Bikash Balami Graph Theory

[Graph Theory]

Discrete Structure

Compiled By

Bikash Balami

Deerwalk Institute Of Technology

Tribhuvan University

Kathmandu, Nepal

1

Bikash Balami Graph Theory

Graphs

Graph is a discrete structure consisting of **vertices** and **edges** that connect these vertices. Problems in almost every conceivable discipline can be solved using graph model. Graph was introduced in eighteenth century by the great Swiss mathematician Leanhard Euler. Graphs are used to solve many problems in many fields. For example,

- Graphs can be used to determine whether a circuit can be implemented on a planar circuit board.
- Graphs can be used to distinguish two chemical compounds with the same molecular formula but different structures.
- Graphs can be used to study the structure of the WWW (World Wide Web).
- Graph model of computer network can be used to determine whether two computers are connected by a communication link.
- Graphs with weights assigned to their edges can be used to solve problems such as finding the shortest path between two cities in a transportation network.
- Graphs can also be used to schedule exams and assign channels to television stations.

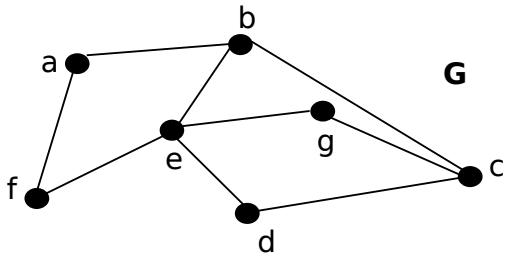
Types of graphs

Simple Graph

We define a simple graph as 2 – tuple consists of a non empty set of vertices V and a set of unordered pairs of distinct elements of vertices called edges. We can represent graph as G = (V, E). A simple graph G = (V, E) consists of V, a nonempty set of vertices, and E, a set of unordered pairs of distinct elements of V called edges. This kind of graph has undirected edges, no loops and no multiple edges. The figure below is an example of simple graph.

2

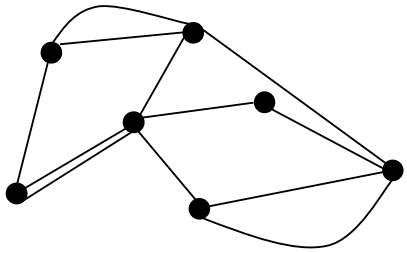
Bikash Balami Graph Theory



In the above graph G = (V, E), the set of vertices, V (G) or V = {a, b, c, d, e, f, g} and the set of edges E (G) or E = {{a, b}, {a, f}, {b, c}, {b, e}, {c, d}, {c, g}, {d, e}, {e, g},{e, f}}.

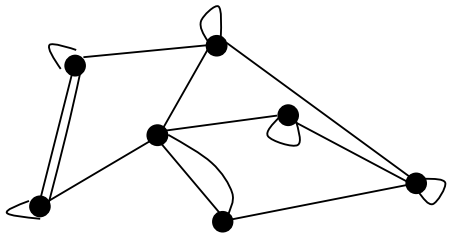
Multigraph

A multigraph G = (V, E) consists of a set of vertices V, a set of edges E, and a function f from E to {{u, v} | u, v ∈ V, u ≠ v}. The edges *e1* and *e2* are called multiple or parallel edges if *f(e1) = f(e2)*. This type of graph can have multiple edges between the same two vertices. This kind of graph has undirected edges, and no loops. Since simple graph has single edge, every simple graph is a multigraph. The figure below is an example of a multigraph.



Pseudograph

A pseudograph G = (V, E) consists of a set of vertices V, a set of edges E, and a function f from E to {{u, v}|u, v ∈ V}. An edge is a loop if *f(e) = {u, u} = {u}* for some u ∈ V. The figure below is an example of a pseudograph.

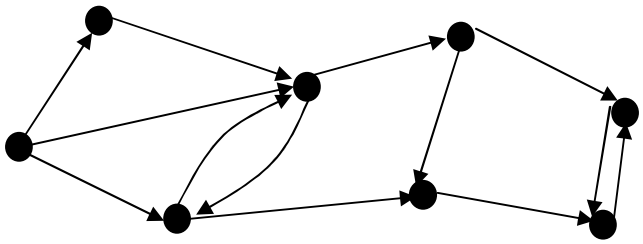


3

Bikash Balami Graph Theory

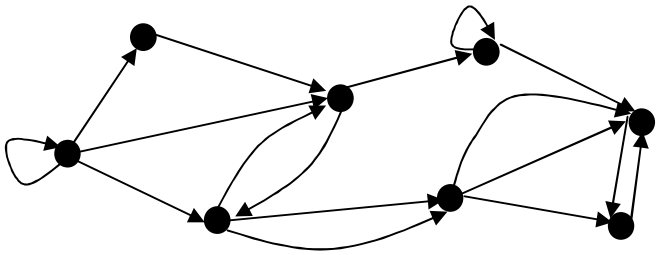
Directed Graph

A directed graph (V, E) consists of a set V of vertices, a set E of edges that are ordered pairs of elements of V. The below figure is a directed graph. In this graph loop is allowed but no two vertices can have multiple edges in same direction.



Directed Multigraph

A directed multigraph G = (V, E) consists of a set of vertices V, a set of edges E, and a function f from E to {(u, v)|u, v ∈ V}. The edges *e1* and *e2* are called multiple edges if *f(e1) = f(e2)*. The figure below is an example of a directed multigraph.



Graph Terminologies

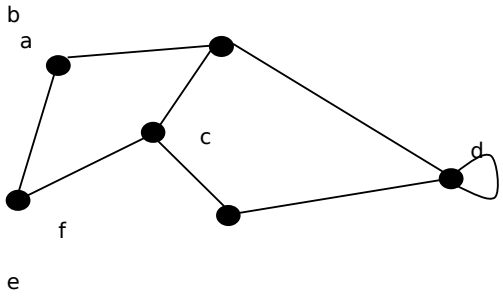
Two vertices u, v in an undirected graph G are called ***adjacent*** or ***neighbors*** in if {u, v} is an edge. The edge e is called ***incident with*** the vertices u and v if e = {u, v}. This edge is also said to connect u and v, where u and v are end points of the edge.

The ***degree of a vertex*** in an undirected graph is the number of edges incident with it, except a loop at a vertex. Loop in a vertex counts twice to the degree. Degree of a vertex v is denoted by deg (v). A vertex of degree zero is called ***isolated*** vertex and a vertex with degree one is called ***pendant*** vertex.

4

Bikash Balami Graph Theory

Example: Find the degrees of the vertices in the following graph.



Solution:

deg(a) = deg(f) = deg(e) = 2 ; deg(b) = deg(c) = 3; deg(d) = 4

Theorem 1: The Handshaking Theorem

Let G = (V, E) be an undirected graph with e edges. Then $2e = \sum_{v \in V} deg(v)$

Theorem 2:

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

Proof:

Take two sets of vertices, V1, a set of vertices with even degree, and V2, a set of vertices with odd degree. In an undirected graph G = (V, E) we have

$2e = \sum_{v \in V} deg(v) = \sum_{v \in V1} deg(v) + \sum_{v \in V2} deg(v)$

From the equality above we can say the left part is even i.e. 2e is even, the sum of deg(v) for $v \in$ V1 is even since every vertices has even degree. So for the left hand to be even sum of deg(v) for $v \in$ V2 must be even. Since all the vertices in the set V2 have odd degree the number of such vertices must be even for the sum to be even. Hence proved.

5

Bikash Balami Graph Theory

Other Terminologies

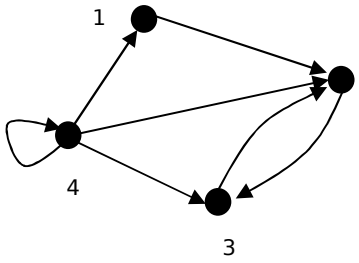
Let (u, v) be an edge representing edge of a directed graph G, *u* is called adjacent to *v* and *v* is called adjacent from *u*. The vertex *u* is called ***initial*** vertex and the vertex *v* is called ***terminal*** or ***end*** vertex. Loop has same initial and terminal vertex.

In directed graph the ***in-degree*** of a vertex *v*, denoted by $\deg^-(v)$, is the number of edges that have *v* as their terminal vertex. The ***out-degree*** of a vertex *v*, denoted by $\deg^+(v)$, is the number of edges that have v as their initial vertex. Loop at a vertex adds up both in-degree and outdegree to one more than calculated in-degree and out-degree.

Example:

Find the in-degree and out-degree of each vertex in the following graph

2



Solution:

In-degrees of a graph are $\deg^-(1) = \deg^-(4) = 1$; $\deg^-(2) = 3$; $\deg^-(3) = 2$ and the out-degrees of a graph are $\deg^+(1) = \deg^+(2) = \deg^+(3) = 1$; $\deg^+(4) = 4$

Walk

A **walk** is an alternating sequence of vertices and edges, beginning and ending with a vertex, where each vertex is incident to both the edge that precedes it and the edge that follows it in the sequence, and where the vertices that precede and follow an edge are the end vertices of that edge. A walk is **closed** if its first and last vertices are the same, and **open** if they are different.

6

Bikash Balami Graph Theory

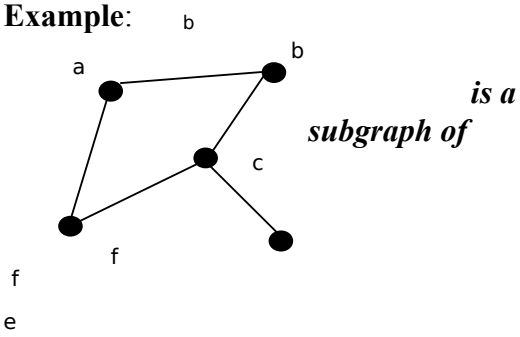
An open walk is called a **path**.

Theorem 3:

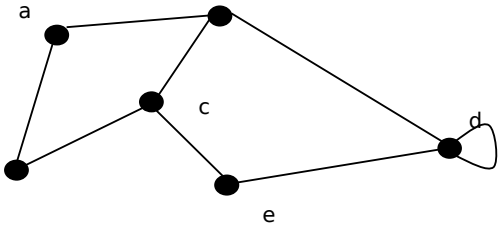
Let G(V, E) be a graph with directed edges. Then $\sum_{v \in V} \deg^-(v) = \sum_{v \in V} \deg^+(v) = |E|$.

A ***subgraph*** of a graph G = (V, E) is a graph H = (W, F) where W ⊆ V and F ⊆ E.

Example:



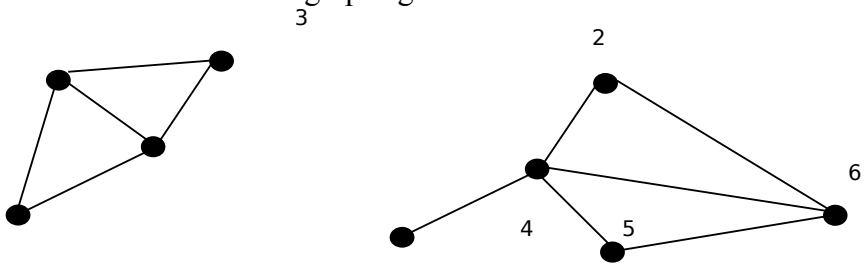
is a subgraph of



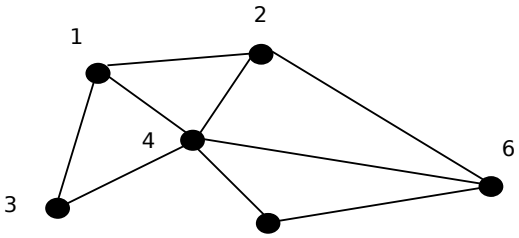
The union of two simple graphs G1 = (V1, E1) and G2 = (V2, E2) is the simple graph with vertex set V1∪V2 and the edge set E1∪E2. The union of G1 and G2 is denoted by G1 ∪ G2.

Example:

Find the union of two graphs given below



Solution



7

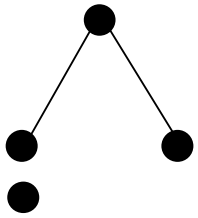
Complete Graphs

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

Example:

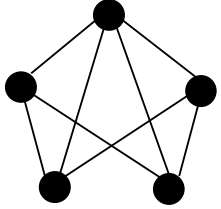
What are K_1, K_3, and K_5?

Solution:



K_1 K_3 K_5

Bikash Balami Graph Theory

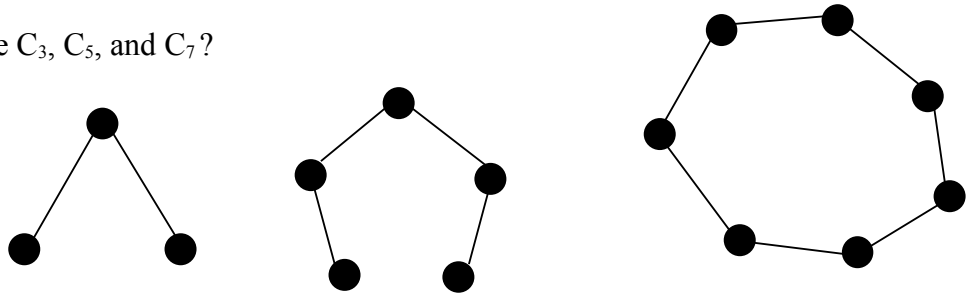


Cycles

The cycle C_n, $n \geq 3$, consists of n vertices v_1, v_2, …, v_n and edges $\{v_1, v_2\}, \{v_2, v_3\}$, …, $\{v_{n-1}, v_n\}$, and $\{v_n, v_1\}$.

Example:

What are C_3, C_5, and C_7?



Solution

C

C

C3 5 7

Wheels

The wheel W_n, for $n \geq 3$, is an union of C_n and additional vertex where the new vertex is connected by each vertex of the cycle.

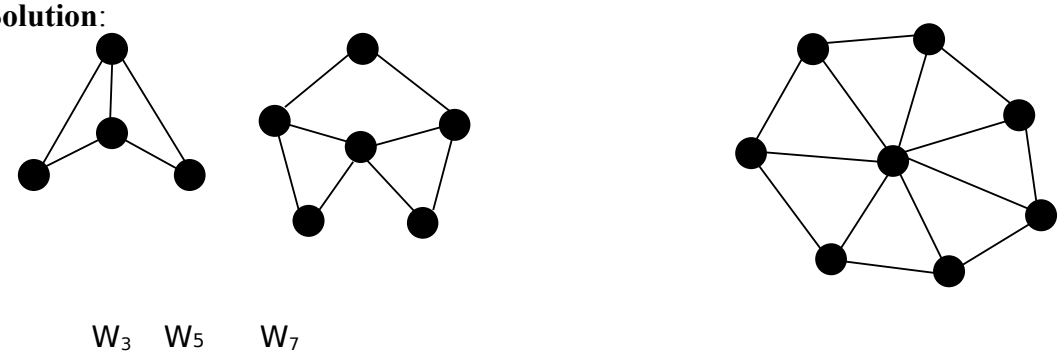
Example:

9

Bikash Balami Graph Theory

What are W_3, W_5, and W_7?

Solution:



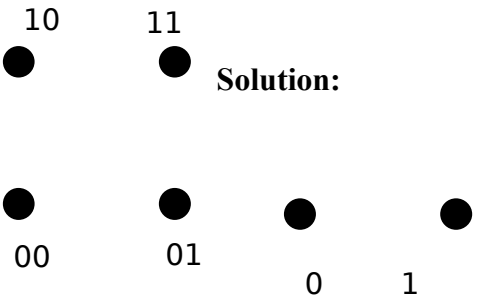
W_3 W_5 W_7

n- Cubes

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

Example:

What are Q_1, Q_2, and Q_3 ? 110 111

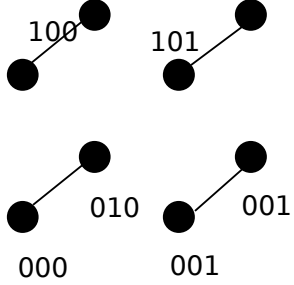


Solution:

Q_1

Q_2

Bikash Balami Graph Theory



Q_3

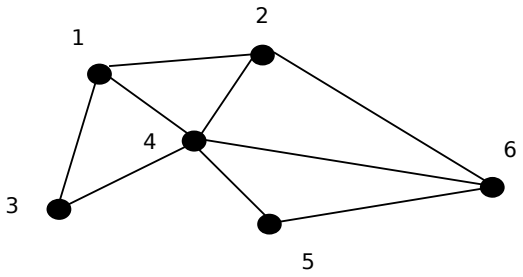
Bipartite Graphs

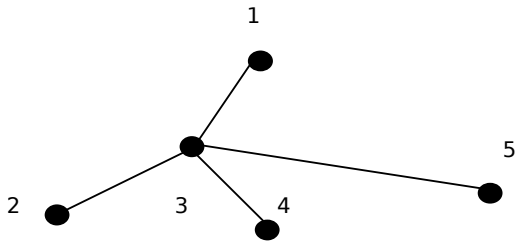
A simple graph G is bipartite if its vertex set V can be partitioned into two disjoint subsets V_1 and V_2 such that every edge in the graph connects a vertex from the set V_1 to the vertex of the set V_2. No two vertices of the same set are connected by an edge. For example, C_6 is bipartite but K_3 is not.

A graph is bipartite if and only if it is possible to color the vertices of the graph with at most two colors such that no two adjacent vertices have the same color.

A graph is bipartite if and only if it is not possible to start at a vertex and return to this vertex by traversing an odd number of distinct edges.

Q. Are the graphs below bipartite





Complete Bipartite Graphs

The complete bipartite graph $K_{m,n}$ is the graph where the vertex set is partitioned into two subsets of m and n vertices, respectively. In this graph there is an edge between two vertices if and only if two vertices are in different subsets of vertices.

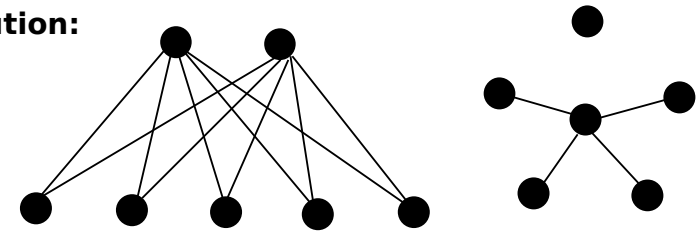
Example:

Sketch $K_{2,5}$, $K_{1,5}$.

11

Bikash Balami Graph Theory

Solution:



$K_{2,5}$ $K_{1,5}$

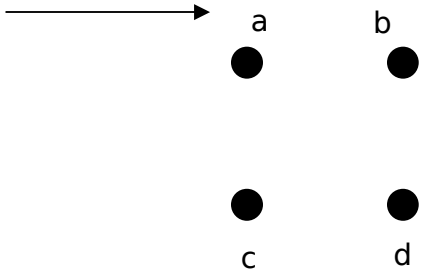
Graph Representations

Graph can be represented in many ways. One of the ways of representing a graph without multiple edges is by listing its edges. Some other ways are described below:

Adjacency List

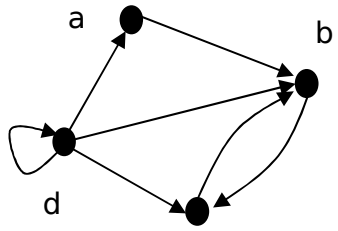
This type of representation is suitable for the undirected graphs without multiple edges, and directed graphs. This representation looks as in the tables below.

Edge List for Simple Graph	
Vertex	Adjacent Vertices
a	b, c
b	a, d
c	a, d
d	b, c



12

Bikash Balami Graph Theory



Edge List for Directed Graph	
Initial Vertex	End Vertices
a	B
b	C
c	B
d	a, b, c, d

c

If we try to apply the algorithms of graph using the representation of graphs by lists of edges, or adjacency lists it can be tedious and time taking if there are high numbers of edges. For the sake of the computation, the graphs with many edges can be represented in other ways. Here we discuss two ways of representing graphs in form of matrix.

Adjacency Matrix

Given a simple graph G = (V, E) with |V| = n. Assume that the vertices of the graph are listed in some arbitrary order like v1, v2, …, vn. The adjacency matrix A of G, with respect to the order of the vertices is n-by-n zero-one matrix (A = [aij]) with the condition,

1 if vi and vj are adjacent
aij =
0 otherwise

Since there are n vertices and we may order vertices in any order there are n! possible order of the vertices. The adjacency matrix depends on the order of the vertices, hence there are n! possible adjacency matrices for a graph with n vertices.

13

Bikash Balami Graph Theory

Adjacency matrix for undirected graph is symmetric, in case of the pseudograph or multigraph the representation is similar but the matrix here is not zero-one matrix rather the $(i, j)^{th}$ entry of the matrix contains the number of edges appearing between that pair of vertices.

In case of the directed graph we can extend the same concept as in undirected graph as dictated by the relation

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

The main difference is that the matrix may not be symmetric.

If the number of edges is few then the adjacency matrix becomes sparse. Sometimes it will be beneficial to represent graph with adjacency list in such a condition.

Incidence Matrix

This is another way of representing graph. Given an undirected graph G = (V, E). Assume that the vertices of the graph are $v_1, v_2, \ldots, v_n$ and the edges of the graph are $e_1, e_2, \ldots, e_m$. The incidence matrix of a graph with respect to the above ordering of V and E is n-by-m matrix M =

$[m_{ij}]$, where

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

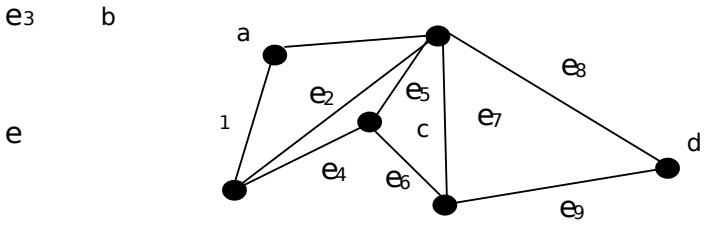
When the graph is not simple then also the graph can be represented by using incidence matrix where multiple edges corresponds to two different columns with exactly same entries. Loops are represented with column with only one entry.

14

Bikash Balami Graph Theory

Example 1:

Represent the following graph using adjacency matrix and incidence matrix.



Solution: f e

Let the order of the vertices be a, b, c, d, e, f and edges order be e1, e2, e3, e4, e5, e6, e7, e8, e9.

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

Adjacency Matrix

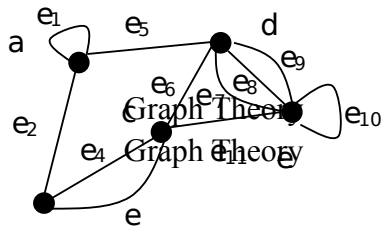
$$\begin{array}{c} \\ a \\ b \\ c \\ d \\ e \\ f \end{array} \begin{array}{c} \begin{array}{ccccccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 & e_9 \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

Incidence Matrix

Example 2:

Represent the following graph using adjacency matrix and incidence matrix.

15



Solution:

Let the order of the vertices be a, b, c, d, e and edges order be e1, e2, e3, e4, e5, e6, e7, e8, e9, e10, e11.

	a	b	c	d	e
a	1	1	0	1	0
b	1	0	2	0	0
c	0	2	0	1	1
d	1	0	1	0	3
e	0	0	1	3	1

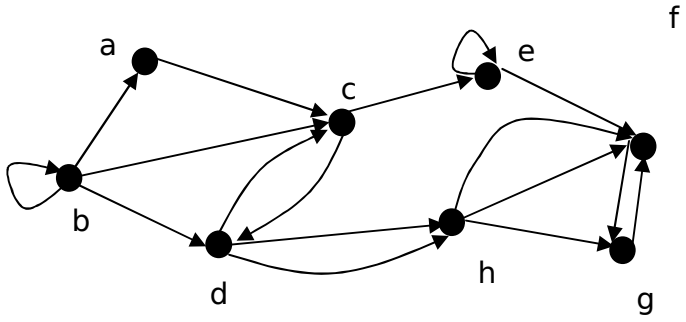
Adjacency Matrix

	e1	e2	e3	e4	e5	e6	e7	e8	e9	e10	e11
a	1	1	0	0	0	0	0	0	0	0	0
b	0	0	1	1	0	0	0	0	0	0	0
c	0	0	0	0	1	0	1	0	1	0	1
d	0	0	0	0	0	1	1	1	0	1	1
e	0	0	1	1	0	1	0	0	0	0	0

Incidence Matrix

Example 3:

Represent the following directed graph using adjacency matrix.



Solution:

Let the order of the vertices be a, b, c, d, e, f, g and h. The adjacency matrix is

Bikash Balami Graph Theory

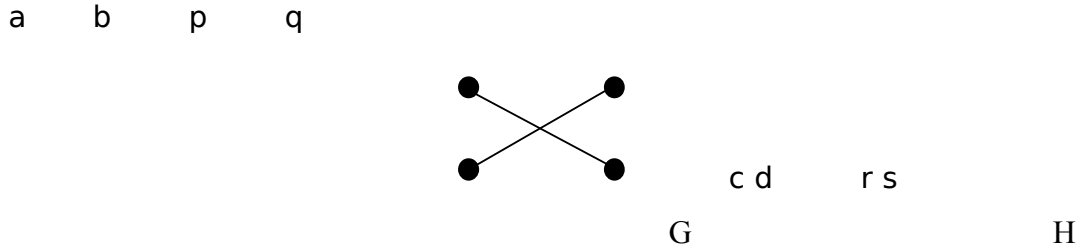
$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 1 & 0 \end{bmatrix}$$

Isomorphism of Graphs

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

In other words, two simple graphs are called isomorphic if there is a one-to-one correspondence between vertices of the two graphs that preserves the adjacency relationship. Also, isomorphism of simple graphs is an equivalence relationship.

Example: Show that the graphs G = (V, E) and H = (W, F) displayed below are isomorphic.



Solution

17

Bikash Balami Graph Theory

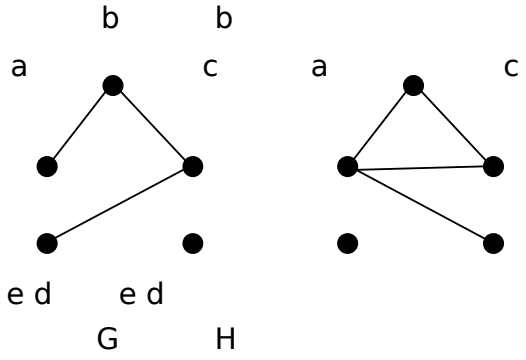
The function f with f(a) = p, f(b) = s, f(c) = r, and f(d) = q is a one-to-one correspondence between V and W. Also, this correspondence preserves adjacency because adjacent vertices in G are a and b, a and c, b and d, and c and d, and each of the pairs f(a) = p and f(b) = s, f(a) = p and f(c) = r, f(b) = s, and f(d) = q, and f(c) = r and f(d) = q are adjacent is H.

Determining whether two graphs are isomorphic or not is a difficult task since there are n! possible one-to-one correspondence between the vertex sets of two simple graphs with n vertices if n is large.

However we can show that two simple graphs are not isomorphic by showing that they do not share a property that isomorphic simple graphs must both have. Such a property is called invariant. These properties are described as follows:

1. Isomorphic simple graphs must have the same number of vertices (one-to-one correspondence between vertices of two graphs is required).
2. Isomorphic simple graphs also must have the same number of edges (due to adjacency preservation).
3. The degrees of the vertices in isomorphic simple graphs must be same because the number of edges from the vertex is determined by degree.

Example: Show that the graphs displayed below are not isomorphic.



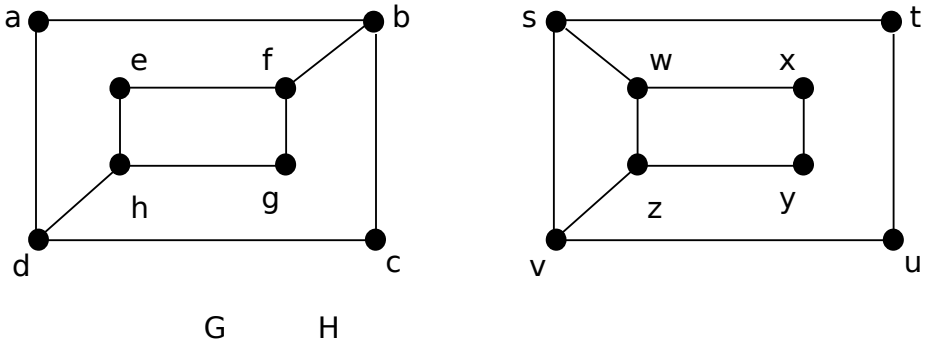
Solution:

18

Bikash Balami Graph Theory

Both graphs have five vertices and six edges. However, H has a vertex of degree one, namely e, whereas G has no vertices of degree one. Hence G and H are not isomorphic.

When these invariants are the same, it does not necessarily mean that the two graphs are isomorphic. There are no useful sets of invariants currently known that can be used to determine whether simple graphs are isomorphic.



Example:

Determine whether the graphs shown below are isomorphic.

Solution:

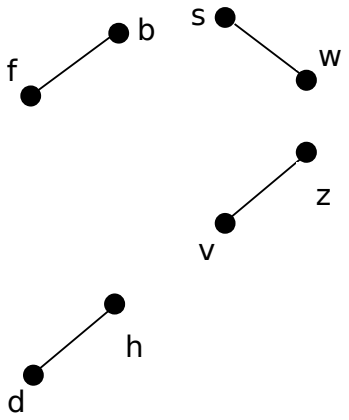
These both graphs have eight vertices and ten edges. Also, they both have four vertices of degree two and four vertices of degree three.

However, G and H are not isomorphic. Since, deg(a) = 2 in G, a must correspondence to either t, u, x, or y in H because these are the vertices of degree two in H. Here, each of these four vertices in H is adjacent to another vertex of degree two in H, which is not true for a in G.

Another way to see that two graphs are not isomorphic is to note that the subgraphs formed by connecting the edges from the vertex with same degree in both the graphs are not isomorphic. For example, subgraphs of G and H made up of vertices of degree three and the edges connecting them in the above figure are not isomorphic. Hence, G and H are not isomorphic. The figure below shows subgraphs of G and H made up of vertices of degree three.

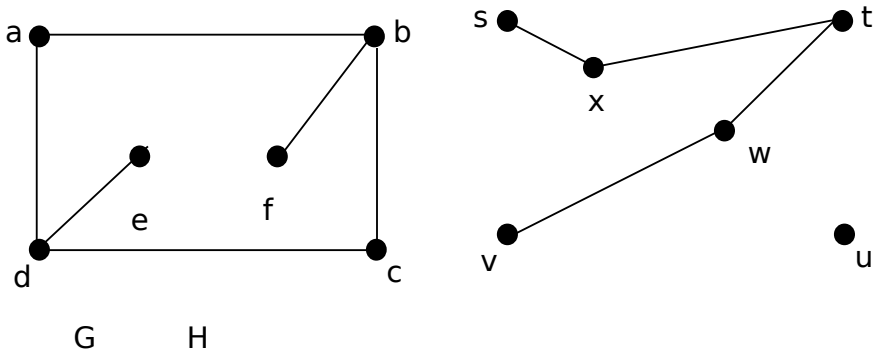
19

Bikash Balami Graph Theory



To show isomorphism of graphs, we can also use adjacency matrix. For this we should show that adjacency matrices of the two graphs are same. For this, we may need to arrange vertices in the adjacency matrix.

Example: Determine whether the graphs displayed below are isomorphic.



Solution:

Both graphs have six vertices and seven edges. Both have four vertices of degree two and two vertices of degree three. Also, subgraphs are isomorphic. Here we cannot say that these graphs are isomorphic or not. For this we can use adjacency matrix with the order of vertices a, b, c, d, e, f and w, t, u, v, s, x. If we draw the adjacency matrices with the above mentioned orders of vertices, we can see similar two matrices. Hence these graphs are isomorphic.

Note:

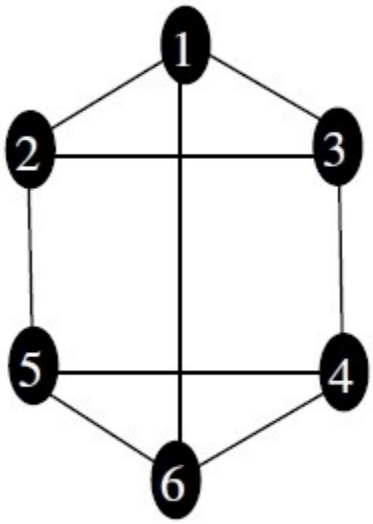
20

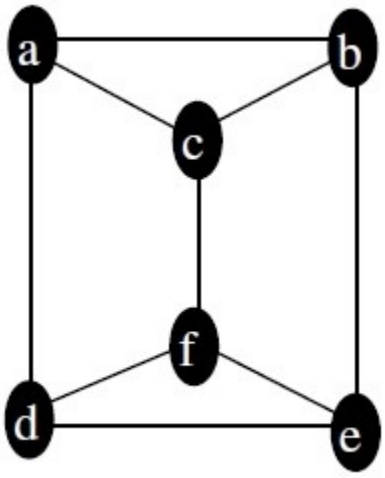
Bikash Balami Graph Theory

If both adjacency matrices are not same, we cannot say that the two graphs are not isomorphic because another correspondence of the vertices may be same.

Example

Determine whether the given two graphs are isomorphic or not?





Solution:

In the above two graphs number of vertices in both graphs is same (i.e. 6), number of edges equal to 9 in both the graphs and all the vertices in both the graphs have degree 3.

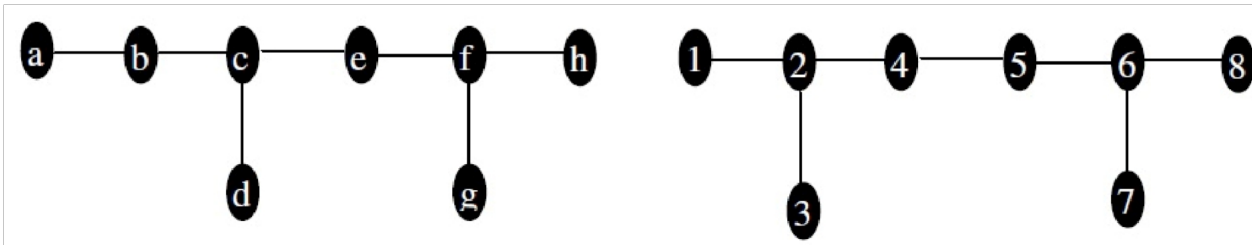
Since the invariants agree in both the graphs, we can try out to find the function that is isomorphism. Take the sequence of vertices from the first graph as 1, 2, 3, 4, 5, and 6. Now define f(1) = c, f(2) = a here there is adjacency preservation since we have {1, 2} as and edge in the first graph where as {f(1), f(2)} ={c, a} is an edge in the second graph. Similarly we can assign f(3) = b, f(4) = e, f(5) = d, f(6) = f. Since we found one to one correspondence between vertices of two graphs persevering the adjacency, the above two graphs are isomorphic. We can note that the adjacency matrices of two isomorphic graphs in which the vertices are ordered in terms of function i.e. in our example 1, 2, 3, 4, 5, and 6 for the first graph and c, a, b, e, d, and f in the second graph are same. In the above two graphs note that the number of circuits of same length in both the graphs is same.

21

Bikash Balami Graph Theory

Example:

Determine whether the given two graphs are isomorphic or not?



Solution:

In the above two graphs number of vertices in both graphs is 8, the number of edges equal to 7 in both the graphs, in both graphs two vertices have degree 3, 4 vertices have degree 1 and the remaining 2 vertices have degree 2. Since the invariants agree in both the graphs, we can continue to get the function such that it is isomorphism. However, in case of first graph the subgraph containing the vertex c (degree 3), with vertices a, b, c, d, and e is not isomorphic with any of the subgraph formed by connecting edges with vertex 2 or 6 (both of degree 3). Hence the two above graphs are not isomorphic.

Graph Connectivity

Many problems can be modeled with paths formed by travelling along the edges of the graphs. For example, problem of determining whether a message can be sent between two computers using intermediate links, problems of efficiently planning routes for mail delivery, garbage pickup, diagnostics in computer networks, and so on.

Path

Informally, a path is a sequence of edges that begins at a vertex of a graph and travels along edges of the graph, always connecting pairs of adjacent vertices. A formal definition is given below.

In an undirected graph G, a path of length *n*, where *n* is a nonnegative integer, from *a* to *b* in G is a sequence of *n* edges $e_1, e_2, \ldots, e_n$ of G such that $f(e_1) = \{x_0, x_1\}$, $f(e_2) = \{x_1, x_2\}$, …, $f(e_n) = \{x_{n1}, x_n\}$, where $a = x_0$ and $b = x_n$. When the graph is simple, we denoted this path by a vertex sequence

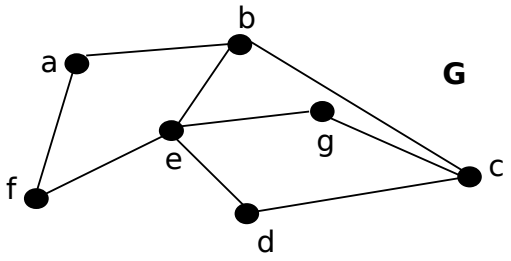
22

Bikash Balami Graph Theory

$x_0, x_1, \ldots, x_n$. If a = b and the path length is greater than zero, then the path is called circuit or cycle. The path or circuit is said to pass through the vertices $x_1, x_2, \ldots, x_{n-1}$ or traverse the edges $e_1, e_2, \ldots, e_n$. A path or circuit is simple if it does not contain the same edge more than once.

When it is not necessary to distinguish between multiple edges, we will denote a path $e_1, e_2, \ldots, e_n$, where $f(e_i) = \{x_{i-1}, x_i\}$ for i = 1, 2, …, n by its vertex sequence $x_0, x_1, \ldots, x_n$. A path of length zero consists of a single vertex.

Example: In the simple graph G shown below a, b, e, g, c is a simple path of length 4 since {a, b}, {b, e}, {e, g}, and {g, c} are all edges. However, a, b, g, c is not a path since {b, g} is not an edge. Note that a, b, e, f, a is a circuit of length 4 since {a, b}, {b, e}, {e, f}, {f, a} are edges, and this path begins and ends at b. The path a, b, e, f, a, b, which is of length 5, is not simple since it contains the {a, b} twice.



Let n be a nonnegative integer and G a directed multigraph. A path of length *n*, where *n* is a nonnegative integer, from a to b in G is a sequence of n edges $e_1, e_2, \ldots, e_n$ of G such that $f(e_1) = (x_0, x_1)$, $f(e_2) = (x_1, x_2)$, …, $f(e_n) = (x_{n-1}, x_n)$, where $a = x_0$ and $b = x_n$. When there is no multiple edges in the directed graph, this path is denoted by its vertex sequence $x_0, x_1, \ldots, x_n$. If a = b and the path length is greater than zero, then the path is called circuit or cycle. A path or circuit is simple if it does not contain the same edge more than once.

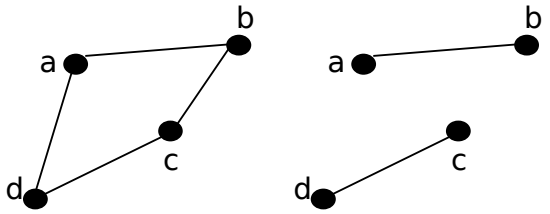
Note that the terminal vertex of an edge in a path is the initial vertex of the next edge in the path. When it is not necessary to distinguish between multiple edges, we will denote a path $e_1, e_2, \ldots, e_n$, where $f(e_i) = (x_{i-1}, x_i)$ for i = 1, 2, …, n by its vertex sequence $x_0, x_1, \ldots, x_n$.

23

Bikash Balami Graph Theory

Connectedness in Undirected Graphs

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



of the graph. The first graph given below is connected whereas the second graph is not.

A graph that is not connected is the union of more than one connected graphs that do not share the common vertex. These disjoint connected subgraphs are called **connected component** of a graph.

Theorem:

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

Proof:

Suppose a and b are two distinct vertices of the connected undirected graph G = (V, E). We know that G is connected so by definition there is at least one path between a and b . Let $x_0, x_1, \ldots, x_n$, where $x_0 = a$ and $x_n = b$, be the vertex sequence of a path of a least length. Now if this path of the least length is not simple then we have $x_i = x_j$, for some i and j with $0 \leq i < j$. This implies that there is a path from a to b of shorter length with the vertex sequence $x_0, x_1, \ldots, x_i, \ldots, x_{j+1}, \ldots, x_n$ obtained by removing the edges corresponding to the vertex sequence $x_{i+1}, \ldots, x_j$. This shows that there is a simple path.

Cut vertices (**articulation points**) are those vertices in the graph whose removal along with the edges incident on them produces subgraph with more connected components than in the original graph.

Cut edge (**bridge**) is an edge whose removal produces a graph with more connected components than in the original graph. A **cut set** of a graph is a set of edges such that the removal of these

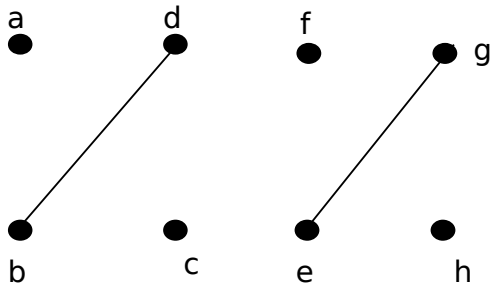
24

Bikash Balami Graph Theory

edges produces a subgraph with more connected components than in the original graph, but no proper subset of this set of edges has this property.

Example:

Find the cut vertices and cut edges in the graph given below.



Solution*:*

The cut vertices are b, c, and e. the removal of one of these vertices and its adjacent edges disconnects the graph. The cut edges are {a, b} and {c, e}. Removing either one of these edges disconnects the graph.

Connectedness in Directed Graphs

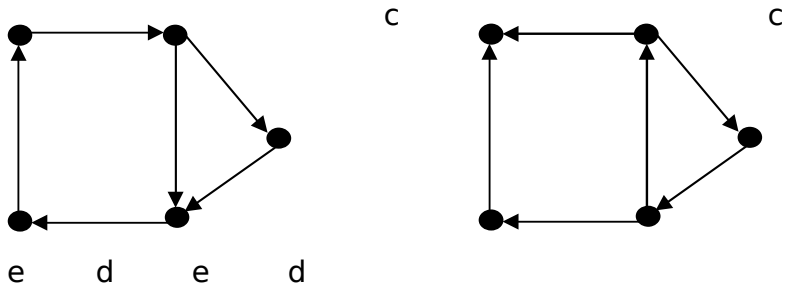
A directed graph is **strongly connected** if there is a path from a to b and from b to a whenever a and b are vertices in the graph. A directed graph is **weakly connected** if there is a path between every two vertices in the underlying undirected graph. The subgraphs of a directed graph G that are strongly connected but not contained in larger strongly connected subgraphs are called **strongly connected components** or **strong components** of G.

Example:

a b a b

25

Bikash Balami Graph Theory



Are the directed graphs shown above strongly connected? Are they weakly connected?

Solution*:*

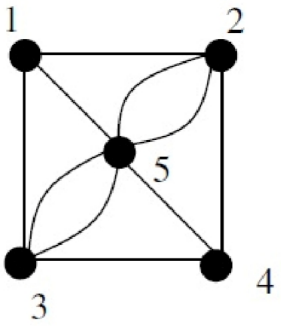
The first graph is strongly connected because there is a path between any two vertices in this directed graph. Hence it is also weakly connected. The second graph is not strongly connected. There is no directed path from a to b in this graph. However, this graph is weakly connected, since there is a path between any two vertices in the underlying undirected graph of this graph.

Euler Paths and Circuits

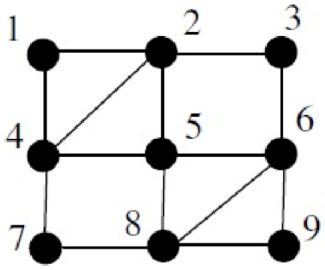
An Euler circuit in a graph G is a simple circuit containing every edge of G. An Euler path in G is a simple path containing every edge of G.

Example:

Find the Euler path or circuit in the following graphs



No Euler circuit exist, but the Euler path is
1,2,5,1,3,5,4,3,5,2,4



The Euler circuit is 1,2,3,6,9,8,7,4,5,8,6,5,2,4,1

26

Bikash Balami Graph Theory

Necessary and Sufficient Conditions for Euler Circuits and Paths

Theorem 5:

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

Proof:

Take a connected multigraph G =(V, E) where V and E are finite. We can prove the theorem in two parts.

First we prove that if a connected multigraph has an Euler circuit, then all the vertices have even degree. For this, take a vertex v, where the Euler circuit begins. There is some edge that is incident to v and some other vertex say u then we have an edge {v, u}. This edge {v, u} contributes one to the degree of v and u both. Again there must be some edge other than {v,u} that is incident to u and some other vertex. In this case the total degree of the vertex u becomes even, so whenever in the circuit the vertex is met the degree of that vertex is even since every time entering and leaving the vertex gives even degree to all the vertices other than the initial vertex. However since the circuit must terminate in the vertex v and the edge that is terminating the circuit contributes one to the degree of the initial vertex v the total degree of the vertex v is also even. Now we have every time the vertex is entered and left it gives even degree and the initial vertex also gives even degree, we can conclude that if a graph has Euler circuit, then all the vertices have even degree. Now we try to prove that if all the vertices in the connected multigraph have even degree, the there exist Euler circuit. For this, take a connected multigraph G with all the vertices having even degree. To make a circuit start at arbitrary vertex, say a of G. now start from the vertex a = x0 and arbitrarily choose other vertex x1 to form and edge {x0, x1}.

Continue building the simple path $\{x_0, x_1\}$, $\{x_1, x_2\}$, …, $\{x_{n-1}, x_n\}$. This path terminates since it has a finite number of edges. It begins at a with an edge {a, x} and terminate at a with some edge {y, a}. This is correct since every vertex has even degree in the graph we are considering, if an edge left some vertex then there must be an edge entering that vertex to make its degree even. Now we have shown that there exists simple circuit in the graph with all the vertices of even degree. If this circuit has all the edges of the graph in it, then the simple circuit is itself an Euler circuit. If all the edges are not in the circuit, then we have next possibility. Now, consider the

27

Bikash Balami Graph Theory

subgraph, say H that is formed by removing all the edges that are already in the simple circuit formed above and by removing the isolated vertices after edges are removed. Since the original graph G is connected, there must be at least one vertex of H that is common with the circuit we have formed. Let w be such a vertex. Every vertex in H has even degree since it is a subgraph of original graph. In case of w, while forming the circuit pairs of incident edges are used up. So the degree of w is again even. Beginning at w we can build a simple circuit as described above. We can continue this process until all edges have been used. Now if we combine the formed circuit in a way that it makes use of common vertex to make a circuit then we can say that the circuit is an Euler circuit. Hence if every vertices of a connected graph has an even degree then it has an Euler circuit. This concludes the proof.

Theorem 6:

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

Proof:

This fact can be proved if we can prove that first, if the connected multigraph has Euler path exactly two vertices have odd degree and second if the connected multigraph has exactly two vertices of odd degree, then it has Euler path.

Now, if the graph (in this proof graph means connected multigraph) has an Euler path say from a to z but not Euler circuit, then it must pass through every edge exactly once. In this scenario the first edge in the path contributes one to the degree of vertex a, and at all other time when other edges pass through vertex a it contributes twice to the degree of a, hence we can say that degree of a is odd. Similarly the last edge in the path coming to z contributes one to the degree of z, all the other edges contributes two one for entering and one for leaving. Here also the degree of last vertex, z is odd. All the other vertices other than a and z must have even degree since the edges in those vertices enter and leave the vertex contributing two to the degree every time the vertices are met. Hence if there is an Euler path but not an Euler circuit, exactly two vertices of the graph have odd degree.

28

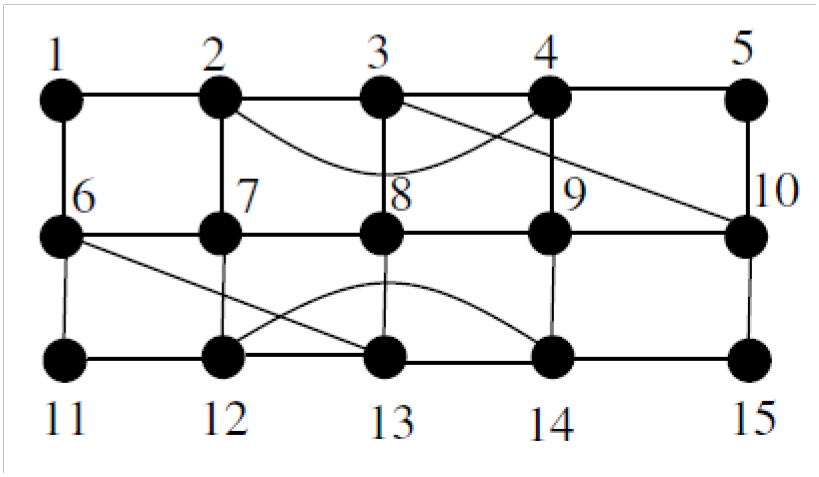
Bikash Balami Graph Theory

Secondly, if exactly two vertices of a graph have odd degree and lets consider they are a and z. Now, consider another graph that adds an edge {a, z} to the original graph, then the newly formed graph will have every vertices of even degree. So there exists Euler circuit in the new graph and the removal of the new edge gives us the Euler path in the original graph. Hence if exactly two vertices of the graph have an odd degree, then the graph has an Euler path but not Euler circuit.

This concludes the proof.

Example:

Find Euler path or circuit?



Solution:

In the above graph, all the vertices have even degree, hence there is an Euler circuit. The Euler circuit is 1,2,3,4,5,10,15,14,13,12,11,6,13,8,9,14,12,7,8,3,10,9,4,2,7,6,1

29

Bikash Balami Graph Theory

Hamilton paths

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

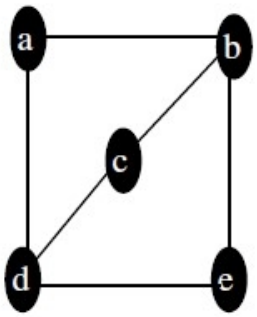
A graph with a vertex of degree one cannot have a Hamilton circuit.

If a vertex in a graph has degree 2, then both edges incident with this vertex must be part of Hamilton cycle.

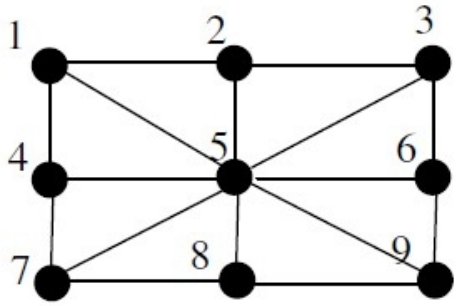
Example:

Find Hamilton circuit from the following graph if exists? What about Hamilton path?

(a)



(b)



Solution:

In graph (a) there is no Hamilton circuit since the node c has degree 2 and both the edges from it must be in Hamilton circuit, which is not possible. One of the Hamilton path in the graph (a) is a, b, c, d, e.

In graph (b) we can find Hamilton circuit, the circuit can be 1,2,3,5,6, 9,8,7,4,1. Since there is circuit we can have path also.

Theorem 7: Dirac's Theorem

If G is a simple graph with n vertices with $n \geq 3$ such that the degree of every vertex in G is at least $\frac{n}{2}$, then G has a Hamilton circuit.

30

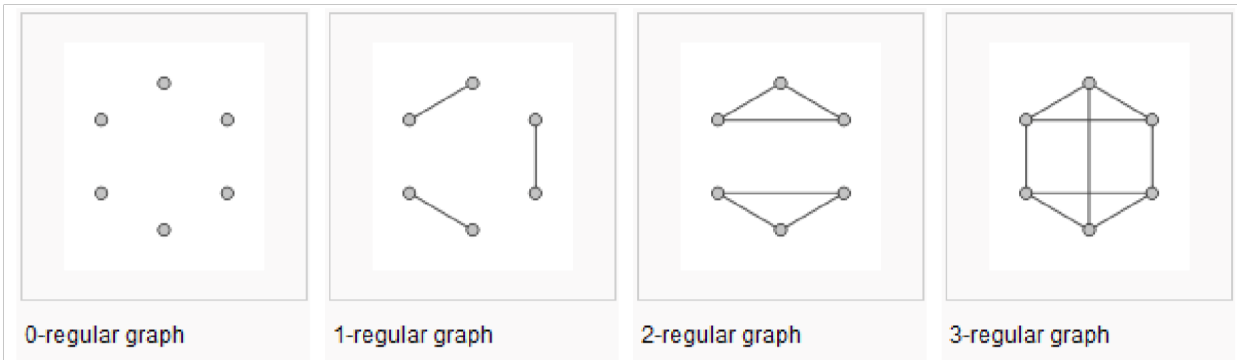
Bikash Balami Graph Theory

Theorem 8: Ore's Theorem

If G is a simple graph with n vertices with n ≥ 3 such that deg(u) + deg(v) ≥ n for every pair of nonadjacent vertices u and v in G, then G has a Hamilton path.

Regular Graphs

In graph theory, a regular graph is a graph where each vertex has the same number of neighbors; i.e. every vertex has the same degree. A regular graph with vertices of degree k is called a k-regular graph or regular graph of degree k. Regular graphs of degree at most 2 are easy to classify: A 0-regular graph consists of disconnected vertices, a 1-regular graph consists of disconnected edges, and a 2-regular graph consists of disconnected cycles. A 3-regular graph is known as a cubic graph.



Planar graphs

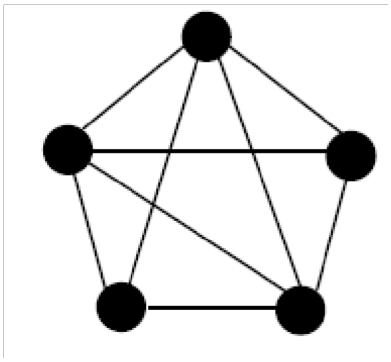
A graph is called a planar if it can be drawn in the plane without any edges crossing. Such drawing is called a planar representation of the graph.

Example:

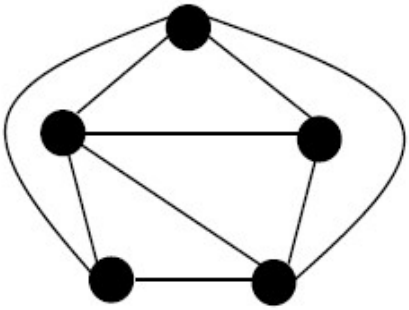
Draw the graph below as planar representation of the graph.

31

Bikash Balami Graph Theory



Solution



Theorem 9: Euler's Formula

Let G be a connected planar simple graph with *e* edges and *v* vertices. Let *r* be the number of regions in a planar representation of G. Then r = e – v + 2.

Example:

Suppose that a connected planar graph has 30 edges. If a planar representation of this graph divides the plane into 20 regions, how many vertices does this graph have?

Solution:

We have, r = 20, e = 30, so by Euler's formula we have v = e – r + 2 = 30 – 20 + 2 = 12. So the number of vertices is 12.

32

Bikash Balami Graph Theory

Corollary 1:

If G is a connected planar simple graph with e edges and v vertices where $v \geq 3$, then $e \leq 3v - 6$.

Corollary 2:

If G is a connected planar simple graph, then G has a vertex of degree not exceeding five.

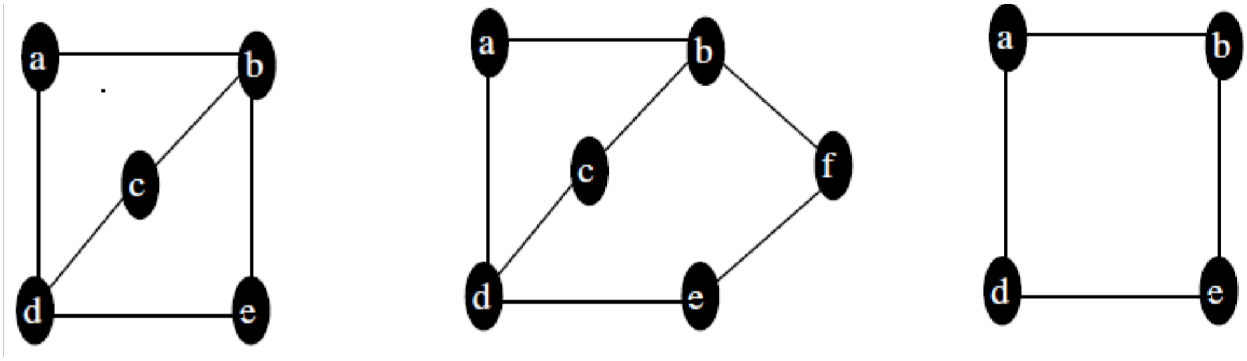
Corollary 3:

If a connected planar simple graph has e edges and v vertices with $v \geq 3$ and no circuits of length three, then $e \leq 2v - 4$.

If a graph is planar, so will be any graph obtained by removing an edge {u, v} and adding a new vertex w together with edges {u, w} and {w, v}. Such an operation is called an elementary subdivision. The graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are called homeomorphic if they can be obtained from the same graph by a sequence of elementary subdivisions.

Example:

The below example graphs are homeomorphic to the third graph.



Theorem 10: Kuratowski's Theorem (Planarity Testing Algorithm)

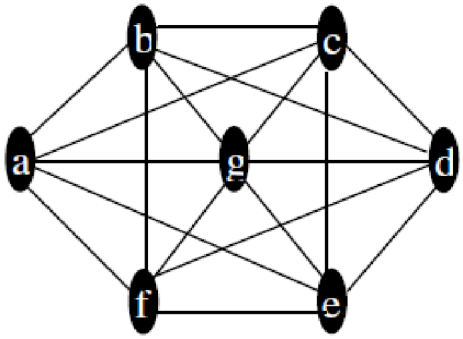
A graph is nonplanar if and only if it contains a subgraph homeomorphic to $K_{3,3}$ or K_5.

Example:

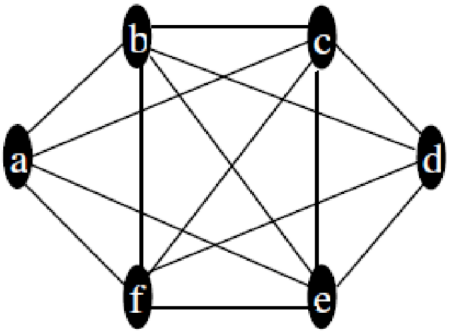
Determine whether the following graph is planar or not?

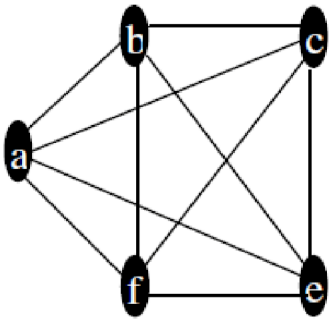
33

Bikash Balami Graph Theory



Solution:





Above we saw that the graph is homeomorphic to K_5, the given graph is not planar.

A Shortest – Path Algorithm

There are several algorithms that find a shortest path between two vertices in a weighted graph.

Graphs that have a number assigned to each edge are called **weighted graphs**.

Algorithm (Dijkstra's Algorithm)

Procedure *Dijkstra(G : weighted connected simple graph, with all weighted positive)*

- *G has vertices* $a = v_0, v_1, \ldots\ldots, v_n = z$ *and weights* $w(v_i, v_j)$ *where* $w(v_i, v_j) = \infty$ *if* (v_i, v_j) *is not an edge in G*
- ***for*** $i = 1$ ***to*** n

  $L(v_i) = \infty$
- $L(a) = 0$
- $S = \varnothing$

34

Bikash Balami Graph Theory

- *{the labels are now initialized so that the label of a is 0 and all others label are ∞, and S is the empty set}*
- ***while*** *z ∉ S*
- ***begin***

  u = a vertex not in S with L(u) minimal
  S = S ∪ {u}

  for *all vertices v not is S*

  if *L(u) + w(u,v) < L(v)* ***then*** *L(v) = L(u) + w(u,v)*

  (this adds a vertex to S with minimum label and updates the labels of vertices not in S)

- ***end*** *{L(z) = length of the shortest path from a to z}*

Algorithm Tracing

Use Dijkstra's algorithm to find the shortest path from the vertices *a* to z in following weighted graph given in figure (a).

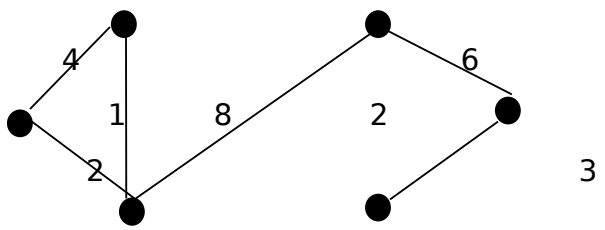
Solution

The steps used by Dijkstra's algorithm to find the shortest path between a and z are shown below.

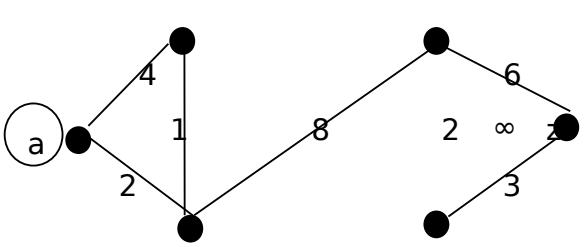
b ∞ 5 d ∞ b 4(a) 5 d ∞

35

Bikash Balami Graph Theory



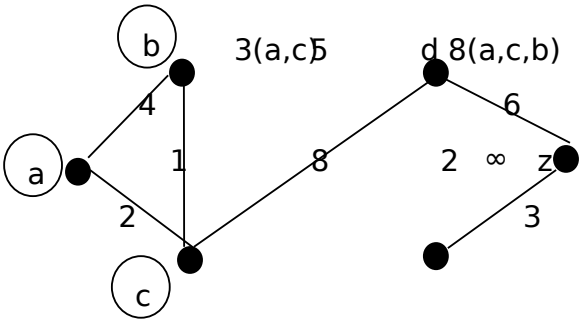
a z ∞

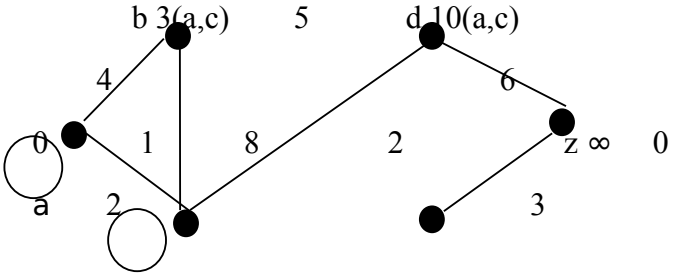


c ∞ 10 e ∞

c 2(a) 10 e∞

(a) **(b)**



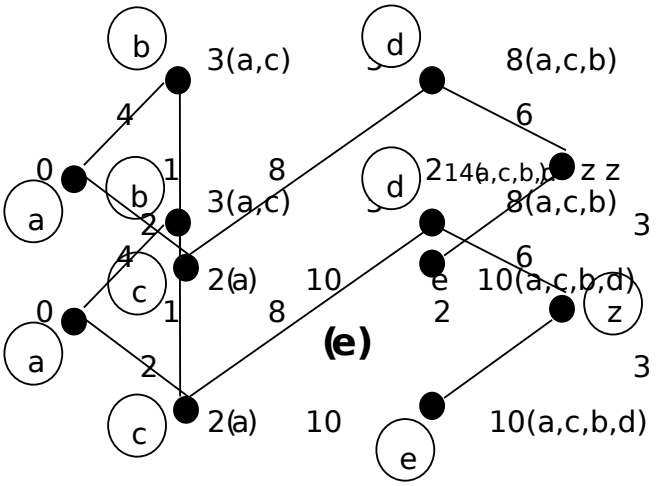


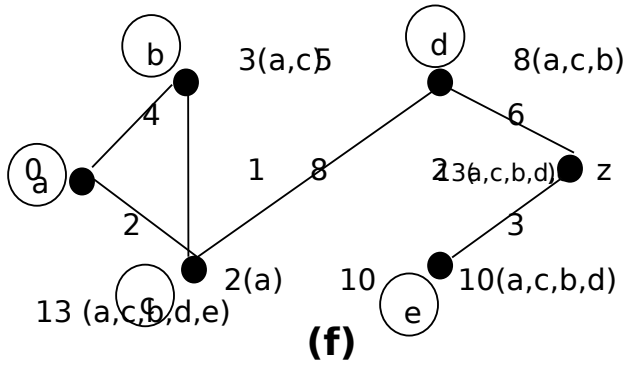
c 2(a) 10 e 12(a,c)

2(a) 10 e 12(a,c)

36

Bikash Balami Graph Theory





(c) **(d)**

(g)

Graph coloring

A coloring of a simple graph is the assignment of a color to each vertex of the graph so that no two adjacent vertices are assigned the same color. A chromatic number of a graph is the least number of colors needed for a coloring of this graph.

Theorem 11:The Four Color Theorem

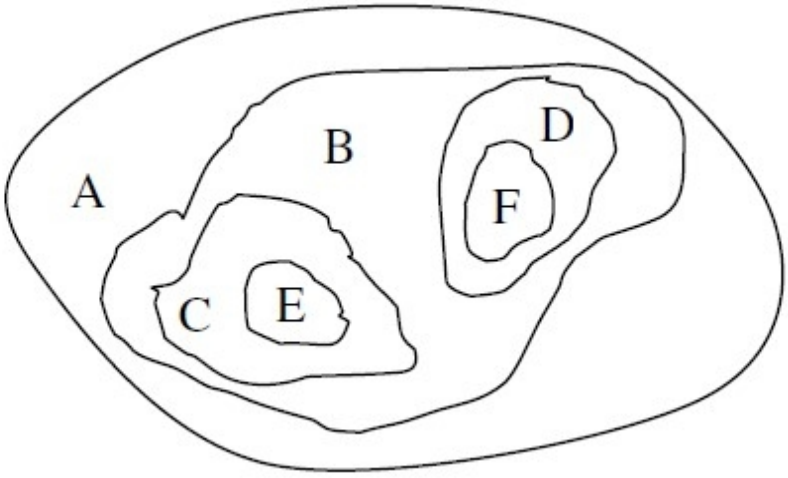
The chromatic number of a planar graph is no greater than four. To show the chromatic number of a graph as *k* we must show that the graph can be colored using *k* colors and the given graph cannot be colored using fewer than k colors.

Example:

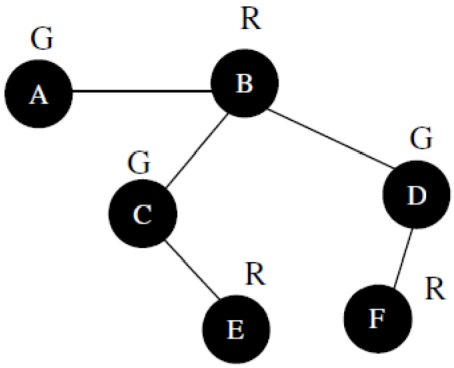
37

Bikash Balami Graph Theory

Construct the dual graph for the map shown. Then find the number of colors needed to color the map so that no two adjacent regions have the same color.



Solutic..



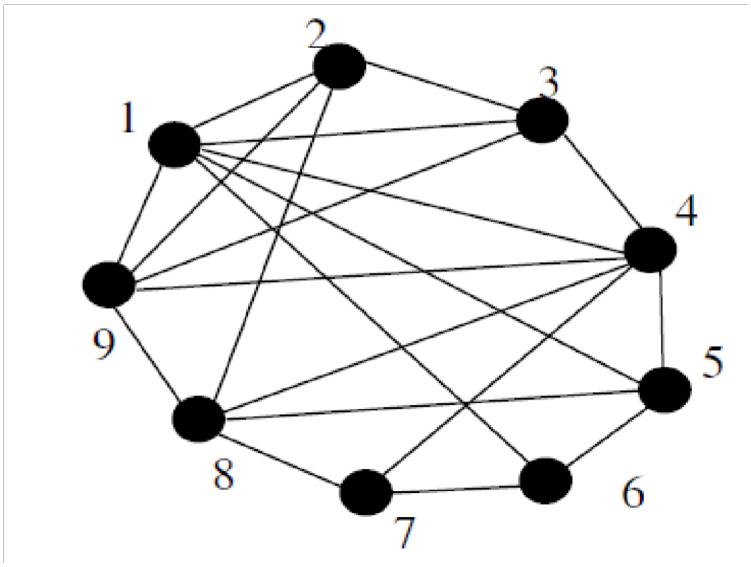
We can color the graph with at most 2 colors as shown in the graph. Take R and G as red and green

Example:

Find the chromatic number of the graph below.

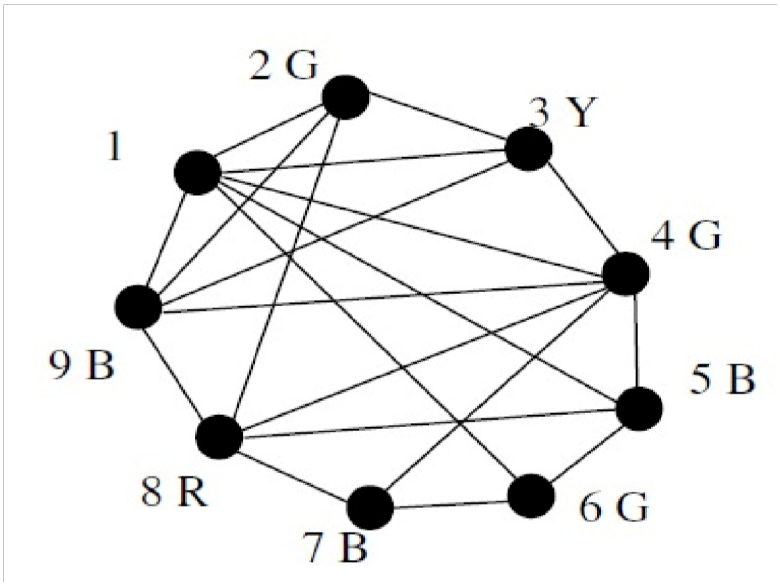
38

Bikash Balami Graph Theory



Solution:

Lets start with vertex 1, it has adjacent vertices as 2, 3, 4, 5, 6, 9 so using only 2 colors would suffice for the graph having the edges from 1 to its adjacent vertices. However since 9 has its adjacent vertices as 1, 2, 3, 4 we cannot just color the above graph with 2 colors because at least 1, 2 and 9 must have different colors. Trying with 3 colors we found that at least 1, 2, 3, and 9 must have different colors. So trying with four colors we can color the graph. Hence the chromatic number of the above graph is 4. Possible coloring is shown in the figure below.

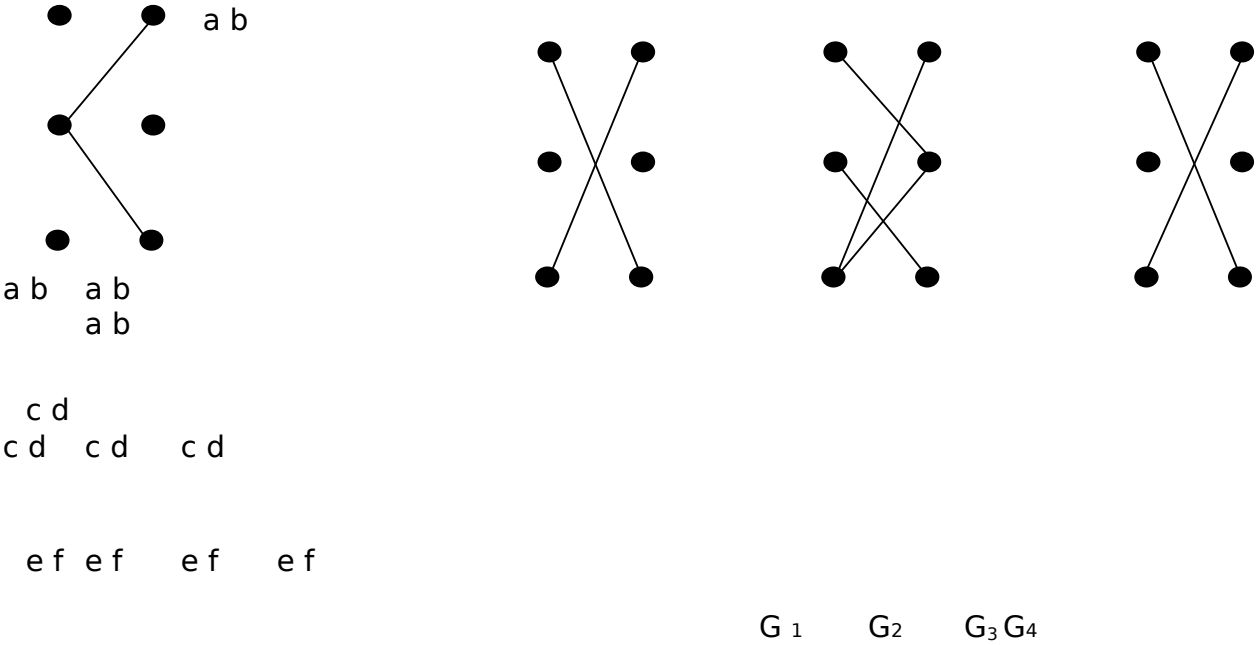


39

Bikash Balami Graph Theory

Trees

A tree is a connected undirected simple graph with no simple circuits. A tree is a particular type of graph. For example, family trees are graphs that represent genealogical charts. Family trees have vertices to represents the members of a family and edges to represents parent-child relationship. Trees are particularly useful in computer science in a wide range of algorithm including searching and sorting.



Q. Which of the graphs shown below are trees?

An undirected graph having no simple circuit and is not connected is called forest. The forest has each of its connected components as tree. For example, the figure G_4 above displays a forest.

Theorem 1:

An undirected graph is a tree if and only if there is a unique simple path between any two of its vertices.

Proof:

40

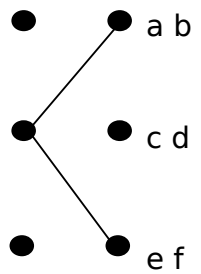
Bikash Balami Graph Theory

Assume that T is a tree. Since T is a tree it is a connected simple graph with no simple circuits. Let x and y be two vertices of T. we know that every connected graph has a simple path between every pair of vertices. So there is a simple path from x to y. This path must be unique because, if the path between x and y is not unique then there is another path between x and y that uses edges different from the path between x and y for first path, then reversing the path i.e. going from x to y from the first path and going from y to x through the second path forms a circuit. This is a contradiction that T is a tree; hence there is a unique simple path between any two vertices of a tree.

Again assume that there is a unique simple path between any two vertices of a graph, say T. Since there is a path between any two vertices of a graph, the graph is connected. Now, we can show that the graph T cannot have simple circuit. Had there been a simple circuit, there would be two simple paths between two vertices, say x and y, and the two simple path between x and y would create a simple circuit where first path goes from x to y and the second path goes from y to x. This violates our assumption that the path is unique. Hence, a graph with a unique simple path between any two vertices is a tree.

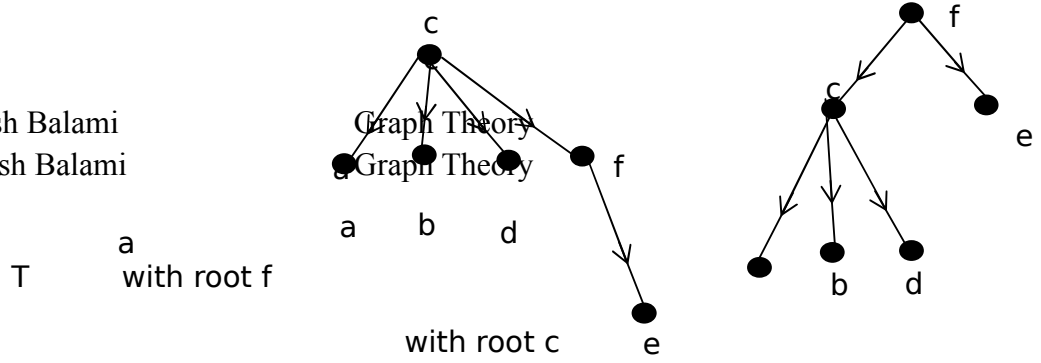
In many applications of trees, a particular vertex of a tree is designated as the root. A rooted tree is a tree in which one vertex has been designated as the root and every edge is directed away from the root.

We can change an un-rooted tree in to a rooted by choosing any vertex as the root. The tree in which root is defined produces a directed graph as shown below:



41

Bikash Balami Graph Theory



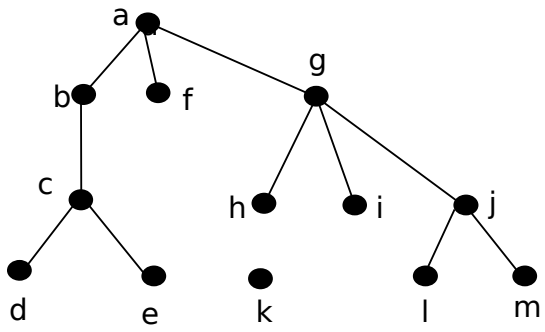
Take a rooted tree T. if v is the vertex in T other than root, then the ***parent*** of v is a vertex u in T such that there is a directed edge from u to v. In this scenario v is called ***child*** of u. Vertices with same parents are called ***siblings***. All the vertices that appear in the path from root to some vertex v in T, including root are called ***ancestors*** of v. The ***descendents*** of a vertex v are those vertices that have v as their ancestor. All the vertices that have children are called ***internal vertices*** (root is also an internal vertex if the tree has more than one vertices).
A ***subtree*** of a rooted tree T, with root a, is a ***subgraph*** of the tree consisting of a and all of its descendents and all the edges incident to these descendents. Here all the vertices must be in T also.

A ***m-ary*** tree is a rooted tree in which every internal vertex has no more than *m* children. It is called ***full m-ary*** tree if every internal vertex has exactly *m* children.

Example: Binary tree i.e. 2-ary tree.

An ***ordered rooted*** tree is a rooted tree where the children of each internal vertex are ordered. For e.g. in ordered binary tree (also called just a binary tree) if an internal vertex has two children then the first child is called ***left child*** and the second child is called ***right child***.

Q. In the rooted tree given below, find the parent of c, the children of g, the sibling of h, all ancestors of e, all descendants of b, all internal vertices, and all leaves. What is the sub tree rooted at g?



42

Bikash Balami Graph Theory

A rooted tree is called an m-ary tree if every internal vertex has no more than m children. The tree is called a full m-ary tree if every internal vertex has exactly m children. An m-ary tree with m = 2 is called a binary tree.

An ordered rooted tree is a rooted tree where the children of each internal vertex are ordered from left to right.

In an ordered binary tree (usually called just a binary tree), if an internal vertex has two children the first child is called the left child and the second child is called the right child. The tree rooted at the left child of a vertex is called the left sub tree of this vertex, and the tree rooted at the right child of a vertex is called the right sub tree of the vertex.

Properties of Trees

Theorem: A tree with n vertices has n-1 edges.

Proof: Here, we use mathematical induction to prove this theorem.

Basis step: When n=1, a tree with n=1 vertex has no edges. It follows that the theorem is true for n=1.

Inductive hypothesis: Assume that the tree with k vertices has k -1 edges, where k is a positive integer.

Inductive step: Suppose that a tree T has k + 1 vertices and that v is a leaf of T. Removing the vertex v and the associated edge from T produces a tree T1 with k vertices, since the resulting graph is still connected and has no simple circuits. By the induction hypothesis, T1 has k-1 edges. Hence, T has k edges since it has one more edge than T1, the edge connecting v to its parent.

Theorem: A full m-ary tree with i internal vertices contain n = mi + 1 vertices.

43

Bikash Balami Graph Theory

The level (depth) of a vertex v in a rooted tree is the length of the unique path from the root to vertex. The level of the root is zero. The height of the rooted tree is the length of the longest path from the root to any vertex. A rooted m-ary tree of height h is balanced if all leaves are at levels h or h – 1.

Application of Trees

We can solve different problems using trees. For example,

- How should items in a list be stored so that an item can be easily located? To solve these problems, we use the concept of binary search trees.
- What series of decisions should be made to find an object with a certain property in a collection of objects of a certain type? To solve these problems, we use the concept of decision trees.
- How should a set of characters be efficiency coded by bit strings? To solve these problems, we use the concept of prefix codes.

Application of Graphs

- Graphs are used to represent networks of communication, data organization, computational devices, flow of computation.
- Graphs can be used to show the link structures of web sites. The vertices are the web pages available at the website and a directed edge from page A to page B exists if and only if A contains a link to B.
- Graph theory is also used to study molecules in chemistry and physics.
- In chemistry a graph makes a natural model for a molecule, where vertices represent atoms and edges bonds. This approach is especially used in computer processing of molecular structures, ranging from chemical editors to database searching.

44

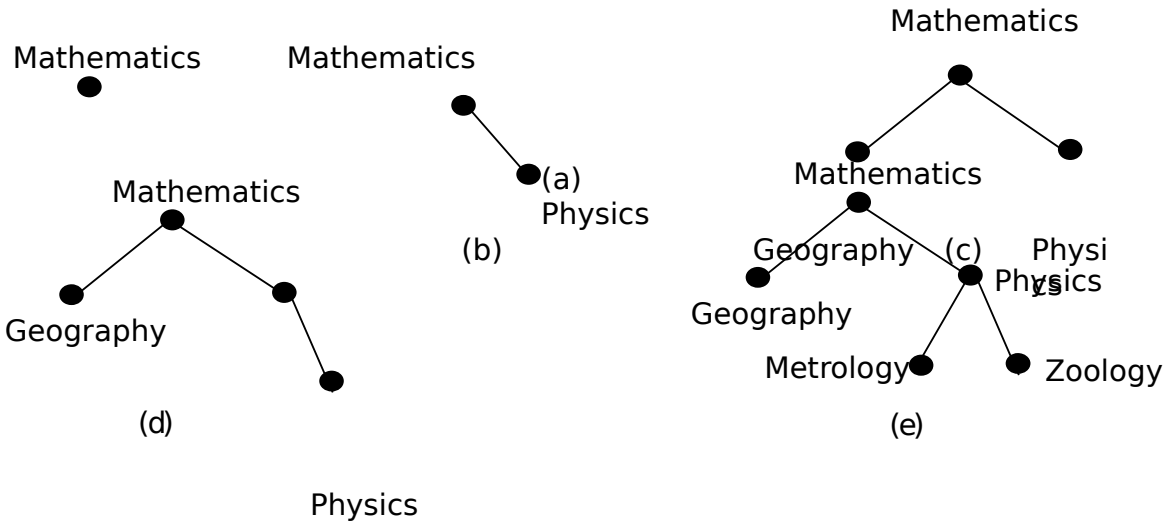
Bikash Balami Graph Theory

Binary Search Trees

A binary search tree (BST) is a binary tree in which each child of a vertex is designated as a right or left child, no vertex has more than one right child or left child and each vertex is labeled with a key, which is one of the items. Furthermore vertices are assigned keys so that the key of a vertex is both larger than the keys of all vertices in its left subtree and smaller than the keys of all vertices in its right subtree.

We use a recursive procedure to form a binary search tree for a list of items. We start with a tree containing just one vertex, namely, the root. The first item in the list is assigned as the key of the root. To add a new item, first compare it with the keys of vertices already in the tree, starting at the root and moving to the left if the item is less than the key of the respective vertex if this vertex has a left child or moving to the right if the item is greater than the key of the respective vertex if this vertex has a right child. When the item is less than the respective vertex and this vertex has no left child, then a new vertex with this item as its key is inserted as a new left child. Similarly, when the item is greater than the respective vertex and this vertex has no right child, then a new vertex with this item as its key is inserted as a new right child.

Example: Form a BST for the words mathematics, physics, geography, zoology, metrology, geology, psychology, and chemistry.

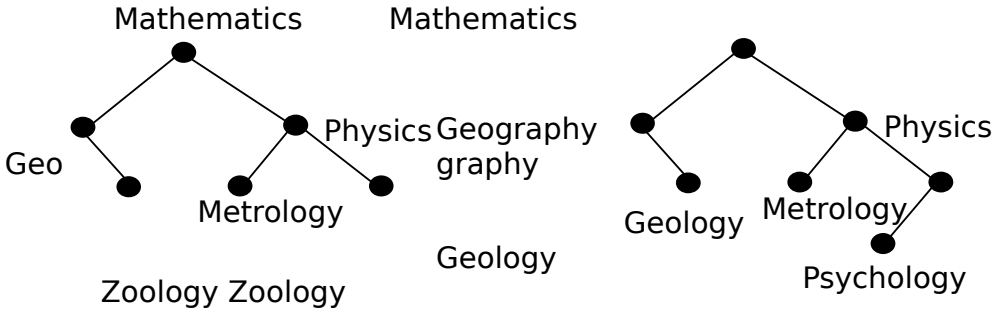


45

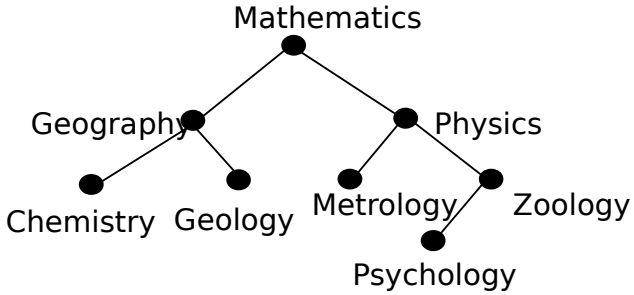
Bikash Balami Graph Theory

Zoology

Solution:



(f) (g)



(h)

Q. Consider a BST whose elements are abbreviated names of chemical elements. Starting with an empty BST, show the effect of successively adding the following elements: H, C, N, O, Al, Si, Fe, Na, P, S, Ni, and Ca.

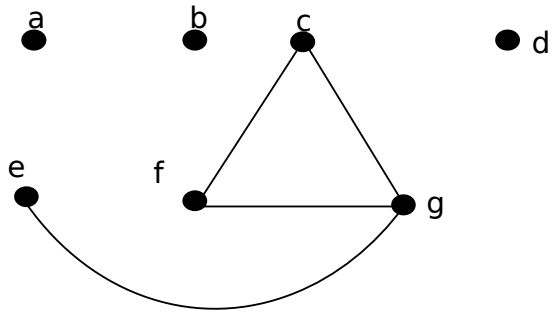
Spanning Trees

Let G be a simple connected graph. A spanning tree of G is a subgraph of G that is a tree containing every vertex of G.

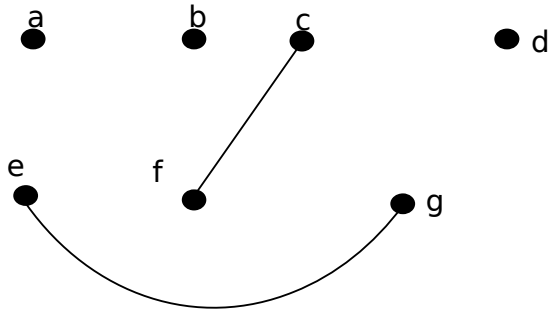
Example: Find a spanning tree of the simple graph G shown below.

46

Bikash Balami Graph Theory



Solution: The above graph is connected but is not tree because it contains simple circuits. By removing the edges {a, e}, {e, f} and {c, g}, we obtain a simple graph with no simple circuits. This subgraph is a spanning tree. The figure below shows this spanning tree. This tree is not the only spanning tree of the graph. The graph can also have other spanning trees.



Theorem: A simple graph is connected if and only if it has a spanning tree.

Proof: First, suppose that a simple graph G has a spanning tree T. Then, T contains every vertex of G. Furthermore, there is a path in T between any two of its vertices. Since T is a subgraph of G, there is a path in G between any two of its vertices. Hence, G is connected.

Now, suppose that G is connected. If G is not a tree, it must contain a simple circuit. Remove an edge from one of these simple circuits. The resulting subgraph has one fewer edge but still contains all the vertices of G and is connected. If this subgraph is not a tree, it has a simple circuit; so as before, remove an edge that is in a simple circuit. Repeat this process until no simple circuits remain. A tree is produced since the graph stays connected as edges are removed.

47

Bikash Balami Graph Theory

And, this tree is a spanning tree since it contains every vertex of G.

Note: Spanning trees are important in data networking.

Minimum Spanning Trees

A minimum spanning tree in a connected weighted graph is a spanning tree that has the smallest possible sum of weights of its edges.

Kruskal's algorithm for constructing minimum spanning trees: To carry out Kruskal's algorithm, choose an edge in the graph with minimum weight. Successively add edges with minimum weight that do not form a simple circuit with those edges already chosen. Stop after n-1 edges have been selected.

Algorithm: *procedure **Kruskal** (G:weighted connected undirected graph with*

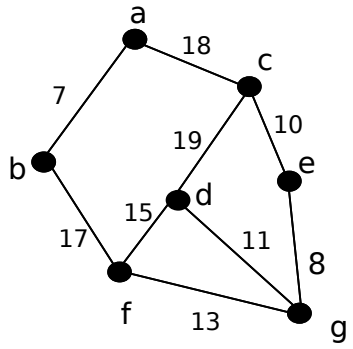
n vertices) T := empty graph

for i := 1 to n - 1 begin e := any edge in graph G with smallest weight that

edges not form a simple circuit when added to T T := T with e

added end {T is a minimum spanning tree of G}

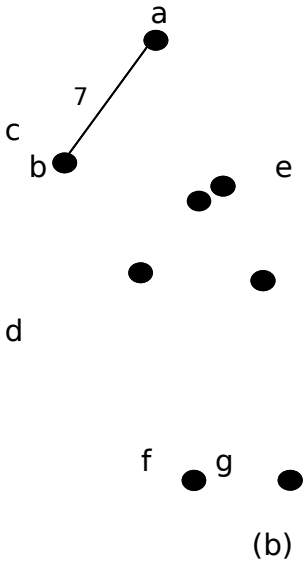
Example: Find the minimum spanning tree of the following graph using Kruskal's algorithm.



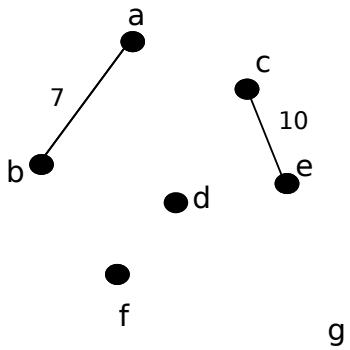
48

Bikash Balami Graph Theory

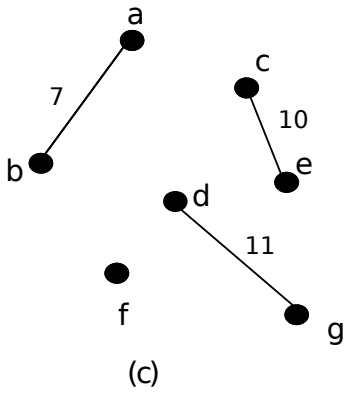
Solution:



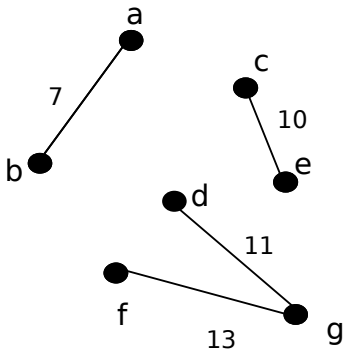
(a)



(b)



(c)

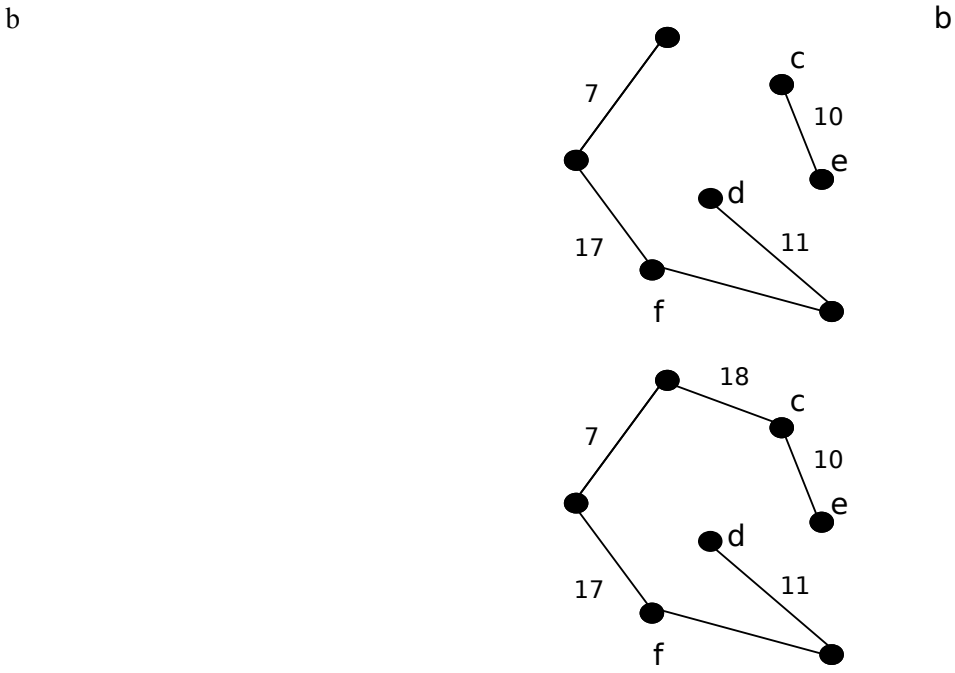


(d)

a a

49

Bikash Balami Graph Theory



(e) (f)

Prim's Algorithm

Working principle: This is a greedy algorithm that chooses optimal solution at a particular instance. Choose an edge of the smallest edge, put it into the spanning tree. Successively add to the tree edges of minimum weight that are incident to the vertex already in the tree and not forming a simple circuit. Stop when n-1 edges are added.

Algorithm:

Tree prim(G: connected weighted undirected graph with n vertices)

{

T = a minimum weight edge.

for i = 1 to n-2

{

50

Bikash Balami Graph Theory

e = an edge of minimum weight incident to a vertex in T and not forming a simple circuit in T if added to T

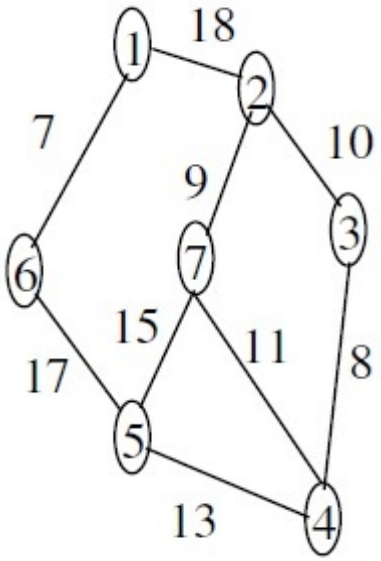
T = T with e added

}

}

Example:

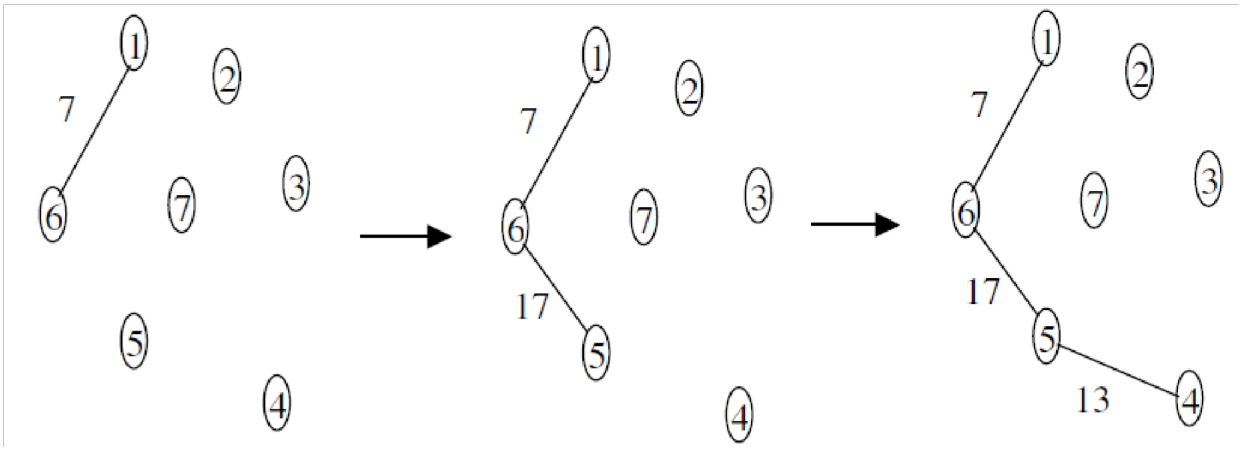
Find the minimum spanning tree of the following graph using Prim's algorithm.

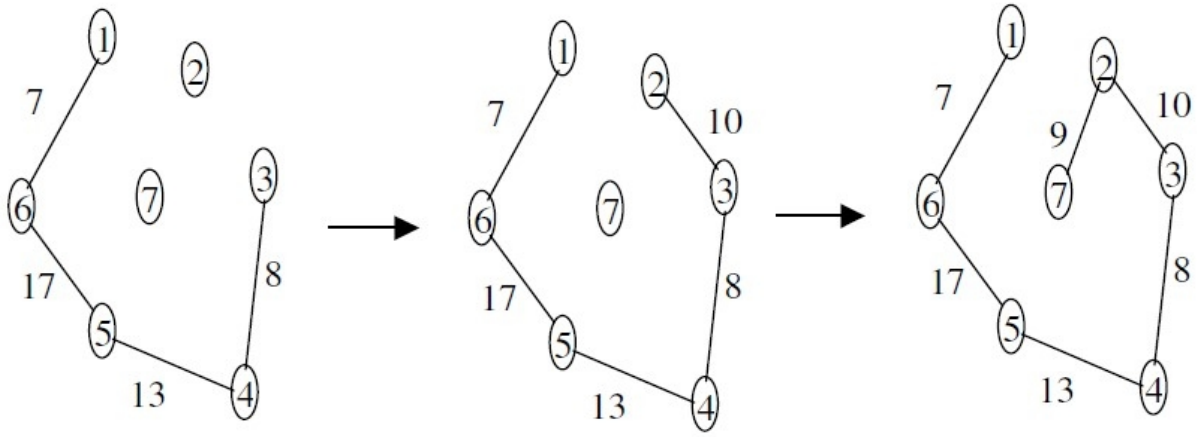


Solution

51

Bikash Balami Graph Theory





Bibliography

- Discrete Mathematics and its Applciations by Kenneth H Rosen
- Notes from Samujjwal Bhandari of CDCSIT, TU
- Notes from Nabaraj Poudel of Patan Multiple Campus, TU

52