R$ implies $(a, c) \in R$ for all $a, b, c \in A$.

Equivalence Relation

- **Definition** A relation R on a set A is called an *equivalence relation* if it is reflexive, symmetric, and transitive.

- **Definition** Let R be an equivalence relation on a set A. The set of all elements that are related to an element a of A is called the *equivalence class* of a, denoted by $[a]_R$. When only one relation is considered, we use the notation $[a]$.

$$[a]_R = \{b : (a, b) \in R\}$$

Equivalence Classes and Partitions

- **Theorem** Let R be an equivalence relation on a set A. Then union of all the equivalence classes of R is A:

$$A = \bigcup_{a \in A} [a]_R$$

Equivalence Classes and Partitions

- **Theorem** Let R be an equivalence relation on a set A. Then union of all the equivalence classes of R is A:

$$A = \bigcup_{a \in A} [a]_R$$

Theorem The equivalence classes form a partition of A.

Equivalence Classes and Partitions

- **Theorem** Let R be an equivalence relation on a set A. Then union of all the equivalence classes of R is A:

$$A = \bigcup_{a \in A} [a]_R$$

Theorem The equivalence classes form a partition of A.

Theorem Let $\{A_1, A_2, \ldots, A_i, \ldots\}$ be a partition of S. Then there is an equivalence relation R on S, that has the sets A_i as its equivalence classes.

Partial Ordering

- **Definition** A relation R on a set S is called a *partial ordering*, or *partial order*, if it is reflexive, antisymmetric, and transitive. A set S together with a partial ordering R is called a *partially ordered set*, or *poset*, denoted by (S, R). Members of S are called *elements of the poset*.

Partial Ordering

- **Definition** A relation R on a set S is called a *partial ordering*, or *partial order*, if it is reflexive, antisymmetric, and transitive. A set S together with a partial ordering R is called a *partially ordered set*, or *poset*, denoted by (S, R). Members of S are called *elements of the poset*.

- **Definition** The elements a and b of a poset $(S, \preccurlyeq)$ are *comparable* if either $a \preccurlyeq b$ or $b \preccurlyeq a$. Otherwise, a and b are called *incomparable*.

Partial Ordering

- **Definition** A relation R on a set S is called a *partial ordering*, or *partial order*, if it is reflexive, antisymmetric, and transitive. A set S together with a partial ordering R is called a *partially ordered set*, or *poset*, denoted by (S, R). Members of S are called *elements of the poset*.

- **Definition** The elements a and b of a poset $(S, \preccurlyeq)$ are *comparable* if either $a \preccurlyeq b$ or $b \preccurlyeq a$. Otherwise, a and b are called *incomparable*.

- **Definition** If $(S, \preccurlyeq)$ is a poset and every two elements of S are comparable, S is called a *totally ordered* or *linearly ordered set*, and $\preccurlyeq$ is called a *total order* or a *linear order*. A totally ordered set is also called a *chain*.

Well-Ordered Set

- **Definition** $(S, \preccurlyeq)$ is a *well-ordered set* if it is a poset such that $\preccurlyeq$ is a total ordering and every nonempty subset of S has a least element.

Well-Ordered Set

- **Definition** $(S, \preccurlyeq)$ is a *well-ordered set* if it is a poset such that $\preccurlyeq$ is a total ordering and every nonempty subset of S has a least element.

The Principle of Well-Ordering Induction Suppose that S is a well-ordered set. Then $P(x)$ is true for all $x \in S$, if

Inductive Step For every $y \in S$, if $P(x)$ is true for all $x \in S$ with $x \prec y$, then $P(y)$ is ture.

Well-Ordered Set

- **Definition** $(S, \preccurlyeq)$ is a *well-ordered set* if it is a poset such that $\preccurlyeq$ is a total ordering and every nonempty subset of S has a least element.

The Principle of Well-Ordering Induction Suppose that S is a well-ordered set. Then $P(x)$ is true for all $x \in S$, if

Inductive Step For every $y \in S$, if $P(x)$ is true for all $x \in S$ with $x \prec y$, then $P(y)$ is ture.

Proof Consider $A = \{x \in S : P(x) \text{ is false}\}$

Well-Ordered Set

- **Definition** $(S, \preccurlyeq)$ is a *well-ordered set* if it is a poset such that $\preccurlyeq$ is a total ordering and every nonempty subset of S has a least element.

The Principle of Well-Ordering Induction Suppose that S is a well-ordered set. Then $P(x)$ is true for all $x \in S$, if

Inductive Step For every $y \in S$, if $P(x)$ is true for all $x \in S$ with $x \prec y$, then $P(y)$ is ture.

Proof Consider $A = \{x \in S : P(x) \text{ is false}\}$

Question: Why don't we need a *basic step* here?

Lexicographic Ordering

- **Definition** Given two posets $(A_1, \preccurlyeq_1)$ and $(A_2, \preccurlyeq_2)$, the *lexicographic ordering* on $A_1 \times A_2$ is defined by specifying that (a_1, a_2) is less than (b_1, b_2), i.e., $(a_1, a_2) \prec (b_1, b_2)$, either if $a_1 \prec_1 b_1$ or if $a_1 = b_1$ then $a_2 \prec_2 b_2$.

Lexicographic Ordering

- **Definition** Given two posets $(A_1, \preccurlyeq_1)$ and $(A_2, \preccurlyeq_2)$, the *lexicographic ordering* on $A_1 \times A_2$ is defined by specifying that (a_1, a_2) is less than (b_1, b_2), i.e., $(a_1, a_2) \prec (b_1, b_2)$, either if $a_1 \prec_1 b_1$ or if $a_1 = b_1$ then $a_2 \prec_2 b_2$.

  Example Consider strings of lowercase English letters. A lexicographic ordering can be defined using the ordering of the letters in the alphabet. This is the same ordering as that used in dictionaries.

Lexicographic Ordering

- **Definition** Given two posets $(A_1, \preccurlyeq_1)$ and $(A_2, \preccurlyeq_2)$, the *lexicographic ordering* on $A_1 \times A_2$ is defined by specifying that (a_1, a_2) is less than (b_1, b_2), i.e., $(a_1, a_2) \prec (b_1, b_2)$, either if $a_1 \prec_1 b_1$ or if $a_1 = b_1$ then $a_2 \prec_2 b_2$.

  Example Consider strings of lowercase English letters. A lexicographic ordering can be defined using the ordering of the letters in the alphabet. This is the same ordering as that used in dictionaries.

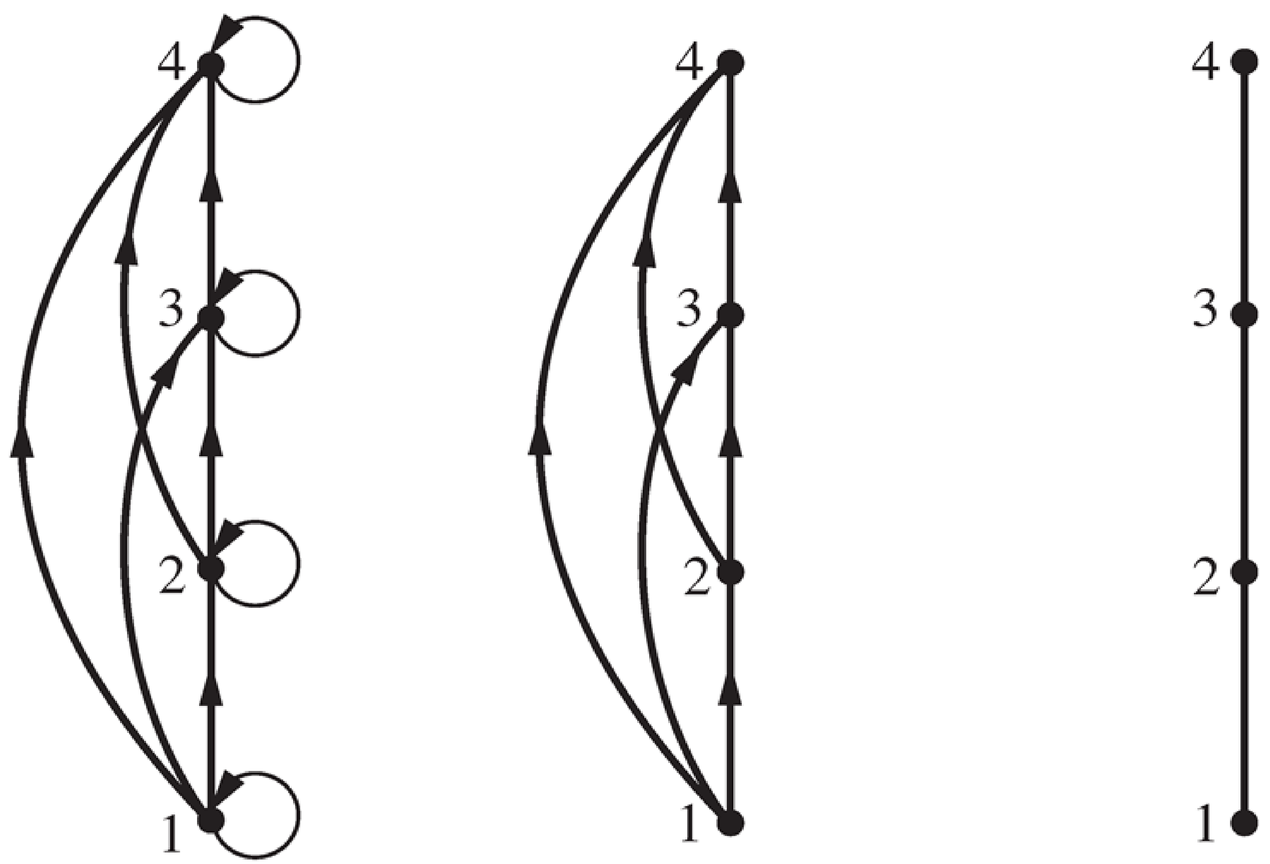
  - $\diamond$ *discreet* $\prec$ *discrete*
  - $\diamond$ *discreet* $\prec$ *discreetness*

Hasse Diagram

- A Hasse diagram is a visual representation of a partial ordering that leaves out edges that must be present because of the reflexive and transitive properties.

Hasse Diagram

- A Hasse diagram is a visual representation of a partial ordering that leaves out edges that must be present because of the reflexive and transitive properties.

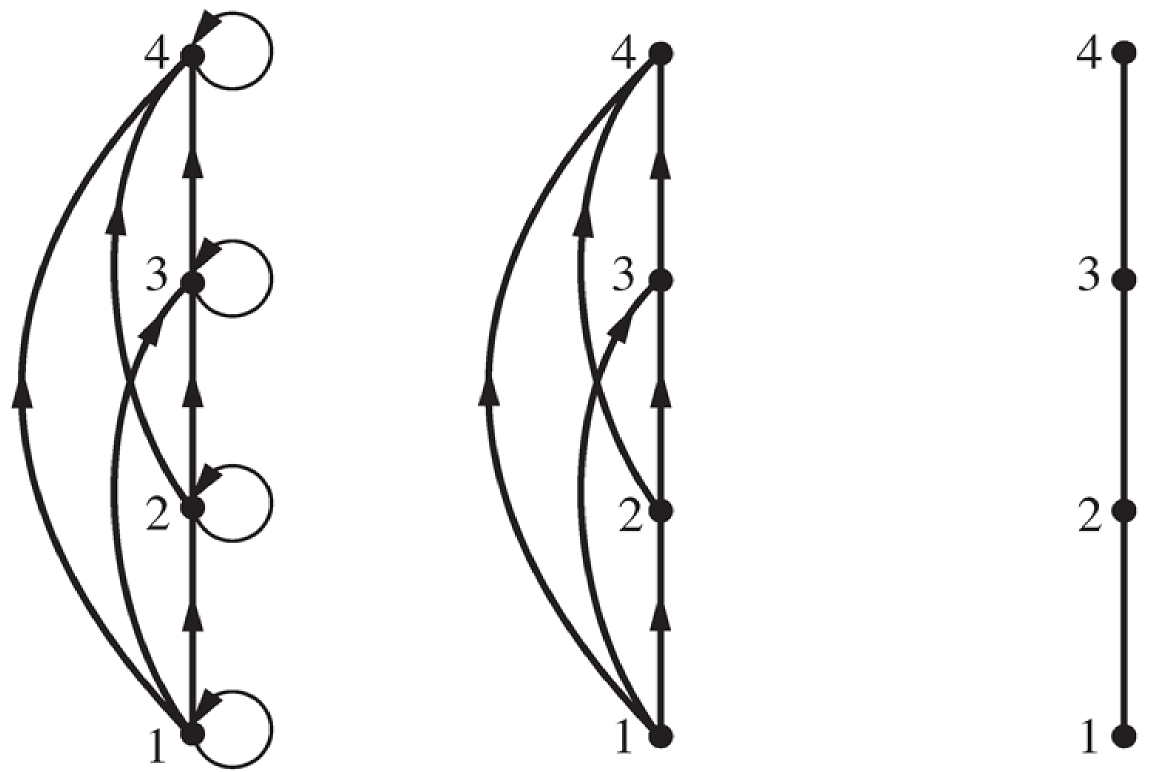


Hasse Diagram

- (a) A partial ordering. The loops are due to the reflexive property

  (b) The edges that must be present due to the transitive property are deleted

  (c) The Hasse diagram for the partial ordering (a)



Procedure for Constructing Hasse Diagram

- Start with the directed graph of the relation:

Procedure for Constructing Hasse Diagram

- Start with the directed graph of the relation:
  - ◇ Remove the loops (a, a) present at every vertex due to the reflexive property

Procedure for Constructing Hasse Diagram

- Start with the directed graph of the relation:
  - ◇ Remove the loops (a, a) present at every vertex due to the reflexive property
  - ◇ Remove all edges (x, y) for which there is an element $z \in S$ s.t. $x \prec z$ and $z \prec y$. These are the edges that must be present due to the transitive property

Procedure for Constructing Hasse Diagram

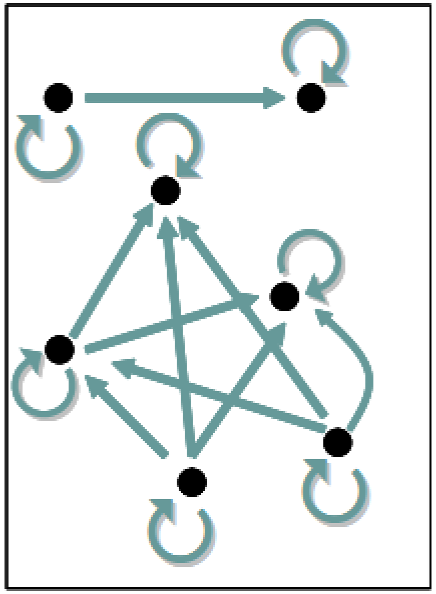
- Start with the directed graph of the relation:

  ◇ Remove the loops (a, a) present at every vertex due to the reflexive property

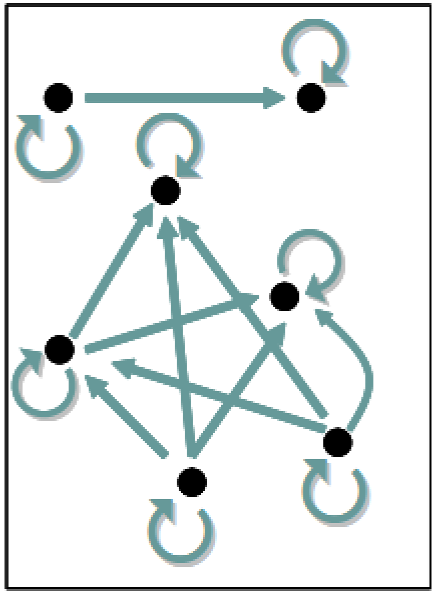
  ◇ Remove all edges (x, y) for which there is an element $z \in S$ s.t. $x \prec z$ and $z \prec y$. These are the edges that must be present due to the transitive property

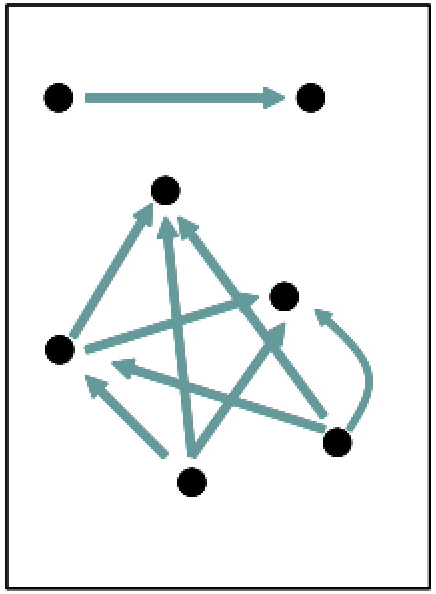
  ◇ Arrange each edge so that its initial vertex is below the terminal vertex. Remove all the arrows, because all edges point upwards toward their terminal vertex.

Hasse Diagram Example

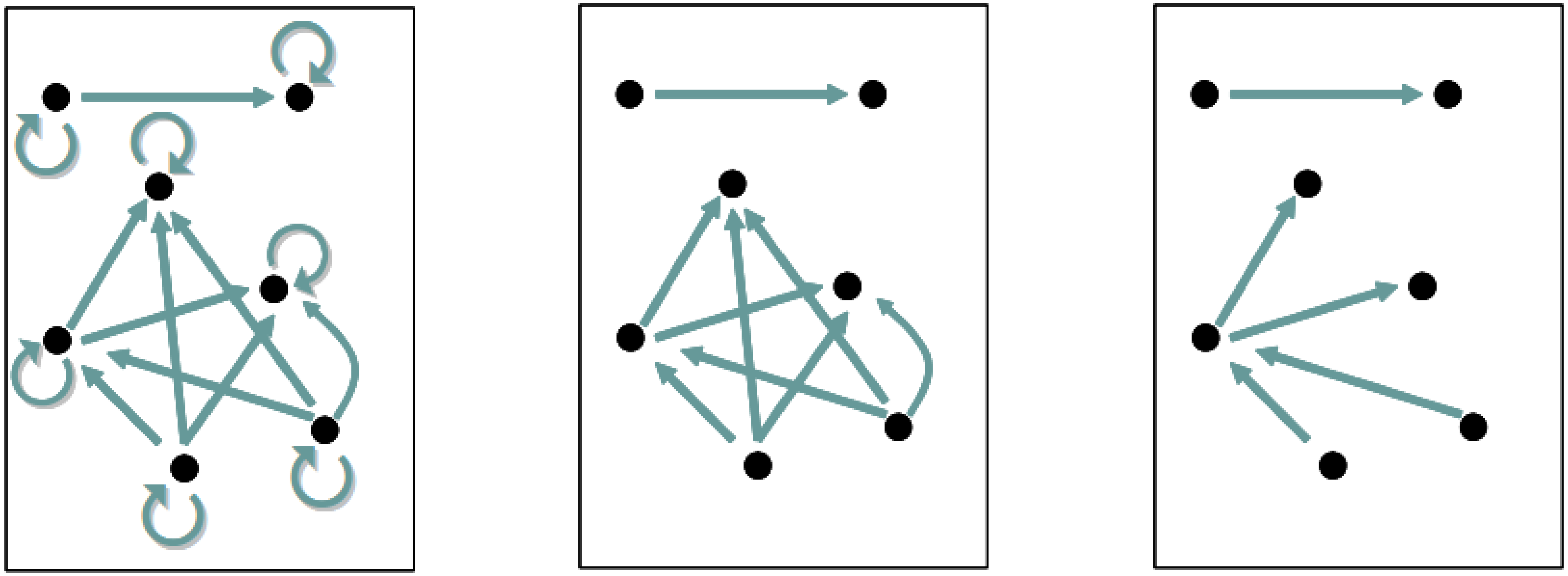


Hasse Diagram Example





Hasse Diagram Example



Maximal and Minimal Elements

- **Definition** a is a *maximal* (resp. *minimal*) element in poset $(S, \preccurlyeq)$ if there is no $b \in S$ such that $a \prec b$ (resp. $b \prec a$).

Maximal and Minimal Elements

- **Definition** a is a *maximal* (resp. *minimal*) element in poset $(S, \preccurlyeq)$ if there is no $b \in S$ such that $a \prec b$ (resp. $b \prec a$).

  Example Which elements of the poset $(\{2, 4, 5, 10, 12, 20, 25\}, |)$ are maximal, and minimal?

Maximal and Minimal Elements

- **Definition** a is a *maximal* (resp. *minimal*) element in poset $(S, \preccurlyeq)$ if there is no $b \in S$ such that $a \prec b$ (resp. $b \prec a$).

  Example Which elements of the poset $(\{2, 4, 5, 10, 12, 20, 25\}, |)$ are maximal, and minimal?

  Definition a is the *greatest* (resp. *least*) element of the poset $(S, \preccurlyeq)$ if $b \preccurlyeq a$ (resp. $a \preccurlyeq b$) for all $b \in S$.

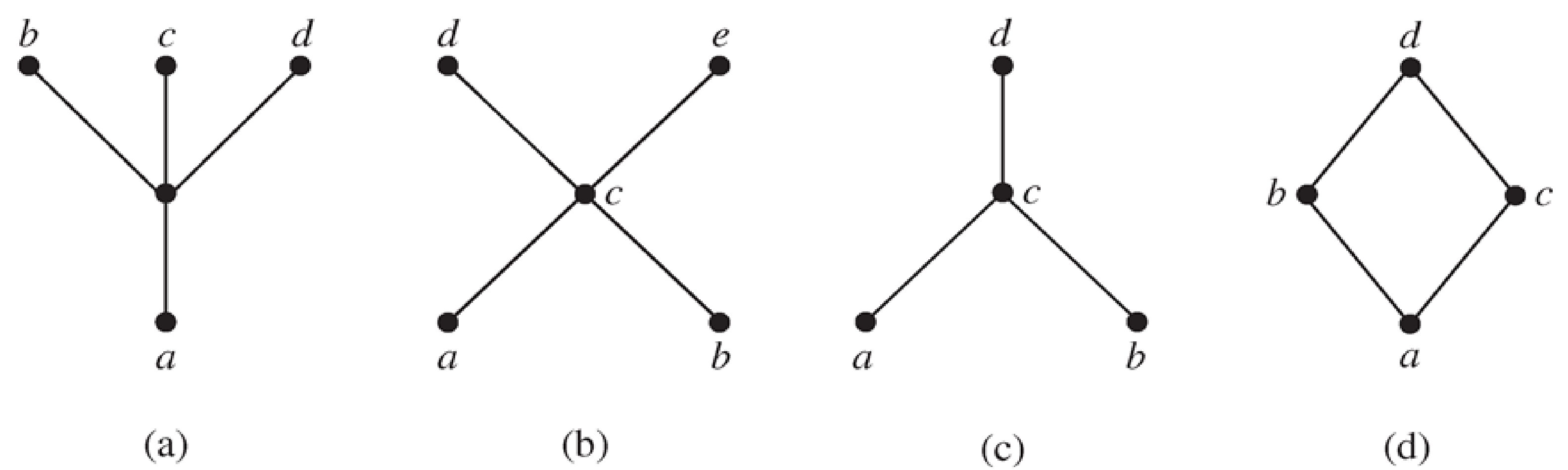
Maximal and Minimal Elements

- **Definition** a is a *maximal* (resp. *minimal*) element in poset $(S, \preccurlyeq)$ if there is no $b \in S$ such that $a \prec b$ (resp. $b \prec a$).

  Example Which elements of the poset $(\{2, 4, 5, 10, 12, 20, 25\}, |)$ are maximal, and minimal?

  Definition a is the *greatest* (resp. *least*) element of the poset $(S, \preccurlyeq)$ if $b \preccurlyeq a$ (resp. $a \preccurlyeq b$) for all $b \in S$.

  Example



Maximal and Minimal Elements

- **Definition** Let A be a subset of a poset $(S, \preccurlyeq)$.
  - $u \in S$ is called an *upper bound* (resp. *lower bound*) of A if $a \preccurlyeq u$ (resp. $u \preccurlyeq a$) for all $a \in A$.
  - $x \in S$ is called the *least upper bound* (resp. *greatest lower bound*) of A if x is an upper bound (resp. lower bound) that is less than (resp. greater than) any other upper bound (resp. lower bound) of A.

Maximal and Minimal Elements

- **Definition** Let A be a subset of a poset $(S, \preccurlyeq)$.
  - $u \in S$ is called an *upper bound* (resp. *lower bound*) of A if $a \preccurlyeq u$ (resp. $u \preccurlyeq a$) for all $a \in A$.
  - $x \in S$ is called the *least upper bound* (resp. *greatest lower bound*) of A if x is an upper bound (resp. lower bound) that is less than (resp. greater than) any other upper bound (resp. lower bound) of A.

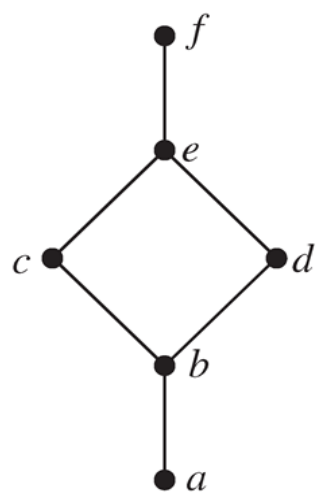
Example Find the greatest lower bound and the least upper bound of the sets $\{3, 9, 12\}$ and $\{1, 2, 4, 5, 10\}$, if they exist, in the poset $(\mathbf{Z}^+, |)$.

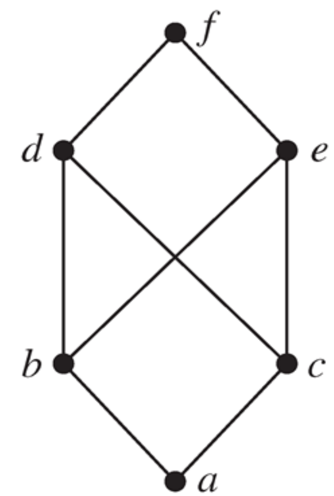
Lattices

- **Definition** A partial ordered set in which every pair of elements has both a least upper bound and a greatest lower bound is called a *lattice*.

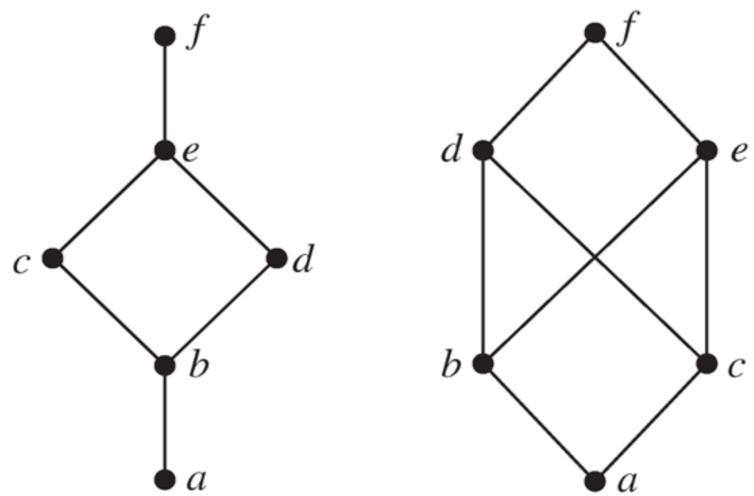
Lattices

- **Definition** A partial ordered set in which every pair of elements has both a least upper bound and a greatest lower bound is called a *lattice*.





- **Definition** A partial ordered set in which every pair of elements has both a least upper bound and a greatest lower bound is called a *lattice*.



Example Determine whether the posets $(\{1,2,3,4,5\},|)$ and $(\{1,2,4,8,16\},|)$ are lattices.

Topological Sorting

- Motivation: A project is made up of 20 different tasks. Some tasks can be completed only after others have been finished. How can an order be found for these tasks?

Topological Sorting

- Motivation: A project is made up of 20 different tasks. Some tasks can be completed only after others have been finished. How can an order be found for these tasks?

Topological sorting: Given a partial ordering R, find a total ordering $\preccurlyeq$ such that $a \preccurlyeq b$ whenever $a \mathrel{R} b$. $\preccurlyeq$ is said *compatible with* R.

Topological Sorting for Finite Posets

procedure topological_sort (S: finite poset)

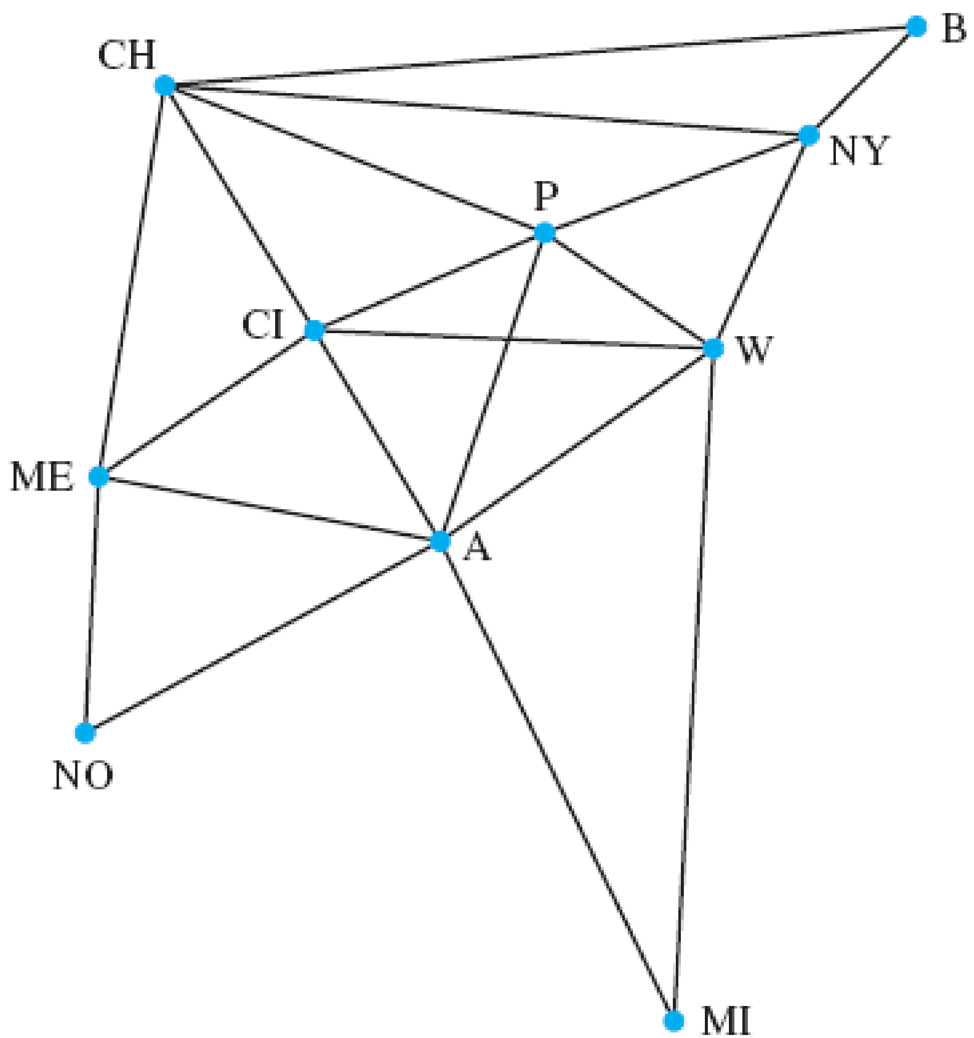
$k := 1$;
while $S \neq \emptyset$
  $a_k :=$ a minimal element of S
  $S := S \setminus \{a_k\}$
  $k := k + 1$
end while
// $\{a_1, a_2, \ldots, a_n\}$ is a compatible total ordering of S

Example

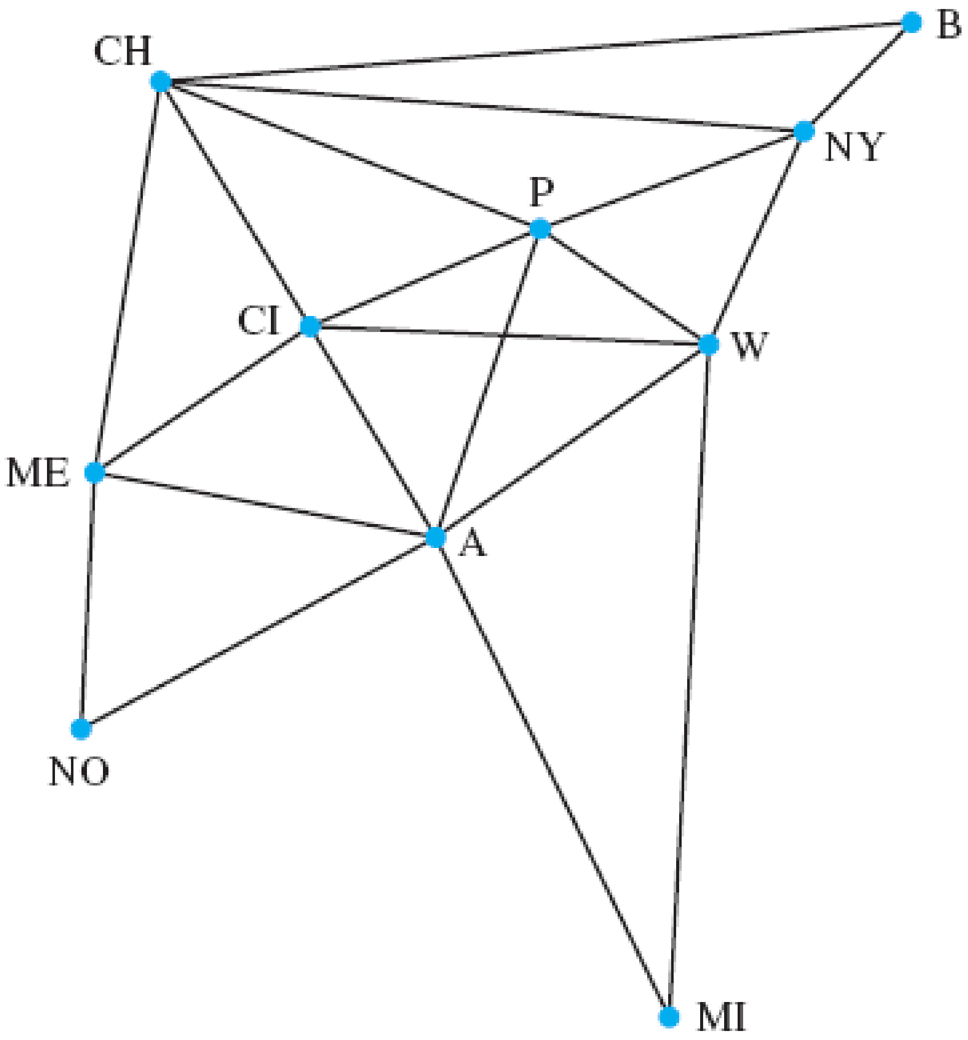
- Map of some cities in Eastern U.S. with communication lines existing between certain pairs of these cities.

Example

- Map of some cities in Eastern U.S. with communication lines existing between certain pairs of these cities.

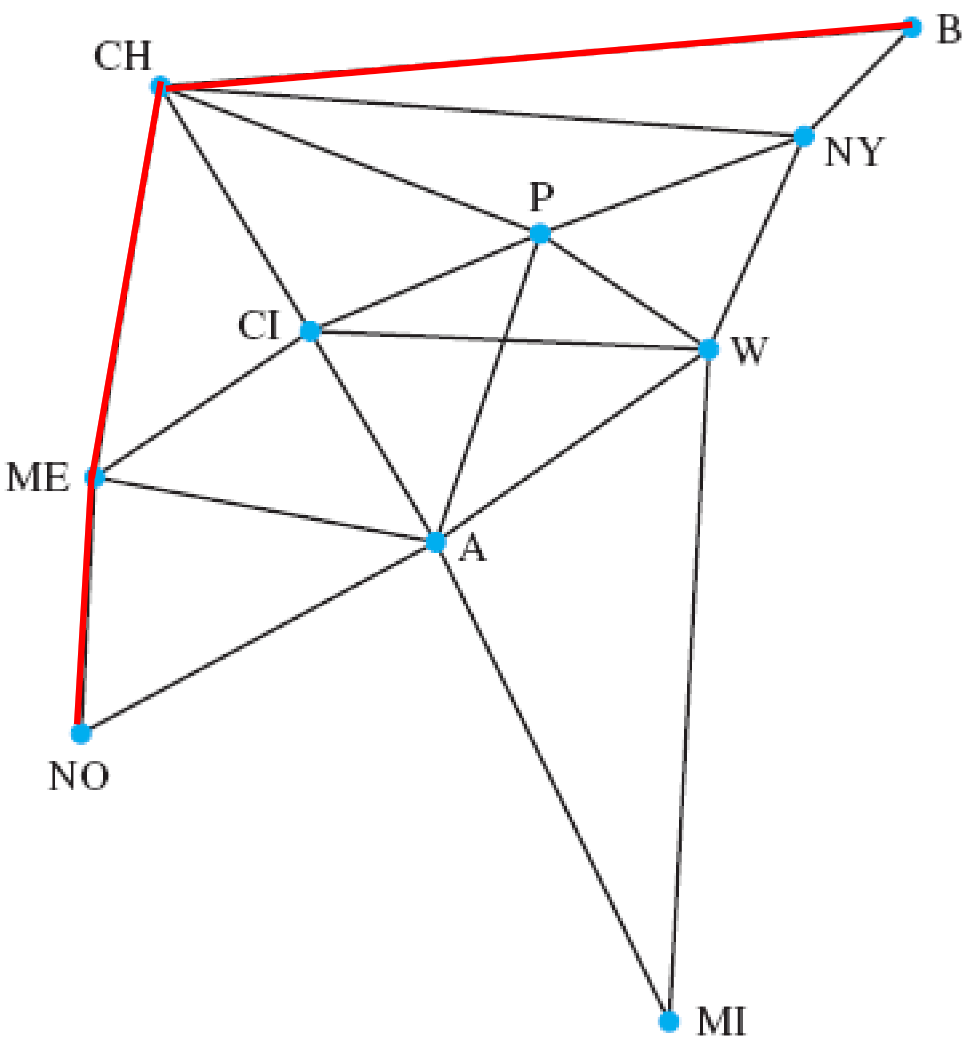


Example



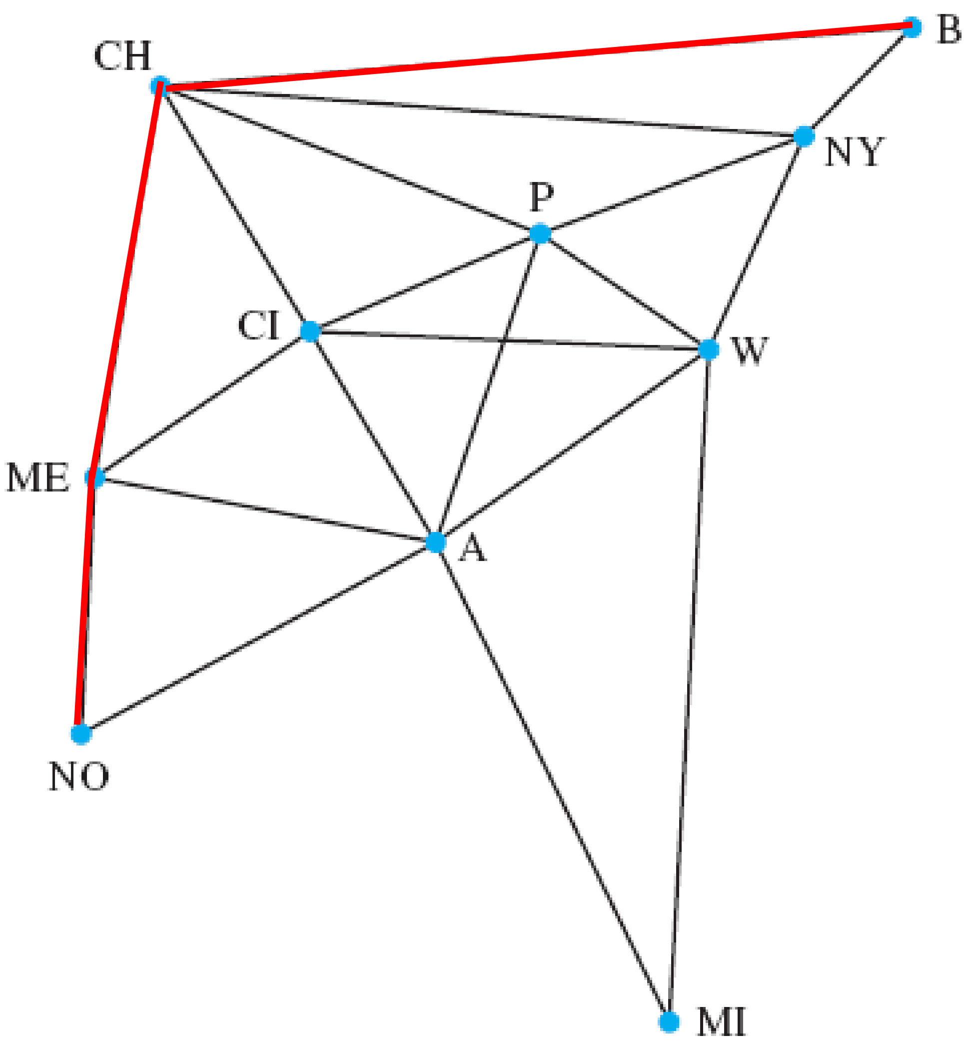
What is the minimum number of links to send a message from B to NO?

Example



What is the minimum number of links to send a message from B to NO?

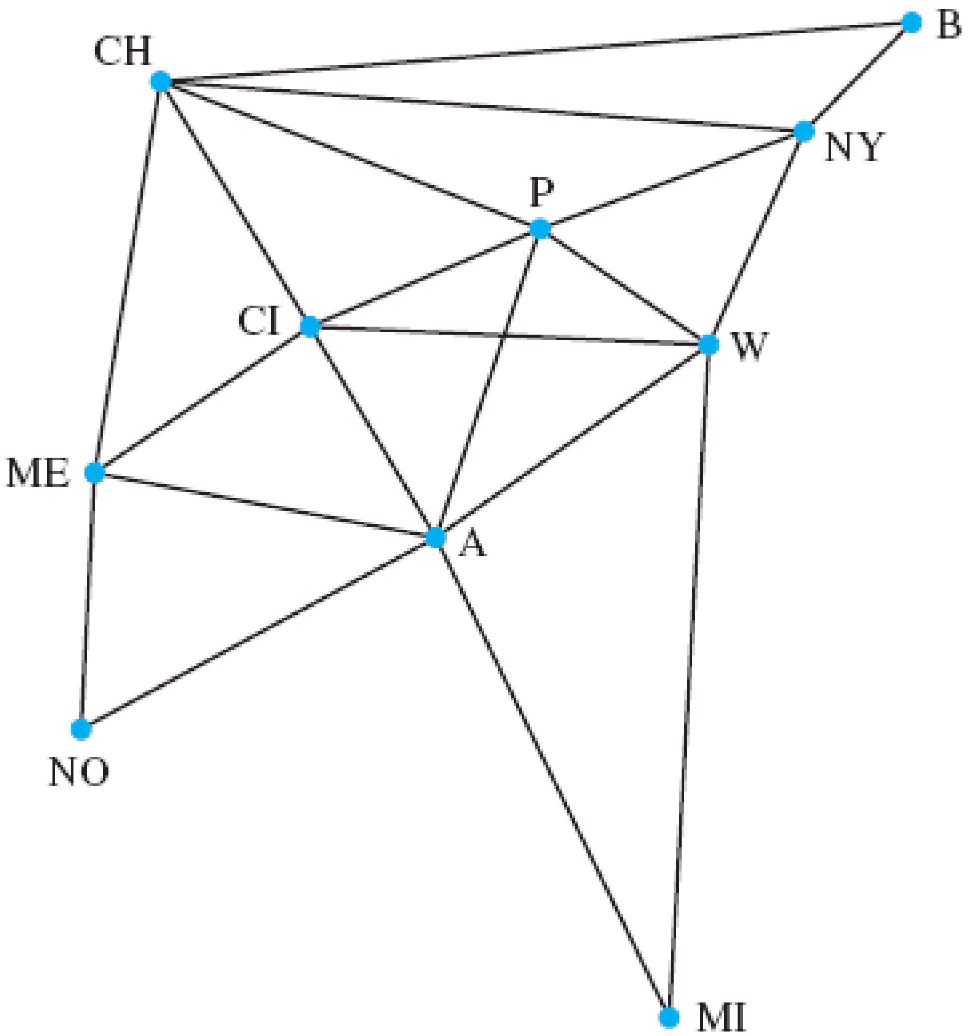
Example



What is the minimum number of links to send a message from B to NO?

3: B - CH - ME - NO

Example

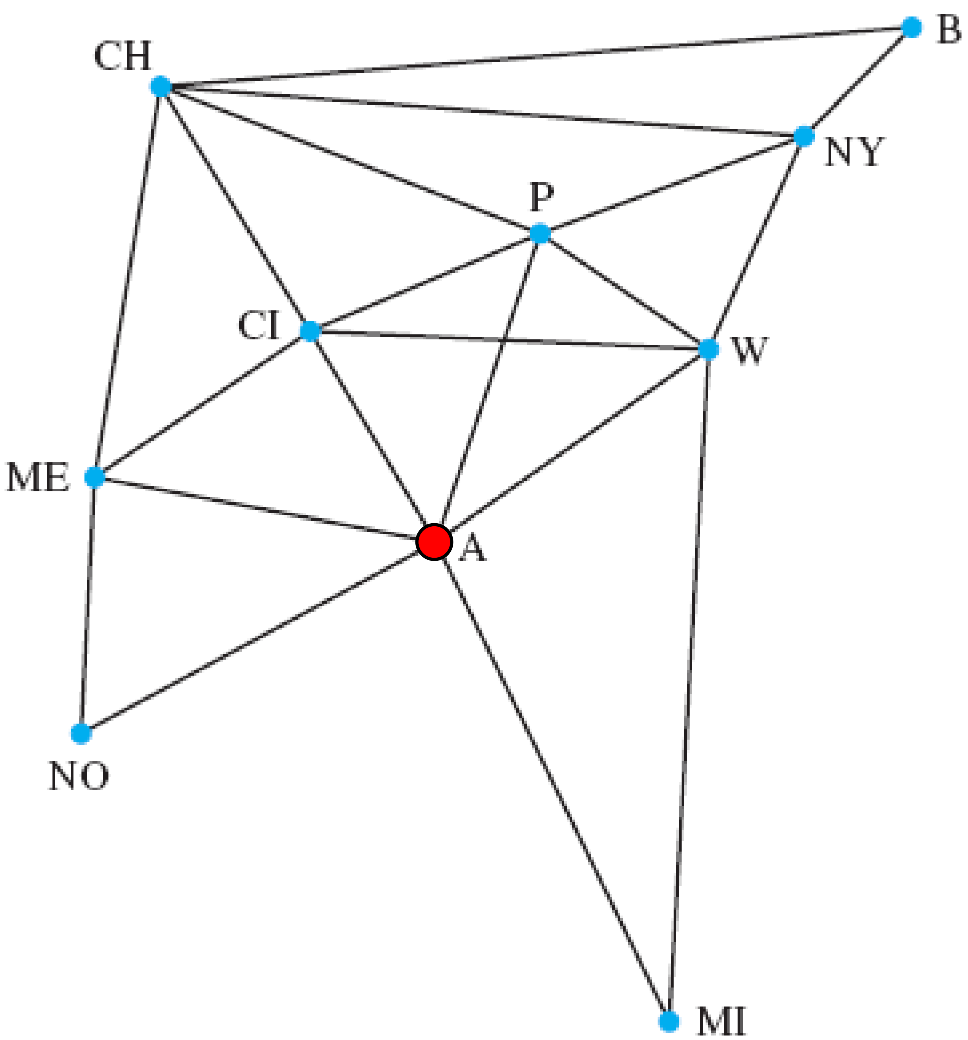


What is the minimum number of links to send a message from B to NO?

3: B - CH - ME - NO

Which city/cities has/have the most communication links emanating from it/them?

Example

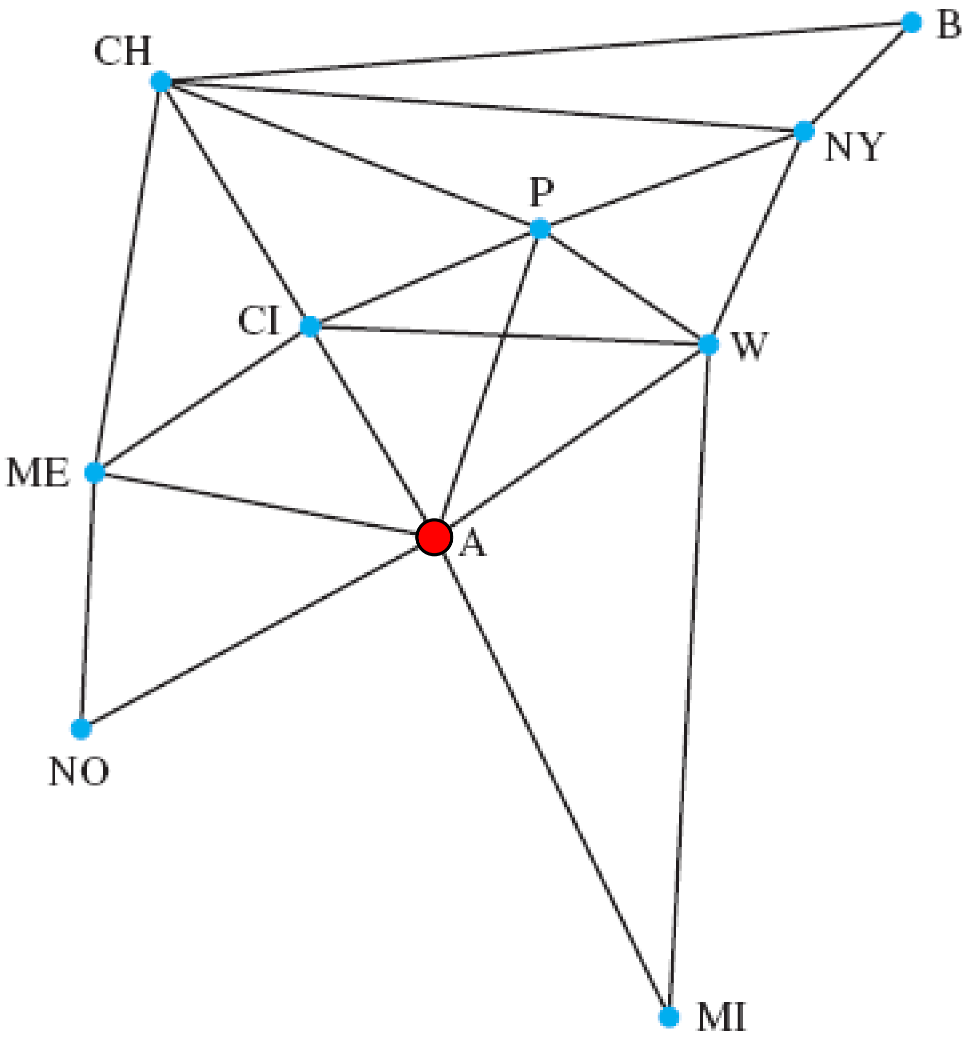


What is the minimum number of links to send a message from B to NO?

3: B - CH - ME - NO

Which city/cities has/have the most communication links emanating from it/them?

Example



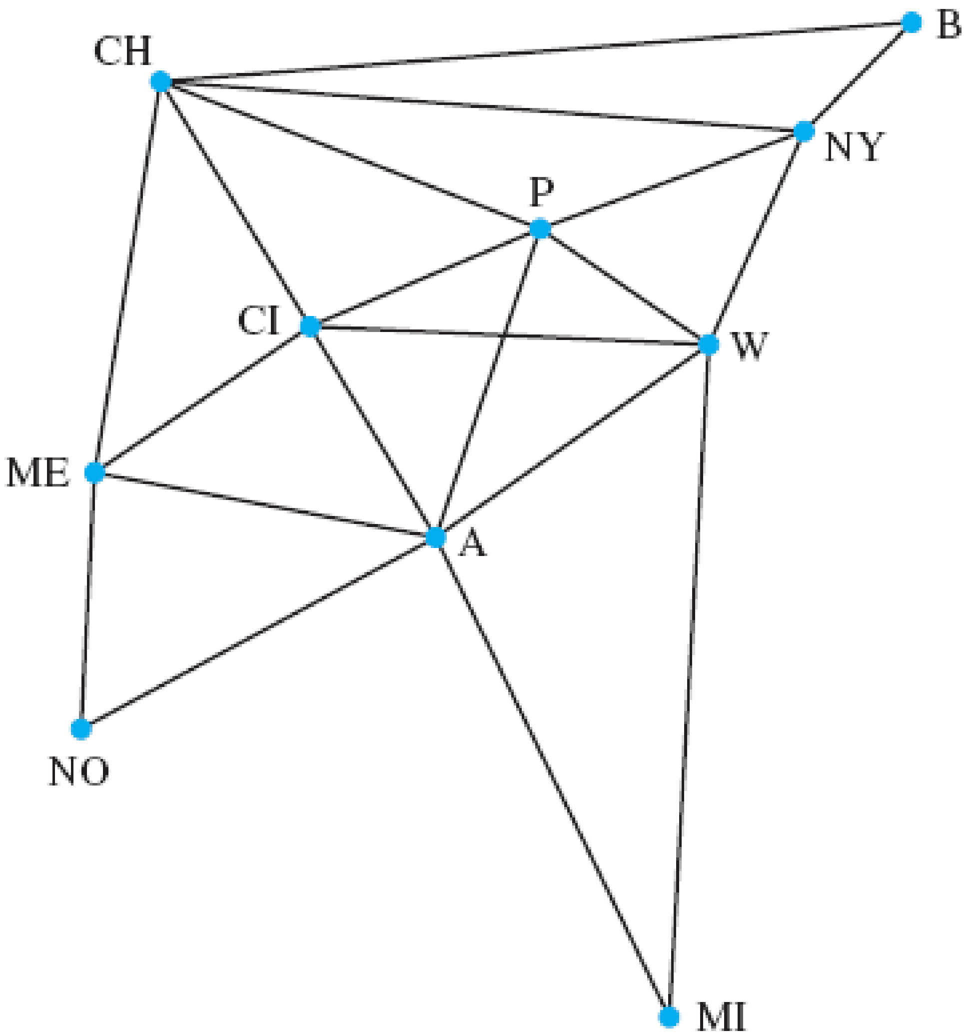
What is the minimum number of links to send a message from B to NO?

3: B - CH - ME - NO

Which city/cities has/have the most communication links emanating from it/them?

A: 6 links

Example



What is the minimum number of links to send a message from B to NO?

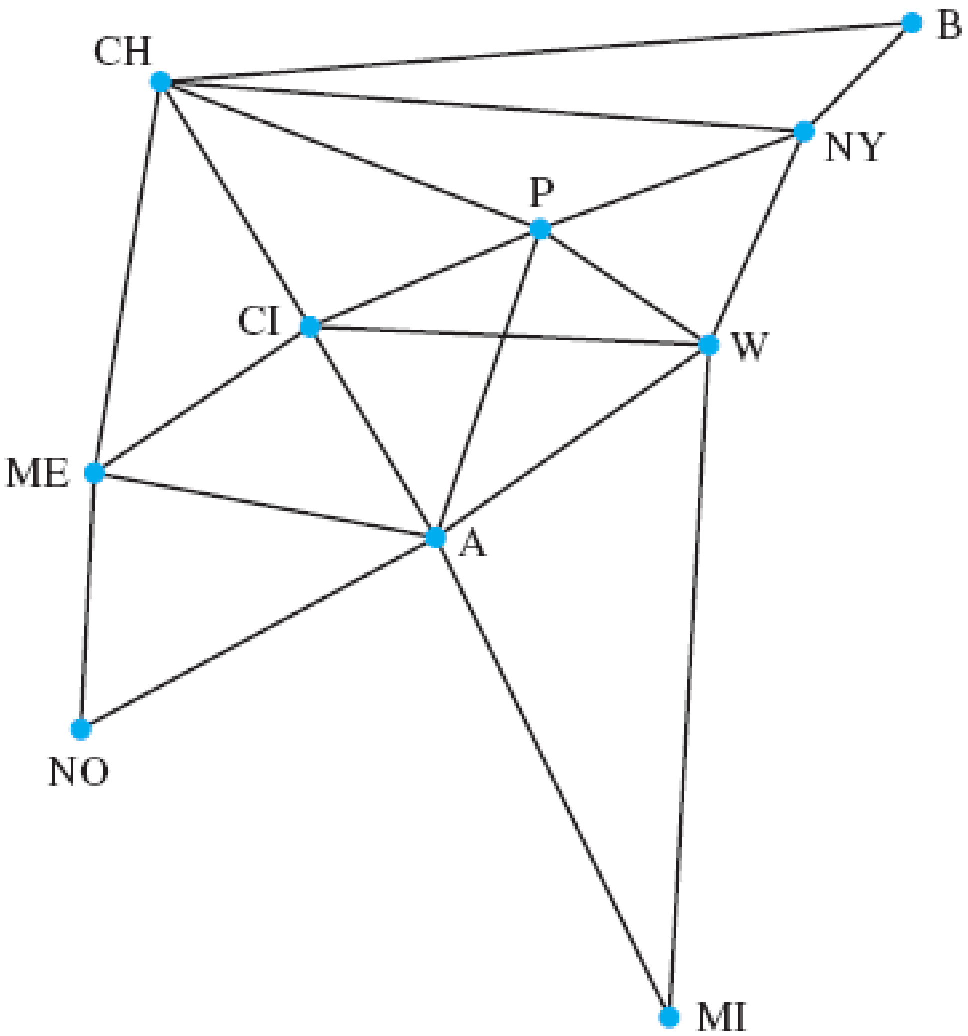
3: B - CH - ME - NO

Which city/cities has/have the most communication links emanating from it/them?

A: 6 links

What is the total number of communication links?

Example



What is the minimum number of links to send a message from B to NO?

3: B - CH - ME - NO

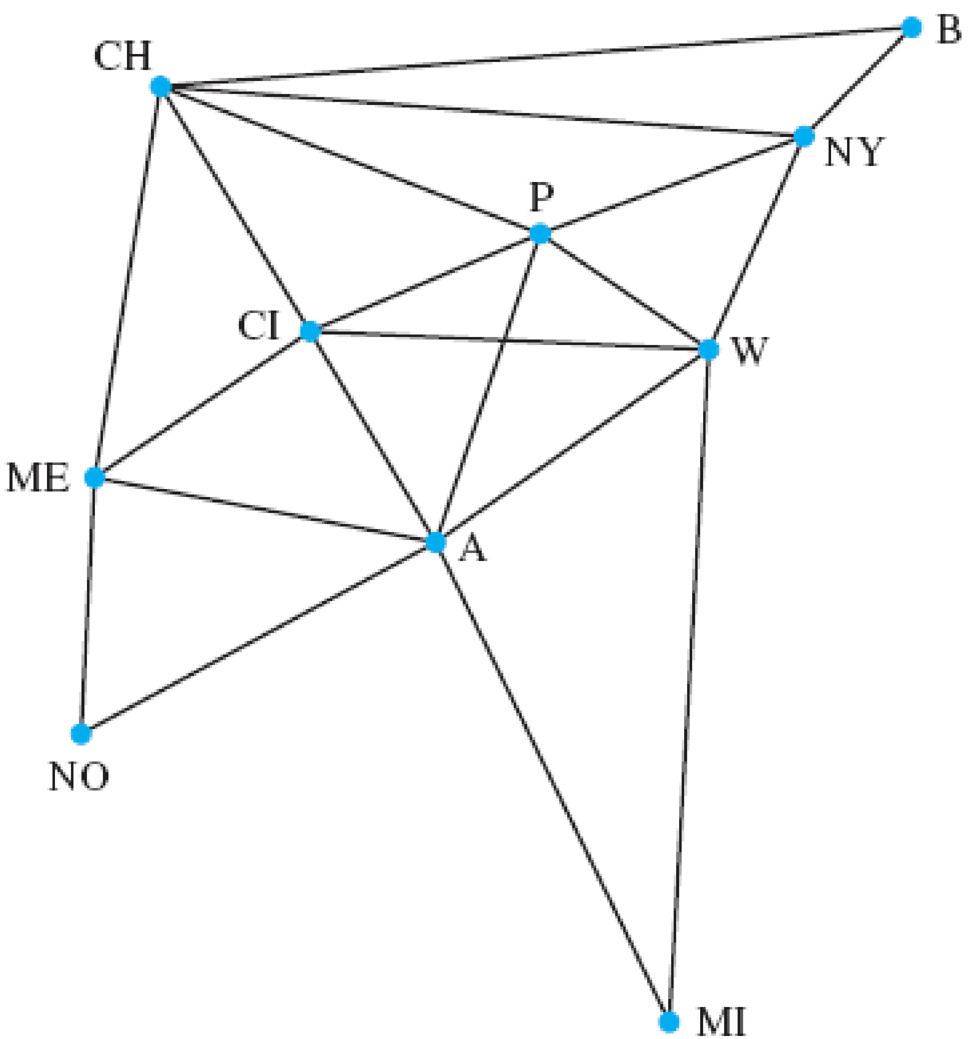
Which city/cities has/have the most communication links emanating from it/them?

A: 6 links

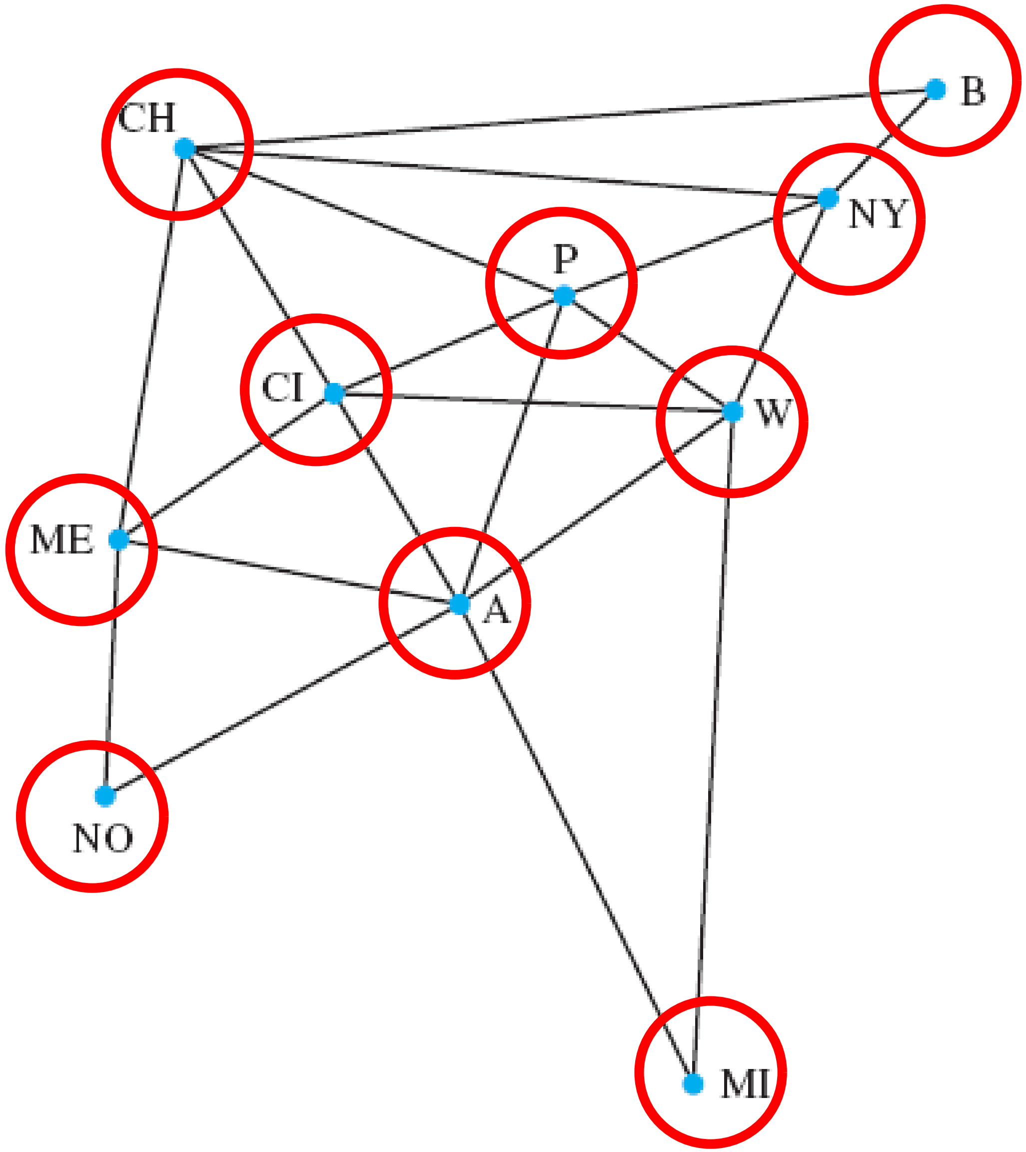
What is the total number of communication links?

20 links

Graph G

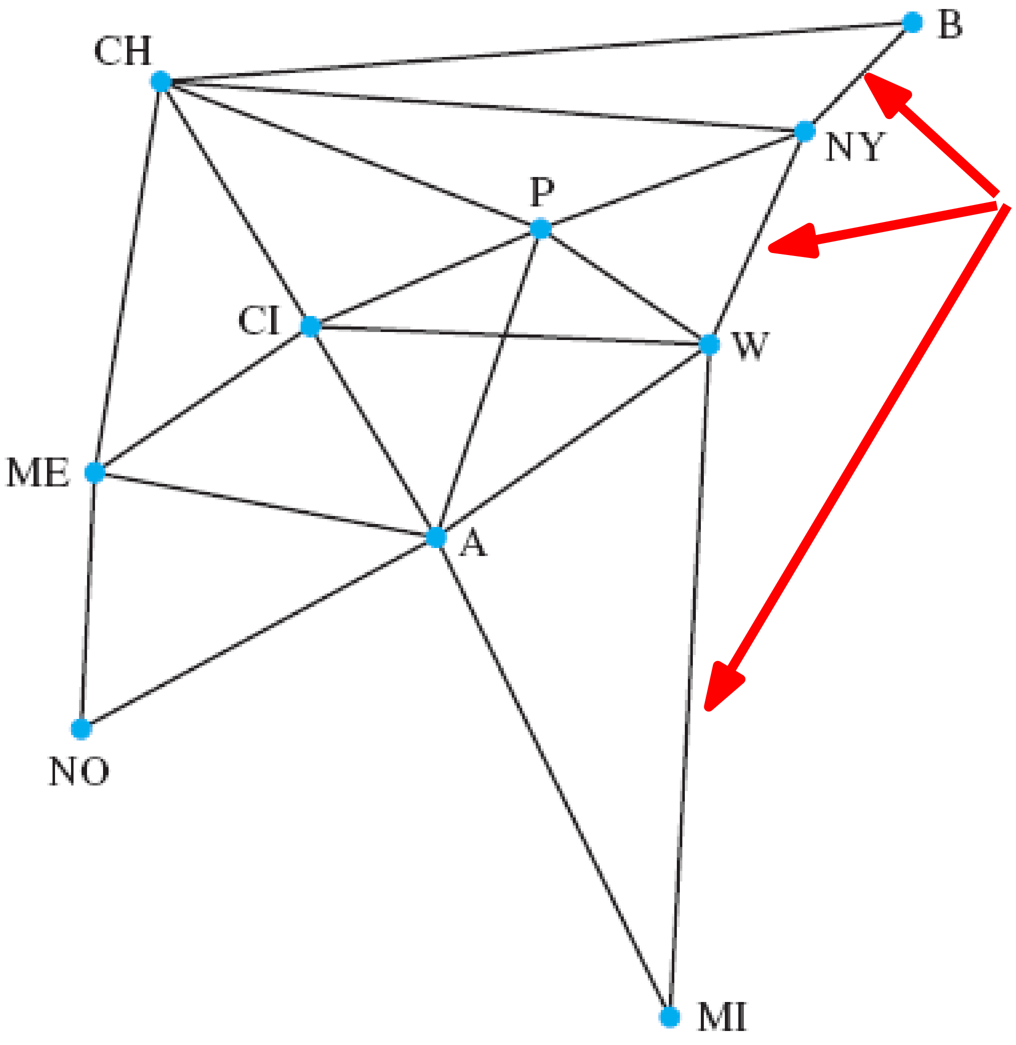


Graph G



consists of a set of vertices V, $|V| = n$

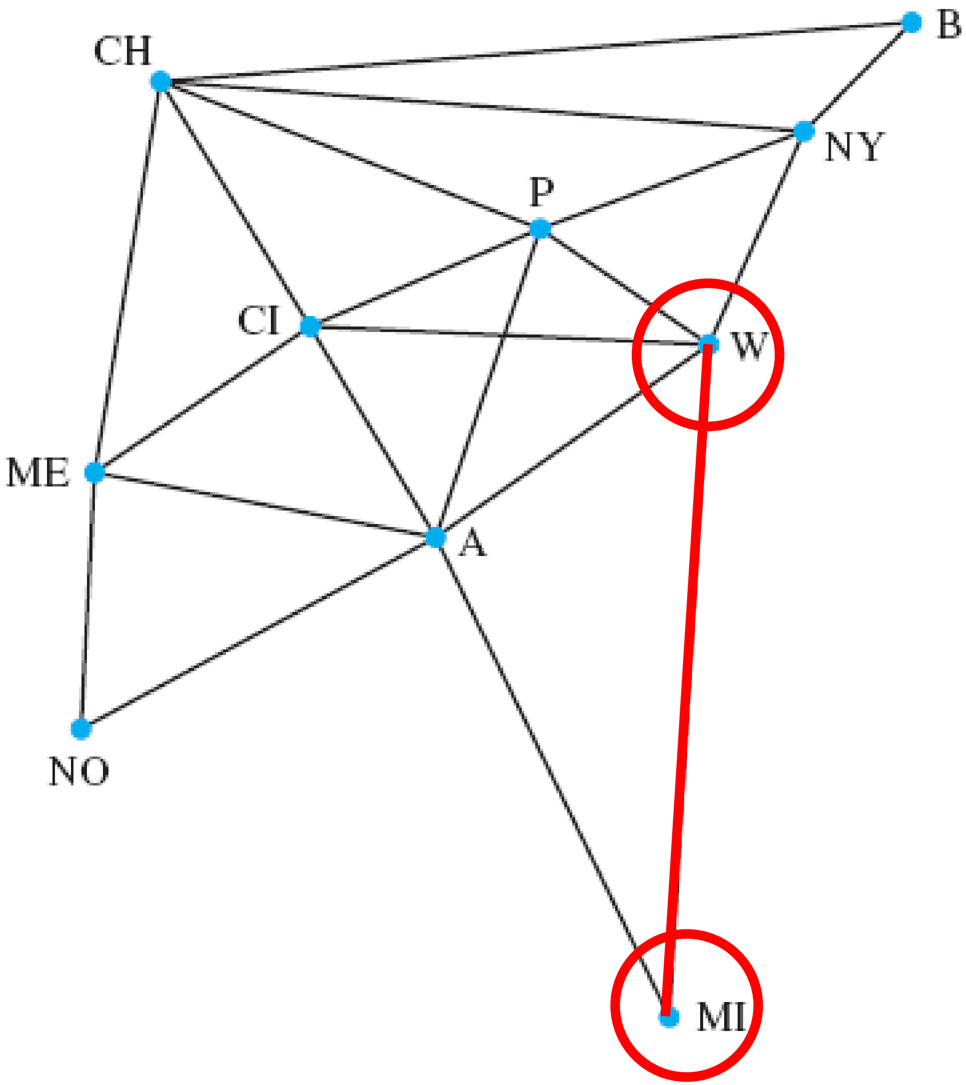
Graph G



consists of a set of vertices V, $|V| = n$

and a set of edges E, $|E| = m$

Graph G

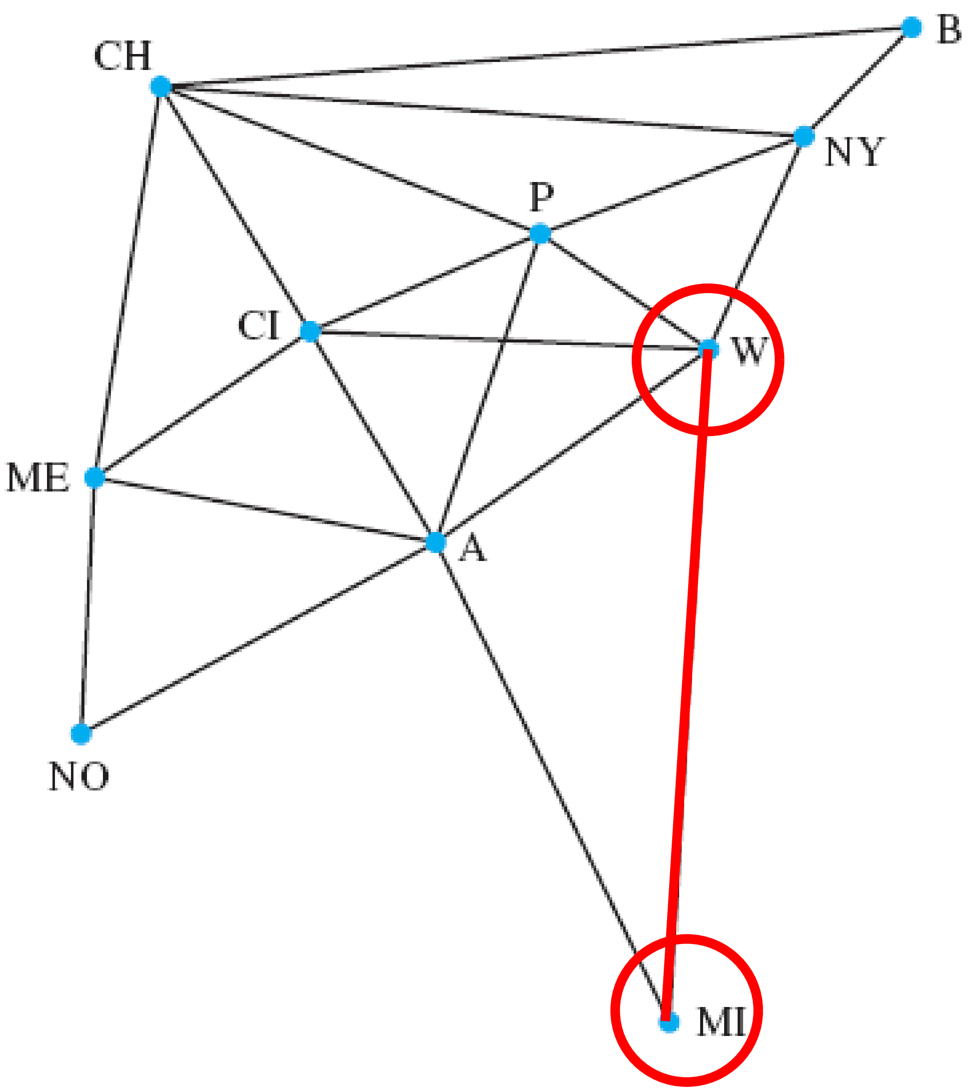


consists of a set of vertices V, $|V| = n$

and a set of edges E, $|E| = m$

Each edge has two endpoints

Graph G



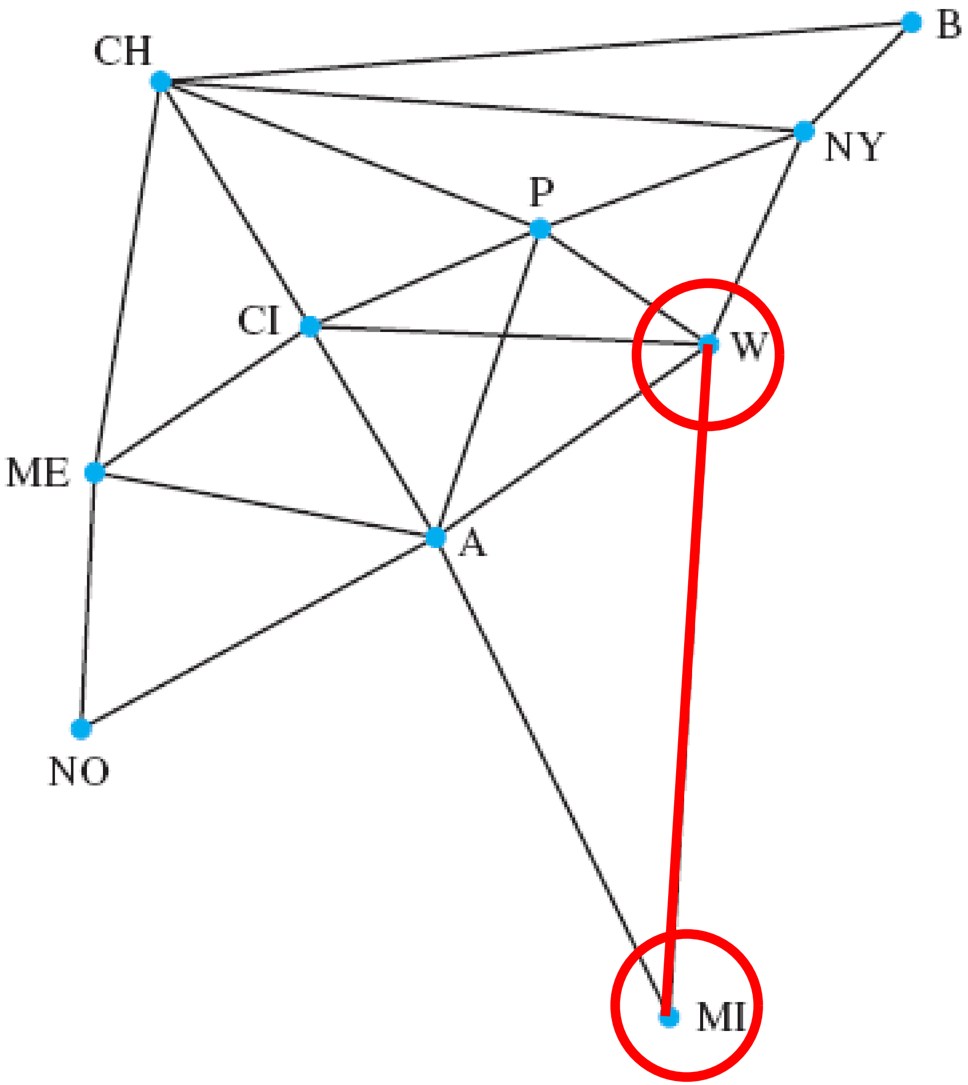
consists of a set of vertices V, $|V| = n$

and a set of edges E, $|E| = m$

Each edge has two endpoints

An edge joins its endpoints, two endpoints are adjacent if they are joined by an edge

Graph G



consists of a set of vertices V, $|V| = n$

and a set of edges E, $|E| = m$

Each edge has two endpoints

An edge joins its endpoints, two endpoints are adjacent if they are joined by an edge

When a vertex is an endpoint of an edge, we say that the edge and the vertex are incident to each other

More Examples

- Vertices: biological species
  Edges: species have a common ancestor

More Examples

- Vertices: biological species
  Edges: species have a common ancestor

  Vertices: people
  Edges: people who know each other

More Examples

- Vertices: biological species
  Edges: species have a common ancestor

  Vertices: people
  Edges: people who know each other

  Vertices: subway stations
  Edges: direct connection

More Examples

- Vertices: biological species
  Edges: species have a common ancestor

  Vertices: people
  Edges: people who know each other

  Vertices: subway stations
  Edges: direct connection

  Vertices: web sites
  Edges: a link from one site to another

More Examples

- Vertices: biological species
  Edges: species have a common ancestor

  Vertices: people
  Edges: people who know each other

  Vertices: subway stations
  Edges: direct connection

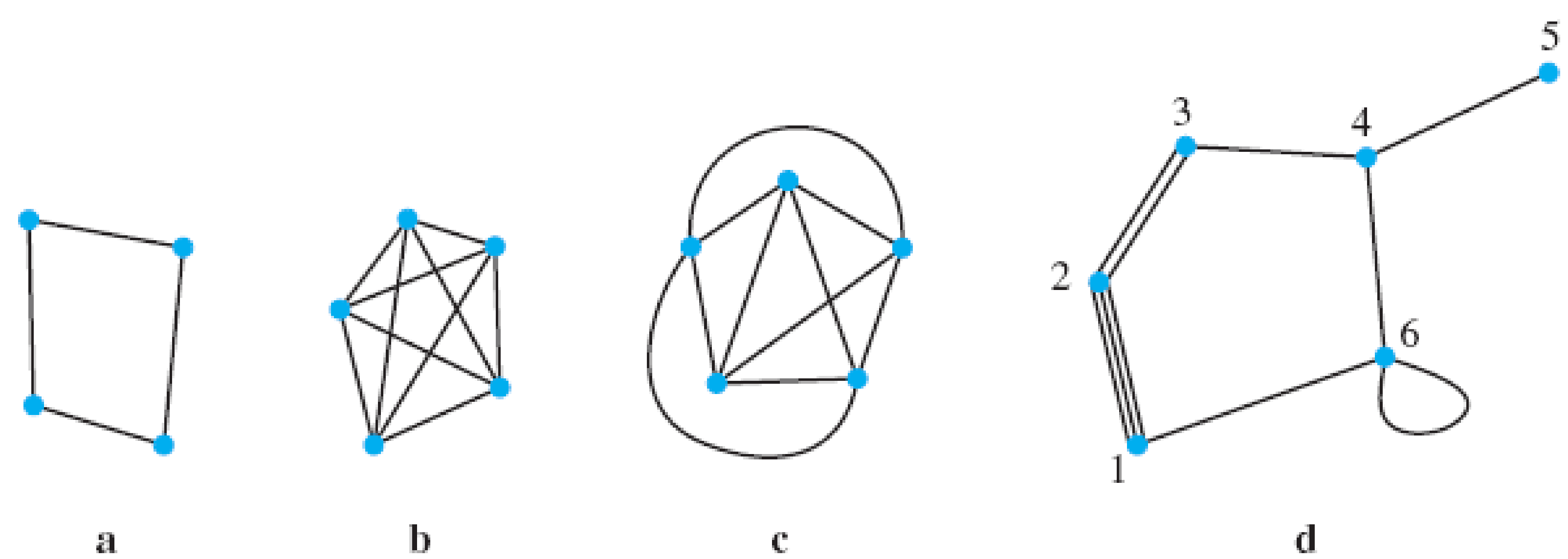
  Vertices: web sites
  Edges: a link from one site to another

How Google models the Internet!

Definition of a Graph

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

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



More Definitions

- *Simple graph* vs. *multigraph pseudograph*

  A graph in which at most one edge joins each pair of distinct vertices (vs. multiple edges) and no edge joins a vertex to itself (= loop)

More Definitions

- *Simple graph* vs. *multigraph* *pseudograph*

  A graph in which at most one edge joins each pair of distinct vertices (vs. multiple edges) and no edge joins a vertex to itself (= loop)

  Complete graph K_n

  A graph with n vertices that has an edge between each pair of vertices

Graphs

- Graphs and graph theory can be used to model:
  - Computer networks
  - Social networks
  - Communication networks
  - Infromation networks
  - Software design
  - Transportation networks
  - Biological networks

Graph Models

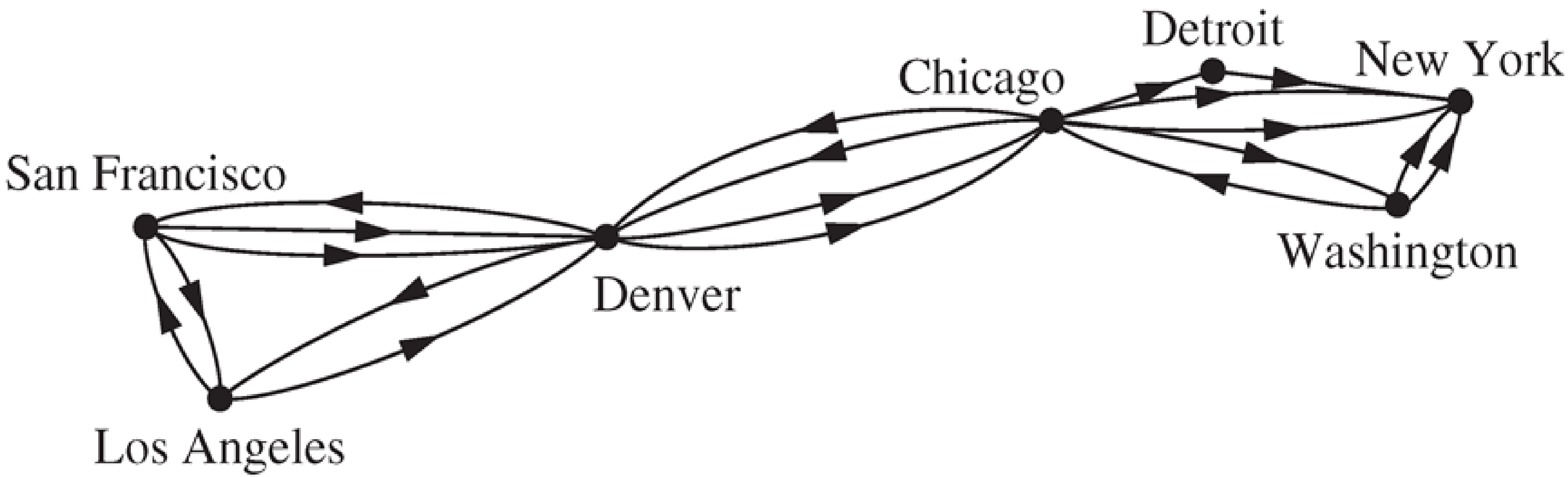
- Computer Networks

  Vertices: computers
  Edges: connections

Graph Models

- Computer Networks

  Vertices: computers
  Edges: connections



Graph Models

- Social Networks

  Vertices: individuals
  Edges: relationships

Graph Models

- Social Networks

  Vertices: individuals
  Edges: relationships

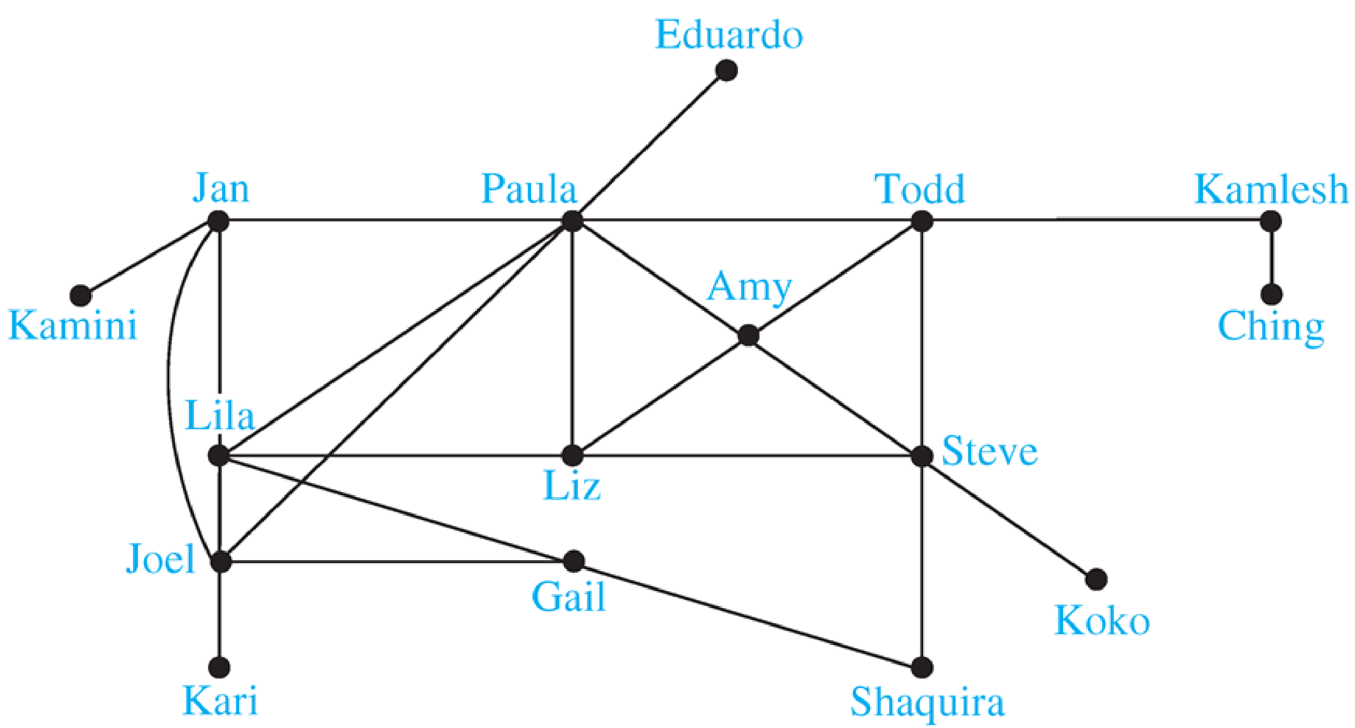
  Friendship graphs: undirected graphs where two people are connected if they are friends (in the real world, wechat, or Facebook, etc.)

Graph Models

- Social Networks

  Vertices: individuals
  Edges: relationships

  Friendship graphs: undirected graphs where two people are connected if they are friends (in the real world, wechat, or Facebook, etc.)



Graph Models

- Influence graphs

  directed graphs where there is an edge from one person to another if the first person can influence the second one

- Influence graphs

  directed graphs where there is an edge from one person to another if the first person can influence the second one

  Collaboration graphs

  undirected graphs where two people are connected if they collaborate in some way

Graph Models

- Influence graphs

  directed graphs where there is an edge from one person to another if the first person can influence the second one

  Collaboration graphs

  undirected graphs where two people are connected if they collaborate in some way

  Example

  - the Hollywood graph
  - the Erdös number

Undirected Graphs

- **Definition** Two vertices u, v in an undirected graph G are called *adjacent* (or *neighbors*) in G if there is an edge e between u and v. Such an edge e is called *incident* with the vertices u and v and e is said to connect u and v.

Undirected Graphs

- **Definition** Two vertices u, v in an undirected graph G are called *adjacent* (or *neighbors*) in G if there is an edge e between u and v. Such an edge e is called *incident* with the vertices u and v and e is said to connect u and v.

  Definition The set of all neighbors of a vertex v of $G = (V, E)$, denoted by $N(v)$, is called *the neightborhood of v*. If A is a subset of V, we denote by $N(A)$ the set of all vertices in G that are adjacent to at least one vertex in A.

Undirected Graphs

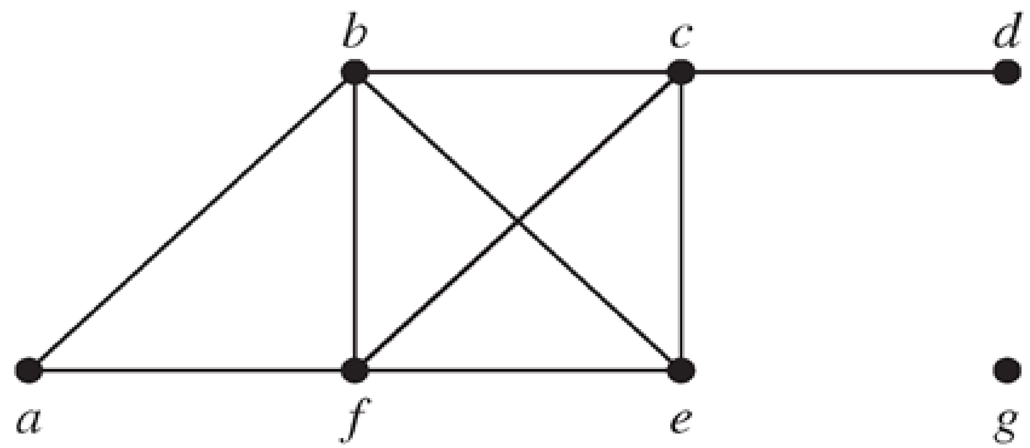
- **Definition** Two vertices u, v in an undirected graph G are called *adjacent* (or *neighbors*) in G if there is an edge e between u and v. Such an edge e is called *incident* with the vertices u and v and e is said to connect u and v.

  Definition The set of all neighbors of a vertex v of $G = (V, E)$, denoted by $N(v)$, is called *the neightborhood of v*. If A is a subset of V, we denote by $N(A)$ the set of all vertices in G that are adjacent to at least one vertex in A.

  Definition 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 two to the degree of that vertex. The degree of the vertex v is denoted by $\deg(v)$.

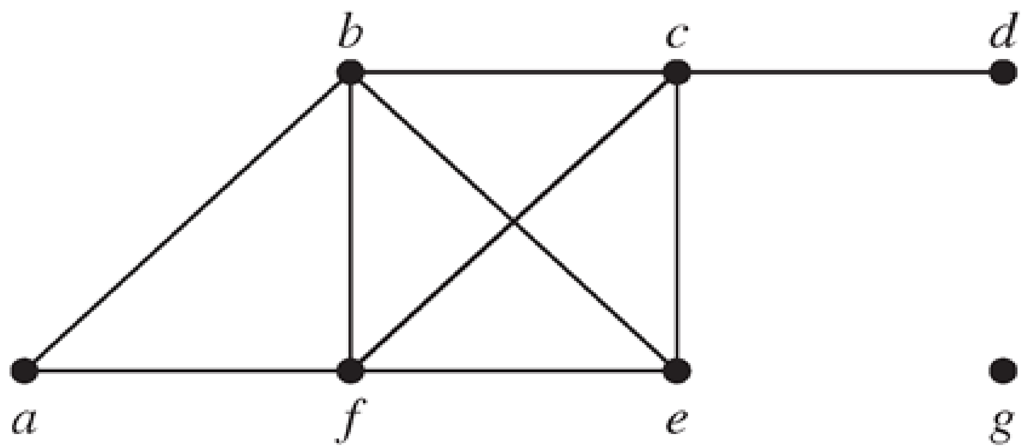
Undirected Graphs

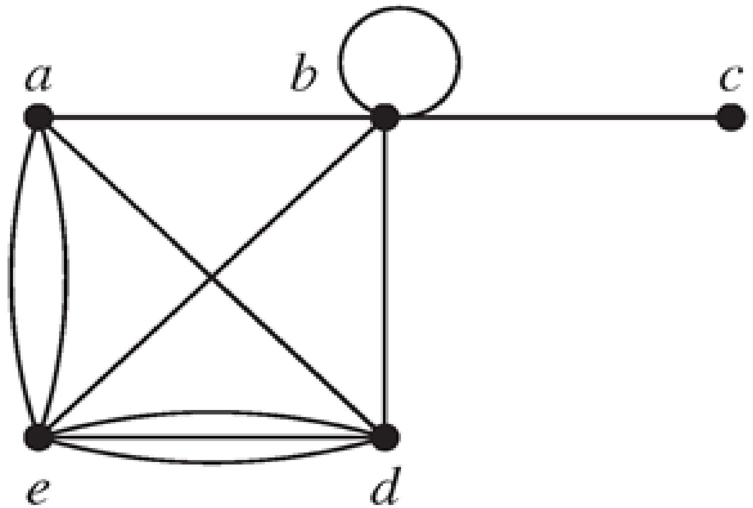
- **Example**: What are the degrees and neighborhoods of the vertices in the graph G?



Undirected Graphs

- **Example**: What are the degrees and neighborhoods of the vertices in the graph G?





Undirected Graphs

- **Theorem 1** (Handshaking Theorem) If $G = (V, E)$ is an undirected graph with m edges, then

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

Proof

Undirected Graphs

- **Theorem 2** An undirected graph has an even number of vertices of odd degree.

Undirected Graphs

- **Theorem 2** An undirected graph has an even number of vertices of odd degree.

  Proof Let V_1 be the vertices of even degrees and V_2 be the vertices of odd degree.

Undirected Graphs

- **Theorem 2** An undirected graph has an even number of vertices of odd degree.

  Proof Let V_1 be the vertices of even degrees and V_2 be the vertices of odd degree.

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

Undirected Graphs

- **Theorem 2** An undirected graph has an even number of vertices of odd degree.

  Proof Let V_1 be the vertices of even degrees and V_2 be the vertices of odd degree.

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

Directed Graphs

- **Definition** An *directed graph* $G = (V, E)$ consists of V, a nonempty set of vertices, and E, a set of directed edges. Each edge is an ordered pair of vertices. The directed edge (u, v) is said to start at u and end at v.

Directed Graphs

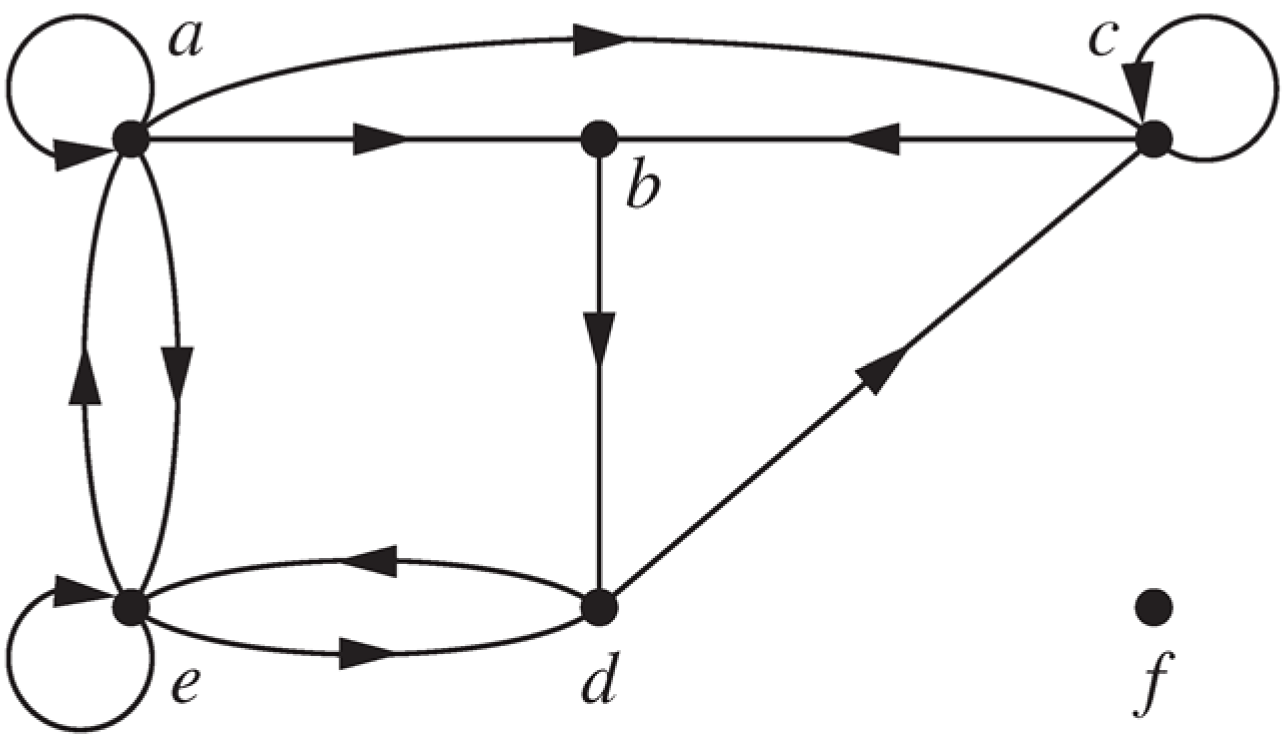
- **Definition** An *directed graph* $G = (V, E)$ consists of V, a nonempty set of vertices, and E, a set of directed edges. Each edge is an ordered pair of vertices. The directed edge (u, v) is said to start at u and end at v.

  Definition Let (u, v) be an edge in G. Then u is the *initial vertex* of the edge and is *adjacent to v* and v is the *terminal vertex* of this edge and is *adjacent from u*. The initial and terminal vertices of a loop are the same.

Directed Graphs

- **Definition** The *in-degree* of a vertex v, denoted by $\deg^{-}(v)$, is the number of edges which terminate at v. 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 the vertex.

- **Definition** The *in-degree* of a vertex v, denoted by $\deg^-(v)$, is the number of edges which terminate at v. 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 the vertex.



- **Theorem 3** Let $G = (V, E)$ be a graph with directed edges. Then
$$|E| = \sum_{v \in V} \deg^{-}(v) = \sum_{v \in V} \deg^{+}(v)$$

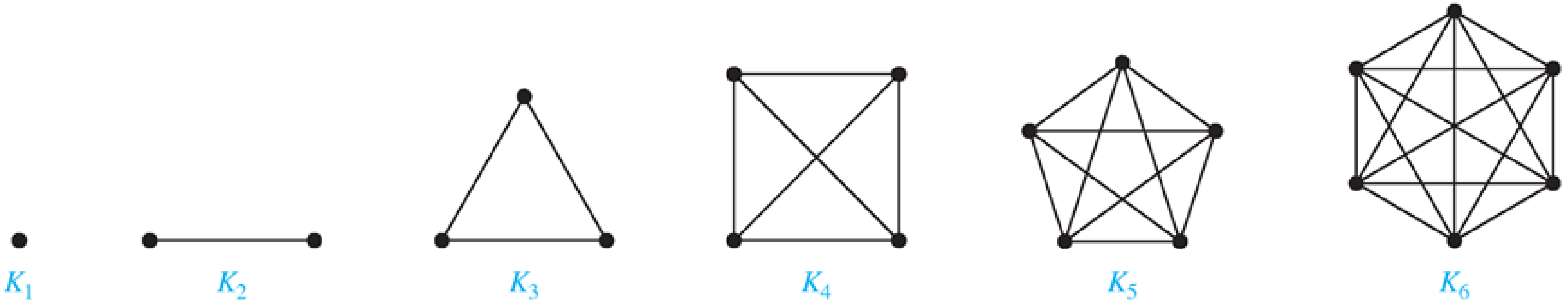
Proof

Complete Graphs

- A *complete graph* on n vertices, denoted by K_n, is the simple graph that contains exactly one edge between each pair of distinct vertices.

Complete Graphs

- A *complete graph* on n vertices, denoted by K_n, is the simple graph that contains exactly one edge between each pair of distinct vertices.

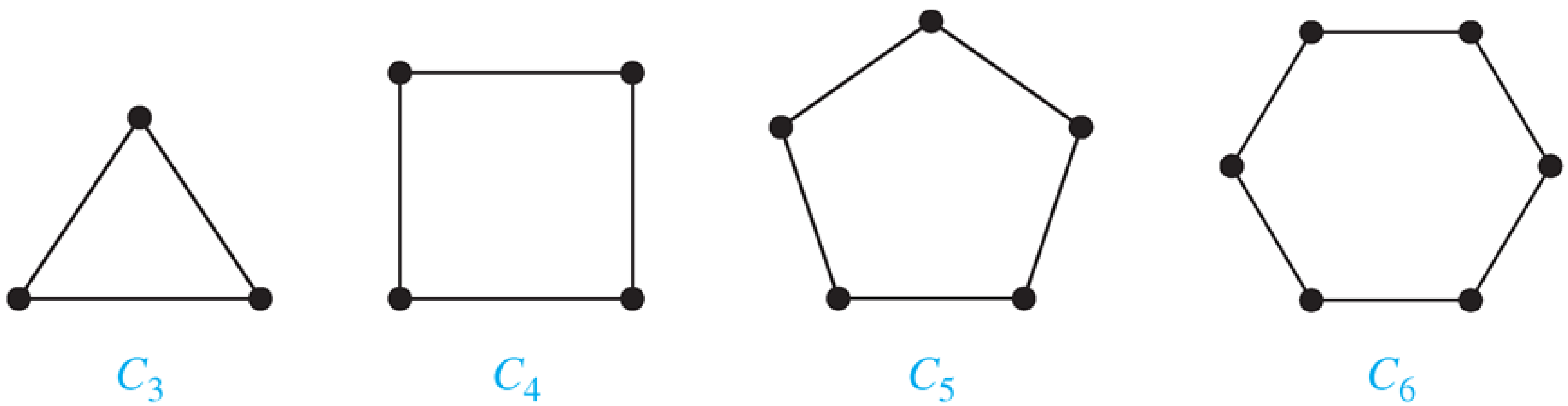


Cycles

- A *cycle* C_n for $n \geq 3$ consists of n vertices $v_1, v_2, \ldots, v_n$, and edges $\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{n-1}, v_n\}, \{v_n, v_1\}$.

Cycles

- A *cycle* C_n for $n \geq 3$ consists of n vertices $v_1, v_2, \ldots, v_n$, and edges $\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{n-1}, v_n\}, \{v_n, v_1\}$.

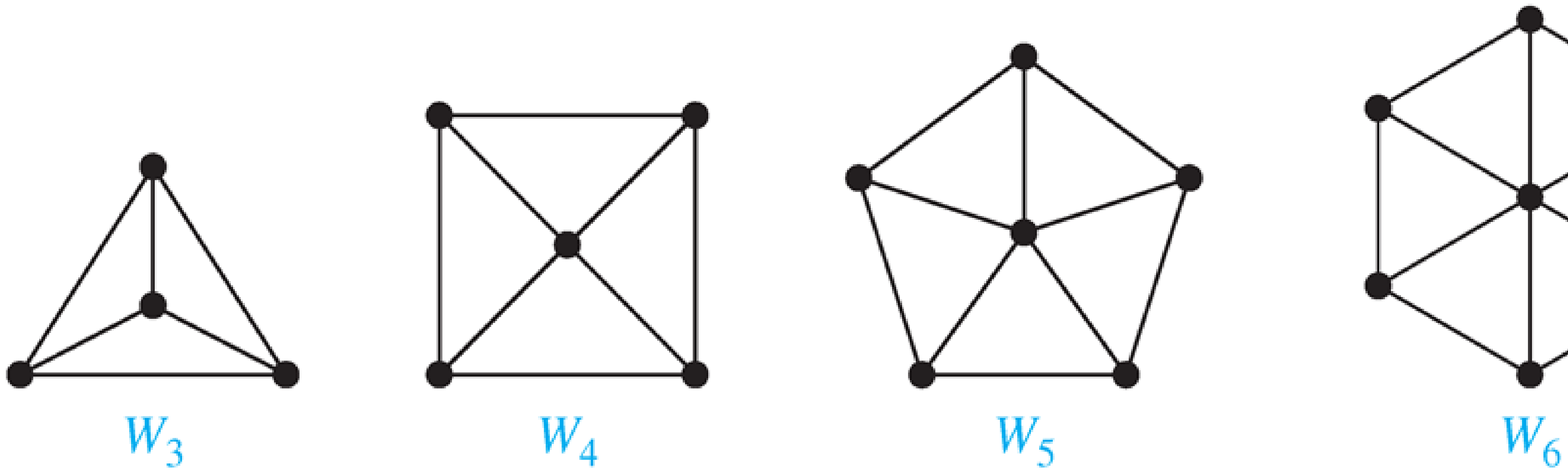


Wheels

- A *wheel* W_n is obtained by adding an additional vertex to a cycle C_n.

Wheels

- A *wheel* W_n is obtained by adding an additional vertex to a cycle C_n.

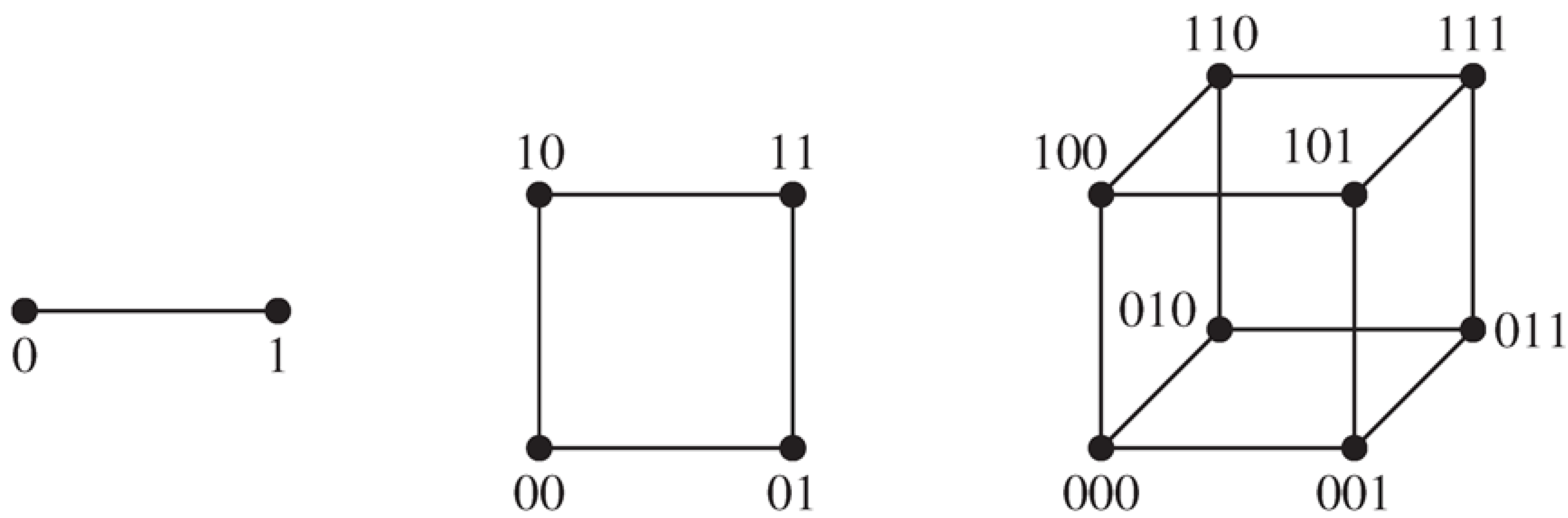


N-dimensional Hypercube

- An *n-dimensional hypercube*, or *n-cube*, Q_n is a graph with 2^n vertices representing all bit strings of length n, where there is an edge between two vertices that differ in exactly one bit position.

N-dimensional Hypercube

- An *n-dimensional hypercube*, or *n-cube*, Q_n is a graph with 2^n vertices representing all bit strings of length n, where there is an edge between two vertices that differ in exactly one bit position.



Q_1 Q_2 Q_3

Bipartite Graphs

- **Definition** A simple graph G is *bipartite* if V can be partitioned into two disjoint subsets V_1 and V_2 such that every edge connects a vertex in V_1 and a vertex in V_2.

Bipartite Graphs

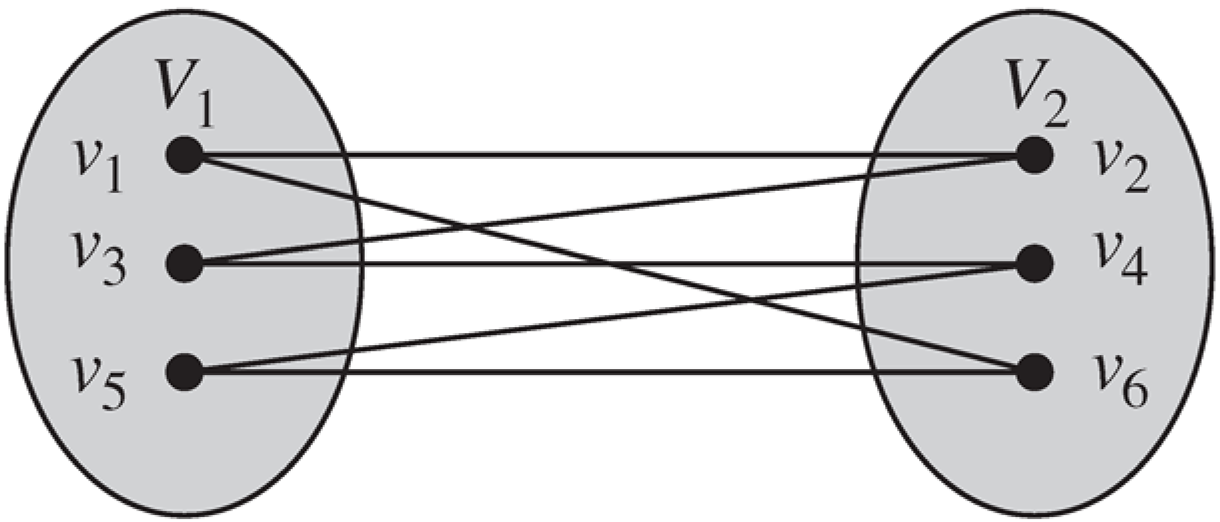
- **Definition** A simple graph G is *bipartite* if V can be partitioned into two disjoint subsets V_1 and V_2 such that every edge connects a vertex in V_1 and a vertex in V_2.

  An equivalent definition of a bipartite graph is a graph where it is possible to color the vertices red or blue so that no two adjacent vertices are of the same color.

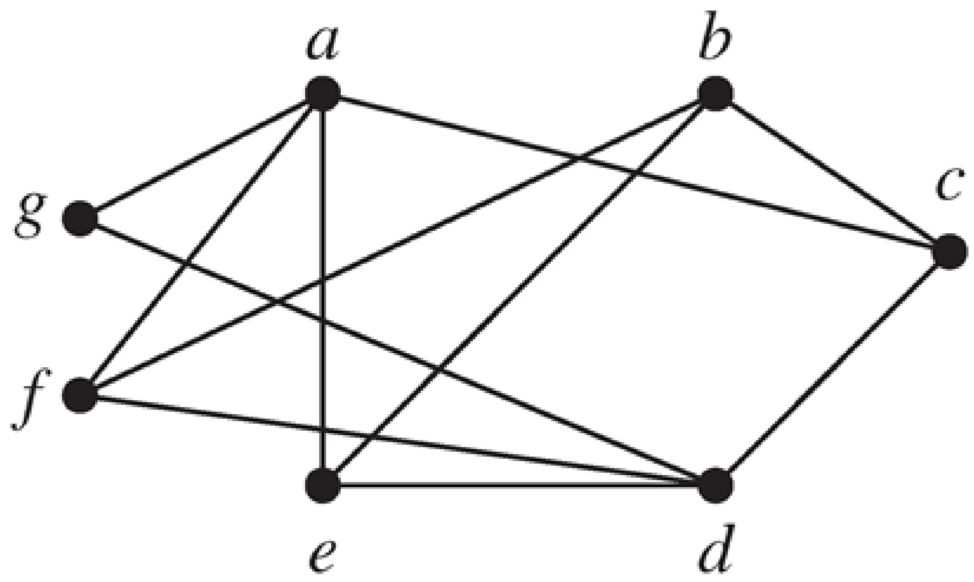
Bipartite Graphs

- **Definition** A simple graph G is *bipartite* if V can be partitioned into two disjoint subsets V_1 and V_2 such that every edge connects a vertex in V_1 and a vertex in V_2.

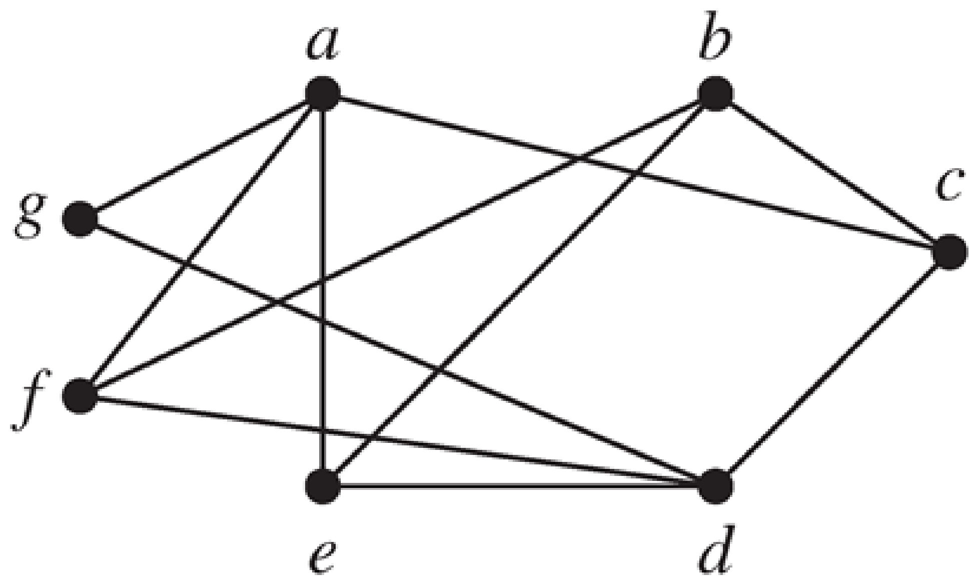
  An equivalent definition of a bipartite graph is a graph where it is possible to color the vertices red or blue so that no two adjacent vertices are of the same color.

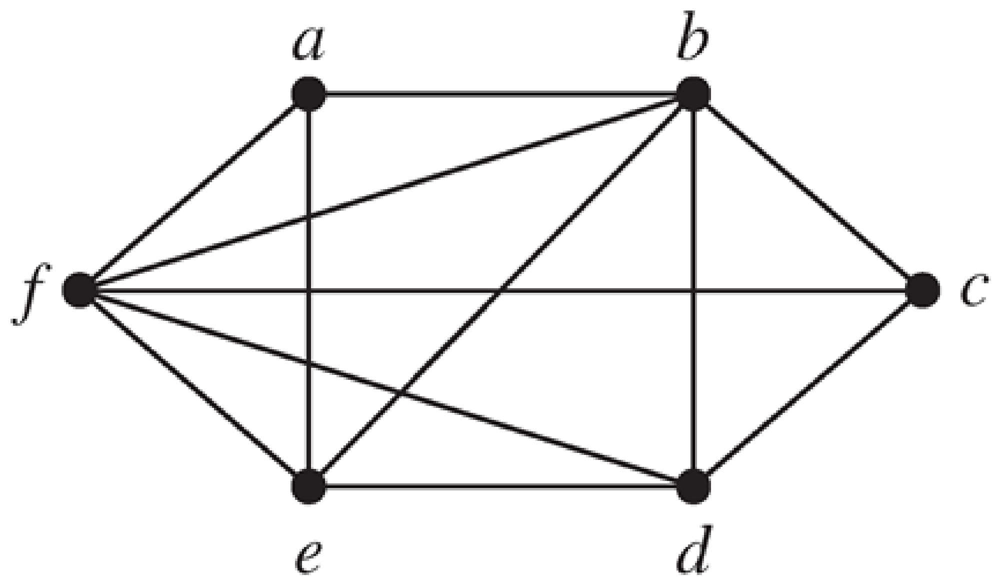


Bipartite Graphs



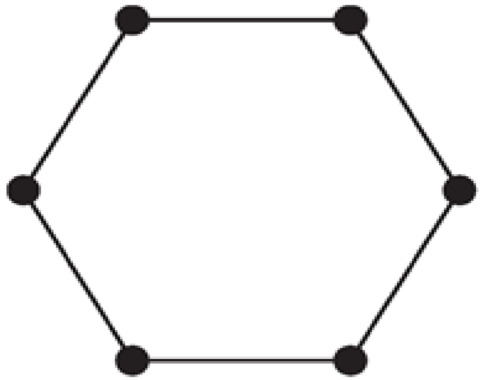
Bipartite Graphs





Bipartite Graphs

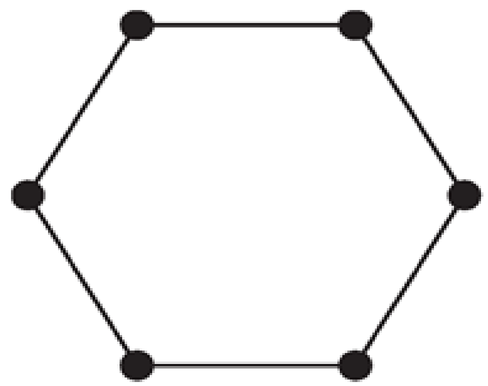
- **Example** Show that C_6 is bipartite.



C_6

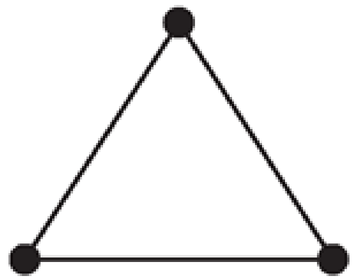
Bipartite Graphs

- **Example** Show that C_6 is bipartite.



C_6

Example Show that C_3 is not bipartite.



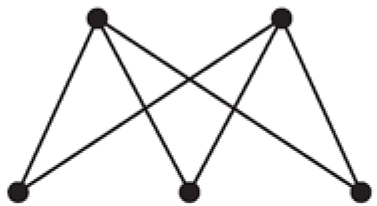
C_3

Complete Bipartite Graphs

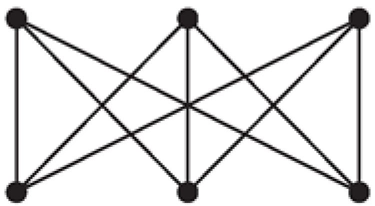
- **Definition** A *complete bipartite graph* $K_{m,n}$ is a graph that has its vertex set partitioned into two subsets V_1 of size m and V_2 of size n such that there is an edge from every vertex in V_1 to every vertex in V_2.

Complete Bipartite Graphs

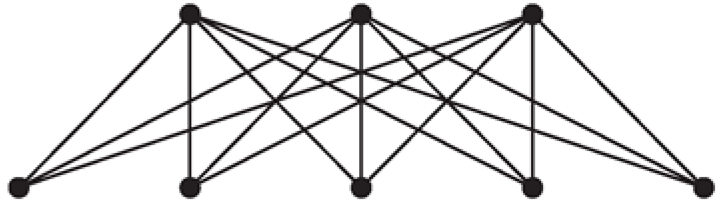
- **Definition** A *complete bipartite graph* $K_{m,n}$ is a graph that has its vertex set partitioned into two subsets V_1 of size m and V_2 of size n such that there is an edge from every vertex in V_1 to every vertex in V_2.



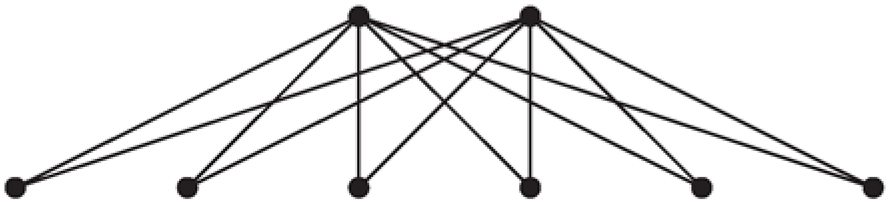
$K_{2,3}$



$K_{3,3}$



$K_{3,5}$



$K_{2,6}$

Bipartite Graphs and Matchings

- *Matching* the elements of one set to elements in another. A *matching* is a subset of E s.t. no two edges are incident with the same vertex.

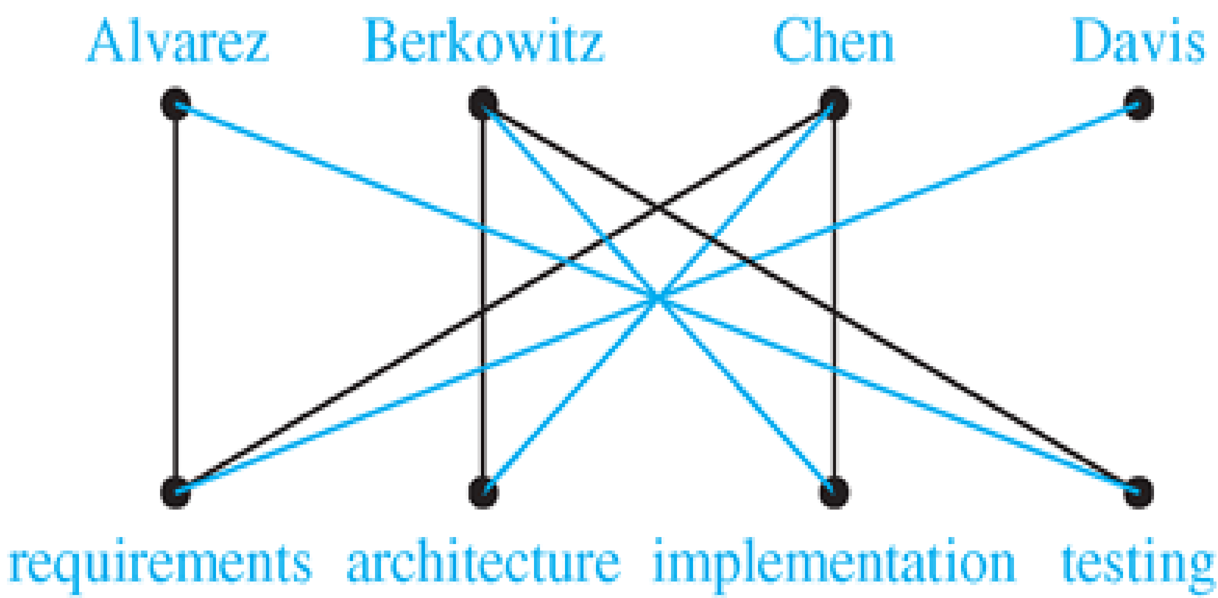
Bipartite Graphs and Matchings

- *Matching* the elements of one set to elements in another. A *matching* is a subset of E s.t. no two edges are incident with the same vertex.

  Job assignments: vertices represent the jobs and the employees, edges link employees with those jobs they have been trained to do. A common goal is to match jobs to employees so that the most jobs are done.

Bipartite Graphs and Matchings

- *Matching* the elements of one set to elements in another. A *matching* is a subset of E s.t. no two edges are incident with the same vertex.
  Job assignments: vertices represent the jobs and the employees, edges link employees with those jobs they have been trained to do. A common goal is to match jobs to employees so that the most jobs are done.



Next Lecture

- graph theory II ...