Discrete Mathematics

Chapter 9: Graphs

Department of Mathematics
The FPT university

Chapter 9: Introduction

Chapter 9: Introduction

Topics covered:

Topics covered:

9.1 Graphs and Graph Models

Chapter 9: Introduction

Topics covered:

Chapter 9: Introduction

Topics covered:

Chapter 9: Introduction

Topics covered:

Topics covered:

9.1 Graphs and Graph Models

9.2 Graph Terminologies and Special Types of Graphs

9.3 Representing Graphs and Graph Isomorphism

9.4 Connectivity

9.5 Euler and Hamilton Paths

Topics covered:

A graph $G = (V, E)$ consists of V a non-empty set of vertices (or nodes) and E, a set of edges. Each edge has one or two vertices associated with it, called endpoints.

A graph $G = (V, E)$ consists of V a non-empty set of vertices (or nodes) and E, a set of edges. Each edge has one or two vertices associated with it, called endpoints.

Graph classification:

9.1 Graphs and Graph Models

A graph $G = (V, E)$ consists of V a non-empty set of vertices (or nodes) and E, a set of edges. Each edge has one or two vertices associated with it, called endpoints.

Graph classification:

- **Undirected graphs:**

9.1 Graphs and Graph Models

A graph $G = (V, E)$ consists of V a non-empty set of vertices (or nodes) and E, a set of edges. Each edge has one or two vertices associated with it, called endpoints.

Graph classification:

- **Undirected graphs:** Simple graphs, Multigraphs and Pseudographs

A graph $G = (V, E)$ consists of V a non-empty set of vertices (or nodes) and E, a set of edges. Each edge has one or two vertices associated with it, called endpoints.

Graph classification:

- **Undirected graphs:** Simple graphs, Multigraphs and Pseudographs
- **Directed graphs:**

9.1 Graphs and Graph Models

A graph $G = (V, E)$ consists of V a non-empty set of vertices (or nodes) and E, a set of edges. Each edge has one or two vertices associated with it, called endpoints.

Graph classification:

- **Undirected graphs:** Simple graphs, Multigraphs and Pseudographs
- **Directed graphs:** Simple directed graphs, Directed multigraphs

Undirected graphs

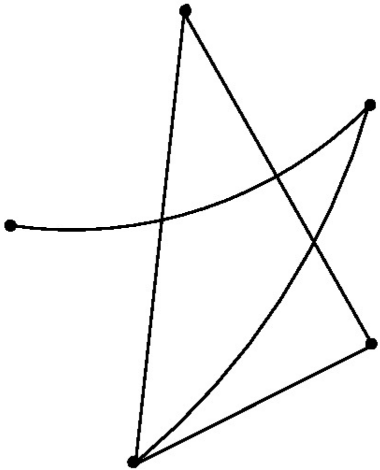
Undirected graphs

Simple graphs:

Undirected graphs

Simple graphs: For any two vertices there is at most one edge connecting them, and there are no loops.

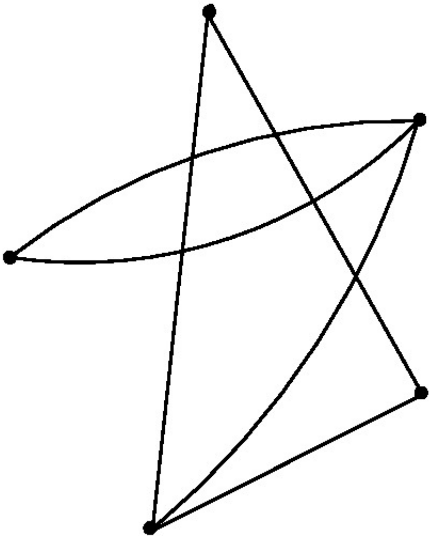
Simple graphs: For any two vertices there is at most one edge connecting them, and there are no loops.



Multigraphs:

Multigraphs: There are possibly multiple edges, but no loops.

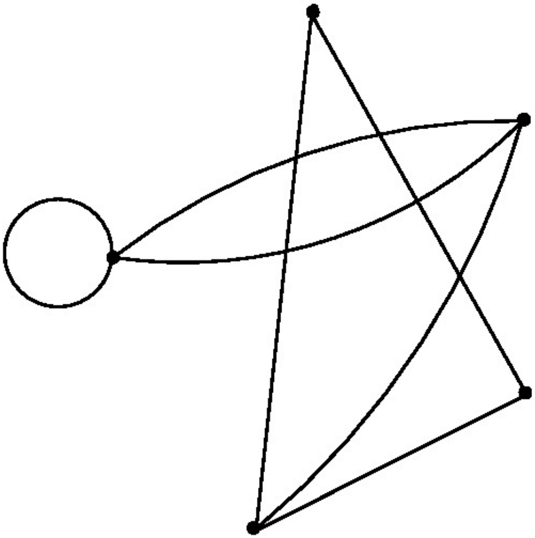
Multigraphs: There are possibly multiple edges, but no loops.



Pseudographs:

Pseudographs: There are possibly multiple edges and loops.

Pseudographs: There are possibly multiple edges and loops.



Directed graphs

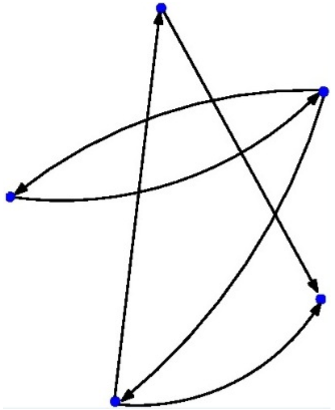
Simple directed graphs:

Directed graphs

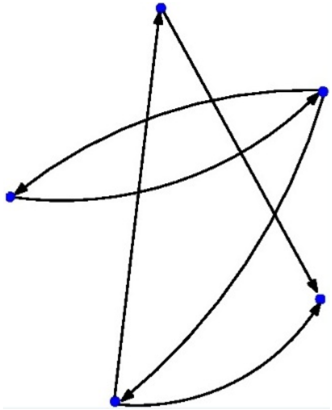
Simple directed graphs: There are no loops and no multiple directed edges.

Directed graphs

Simple directed graphs: There are no loops and no multiple directed edges.

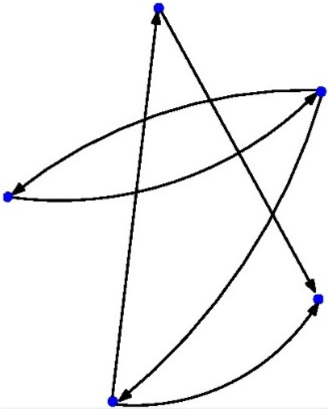


Simple directed graphs: There are no loops and no multiple directed edges.



Directed multigraphs:

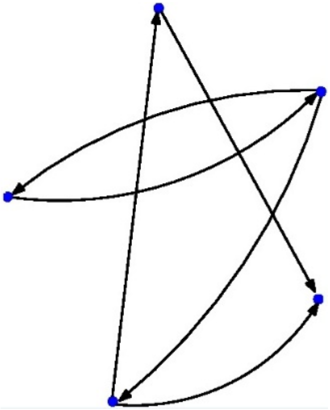
Simple directed graphs: There are no loops and no multiple directed edges.



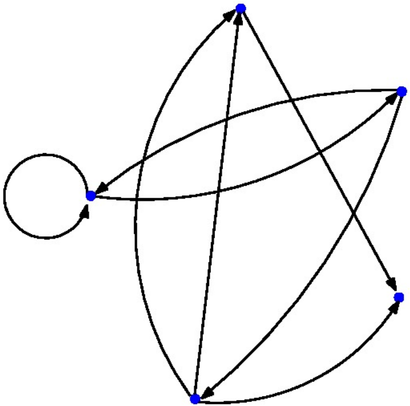
Directed multigraphs: There are possibly multiple directed edges and loops.

Directed graphs

Simple directed graphs: There are no loops and no multiple directed edges.

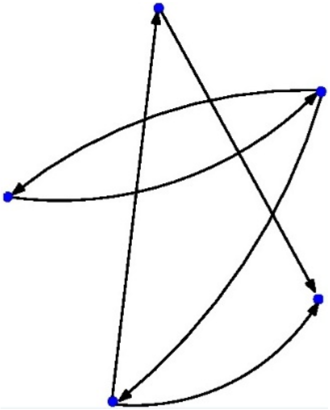


Directed multigraphs: There are possibly multiple directed edges and loops.

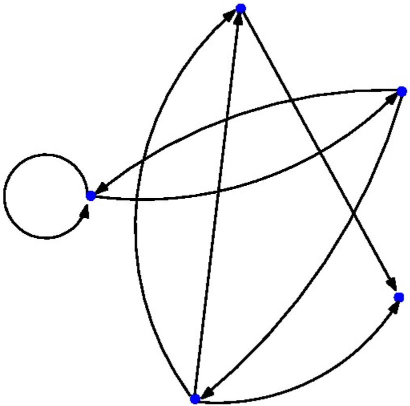


Directed graphs

Simple directed graphs: There are no loops and no multiple directed edges.



Directed multigraphs: There are possibly multiple directed edges and loops.



9.2 Graph Terminologies and Special Types of Graphs

- In an undirected graph, two vertices u and v are called adjacent if they are endpoints of an edge e, and e is called incident with u and v.

9.2 Graph Terminologies and Special Types of Graphs

- In an undirected graph, two vertices u and v are called adjacent if they are endpoints of an edge e, and e is called incident with u and v.
- 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.

9.2 Graph Terminologies and Special Types of Graphs

- In an undirected graph, two vertices u and v are called adjacent if they are endpoints of an edge e, and e is called incident with u and v.
- 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.

9.2 Graph Terminologies and Special Types of Graphs

- In an undirected graph, two vertices u and v are called adjacent if they are endpoints of an edge e, and e is called incident with u and v.
- 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 Handshaking Theorem for Undirected graphs

- In an undirected graph, two vertices u and v are called adjacent if they are endpoints of an edge e, and e is called incident with u and v.
- 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 Handshaking Theorem for Undirected graphs

In an undirected graph G with e edges,

$$\sum \deg(v) = 2e$$

- In an undirected graph, two vertices u and v are called adjacent if they are endpoints of an edge e, and e is called incident with u and v.
- 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 Handshaking Theorem for Undirected graphs

In an undirected graph G with e edges,

$$\sum \deg(v) = 2e$$

Therefore, the number of vertices of odd degrees is an even number.

- In an undirected graph, two vertices u and v are called adjacent if they are endpoints of an edge e, and e is called incident with u and v.
- 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 Handshaking Theorem for Undirected graphs

In an undirected graph G with e edges,

$$\sum \deg(v) = 2e$$

Therefore, the number of vertices of odd degrees is an even number.

Example. Do there exist simple graphs with 5 vertices of degrees:

9.2 Graph Terminologies and Special Types of Graphs

- In an undirected graph, two vertices u and v are called adjacent if they are endpoints of an edge e, and e is called incident with u and v.
- 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 Handshaking Theorem for Undirected graphs

In an undirected graph G with e edges,

$$\sum \deg(v) = 2e$$

Therefore, the number of vertices of odd degrees is an even number.

Example. Do there exist simple graphs with 5 vertices of degrees:

(a) 1, 2, 3, 3, 4

9.2 Graph Terminologies and Special Types of Graphs

- In an undirected graph, two vertices u and v are called adjacent if they are endpoints of an edge e, and e is called incident with u and v.
- 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 Handshaking Theorem for Undirected graphs

In an undirected graph G with e edges,

$$\sum \deg(v) = 2e$$

Therefore, the number of vertices of odd degrees is an even number.

Example. Do there exist simple graphs with 5 vertices of degrees:

(a) 1, 2, 3, 3, 4 (b) 1, 2, 3, 3, 3

9.2 Graph Terminologies and Special Types of Graphs

- In an undirected graph, two vertices u and v are called adjacent if they are endpoints of an edge e, and e is called incident with u and v.
- 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 Handshaking Theorem for Undirected graphs

In an undirected graph G with e edges,

$$\sum \deg(v) = 2e$$

Therefore, the number of vertices of odd degrees is an even number.

Example. Do there exist simple graphs with 5 vertices of degrees:

(a) 1, 2, 3, 3, 4 (b) 1, 2, 3, 3, 3 (c) 1, 2, 3, 4, 4

- If $e = (u, v)$ is an edge of a directed graph, u is said to be adjacent to v and v is adjacent from u. The vertex u is called the initial and v is called the terminal or end vertex of e.

- If $e = (u, v)$ is an edge of a directed graph, u is said to be adjacent to v and v is adjacent from u. The vertex u is called the initial and v is called the terminal or end vertex of e.
- The in-degree of a vertex v in a directed graph, denoted by $\deg^{-}(v)$, is the number of edges with v as their terminal vertex. The out-degree of u, denoted by $\deg^{+}(v)$, is the number of edges with v as their initial vertex.

- If $e = (u, v)$ is an edge of a directed graph, u is said to be adjacent to v and v is adjacent from u. The vertex u is called the initial and v is called the terminal or end vertex of e.
- The in-degree of a vertex v in a directed graph, denoted by $\deg^-(v)$, is the number of edges with v as their terminal vertex. The out-degree of u, denoted by $\deg^+(v)$, is the number of edges with v as their initial vertex.

The Handshaking Theorem for directed graphs

- If $e = (u, v)$ is an edge of a directed graph, u is said to be adjacent to v and v is adjacent from u. The vertex u is called the initial and v is called the terminal or end vertex of e.
- The in-degree of a vertex v in a directed graph, denoted by $\deg^{-}(v)$, is the number of edges with v as their terminal vertex. The out-degree of u, denoted by $\deg^{+}(v)$, is the number of edges with v as their initial vertex.

The Handshaking Theorem for directed graphs

In a directed graph G,

$$\sum \deg^{+}(v) = \sum \deg^{-}(v)$$

- If $e = (u, v)$ is an edge of a directed graph, u is said to be adjacent to v and v is adjacent from u. The vertex u is called the initial and v is called the terminal or end vertex of e.
- The in-degree of a vertex v in a directed graph, denoted by $\deg^-(v)$, is the number of edges with v as their terminal vertex. The out-degree of u, denoted by $\deg^+(v)$, is the number of edges with v as their initial vertex.

The Handshaking Theorem for directed graphs

In a directed graph G,

$$\sum \deg^+(v) = \sum \deg^-(v)$$

- If $e = (u, v)$ is an edge of a directed graph, u is said to be adjacent to v and v is adjacent from u. The vertex u is called the initial and v is called the terminal or end vertex of e.
- The in-degree of a vertex v in a directed graph, denoted by $\deg^-(v)$, is the number of edges with v as their terminal vertex. The out-degree of u, denoted by $\deg^+(v)$, is the number of edges with v as their initial vertex.

The Handshaking Theorem for directed graphs

In a directed graph G,

$$\sum \deg^+(v) = \sum \deg^-(v)$$

Some Special Simple Graphs

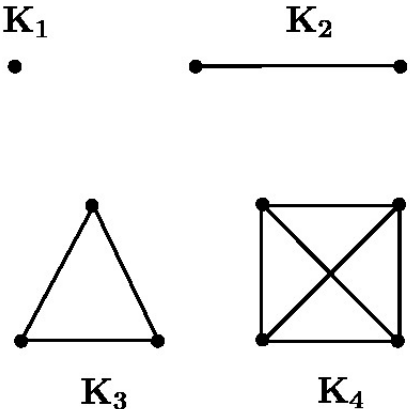
Some Special Simple Graphs

Complete Graphs K_n, $n \geq 1$:

Some Special Simple Graphs

Complete Graphs K_n, $n \geq 1$: n vertices, any two distinct vertices are connected by only one edge.

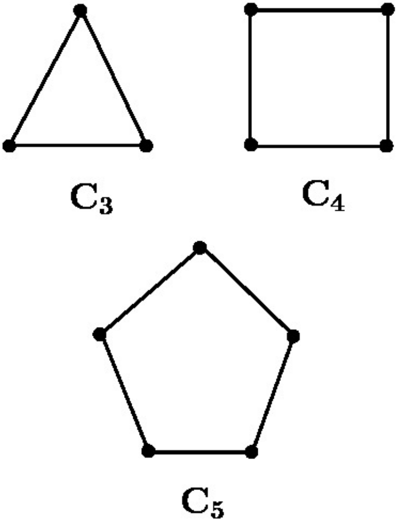
Complete Graphs K_n, $n \geq 1$: n vertices, any two distinct vertices are connected by only one edge.



Cycles C_n, $n \geq 3$:

Cycles C_n, $n \geq 3$: n vertices $v_1, v_2, \ldots, v_n$ and edges $v_1 v_2, v_2 v_3, \ldots, v_{n-1} v_n, v_n v_1$.

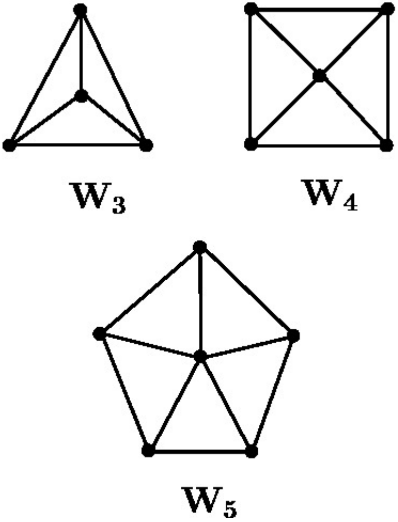
Cycles C_n, $n \geq 3$: n vertices $v_1, v_2, \ldots, v_n$ and edges $v_1v_2, v_2v_3, \ldots, v_{n-1}v_n, v_nv_1$.



Wheels W_n, $n \geq 3$:

Wheels W_n, $n \geq 3$: Add one vertex to C_n and connect it with the remaining vertices.

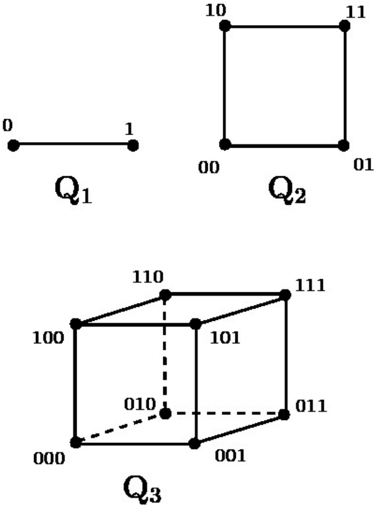
Wheels W_n, $n \geq 3$: Add one vertex to C_n and connect it with the remaining vertices.



n-**Cubes** Q_n, $n \geq 1$:

n-**Cubes** Q_n, $n \geq 1$: 2^n vertices, and the edges are drawn by the following rule: represent each vertex by a bit string of length n, and two vertices are connected if their bit strings differ in exactly one position.

n-**Cubes** Q_n, $n \geq 1$: 2^n vertices, and the edges are drawn by the following rule: represent each vertex by a bit string of length n, and two vertices are connected if their bit strings differ in exactly one position.



Question. How many edges each of the graphs K_n, C_n, W_n, Q_n has?

Bipartite graphs

Bipartite graphs

Bipartite graphs

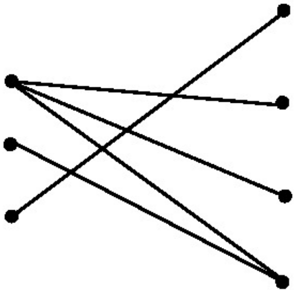
A simple graph G is called bipartite if the vertex set can be divided in two disjoint subsets such that each edge connects one vertex from one of these two subsets to another vertex of the other subset.

Bipartite graphs

A simple graph G is called bipartite if the vertex set can be divided in two disjoint subsets such that each edge connects one vertex from one of these two subsets to another vertex of the other subset.

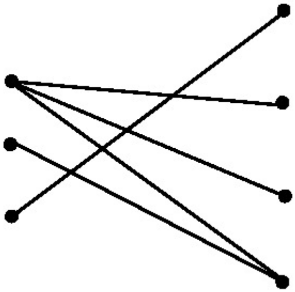
Bipartite graphs

A simple graph G is called bipartite if the vertex set can be divided in two disjoint subsets such that each edge connects one vertex from one of these two subsets to another vertex of the other subset.



Bipartite graphs

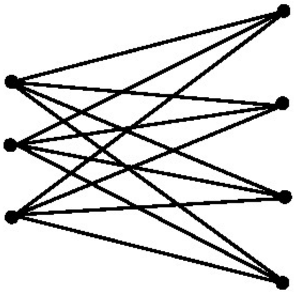
A simple graph G is called bipartite if the vertex set can be divided in two disjoint subsets such that each edge connects one vertex from one of these two subsets to another vertex of the other subset.



Question. Which graphs K_n, C_n, W_n, Q_n are bipartite?

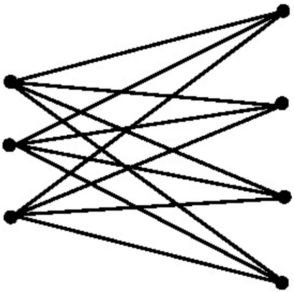
The complete bipartite graph K_{mn} is the graph whose vertex set is divided to two disjoint subsets of m and n vertices, such that two vertices are connected if and only if they do not belong to the same subset.

The complete bipartite graph K_{mn} is the graph whose vertex set is divided to two disjoint subsets of m and n vertices, such that two vertices are connected if and only if they do not belong to the same subset.



$\mathbf{K_{3,4}}$

The complete bipartite graph K_{mn} is the graph whose vertex set is divided to two disjoint subsets of m and n vertices, such that two vertices are connected if and only if they do not belong to the same subset.



$\mathbf{K_{3,4}}$

Question. How many edges does the graph K_{mn} have?

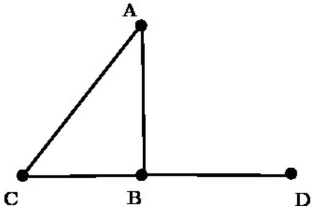
New Graphs from Olds

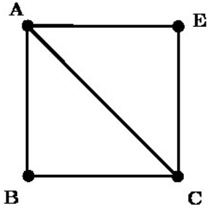
New Graphs from Olds

The union of 2 graphs G and H is a new graph whose vertex set consists of vertices of G and H, and whose edge set consists of edges of G and H.

The union of 2 graphs G and H is a new graph whose vertex set consists of vertices of G and H, and whose edge set consists of edges of G and H.

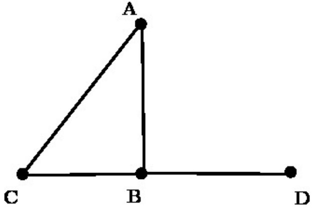
Example. Union of 2 graphs

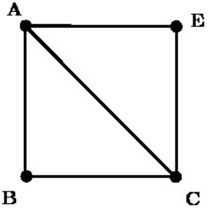




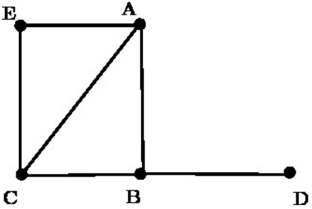
The union of 2 graphs G and H is a new graph whose vertex set consists of vertices of G and H, and whose edge set consists of edges of G and H.

Example. Union of 2 graphs



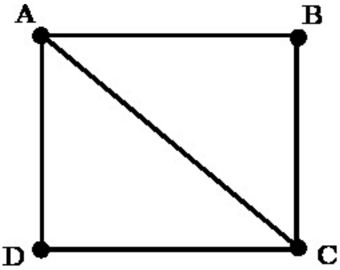


is the graph



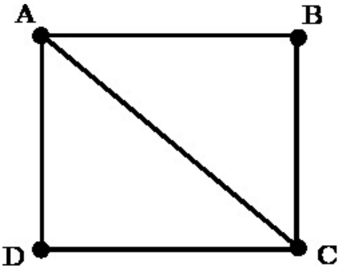
A subgraph of the graph G is a graph whose vertex set is a subset of the vertex set of G, and whose edge set is a subset of the edge set of G.

A subgraph of the graph G is a graph whose vertex set is a subset of the vertex set of G, and whose edge set is a subset of the edge set of G.

Example. Given the graph 

A subgraph of the graph G is a graph whose vertex set is a subset of the vertex set of G, and whose edge set is a subset of the edge set of G.

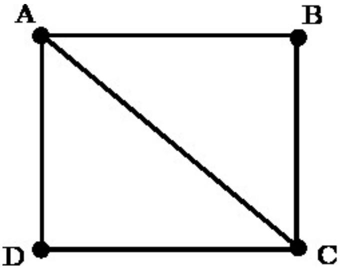
Example. Given the graph



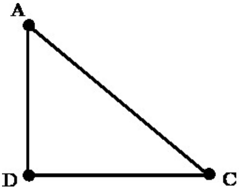
Then:

A subgraph of the graph G is a graph whose vertex set is a subset of the vertex set of G, and whose edge set is a subset of the edge set of G.

Example. Given the graph

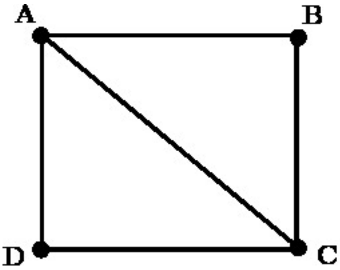


Then:

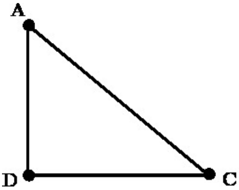


A subgraph of the graph G is a graph whose vertex set is a subset of the vertex set of G, and whose edge set is a subset of the edge set of G.

Example. Given the graph



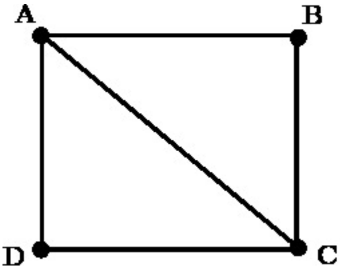
Then:



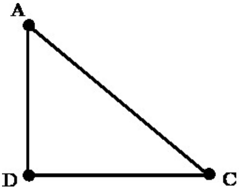
is a subgraph of G

A subgraph of the graph G is a graph whose vertex set is a subset of the vertex set of G, and whose edge set is a subset of the edge set of G.

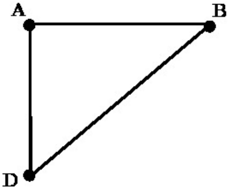
Example. Given the graph



Then:

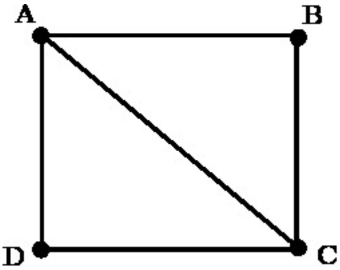


is a subgraph of G

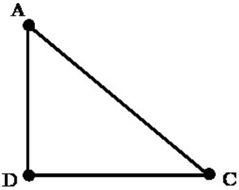


A subgraph of the graph G is a graph whose vertex set is a subset of the vertex set of G, and whose edge set is a subset of the edge set of G.

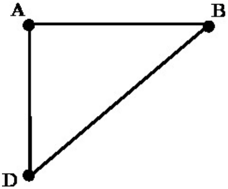
Example. Given the graph



Then:



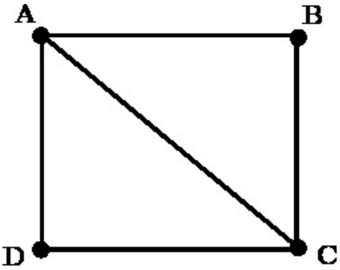
is a subgraph of G



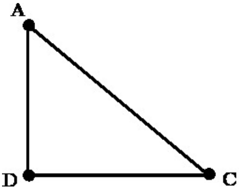
is not a subgraph of G

A subgraph of the graph G is a graph whose vertex set is a subset of the vertex set of G, and whose edge set is a subset of the edge set of G.

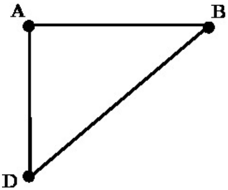
Example. Given the graph



Then:



is a subgraph of G



is not a subgraph of G

Adjacency matrices for Undirected graphs.

Adjacency matrices for Undirected graphs.

Let G be a pseudograph with vertices $v_1, v_2, \ldots, v_n$.

Adjacency matrices for Undirected graphs.

Let G be a pseudograph with vertices $v_1, v_2, \ldots, v_n$. We can represent G by a square matrix $[a_{ij}]$ of order n,

Adjacency matrices for Undirected graphs.

Let G be a pseudograph with vertices $v_1, v_2, \ldots, v_n$. We can represent G by a square matrix $[a_{ij}]$ of order n, whose entries are determined as follows:

Adjacency matrices for Undirected graphs.

Let G be a pseudograph with vertices $v_1, v_2, \ldots, v_n$. We can represent G by a square matrix $[a_{ij}]$ of order n, whose entries are determined as follows:

a_{ij} = the number of edges in G connecting v_i and v_j .

Adjacency matrices for Undirected graphs.

Let G be a pseudograph with vertices $v_1, v_2, \ldots, v_n$. We can represent G by a square matrix $[a_{ij}]$ of order n, whose entries are determined as follows:

$a_{ij} =$ the number of edges in G connecting v_i and v_j .

Adjacency matrices for Directed graphs.

Adjacency matrices for Undirected graphs.

Let G be a pseudograph with vertices $v_1, v_2, \ldots, v_n$. We can represent G by a square matrix $[a_{ij}]$ of order n, whose entries are determined as follows:

$a_{ij} =$ the number of edges in G connecting v_i and v_j .

Adjacency matrices for Directed graphs.

Let G be a directed graph with vertices $v_1, v_2, \ldots, v_n$.

Adjacency matrices for Undirected graphs.

Let G be a pseudograph with vertices $v_1, v_2, \ldots, v_n$. We can represent G by a square matrix $[a_{ij}]$ of order n, whose entries are determined as follows:

$a_{ij} =$ the number of edges in G connecting v_i and v_j .

Adjacency matrices for Directed graphs.

Let G be a directed graph with vertices $v_1, v_2, \ldots, v_n$. We can represent G by a square matrix $[a_{ij}]$ of order n,

Adjacency matrices for Undirected graphs.

Let G be a pseudograph with vertices $v_1, v_2, \ldots, v_n$. We can represent G by a square matrix $[a_{ij}]$ of order n, whose entries are determined as follows:

$a_{ij} =$ the number of edges in G connecting v_i and v_j .

Adjacency matrices for Directed graphs.

Let G be a directed graph with vertices $v_1, v_2, \ldots, v_n$. We can represent G by a square matrix $[a_{ij}]$ of order n, whose entries are determined as follows:

Adjacency matrices for Undirected graphs.

Let G be a pseudograph with vertices $v_1, v_2, \ldots, v_n$. We can represent G by a square matrix $[a_{ij}]$ of order n, whose entries are determined as follows:

$a_{ij} =$ the number of edges in G connecting v_i and v_j .

Adjacency matrices for Directed graphs.

Let G be a directed graph with vertices $v_1, v_2, \ldots, v_n$. We can represent G by a square matrix $[a_{ij}]$ of order n, whose entries are determined as follows:

$a_{ij} =$ the number of edges in G whose initial vertex is v_i and whose end vertex is v_j .

Adjacency matrices for Undirected graphs.

Let G be a pseudograph with vertices $v_1, v_2, \ldots, v_n$. We can represent G by a square matrix $[a_{ij}]$ of order n, whose entries are determined as follows:

$a_{ij} =$ the number of edges in G connecting v_i and v_j .

Adjacency matrices for Directed graphs.

Let G be a directed graph with vertices $v_1, v_2, \ldots, v_n$. We can represent G by a square matrix $[a_{ij}]$ of order n, whose entries are determined as follows:

$a_{ij} =$ the number of edges in G whose initial vertex is v_i and whose end vertex is v_j .

Incident matrices.

Incident matrices.

Let G be a pseudograph with vertices $v_1, v_2, \ldots, v_n$ and edges $e_1, e_2, \ldots, e_m$.

Incident matrices.

Let G be a pseudograph with vertices $v_1, v_2, \ldots, v_n$ and edges $e_1, e_2, \ldots, e_m$. We can represent G by an incident matrix $[a_{ij}]$ of size $n \times m$,

Incident matrices.

Let G be a pseudograph with vertices $v_1, v_2, \dots, v_n$ and edges $e_1, e_2, \dots, e_m$. We can represent G by an incident matrix $[a_{ij}]$ of size $n \times m$, whose entries are determined as follows:

Incident matrices.

Let G be a pseudograph with vertices $v_1, v_2, \ldots, v_n$ and edges $e_1, e_2, \ldots, e_m$. We can represent G by an incident matrix $[a_{ij}]$ of size $n \times m$, whose entries are determined as follows:

$a_{ij} = 1$ if e_j is incident to v_i,

Incident matrices.

Let G be a pseudograph with vertices $v_1, v_2, \ldots, v_n$ and edges $e_1, e_2, \ldots, e_m$. We can represent G by an incident matrix $[a_{ij}]$ of size $n \times m$, whose entries are determined as follows:

$a_{ij} = 1$ if e_j is incident to v_i, and
$a_{ij} = 0$ otherwise.

Graph Isomorphism

Graph Isomorphism

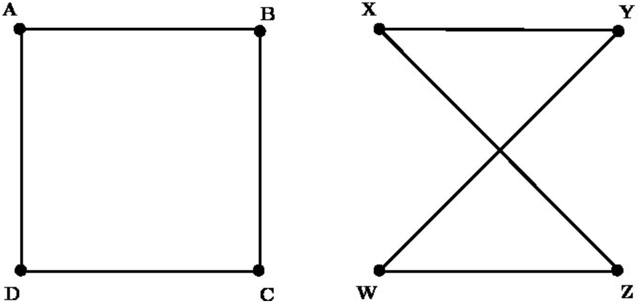
Graph Isomorphism

Two graphs G and H are isomorphic if there is a bijection f between the vertex sets of two graphs with the property that, u and v are adjacent in G if and only if $f(u)$ and $f(v)$ are adjacent in V.

Graph Isomorphism

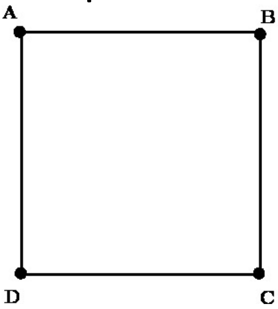
Two graphs G and H are isomorphic if there is a bijection f between the vertex sets of two graphs with the property that, u and v are adjacent in G if and only if $f(u)$ and $f(v)$ are adjacent in V.

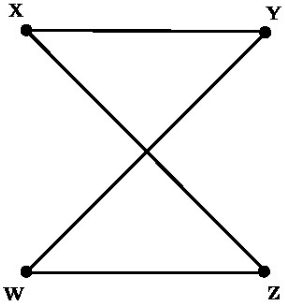
Example 1.



Two graphs G and H are isomorphic if there is a bijection f between the vertex sets of two graphs with the property that, u and v are adjacent in G if and only if $f(u)$ and $f(v)$ are adjacent in V.

Example 1.



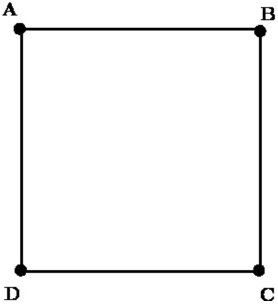


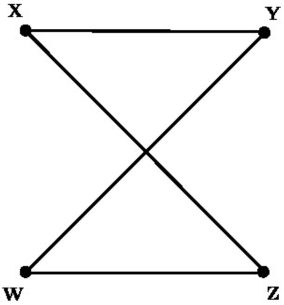
Isomorphism:

Graph Isomorphism

Two graphs G and H are isomorphic if there is a bijection f between the vertex sets of two graphs with the property that, u and v are adjacent in G if and only if $f(u)$ and $f(v)$ are adjacent in V.

Example 1.





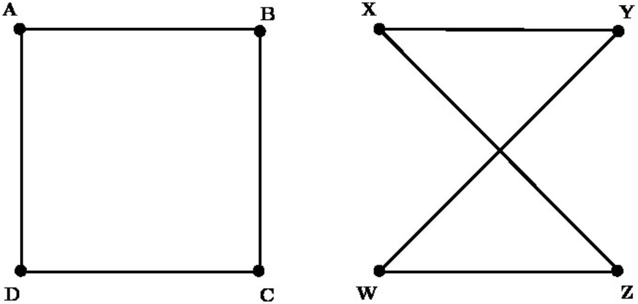
Isomorphism:

$f(A) = X$

Graph Isomorphism

Two graphs G and H are isomorphic if there is a bijection f between the vertex sets of two graphs with the property that, u and v are adjacent in G if and only if $f(u)$ and $f(v)$ are adjacent in V.

Example 1.



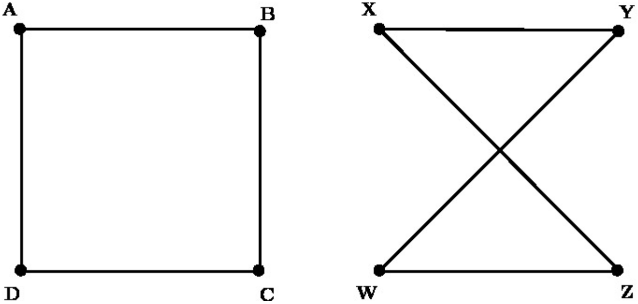
Isomorphism:

$$f(A) = X$$
$$f(B) = Y$$

Graph Isomorphism

Two graphs G and H are isomorphic if there is a bijection f between the vertex sets of two graphs with the property that, u and v are adjacent in G if and only if $f(u)$ and $f(v)$ are adjacent in V.

Example 1.



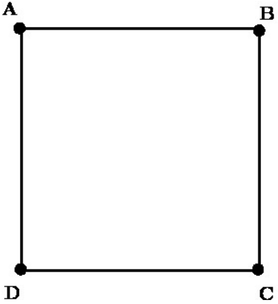
Isomorphism:

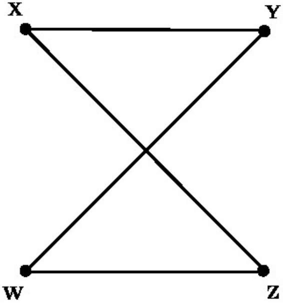
$f(A) = X$
$f(B) = Y$
$f(C) = W$

Graph Isomorphism

Two graphs G and H are isomorphic if there is a bijection f between the vertex sets of two graphs with the property that, u and v are adjacent in G if and only if $f(u)$ and $f(v)$ are adjacent in V.

Example 1.





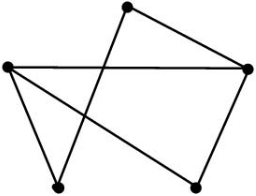
Isomorphism:

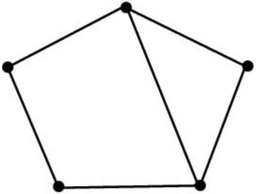
$$f(A) = X$$
$$f(B) = Y$$
$$f(C) = W$$
$$f(D) = Z$$

Example 2.

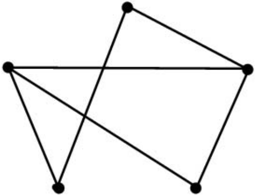
Example 2. Determine if the two graphs are isomorphic:

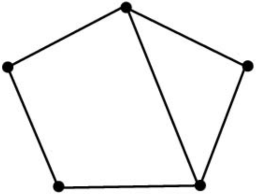
Example 2. Determine if the two graphs are isomorphic:





Example 2. Determine if the two graphs are isomorphic:





Problem.

Example 2. Determine if the two graphs are isomorphic:

Problem. Find an algorithm to check if two graphs are isomorphic.

Graph Invariants

Graph Invariants

If there are no correspondences between two vertex sets preserving the adjacency, then the two graphs are not isomorphic.

Graph Invariants

If there are no correspondences between two vertex sets preserving the adjacency, then the two graphs are not isomorphic.

In some cases, it is easy to conclude that the given graphs are not isomorphic if we can find some properties of the graphs that are not the same.

Graph Invariants

If there are no correspondences between two vertex sets preserving the adjacency, then the two graphs are not isomorphic.

In some cases, it is easy to conclude that the given graphs are not isomorphic if we can find some properties of the graphs that are not the same.

These properties are called graph invariants.

Graph Invariants

If there are no correspondences between two vertex sets preserving the adjacency, then the two graphs are not isomorphic.

In some cases, it is easy to conclude that the given graphs are not isomorphic if we can find some properties of the graphs that are not the same.

These properties are called graph invariants. They can be:

Graph Invariants

If there are no correspondences between two vertex sets preserving the adjacency, then the two graphs are not isomorphic.

In some cases, it is easy to conclude that the given graphs are not isomorphic if we can find some properties of the graphs that are not the same.

These properties are called graph invariants. They can be:

- The number of vertices

Graph Invariants

If there are no correspondences between two vertex sets preserving the adjacency, then the two graphs are not isomorphic.

In some cases, it is easy to conclude that the given graphs are not isomorphic if we can find some properties of the graphs that are not the same.

These properties are called graph invariants. They can be:

- The number of vertices
- The number of edges

Graph Invariants

If there are no correspondences between two vertex sets preserving the adjacency, then the two graphs are not isomorphic.

In some cases, it is easy to conclude that the given graphs are not isomorphic if we can find some properties of the graphs that are not the same.

These properties are called graph invariants. They can be:

- The number of vertices
- The number of edges
- Degrees

Graph Invariants

If there are no correspondences between two vertex sets preserving the adjacency, then the two graphs are not isomorphic.

In some cases, it is easy to conclude that the given graphs are not isomorphic if we can find some properties of the graphs that are not the same.

These properties are called graph invariants. They can be:

- The number of vertices
- The number of edges
- Degrees
- Circuits, paths (to be studied later)

Graph Invariants

If there are no correspondences between two vertex sets preserving the adjacency, then the two graphs are not isomorphic.

In some cases, it is easy to conclude that the given graphs are not isomorphic if we can find some properties of the graphs that are not the same.

These properties are called graph invariants. They can be:

- The number of vertices
- The number of edges
- Degrees
- Circuits, paths (to be studied later)
- ...

Graph Invariants

If there are no correspondences between two vertex sets preserving the adjacency, then the two graphs are not isomorphic.

In some cases, it is easy to conclude that the given graphs are not isomorphic if we can find some properties of the graphs that are not the same.

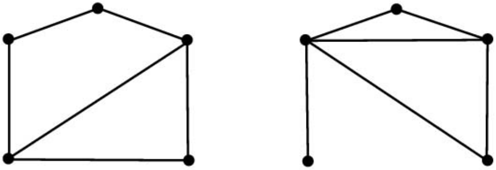
These properties are called graph invariants. They can be:

- The number of vertices
- The number of edges
- Degrees
- Circuits, paths (to be studied later)
- ...

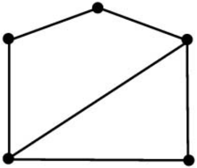
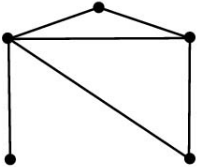
Example.

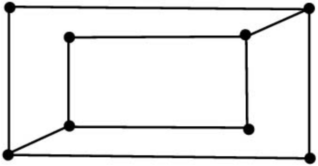
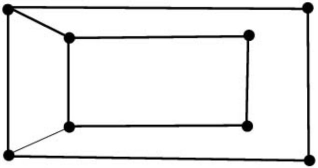
Example. Are the following pairs of graph isomorphic?

Example. Are the following pairs of graph isomorphic?

(a) 

Example. Are the following pairs of graph isomorphic?

(a)  

(b)  

9.4 Connectivity

- Let G be an undirected graph. A path of length n from u to v is a sequence of edges $x_0x_1, x_1x_2, \ldots, x_{n-1}x_n$, in which x_i and x_j are not necessarily distinct, and $x_0 = u$, $x_n = v$.

9.4 Connectivity

- Let G be an undirected graph. A path of length n from u to v is a sequence of edges $x_0x_1, x_1x_2, \ldots, x_{n-1}x_n$, in which x_i and x_j are not necessarily distinct, and $x_0 = u$, $x_n = v$.
- A path is called a circuit if it starts and ends at the same vertex, and has length greater than 0.

9.4 Connectivity

- Let G be an undirected graph. A path of length n from u to v is a sequence of edges $x_0x_1, x_1x_2, \ldots, x_{n-1}x_n$, in which x_i and x_j are not necessarily distinct, and $x_0 = u$, $x_n = v$.
- A path is called a circuit if it starts and ends at the same vertex, and has length greater than 0.
- A path, or a circuit, is simple if it does not contain the same edge more than once.

9.4 Connectivity

- Let G be an undirected graph. A path of length n from u to v is a sequence of edges $x_0x_1, x_1x_2, \ldots, x_{n-1}x_n$, in which x_i and x_j are not necessarily distinct, and $x_0 = u$, $x_n = v$.
- A path is called a circuit if it starts and ends at the same vertex, and has length greater than 0.
- A path, or a circuit, is simple if it does not contain the same edge more than once.

Connectedness in Undirected Graphs

Connectedness in Undirected Graphs

- An undirected graph is connected if there is a path between any pair of distinct vertices.

Connectedness in Undirected Graphs

- An undirected graph is connected if there is a path between any pair of distinct vertices.
- A connected component of an undirected graph is a maximal subgraph that is connected.

Connectedness in Undirected Graphs

- An undirected graph is connected if there is a path between any pair of distinct vertices.
- A connected component of an undirected graph is a maximal subgraph that is connected.
- A cut vertex, or an articulation point, is a vertex that if we remove it and all the edges incident with it we will obtain a subgraph having more connected components than the original graph.

Connectedness in Undirected Graphs

- An undirected graph is connected if there is a path between any pair of distinct vertices.
- A connected component of an undirected graph is a maximal subgraph that is connected.
- A cut vertex, or an articulation point, is a vertex that if we remove it and all the edges incident with it we will obtain a subgraph having more connected components than the original graph.
- A cut edge, or a bridge, is an edge that if we remove it we will obtain a subgraph having more connected components than the original graph.

Connectedness in Undirected Graphs

- An undirected graph is connected if there is a path between any pair of distinct vertices.
- A connected component of an undirected graph is a maximal subgraph that is connected.
- A cut vertex, or an articulation point, is a vertex that if we remove it and all the edges incident with it we will obtain a subgraph having more connected components than the original graph.
- A cut edge, or a bridge, is an edge that if we remove it we will obtain a subgraph having more connected components than the original graph.

Connectedness in Directed Graphs

Connectedness in Directed Graphs

- A directed graph is strongly connected if for all pairs of vertices u and v there is a path from u to v and vice versa.

Connectedness in Directed Graphs

- A directed graph is strongly connected if for all pairs of vertices u and v there is a path from u to v and vice versa. A directed graph is weakly connected if the underlying undirected graph is connected.

Connectedness in Directed Graphs

- A directed graph is strongly connected if for all pairs of vertices u and v there is a path from u to v and vice versa. A directed graph is weakly connected if the underlying undirected graph is connected.
- A strongly connected component of a directed graph G is a maximal subgraph of G that is strongly connected.

Connectedness in Directed Graphs

- A directed graph is strongly connected if for all pairs of vertices u and v there is a path from u to v and vice versa. A directed graph is weakly connected if the underlying undirected graph is connected.
- A strongly connected component of a directed graph G is a maximal subgraph of G that is strongly connected.

Connectedness in Directed Graphs

- A directed graph is strongly connected if for all pairs of vertices u and v there is a path from u to v and vice versa. A directed graph is weakly connected if the underlying undirected graph is connected.
- A strongly connected component of a directed graph G is a maximal subgraph of G that is strongly connected.

Example.

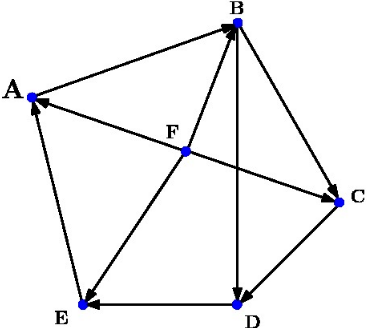
- A directed graph is strongly connected if for all pairs of vertices u and v there is a path from u to v and vice versa. A directed graph is weakly connected if the underlying undirected graph is connected.
- A strongly connected component of a directed graph G is a maximal subgraph of G that is strongly connected.

Example. Given the graph

Connectedness in Directed Graphs

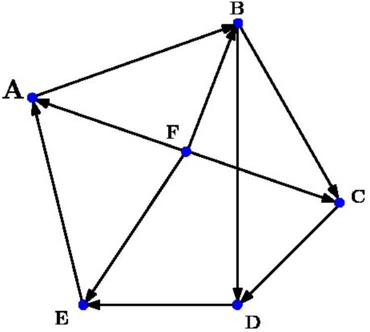
- A directed graph is strongly connected if for all pairs of vertices u and v there is a path from u to v and vice versa. A directed graph is weakly connected if the underlying undirected graph is connected.
- A strongly connected component of a directed graph G is a maximal subgraph of G that is strongly connected.

Example. Given the graph



- A directed graph is strongly connected if for all pairs of vertices u and v there is a path from u to v and vice versa. A directed graph is weakly connected if the underlying undirected graph is connected.
- A strongly connected component of a directed graph G is a maximal subgraph of G that is strongly connected.

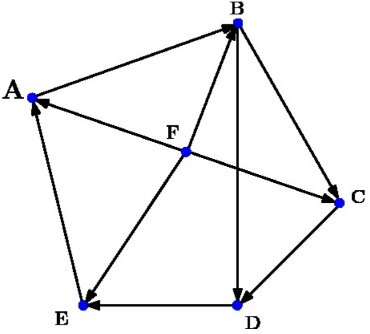
Example. Given the graph



Determine if the graph is strongly connected, weakly connected, and find the number of strongly connected components.

- A directed graph is strongly connected if for all pairs of vertices u and v there is a path from u to v and vice versa. A directed graph is weakly connected if the underlying undirected graph is connected.
- A strongly connected component of a directed graph G is a maximal subgraph of G that is strongly connected.

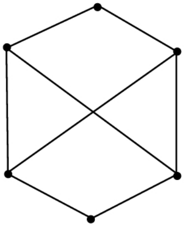
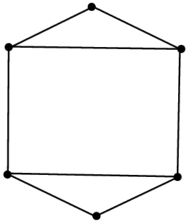
Example. Given the graph



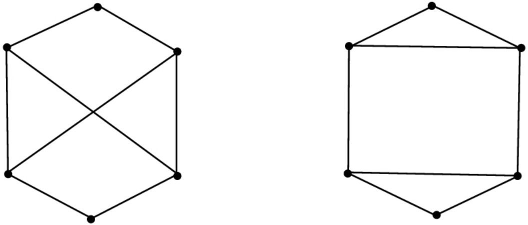
Determine if the graph is strongly connected, weakly connected, and find the number of strongly connected components.

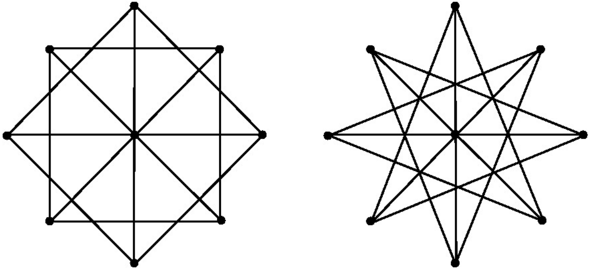
Use graph invariants of paths and circuits to check if the two graphs are isomorphic:

Use graph invariants of paths and circuits to check if the two graphs are isomorphic:

(a)  

Use graph invariants of paths and circuits to check if the two graphs are isomorphic:

(a) 

(b) 

Counting Paths Between Vertices

Counting Paths Between Vertices

Counting Paths Between Vertices

Let G be a graph (undirected, directed) whose adjacency matrix with respect to the ordering of vertices $v_1, v_2, , \ldots, v_n$ is A.

Counting Paths Between Vertices

Let G be a graph (undirected, directed) whose adjacency matrix with respect to the ordering of vertices $v_1, v_2, , \ldots, v_n$ is A. The number of paths of length r from v_i to v_j is the (i, j)-entry of the matrix A^r.

Counting Paths Between Vertices

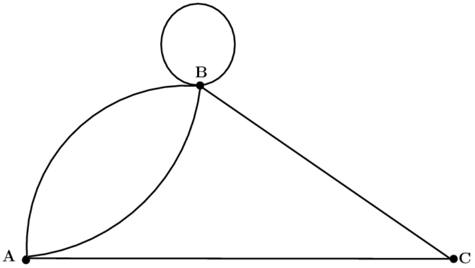
Let G be a graph (undirected, directed) whose adjacency matrix with respect to the ordering of vertices $v_1, v_2, , \ldots, v_n$ is A. The number of paths of length r from v_i to v_j is the (i, j)-entry of the matrix A^r.

Example. Count the number of paths of length 3 between A and C in the graph

Counting Paths Between Vertices

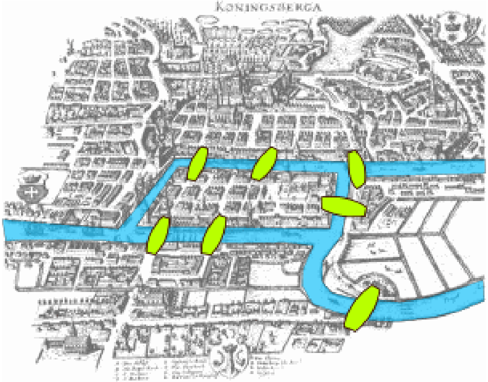
Let G be a graph (undirected, directed) whose adjacency matrix with respect to the ordering of vertices $v_1, v_2, , \ldots, v_n$ is A. The number of paths of length r from v_i to v_j is the (i, j)-entry of the matrix A^r.

Example. Count the number of paths of length 3 between A and C in the graph

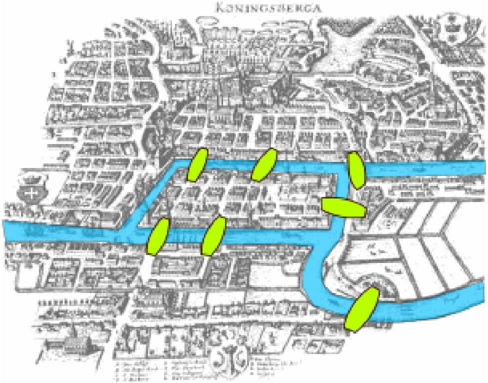


The 7 bridges problem.

The 7 bridges problem.



The 7 bridges problem.



Question. Is this possible to start at some location, travel across all bridges without crossing any bridge twice, then return to the starting point?

Euler Paths and Circuits

Euler Paths and Circuits

Euler Paths and Circuits

A simple circuit containing all edges of a graph is called Euler circuit.

Euler Paths and Circuits

A simple circuit containing all edges of a graph is called Euler circuit. A simple path containing all edges of a graph is called Euler path.

Euler Paths and Circuits

A simple circuit containing all edges of a graph is called Euler circuit. A simple path containing all edges of a graph is called Euler path.

Conditions for existence of Euler paths and circuits

Euler Paths and Circuits

A simple circuit containing all edges of a graph is called Euler circuit. A simple path containing all edges of a graph is called Euler path.

Conditions for existence of Euler paths and circuits

- A connected multigraph G has Euler circuits if and only if every vertex has even degree.

Euler Paths and Circuits

A simple circuit containing all edges of a graph is called Euler circuit. A simple path containing all edges of a graph is called Euler path.

Conditions for existence of Euler paths and circuits

- A connected multigraph G has Euler circuits if and only if every vertex has even degree.
- If G does not have Euler circuits, then it has Euler paths if and only if it has exactly two vertices of odd degrees.

Euler Paths and Circuits

A simple circuit containing all edges of a graph is called Euler circuit. A simple path containing all edges of a graph is called Euler path.

Conditions for existence of Euler paths and circuits

- A connected multigraph G has Euler circuits if and only if every vertex has even degree.
- If G does not have Euler circuits, then it has Euler paths if and only if it has exactly two vertices of odd degrees.

Example. The answer to the 7 bridges problem is NO.

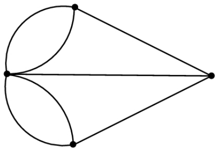
Euler Paths and Circuits

A simple circuit containing all edges of a graph is called Euler circuit. A simple path containing all edges of a graph is called Euler path.

Conditions for existence of Euler paths and circuits

- A connected multigraph G has Euler circuits if and only if every vertex has even degree.
- If G does not have Euler circuits, then it has Euler paths if and only if it has exactly two vertices of odd degrees.

Example. The answer to the 7 bridges problem is NO.



Question.

Question. For what values of m, n do the special graphs $K_n, C_n, W_n, Q_n, K_{mn}$ has Euler paths/circuits?

Question. For what values of m, n do the special graphs $K_n, C_n, W_n, Q_n, K_{mn}$ has Euler paths/circuits?

Problem 1. Find an algorithm to find Euler circuits/paths.

Question. For what values of m, n do the special graphs $K_n, C_n, W_n, Q_n, K_{mn}$ has Euler paths/circuits?

Problem 1. Find an algorithm to find Euler circuits/paths.

Problem 2. Find conditions for the existence of Euler paths/circuits in directed graphs.

Question. For what values of m, n do the special graphs $K_n, C_n, W_n, Q_n, K_{mn}$ has Euler paths/circuits?

Problem 1. Find an algorithm to find Euler circuits/paths.

Problem 2. Find conditions for the existence of Euler paths/circuits in directed graphs.

Hamilton Paths and Circuits

Hamilton Paths and Circuits

Hamilton Paths and Circuits

- A simple path that passes through all vertices exactly once is called Hamilton path.

Hamilton Paths and Circuits

- A simple path that passes through all vertices exactly once is called Hamilton path.
- A simple circuit that passes through all vertices exactly once is called Hamilton circuit.

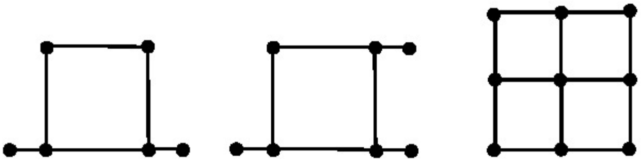
Hamilton Paths and Circuits

- A simple path that passes through all vertices exactly once is called Hamilton path.
- A simple circuit that passes through all vertices exactly once is called Hamilton circuit.

Example. Find Hamilton paths/circuits of the graphs:

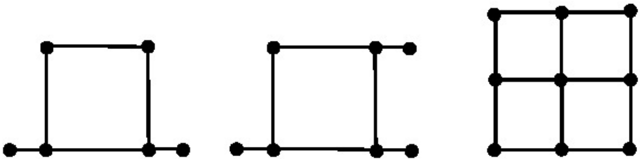
- A simple path that passes through all vertices exactly once is called Hamilton path.
- A simple circuit that passes through all vertices exactly once is called Hamilton circuit.

Example. Find Hamilton paths/circuits of the graphs:



- A simple path that passes through all vertices exactly once is called Hamilton path.
- A simple circuit that passes through all vertices exactly once is called Hamilton circuit.

Example. Find Hamilton paths/circuits of the graphs:



Question. For what values of m, n do the special graphs $K_n, C_n, W_n, Q_n, K_{mn}$ has Hamilton paths/circuits?

9.6 Shortest-Path Problem

9.6 Shortest-Path Problem

9.6 Shortest-Path Problem

A graph that has a number assigned to each edge is called a weighted graph.

9.6 Shortest-Path Problem

A graph that has a number assigned to each edge is called a weighted graph. The length of a path in a weighted graph is the sum of all weights of the edges of this path.

A graph that has a number assigned to each edge is called a weighted graph. The length of a path in a weighted graph is the sum of all weights of the edges of this path.

Problem.

9.6 Shortest-Path Problem

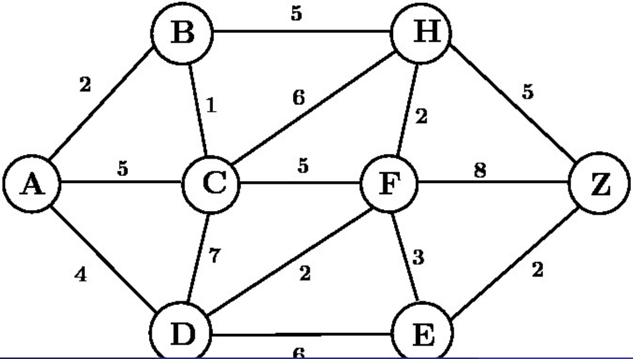
A graph that has a number assigned to each edge is called a weighted graph. The length of a path in a weighted graph is the sum of all weights of the edges of this path.

Problem. Find the path of shortest length between two vertices in a weighted graph.

9.6 Shortest-Path Problem

A graph that has a number assigned to each edge is called a weighted graph. The length of a path in a weighted graph is the sum of all weights of the edges of this path.

Problem. Find the path of shortest length between two vertices in a weighted graph.



Dijkstra's Algorithm

Dijkstra's Algorithm

Let G be a weighted graph.

Dijkstra's Algorithm

Let G be a weighted graph. To find the shortest path between A and Z in G, Dijkstra's algorithm:

Dijkstra's Algorithm

Let G be a weighted graph. To find the shortest path between A and Z in G, Dijkstra's algorithm:

- Finds the length of the shortest path from A to the first vertex.

Dijkstra's Algorithm

Let G be a weighted graph. To find the shortest path between A and Z in G, Dijkstra's algorithm:

- Finds the length of the shortest path from A to the first vertex.
- Finds the length of the shortest path from A to the second vertex.

Dijkstra's Algorithm

Let G be a weighted graph. To find the shortest path between A and Z in G, Dijkstra's algorithm:

- Finds the length of the shortest path from A to the first vertex.
- Finds the length of the shortest path from A to the second vertex.
- Finds the length of the shortest path from A to the third vertex.

Dijkstra's Algorithm

Let G be a weighted graph. To find the shortest path between A and Z in G, Dijkstra's algorithm:

- Finds the length of the shortest path from A to the first vertex.
- Finds the length of the shortest path from A to the second vertex.
- Finds the length of the shortest path from A to the third vertex.
- ...

Dijkstra's Algorithm

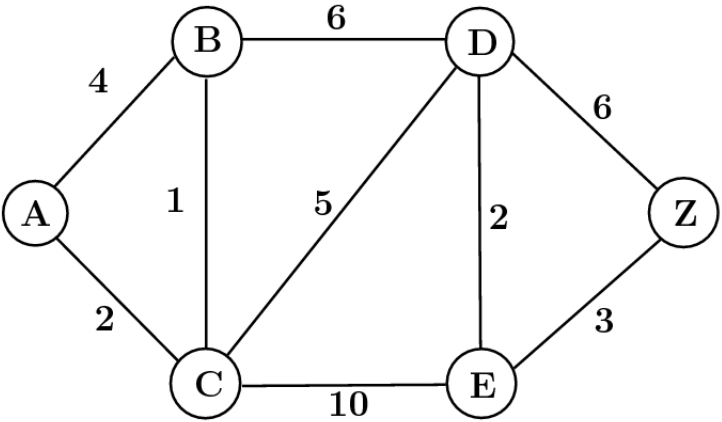
Let G be a weighted graph. To find the shortest path between A and Z in G, Dijkstra's algorithm:

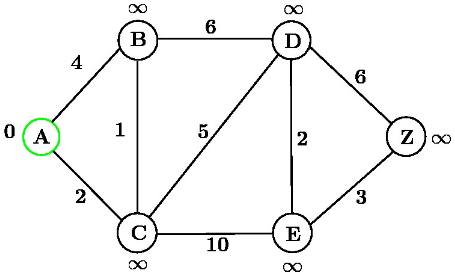
- Finds the length of the shortest path from A to the first vertex.
- Finds the length of the shortest path from A to the second vertex.
- Finds the length of the shortest path from A to the third vertex.
- ...
- Continue the process until Z is reached.

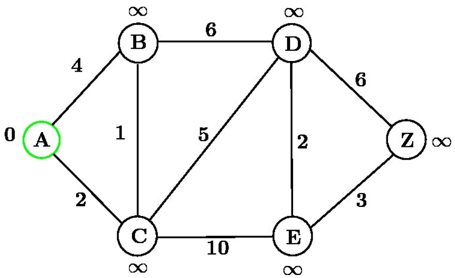
Example.

Example. Find the shortest path from A to Z in the weighted graph

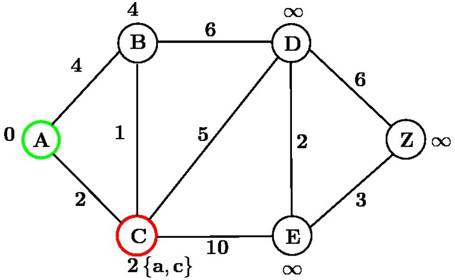
Example. Find the shortest path from A to Z in the weighted graph



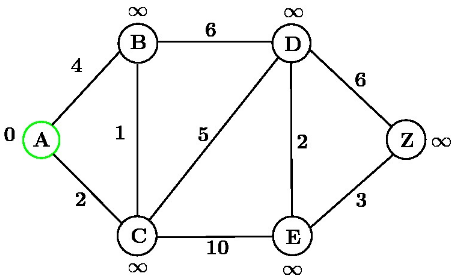

∞
6
∞
B
D
4
6
0
A
1
5
2
Z
∞
2
3
C
10
E
∞
∞

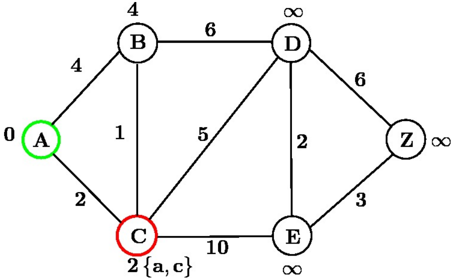


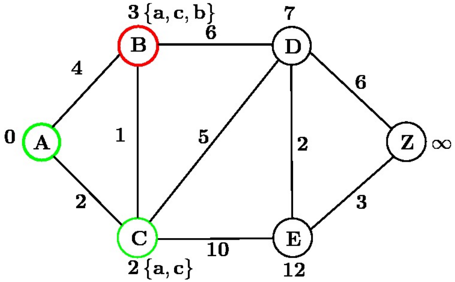
∞
∞
6
B
D
4
6
0
A
1
5
2
Z
∞
2
3
C
10
E
∞
∞

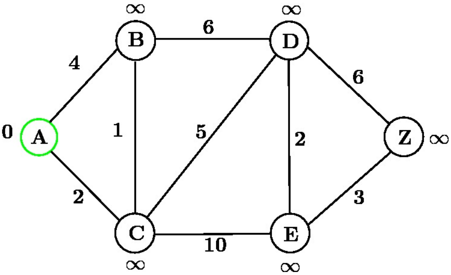


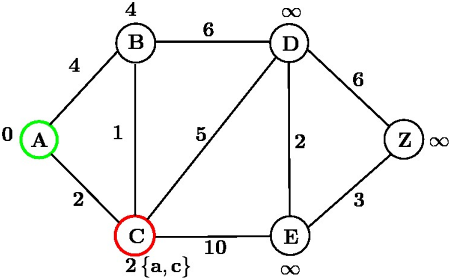
4
∞
6
B
D
4
6
0
A
1
5
2
Z
∞
2
3
C
10
E
2 {a, c}
∞

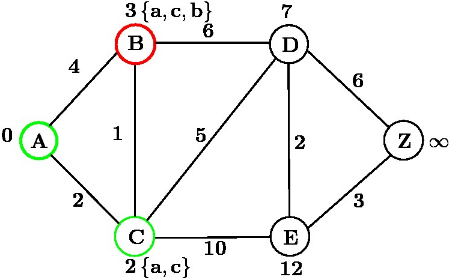

∞
∞
B
D
6
4
6
0
A
1
5
2
Z
∞
2
3
C
10
E
∞
∞

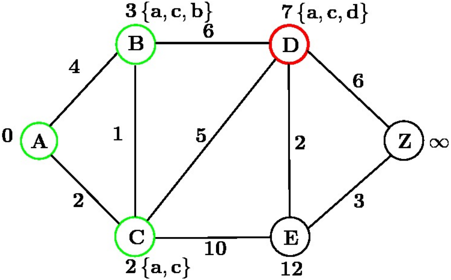

4
∞
B
D
6
4
6
0
A
1
5
2
Z
∞
2
3
C
10
E
2 {a, c}
∞

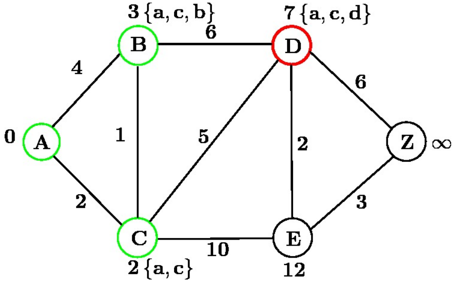

3 {a, c, b}
7
B
6
D
4
6
0
A
1
5
2
Z
∞
2
3
C
10
E
2 {a, c}
12

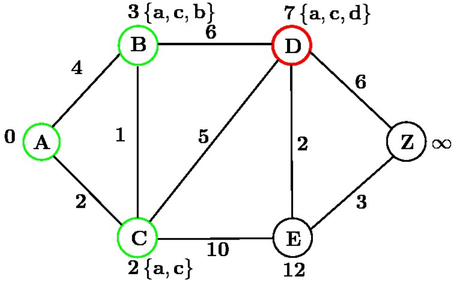

∞
B
6
∞
D
4
6
0
A
1
5
2
Z
∞
2
3
C
10
E
∞
∞

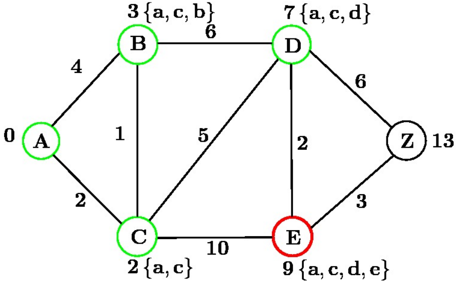

4
B
6
∞
D
4
6
0
A
1
5
2
Z
∞
2
3
C
10
E
2 {a, c}
∞

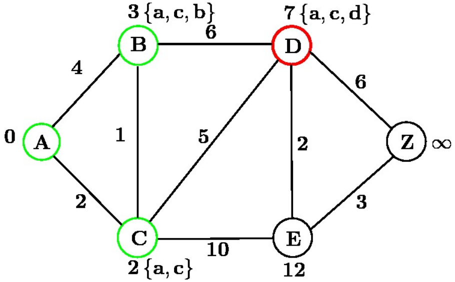

3 {a, c, b}
B
6
7
D
4
6
0
A
1
5
2
Z
∞
2
3
C
10
E
2 {a, c}
12

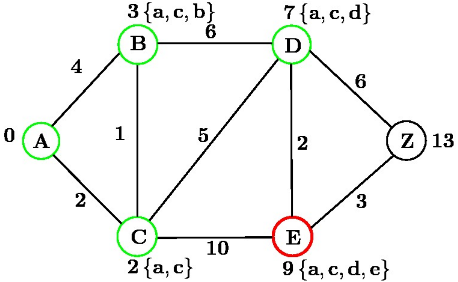

3 {a, c, b}
B
6
7 {a, c, d}
D
4
6
0
A
1
5
2
Z
∞
2
3
C
10
E
2 {a, c}
12

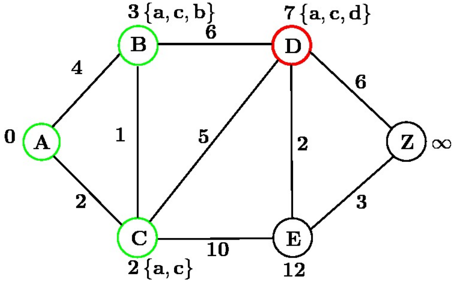

3 {a, c, b}
7 {a, c, d}
B
6
D
4
6
0
A
1
5
2
Z
∞
2
3
C
10
E
2 {a, c}
12


3 {a, c, b}
7 {a, c, d}
B
D
6
4
6
0
A
1
5
2
Z
∞
2
3
C
E
10
2 {a, c}
12

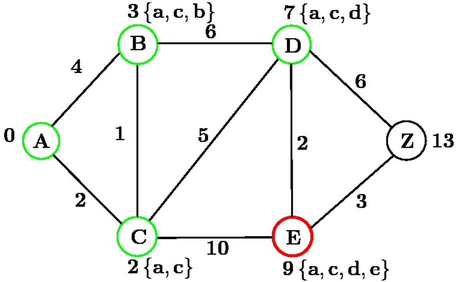

3 {a, c, b}
7 {a, c, d}
B
D
6
4
6
0
A
1
5
2
Z
13
2
3
C
E
10
2 {a, c}
9 {a, c, d, e}


3 {a, c, b}
7 {a, c, d}
B
D
6
4
6
0
A
1
5
2
Z
∞
2
3
C
E
10
2 {a, c}
12

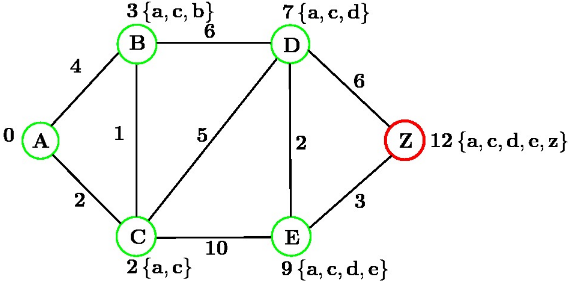

3 {a, c, b}
7 {a, c, d}
B
D
6
4
6
0
A
1
5
2
Z
13
2
3
C
E
10
2 {a, c}
9 {a, c, d, e}



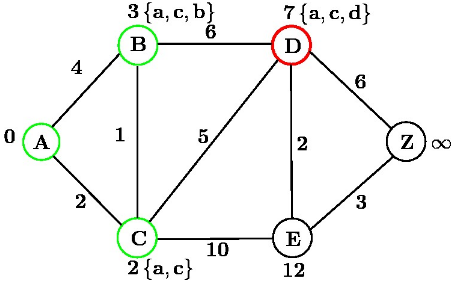
3 {a, c, b}
7 {a, c, d}
B
D
6
4
6
0
A
1
5
2
Z
∞
2
3
C
E
10
2 {a, c}
12

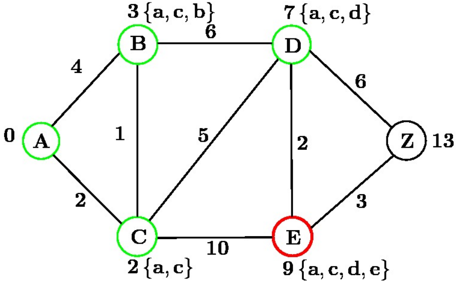


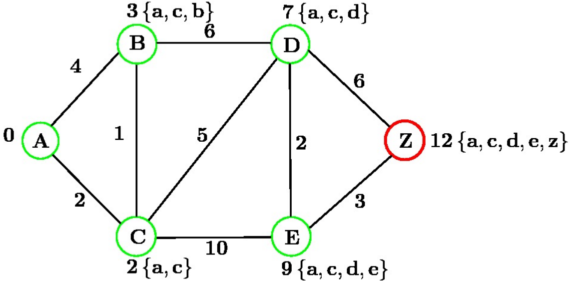
3 {a, c, b}
7 {a, c, d}
B
D
6
4
6
0
A
1
5
2
Z
13
2
3
C
E
10
2 {a, c}
9 {a, c, d, e}



3 {a, c, b}
7 {a, c, d}
B
D
6
4
6
0
A
1
5
2
Z
12 {a, c, d, e, z}
2
3
C
E
10
2 {a, c}
9 {a, c, d, e}


3 {a, c, b}
7 {a, c, d}
B
D
6
4
6
0
A
1
5
2
Z
∞
2
3
C
E
10
2 {a, c}
12


3 {a, c, b}
7 {a, c, d}
B
D
6
4
6
0
A
1
5
2
Z
13
2
3
C
E
10
2 {a, c}
9 {a, c, d, e}


3 {a, c, b}
7 {a, c, d}
B
D
6
4
6
0
A
1
5
2
Z
12 {a, c, d, e, z}
2
3
C
E
10
2 {a, c}
9 {a, c, d, e}

Procedure Dijkstra(G: weighted connected simple graph with n vertices $v_1, v_2, \ldots, v_n$)

for $i := 1$ **to** n

  $L(v_i) := \infty$

$L(A) := 0$

$S := \emptyset$

while $Z \notin S$

begin

  $u :=$ vertex not in S with minimum label

  $S := S \cup \{u\}$

  for all vertices v not in S

    $L(v) := \min\{L(v), L(u) + distance(u, v)\}$

end