COURSE PACK

FOR

GRAPH THEORY (2018 course)

COURSE CODE: 504
COURSE: BCA
SEMESTER: V
YEAR: 2021-22

Course Instructor: Mr. Amit Grover

Course Leader: Mr. Amit Grover

Forwarded by **Approved By**

HOD **Director(I/C)**

BharatiVidyapeeth(Deemed to be University)
Institute of Management & Research, New Delhi
An ISO 9001: 2015 Certified Institute
"A+" grade accreditations by NAAC

Note: "Strictly for Internal academic use only"

Table of content

S.No.	Content	Page Nos.
1	Latest course of Subject	3
2	Introduction ,Course Objective and LO	4
3	Course outline and schedule	5-7
4	Evaluation Criteria	8
5	Brief Profile of faculty	9
6	Unit- 1	12-39
7	Unit- 2	40-59
8	Unit- 3	60-85
9	Unit- 4	86-96
10	Unit- 5	97-143
11	Unit- 6	144-149

12	Question Papers	150-157

Course Objective :

The aims of this Graph theory is a delightful playground for the exploration of proof techniques in discrete mathematics and its results have applications in many areas of the computing ,social and natural science

Expected Outcome :

At the end of the course student should be able to:

- Use graphs as models in a variety of areas.
- Formulate several real world problems in mathematical terms

References (Books, Websites etc) :

Introduction to Graph theory - PHI by Douglas B.West

Discrete Mathematics and its Applications Edition 6th - Tata McGraw Hill by Kenneth H. Rosen

Suggested MOOC :

NPTEL

Course Plan

Unit	Contents
1	**Fundamental Concepts** : Definition, Graph Models, Sub Graph, Decomposition and special Graphs, Connection in Graphs, Bipartite Graph, Degree, Directed Graph, Undirected Graph, weighted graph, Regular Graph, dual graph, Representing Graph in computer memory, **Examples**
2	**Connectivity**: Walk, paths, trail, circuits, Connected Graph, Bridge, Isomorphism, Eulerian Circuits, Euler's path, Euler graph, Hamiltonian Graph and Graph Algorithm, Konigsberge Bridge problem, shortest path problems, city route,puzzle problem, Seating arrangement problem, Travelling salesman problem, **Examples**
3	**Algorithms** : Fleury's algorithm, Warshall's algorithm, Floyde's algorithm, Dijkstra's algorithm, Depth-First Search/ Breadth First search in Directed Graph, **Examples**
4	**Coloring of Graphs and planarity**: Vertex Coloring and upper bonds, Graph with Large Chromatic Number, 4 color theorem,Applications of graph coloring, Planar Graph, Euler's Formula, Homomorphism, Theorems, **Examples**
5	**Trees and Distance**: Concept of Trees, Definition and properties of Trees, Application of Trees, Trees as Models, Game Trees, Tree Traversal, Infix and Postfix notation of arithmetic expression, Binary Trees and its Properties, Binary Search Trees, Spanning Tree, Minimum spanning Tree, Depth First search, Breadth –First search, Back tracking applications, Kruskal algorithm, Prims algorithm, Huffman's algorithm **Excercises**
6	**Matchings** : Matching, Hall's Condition, MinMax Theorem, covers, Maximum Bipartite Matching, Weighted Bipartite Matching, Maximum Networks Flow, Examples

Introduction to course: It a fascinating journey to Graph Theory — an area which connects the elegance of painting and the rigor of mathematics; is simple, but not unsophisticated. Graph Theory gives us, both an easy way to pictorially represent many major mathematical results, and insights into the deep theories behind them. In this course, among other intriguing applications, we will see how GPS systems find shortest routes, how engineers design integrated circuits, how biologists assemble genomes, why a political map can always be colored using a few colors. We will study Ramsey Theory which proves that in a large system, complete disorder is impossible! By the end of the course, we will implement an algorithm which finds an optimal assignment of students to schools.

Course objectives:

1. To understand and apply the fundamental concepts in graph theory.
2. To apply graph theory based tools in solving practical problems.
3. To improve the proof writing skills.

Learning outcomes:

1. Ability to learn about different graphs and their degrees, also to represent graphs in computer memory
2. To solve different puzzles,travelling problems, paths and seating problems..
3. To learn algorithms to find the shortest paths.
4. To color graphs and also theorems which explains isomorphism.
5. Ability to know the concepts of trees.
6. Concepts of matching and how to implement it.

Course Outline and Schedule				
Schedule	**Topic to be discussed**	**Pedagogy**	**Required Reading/ Cases**	**Learning Outcome**
1	Introduction to Graphs	Introduction	Book Ch-13, 13.1	To understand about graphs.LO1/ CO1
2	Graphs,directed graphs,undirected graphs	Presentation	Ch-13, 13.3	To Learn about different graphs.LO1
3	undirected graphs, Degree Questions	Presentation/ Discussion	Ch-13, 13.4	Computation of degree.LO1/ CO2
4	Types of graphs	Discussion	Ch-13, 13.5	LO1
5	Subgraphs, Isomorphics graphs	Presentation	Ch-13, 13.6	To know mapping of graphs.LO1/ CO3
6	Representation of graphs	Presentation	Ch-13, 13.11	To represent graphs in computer memory. LO1
7	Questions	Practice questions	Course pack	LO1/CO2
8	Walks,trail,circuit	Discussion	Ch-13, 13.8	LO2
9	Paths, Connectedness	Discussion/ Questions	Ch-13,13.8	LO2/CO2
10	Konigsberg bridges, Euler graph	Presentation	Course pack	LO2
11	Hamiltonian graphs, Travelling salesman problems	Practical Applications	Notes	LO2/CO2
12	Seating arrangement problems, puzzles	Practical Applications	Notes	LO2/CO2

13	Shortest path problems	Practical Applications	Notes	Get to know about shortest paths.LO2
14	Fleury's Algorithm	Presentation/Question	Course pack	LO3
15	Djikstra's Algorithm	Presentation/ Question	Ch-13, 13.10	LO3/CO3
16	Breadth first search algorithm	Presentation/ Question	Course pack	LO3
17	Depth first search algorithm	Presentation	Course pack	To implement algorithms. LO3
18	Depth first search algorithm	Discussed Question	Course pack/Notes	LO3
19	Floyd Warshall's Algorithm	Presentation	Notes	LO3/CO3
20	Floyd Warshall's Algorithm	Discussed Question	Notes	LO3
21	Vertex Coloring and upper bounds	Presentation	Ch-13, 13.13	Know to color graphs.LO4/ CO1
22	Graphs with large chromatic numbers	Discussion	Course Pack	LO4
23	4 Color Theorem	Discussion	Course Pack	LO4/CO3
24	Applications of graph coloring	Application Based	Notes	To implement coloring.LO 4/CO2
25	Planar graphs, Euler's formula	Presentation	Ch-13, 13.12	LO4
26	Homomorphism theorem	Presentation/Question	Course Pack	LO4/CO3
27	Concept of trees, Definition and Properties	Presentation/ Discussion	Ch-14, 14.1,14.2	Ability to know about trees. LO5/CO1

28	Application of trees, Trees as models	Application Based	Course Pack	To implement coloring. LO5
29	Game tree, Tree traversal, Infix and postfix notations	Presentation	Ch-14, 14.5	LO5
30	Binary search trees, Spanning tree	Presentation	Ch-14, 14.4	LO5
31	Minimum spanning tree, DFS, BFS	Presentation	Ch-14, 14.3	LO5
32	Backtracking applications, Kruskal Algorithm	Presentation/ Discussion	Course Pack	To understand applications. LO5/CO1
33	Prim's Algorithm, Huffman's Algorithm	Presentation/ Discussion	Notes	LO5
34	Matching, Hall's condition, Covers	Presentation/ Discussion	13.15/ Course Pack	Ability to match things.LO6/ CO1
35	MinMax theorem, Maximum Bipartite matching	Presentation/ Discussion Questions	Notes	Knowledge of theorms.LO 6
36	Weighted Bipartite matching	Presentation	Ch-13, 13.15	LO6
37	Maximum Network flow	Presentation/ Questions	Ch-13, 13.15/Notes	LO6
38	Practice Test	Question Bank	Course Pack	CO2
39	Revision	Notes	Course Pack	Revisionary Session
40	Revision	Notes	Course Pack	Revisionary Session

Evaluation Criteria

- Three CES – 10 Marks each (Moodle quiz, Case study, Self-made presentation etc.) (Weightage 10%)
- Class attendance- 10 Marks (More than 75% in aggregate)
- Internal I Exam- 40 marks (Weightage- 10%)
- Internal II Exam- 40 marks (Weightage- 10%)
- University Exams- 100 marks (Weightage- 60%)

Note: Best of two CES will be counted if student will appear in all three CES

Recommended text book and reference books:

1. A Textbook of Discrete Mathematics by Swapan Kumar Sarkar.
2. Discrete Mathematics and its Applications Edition 6- Tata McGraw Hill by Kenneth H. Rosen.
3. Introduction to Graph Theory- PHI by Douglas B. West.
4. Discrete Mathematics and its Applications by Kenneth H. Rosen.

Online references:

For MOOCs:

- NPTEL
- https://alison.com/

Web resources:

- http://www.cs.cornell.edu/home/kleinber/networks-book
- www.graphtheory.com

Brief Profile of Mr. Amit Grover

Mr Amit Grover is a dedicated and experienced educational professional, currently working as a visiting faculty at Bharati VidyaPeeth Deemed University, New Delhi since 2008. A post graduate in commerce, with 20+ years of experience in teaching young adults across a wide range of age groups and diverse backgrounds. Adept at creating powerful curriculum in the field of Statistics and Mathematics.

Knowledgeable and experienced in various educational philosophies, which best promote the overall experience of a student. A committed faculty member, passionate about working to further enhance the educational offerings of an institution.

Graph Theory

Unit-1

13.1. Introduction

Many situations that occur in computer Science, Physical Science, Communication Science, Economics and many other areas can be analysed by using techniques found in a relatively new area of mathematics called graph theory. The graphs can be used to represent almost any problem involving discrete arrangements of objects, where concern is not with the internal properties of these objects but with relationship among them. In this chapter, we begin with some basic graph terminology and then discuss some important concepts in graph theory with many applications of graphs.

13.2. Basic Terminology

A graph G consists of two sets :

(*i*) a non-empty set V whose elements are called **vertices, nodes or points** of G. The set $V(G)$ is called the **vertex set** of G.

(*ii*) a set E of edges such that each edge $e \in E$ associated with ordered or unordered pairs of elements of V. The set E(G) is called the **edge set** of G.

The graph G with vertices V and edges E is written as G = (V, E) or G(V, E).

If an edge $e \in E$ is associated with an ordered pair (u,v) or an unordered pair (u,v), where $u, v \in V$, then e is said to **connect** u and v and u and v are called end points of e. An edge is said to be incident with the vertices it joins. Thus, the edge e that joins the nodes u and v is said to be **incident** on each of its end points u and v. Any pair of nodes that is connected by an edge in a graph is called **adjacent nodes.**

In a graph a node that is not adjacent to another node is called an **isolated node**.

A graph G (V, E) is said to be **finite** if it has a finite number of vertices and finite number of edges. (A graph with a finite number of vertices must also have finite number of edges): otherwise, it is a **infinite** graph, if G is a finite, | V (G) | denotes the number of vertices in G and is called the **order** of G. Similarly if G is finite, | E (G) | denotes the number of edges in G and is called the size of G. We shall often refer to a graph of order n and size m an (n, m) graph. If G be a (p, q) graph then G has p vertices and q edges.

Although graphs are frequently stored in a computer as list of vertices and edges, they are pictured as diagrams in the plane in a natural way. Vertex set of graph is represented as a set of points in a plane and edge is represented by a line segment or an arc (not necessarily straight). The objects shown in Fig. 13.1 are graphs.

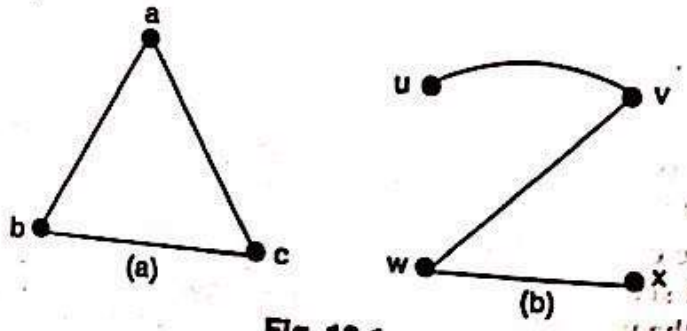


Fig. 13.1

Scanned with CamScanner

It helps when discussing a graph to label each vertex, often with lower case letters as shown above. In Fig. 13.1 (*a*), V = {*a*, *b*, *c*,} and E = {(*a*, *b*), (*a*, *c*), (*b*, *c*)} the member of vertices and edges are |v(G)| = 3 and |E(G)| = 3 in this graph, the vertices *a* and *b*, *a* and *c* and *b* and *c* are adjacent vertices.

In Fig. 13.1 (*b*), $V = \{u, v, w, x\}$ and $E = \{(u, v), (v, w), (w, x)\}$

Here vertices *u* and *v*, *v* and *w*, *w* and *x* are adjacent, whereas *u* and *w*, *u* and *x* and *v* and *x* are non adjacent. The number of vertices and edges are |v(G)| = 4 and |E(G)| = 3.

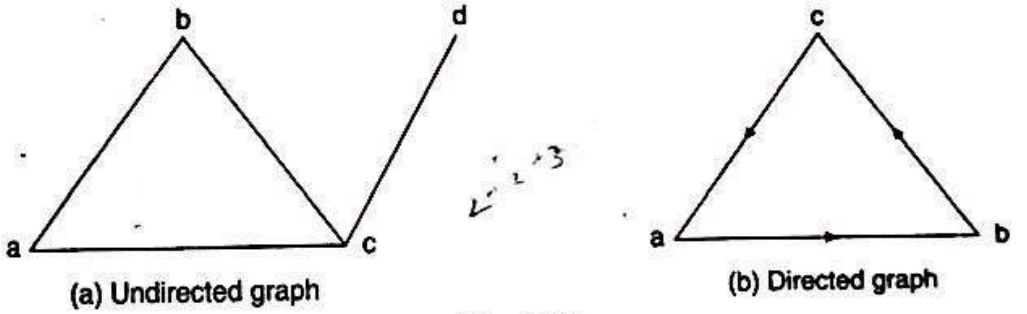
The definition of a graph contains no reference to the length or the shape and the positioning of the edge or arc joining any pair of nodes, nor does it prescribe any ordering of positions of the nodes. Therefore, for a given path, there is no unique diagram that represents the graph, and it can happen that two diagrams that look entirely different from one another may represent the same graph. It is to be noted that, in drawing a graph, it is immaterial whether the lines are drawn straight or curved, long or short, what is important is the incidence between edges and vertices are the same in both cases.

Undirected and Directed Graph

An **undirected** graph G consists of set V of vertices and a set E of edges such that each edge $e \in E$ is associated with an unordered pair of vertices.

Fig. 13.2 (*a*) is an example of an undirected graph we can refer to an edge joining the vertex pair *i* and *j* as either (*i*, *j*) or (*j*, *i*).

A **directed graph (or digraph)** *G* consists of a set V of vertices and a set E of edges such that $e \in E$ is associated with an ordered pair of vertices. In other words, if each edge of the graph G has a direction then the graph is called directed graph. In the diagram of directed graph, each edge $e = (u, v)$ is represented by an arrow or directed curve from initial point *u* of *e* to the terminal point *v*. Fig. 13.2 (*b*) is an example of a directed graph.



(a) Undirected graph (b) Directed graph

Fig. 13.2

Suppose $e = (u,v)$ is a directed edge in a digraph, then

(*i*) *u* is called the **initial vertex** of *e* and v is the **terminal vertex** of *e*

(*ii*) *e* is said to be **incident from** *u* and to be **incident** to v.

(*iii*) *u* is adjacent to v, and v is **adjacent from** *u*

In specifying any edge of a digraph by its end-points, the edge is understood to be directed from the first vertex towards the second.

Graphs, both directed and undirected, occur widely in all sorts of problems and before introducing more terminology we give examples of how graph arise in some familiar contexts.

Scanned with CamScanner

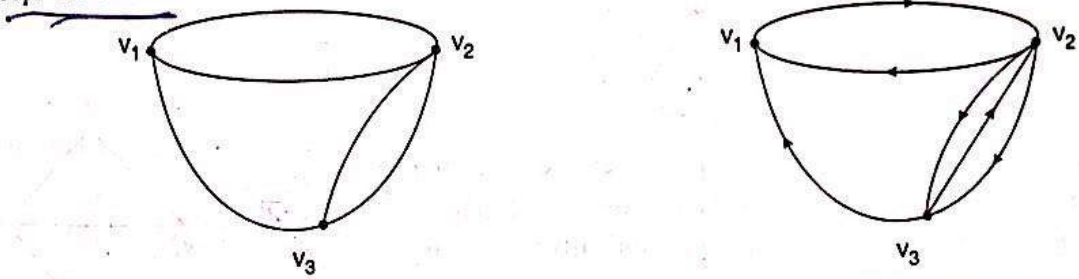
13.3. Simple Graph, Multigraph and Psuedograph

An edge of a graph that joins a node to itself is called a **loop** or **self loop** *i.e.*, a loop is an edge (v_i, v_j) where $v_i = v_j$.

In some directed as well as undirected graphs, we may have contain pair of nodes joined by more than one edges, such edges are called **multiple** or **parallel** edges. Two edges (v_i, v_j) and (v_f, v_r) are parallel edges if $v_i = v_f$ and $v_j = v_r$. Note that in case of directed edges, the two possible edges between a pair of nodes which are opposite in direction are considered distinct. So more than one directed edge in a particular direction in the case of a directed graph is considered parallel.

A graph which has neither loops nor multiple edges *i.e.*, where each edge connects two distinct vertices and no two edges connect the same pair of vertices is called a **simple graph**. Fig. 13.2 (*a*) and (*b*) represents simple undirected and directed graph because the graphs do not contain loops and the edges are all distinct.

Any graph which contains some multiple edges is called a **multigraph.** In a multigraph, no loops are allowed.



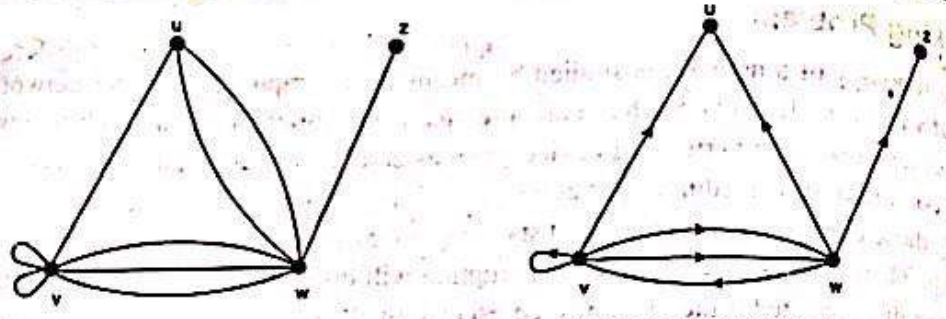
(a) Undirected multigraph (b) Directed multigraph

Fig. 13.7

In Fig. 13.7 (*a*) there are two parallel edges joining nodes v_1 and v_2 and v_2 and v_3. In Fig. 13.7. (*b*), there are two parallel edges associated with v_2 and v_3.

A graph in which loops and multiple edges are allowed, is called a **pseudograph.**

Scanned with CamScanner



(a) Undirected pseudograph (b) Directed pseudograph

Fig. 13.8

It may be noted that there is some lack of standardisation of terminology in graph theory. Many words have almost obvious meaning, which are the same from book to book, but other terms are used differently by different authors.

13.4. Degree of a Vertex

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

The degrees of vertices in the graph G and H in Fig. 13.9 are given below.

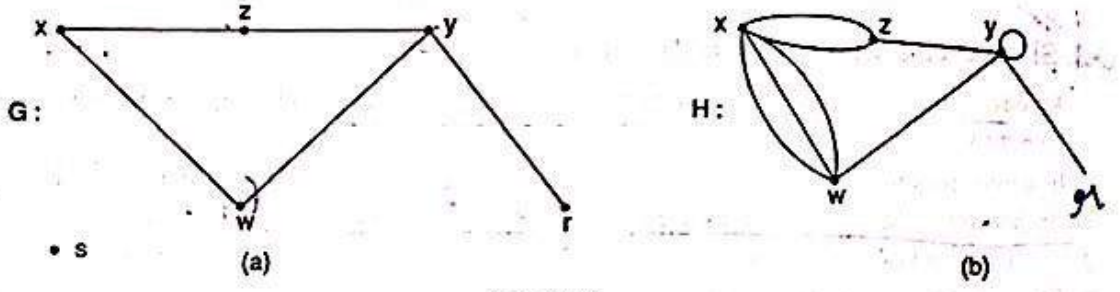


Fig. 13.9

In G as shown in Fig. 13.9 (*a*)

$\deg_G(x) = 2 = \deg_G(z) = \deg_G(w), \deg_G(y) = 3$ and $\deg_G(r) = 1$ and in H as shown in Fig. 13.9 (*b*), $\deg_H(x) = 5$, $\deg_H(z) = 3$, $\deg_H(y) = 5$, $\deg_H(w) = 4$ and $\deg_H(r) = 1$.

A vertex of degree 0 is called **isolated vertex**. A vertex is **pendant** if and only if it has a degree 1. Vertex s in the graph G is isolated and vertex r is pendant. A vertex of a graph is called odd vertex or even vertex depending on whether its degree is odd or even.

In any graph G, we define

$$\delta(G) = \min\{\deg v : \in V(G)\} \text{ and}$$

$$\Delta(G) = \max\{\deg v : \in V(G)\}$$

If $v_1, v_2, \ldots\ldots v_n$ are the n vertices of G, then the sequence $(d_1, d_2, \ldots\ldots d_n)$ where $d_i = \deg(v_i)$ is the degree sequence of G. In general, we order the vertices so that the degree sequence is monotonically increasing *i.e.*

$$\delta(G) = d_1 \le d_2 \le \ldots\ldots \le d_n = \Delta(G).$$

For example, the degree sequence of the graph shown in Fig. 13.10 is (2,2,3,5) as $\deg(v_2) = \deg(v_4) = 2$, $\deg(v_3) = 3$ and $\deg(v_1) = 5$.

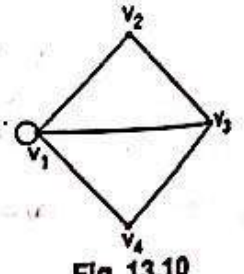


Fig. 13.10

Theorem 13.1 A simple graph with at least two vertices has at least two vertices of same degree.

Proof. Let G be a simple graph with $n \ge 2$ vertices. The graph G has no loop and parallel edges.

Scanned with CamScanner

Hence the degree of each vertices is $\leq n-1$. Suppose all the vertices of G are of different degrees. Hence the following degrees

$$0, 1, 2, 3, \ldots\ldots, n-1$$

are possible for n vertices of G. Let u be the vertex with degree 0. Then u is an isolated vertex. Let v be the vertex with degree $n-1$ then v has $n-1$ adjacent vertices. Since v is not an adjacent vertex of itself, therefore every vertex of G other than u is an adjacent vertex of G. Hence u cannot be an isolated vertex, this contradiction proves that a simple graph contains two vertices of same degree.

The converse of the above theorem is not true.

Theorem (the Handshaking theorem) 13.2. If G = (V, E) be an undirected graph with e edges. Then

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

i.e., the sum of degrees of the vertices in an undirected graph is even.

Proof: Since the degree of a vertex is the number of edges incident with that vertex, the sum of the degree counts the total number of times an edge is incident with a vertex. Since every edge is incident with exactly two vertices, each edge gets counted twice, once at each end. Thus the sum of the degrees is equal twice the number of edges.

Note: This theorem applies even if multiple edges and loops are present. The above theorem holds this rule that if several people shake hands, the total number of hands shake must be even that is why the theorem is called handshaking theorem.

Corollary: In a non directed graph, the total number of odd degree vertices is even.

Proof: Let G = (V, E) a non directed graph. Let U denote the set of even degree vertices in G and W denote the set of odd degree vertices.

Then $\sum_{v_i \in V} \deg_G(v_i) = \sum_{v_i \in U} \deg_G(v_i) + \sum_{v_i \in W} \deg_G(v_i)$

$\Rightarrow$ $2e - \sum_{v_i \in U} \deg_G(v_i) = \sum_{v_i \in W} \deg_G(v_i)$ ...(1)

Now $\sum_{v_i \in U} \deg_G(v_i)$ is even as the sum of degrees of even degree vertices is always even.

Therefore, from (1)

$\sum_{v_i \in W} \deg_G(v_i)$ is even

$\therefore$ Since for each $v_i \in W$, $\deg_G(v_i)$ is odd, the number of odd vertices in G must be even.

In degree and out degree

In a directed graph G, the out degree of a vertex v of G, denoted by $\text{outdeg}_G(v)$ or $\deg_G^+(v)$, is the number of edges beginning at v and the indegree of v, denoted by $\text{indeg}_G(v)$ or $\deg_G^{-1}(v)$, is the number of edges ending at v. The sum of the in degree and out degree of a vertex is called the **total degree** of the vertex. A vertex with zero in degree is called a **source** and a vertex with zero out degree is called a **sink.** Since each edge has an initial vertex and terminal vertex, the immediate theorem follows:

Theorem 13.3. If G = (V,E) be a directed graph with e edges, then

$$\sum_{v \in V} \deg_G^+(v) = \sum_{v \in V} \deg_G^-(v) = e$$

Scanned with CamScanner

i.e., the sum of the outdegrees of the vertices of a diagraph G equals the sum of in degrees of the vertices which equals the number of edges in G.

Proof : Any directed edge (u,v) contributes 1 to the in degree of u and 1 to the out degree of v. Further, a loop at v contributes 1 to the in degree and 1 to the out degree of v. Hence the proof.

In the directed graph G in Fig. 13.11.

$\text{Indeg}_G(a) = 2$, $\text{Indeg}_G(b) = 1$, $\text{Indeg}_G(c) = 2$, $\text{Indeg}_G(d) = 3$.

$\text{Outdeg}_G(a) = 1$, $\text{Outdeg}_G(b) = 5$, $\text{Outdeg}_G(c) = 1$, $\text{Outdeg}_G(d) = 1$.

Note that, the sum of the in degrees and the sum of the out degrees each equal to 8, the number of edges.

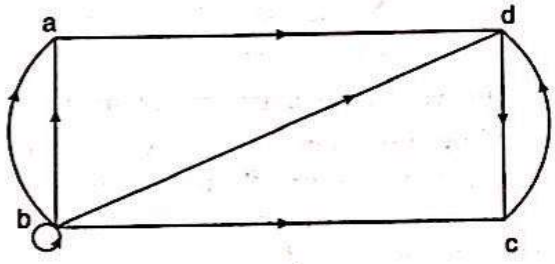
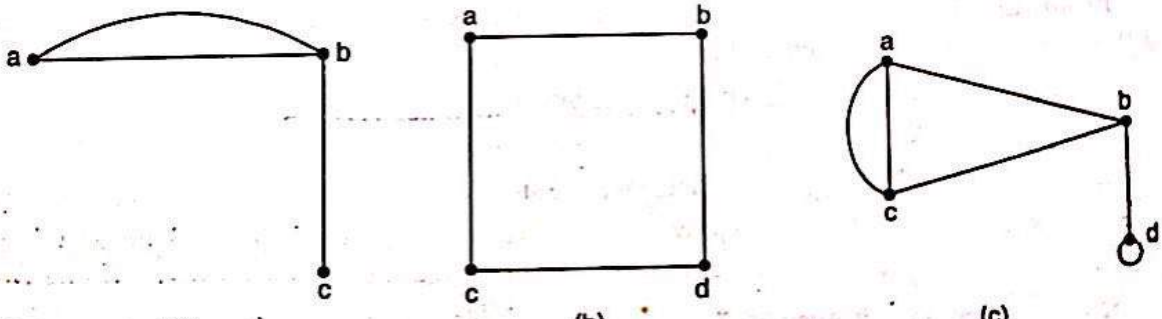


Fig. 13.11

SOLVED EXAMPLES

Example 1. State which of the following graphs are simple ?



(a) (b) (c)

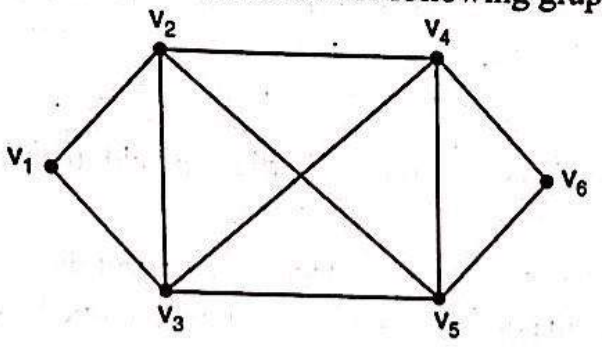
Solution. (*a*) The graph is not a simple graph, since it contains parallel edge between two vertices *a* and *b*.

(*b*) The graph is a simple graph, it does not contain loop and parallel edge.

(*c*) The graph is not a simple graph since it contains parallel edge and a loop.

Scanned with CamScanner

Example 3. Find the degree of each vertex of the following graph.

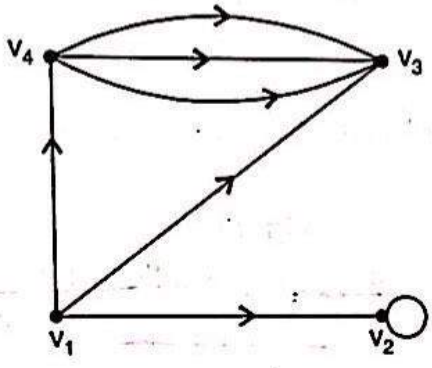


Solution. It is an undirected graph. Then

$\deg(v_1) = 2$, $\deg(v_2) = 4$, $\deg(v_3) = 4$

$\deg(v_4) = 4$ $\deg(v_5) = 4$, $\deg(v_6) = 2$

Example 4. Find the in degree out degree and of total degree of each vertex of the following graph.



Solution. It is a directed graph

in deg $(v_1) = 0$	out degree $(v_1) = 3$	total deg $(v_1) = 3$
in deg $(v_2) = 2$	out degree $(v_2) = 1$	total deg $(v_2) = 3$
in deg $(v_3) = 4$	out degree $(v_3) = 0$	total deg $(v_3) = 4$
in deg $(v_4) = 1$	out degree $(v_4) = 3$	total deg $(v_4) = 4$

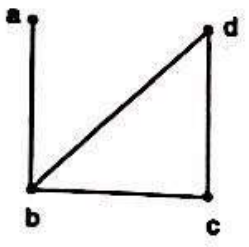
Scanned with CamScanner

GRAPH THEORY PROBLEM SET

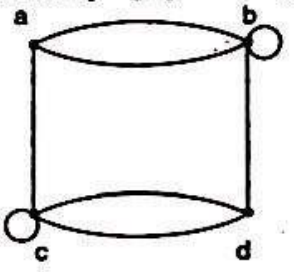
Problem Set

1. Determine whether the graphs shown is a simple graph a multigraph, a pseudograph.

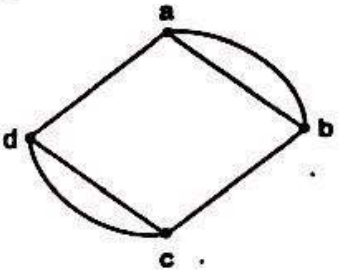
(*a*)



(*b*)

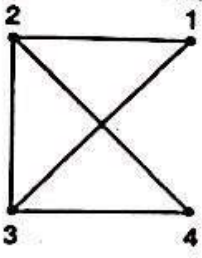


(*c*)

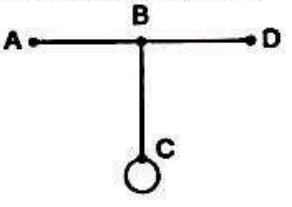


2. Consider the following graphs, determine the (*i*) vertex set and (*ii*) edge set.

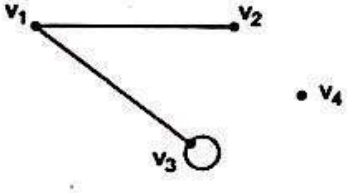
(*a*)



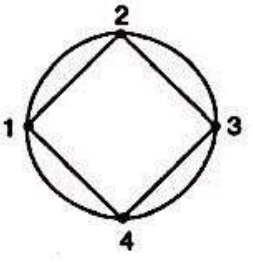
(*b*)



(*c*)

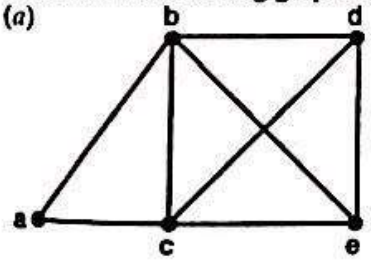


(*d*)

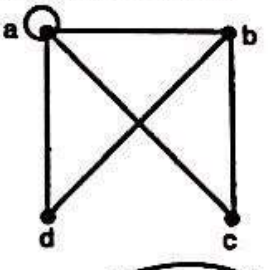


3. Consider the following graphs and determine the degree of each vertex.

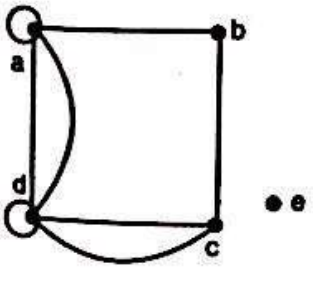
(*a*)



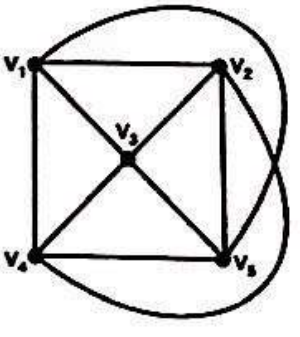
(*b*)



(*c*)



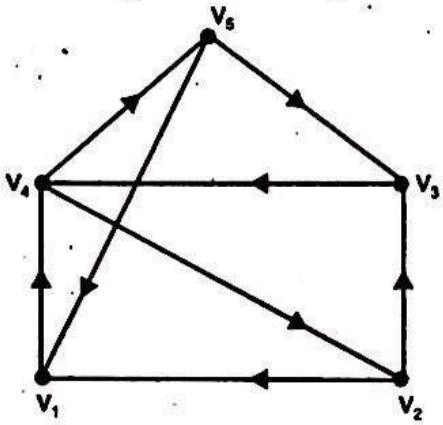
(*d*)



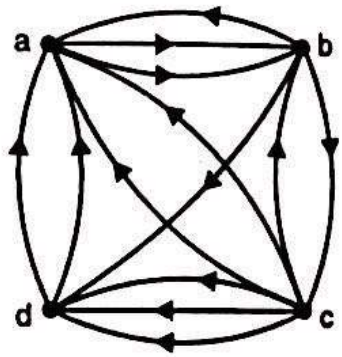
Scanned with CamScanner

4. Find the in-degree and out-degree of each vertex of the following directed graphs

(*c*)



(*d*)



5. Draw a graph having the given properties or explain why no such graph exists.
   (*a*) Graph with four vertices of degree 1,1,2 and 3.
   (*b*) Graph with four vertices of degree 1,1,3 and 3.
   (*c*) Simple graph with four vertices of degree 1,1,3 and 3.
   (*d*) Graph with six vertices each of degree 3
   (*e*) Graph with six vertices and four edges.
   (*f*) Graph with five vertices of degree 3, 3, 3, 3, 2
   (*g*) Graph with five vertices of degree 0, 1, 2, 2, 3 20

Scanned with CamScanner

13.5. Types of Graphs

Some important types of graphs are introduced here. These graphs are often used as examples and arise in many applications.

Null Graph

A graph which contains only isolated node is called a **null graph** *i.e.*, the set of edges in a null graph is empty. Null graph is denoted on n vertices by N_n : N_4 is shown in Fig. 13.12. Note that each vertex of a null graph is isolated.

n = max. edge $\frac{n(n-1)}{2}$

degree = (n−1)

Fig. 13.12

Complete Graph

A simple graph G is said to be complete if every vertex in G is connected with every other vertex *i.e.*, if G contains exactly one edge between each pair of distinct vertices. A complete graph is usually denoted by K_n. It should be noted that K_n has exactly $\frac{n(n-1)}{2}$ edges. The graphs K_n for $n = 1,2,3,4,5,6$ are shown in Fig. 13.13.

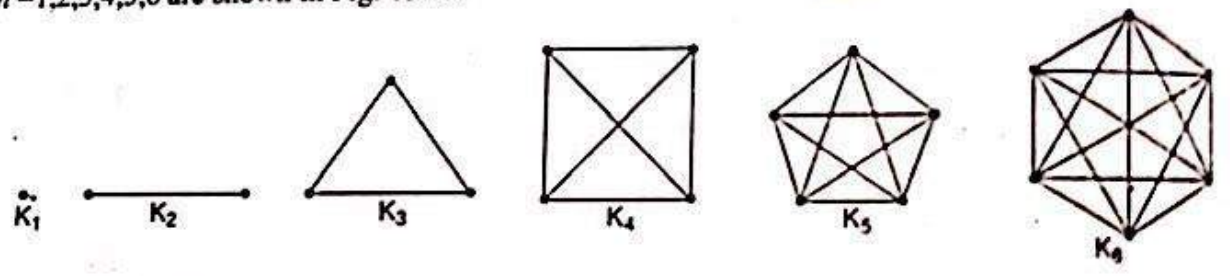
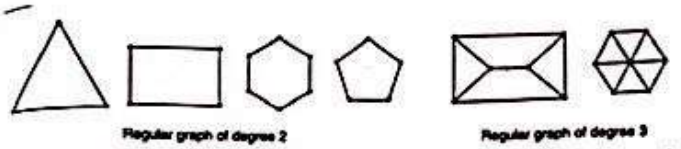


Fig. 13.13

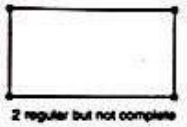
Regular Graph (equal vertex degree)

A graph in which all vertices are of equal degree is called a regular graph. If the degree of each vertex is r, then the graph is called a **regular graph of degree** r. Note that every null graph is regular of degree zero, and that the complete graph K_n is a regular of degree $n-1$. Also, note that if G has n vertices and is regular of degree r, then G has $(1/2)\,r\,n$ edges.

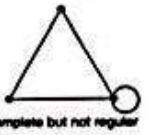


Regular graph of degree 2 Regular graph of degree 3

Note that complete graph need not be regular and a regular graph need not be complete as the following examples illustrate.



2 regular but not complete



Complete but not regular

Scanned with CamScanner

Cycles $n \geq 3$

The cycle graph C_n ($n \geq 3$), of length n is a connected graph which consists of n vertices $v_1, v_2 \ldots\ldots v_n$ and n edges $\{v_1, v_2\}$, $\{v_2, v_3\}$, ... $\{v_{n-1}, v_n\}$, and $\{v_n, v_1\}$. The cycles C_3, C_4, C_5, and C_6 are shown in Fig. 13.15.

C_n

n vertices
n edges

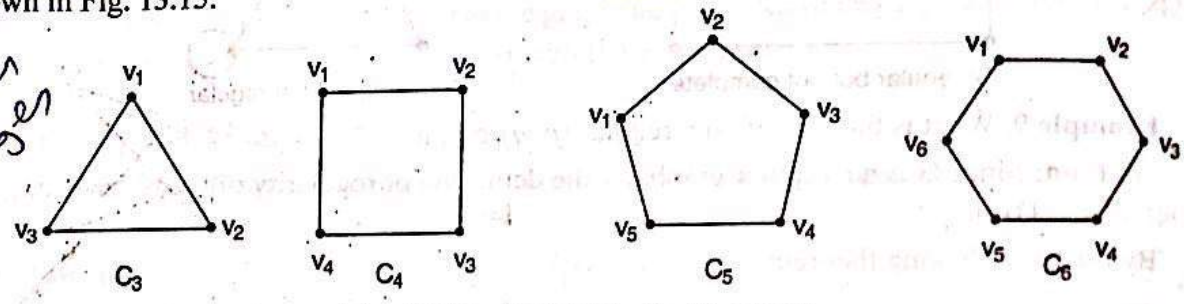


Fig. 13.15. Cycles C_3, C_4, C_5 and C_6

C_n is a regular graph of degree 2.

Wheels $n > 3$ W_n $n+1$ vertices. $2n$ edges

The wheel graph W_n ($n > 3$) is obtained from C_n by adding a vertex v inside C_n and connecting it to every vertex in C_n. The wheels W_3, W_4, W_5, and W_6 are displayed in Fig. 13.16.

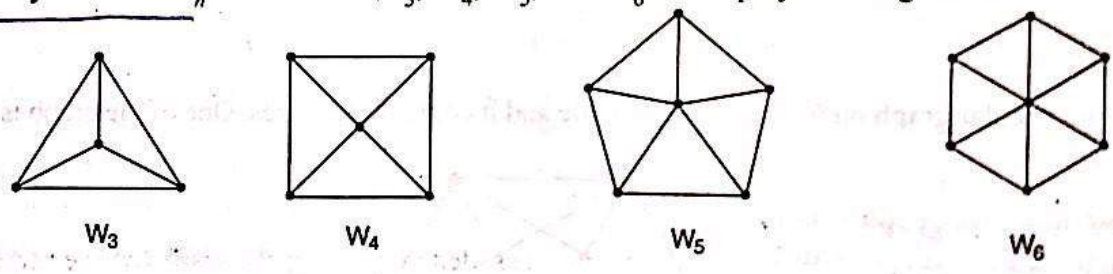


Fig. 13.16. The Wheels W_3, W_4, W_5 and W_6

W_n is a regular graph for $n = 3$. It has $n + 1$ vertices and $2n$ edges.

Scanned with CamScanner

Bipartite Graph

A graph $G = (V, E)$ is bipartite if the vertex set V can be partitioned ino two subsets (disjoint) V_1 and V_2 such that every edge in E connects a vertex in V_1 and a vertex V_2 (so that no edge in G connects either two vertices in V_1 or two vertices in V_2). (V_1, V_2) is called a bipartition of G. Obviously, a bipartite graph can have no loop.

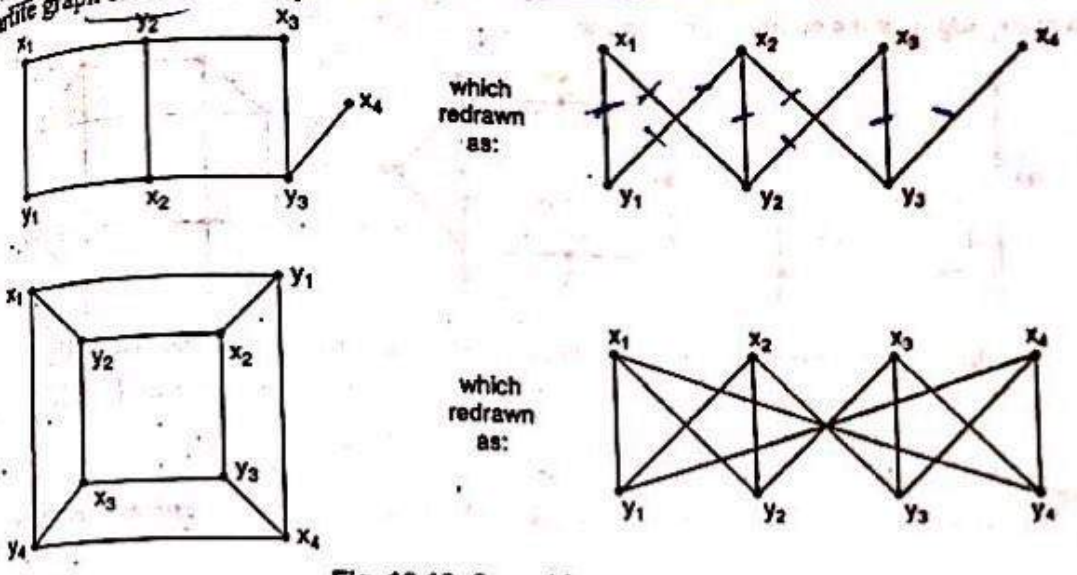
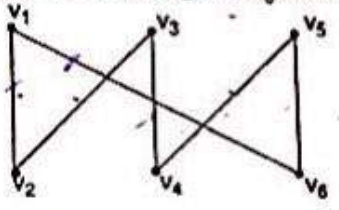


Fig. 13.18. Some bipartite graphs

Example 11. Show that C_6 is a bipartite graph.

Solution. C_6 in Fig. 13.15 is a bipartite graph since its vertex set can be partitioned into the two sets $V_1 = \{v_1, v_3, v_5\}$ and $V_2 = \{v_2, v_4, v_6\}$, and every edge of C_6 connects a vertex in V_1 and a vertex in V_2.



Note : Qn is a bipartite graph

Complete bipartite graph $K_{m,n}$

r + s vertices
rs edges.

The complete bipartite graph on m and n vertices, denoted $K_{m,n}$ is the graph, whose vertex set is partitioned into sets V_1 with m vertices and V_2 with n vertices in which there is an edge between each pair of vertices v_1 and v_2 where v_1 is in V_1 and v_2 is in V_2. The complete bipartite graphs $K_{2,3}$, $K_{2,4}$, $K_{3,3}$, $K_{3,5}$ and $K_{2,6}$ are shown in Fig. 13.19. Note that $K_{r,s}$ has $r + s$ vertices and rs edges.

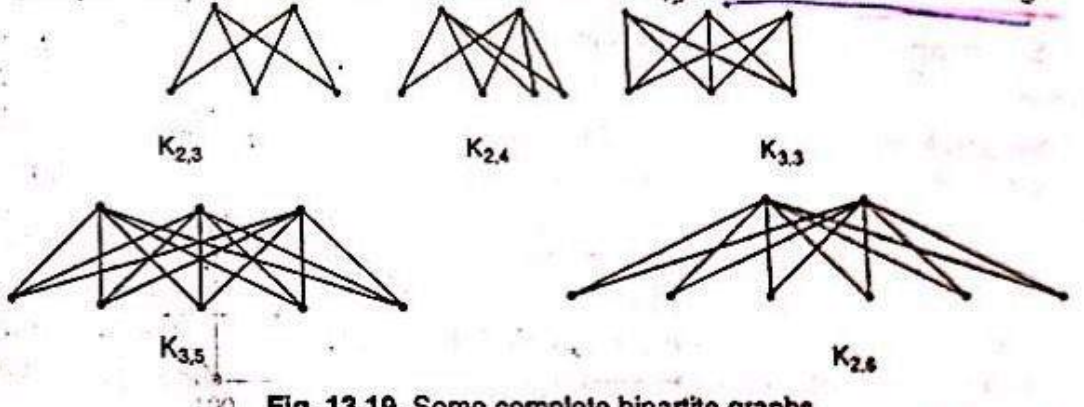


Fig. 13.19. Some complete bipartite graphs

Note :

1. Any graph $K_{1,n}$ is called a star graph.
2. A complete bipartite graph $K_{m,n}$ is not a regular if $m \neq n$.
3. K_5 and $K_{3,3}$ are called kuaratowski graphs.

Scanned with CamScanner

13.6. Subgraphs and Isomorphic Graphs

Subgraph

Some graph applications are concerned with only parts of a graph. For instance, we may be interested about the part of a large computer network that involves the computer centres. In the graph model, we can remove the vertices corresponding to the computer centre other than of our interest and we can remove all edges incident with a vertex that was removed. A smaller graph is obtained. Such graph is called a subgraph of the original graph.

Consider a graph $G = (V, E)$. A graph $H = (V', E')$ is called a subgraph of G if the vertices and edges of H are contained in the vertices and edges of G, that is, if $V' \subseteq V$ and $E' \subseteq E$. In particular.

(*i*) If $V' \subset V$ and $E' \subset E$, then H is called a **proper subgraph** of G.

(*ii*) A subgraph H of G, is called a **spanning subgraph** of G if and only if $V(H) = V(G)$.

(*iii*) A subgraph H (V', E') of $G(V, E)$ is called the **induced subgraph** of G if its edge set E' contain all edges in G whose end points belong to vertices in G.

GRAP

(*iv*) **If a subset U of V and all the edges incident on the elements of U are deleted from a graph $G(V, E)$, then the resulting subgraph is called a vertex deleted subgraph of $G(V, E)$.**

(*v*) **If a subset S of E from a graph $G(V, E)$ is deleted, then the resulting subgraph is called an edge deleted subgraph *of* G.**

For a given graph G, there can be many subgraphs. Let $|V| = m$ and $|E| = n$. The total non-empty subsets of V is $2^m - 1$ and total subsets of E is 2^n. Thus, number of subgraph is equal to $(2^m - 1) \times 2^n$.

Then number of spanning subgraph is equal to 2^n because all vertices are to be included in a spanning subgraph. For example, in Q_3 we have $|V| = 8$, $|E| = 12$. Then total number of subgraphs is

$$(2^8 - 1) \times 2^{12} = 127 \times 4096 = 520192$$

and the total number of spanning subgraphs is $2^{12} = 4096$.

So, if H is a subgraph of G, then Note .

(*i*) All the vertices of H are in G.

(*ii*) All the edges of H are in G and

(*iii*) Each edge of H has the same end points in H as in G.

Any subgraph of a graph G can be obtained by removing certain vertices and edges from G. It is understood that the removal of an edge leaves its points in place, whereas the removal of a vertex necessitates the removal of any edges with that vertex as an end point.

Ex. 14. Consider the graph shown in Fig. 13.20. Find the different subgraphs of this graph.

Scanned with CamScanner

Example 14. Consider the graph shown in Fig. 13.20. Find the different subgraphs of this graph.

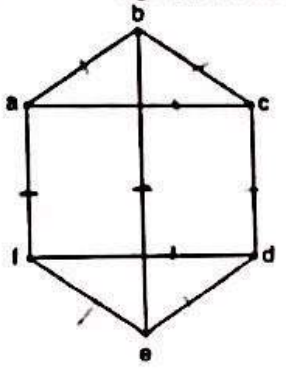
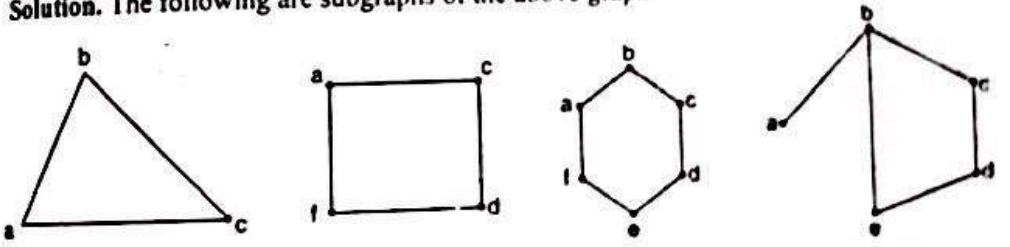


Fig. 13.20

Solution. The following are subgraphs of the above graph.



Example 15. Find the two subgraphs of Fig. 13.21(a) and one subgraph of Fig. 13.21(b).

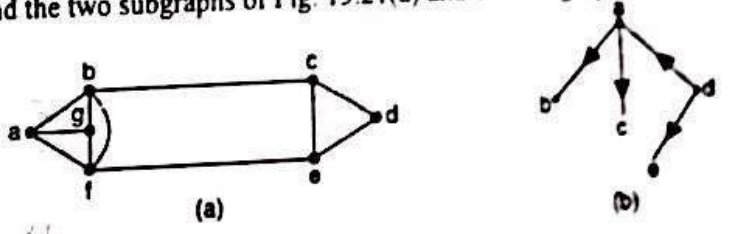


Fig. 13.21

Solution : Two subgraphs of the undirected graph and one subgraph of the directed graph are shown in Fig. 13.22.

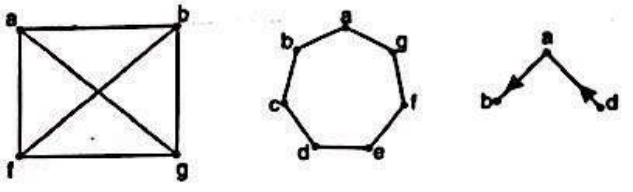


Fig. 13.22

Note. that the subgraphs do not have to be drawn the same way they appear in the presentation of G.

Scanned with CamScanner

Isomorphic Graph

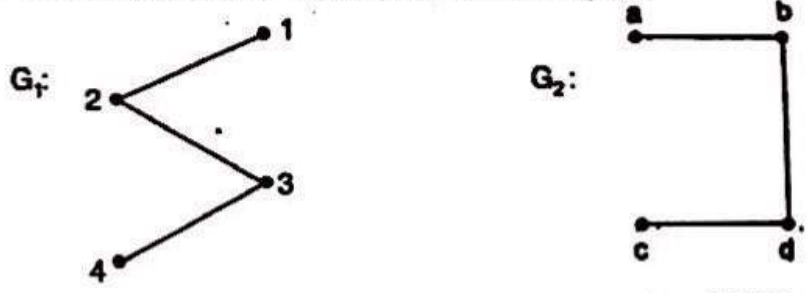
Two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are said to be isomorphic if there exists a functions $f: V_1 \to V_2$ such that

(*i*) f is one-to-one onto *i.e.*, f is bijective.

(*ii*) $\{a, b\}$ is an edge in E_1, if and only if $\{f(a), f(b)\}$ is an edge in E_2 for any two elements $a, b \in V_1$.

The condition (*ii*) says that if vertices a and b are adjacent in G_1 then $f(a)$ and $f(b)$ are adjacent in G_2. In other words the function f preserves adjacency relationship and consequently the corresponding vertices in G_1 and G_2 will have the same degree. Any function f with the above properties is called an **isomorphism** between G_1 and G_2.

Example 19. Show that the given pair of graphs are isomorphic



Solution. Here, $V(G_1) = \{1,2,3,4\}$, $V(G_2) = \{a,b,c,d\}$, $E(G_1) = \{\{1,2\}, \{2,3\}, \{3,4\}\}$ and $E(G_2) = \{\{a,b\}, \{b,d\}, \{d,c\}\}$. Hence $|V(G_1)| = |V(G_2)|$ and $|E(G_1)| = |E(G_2)|$

The vertices of degree 1 in G_1 are $\{1, 4\}$ and

The vertices of degree 1 in G_2 are $\{a, c\}$

The vertices of degree 2 in G_1 are $\{2, 3\}$ and

The vertices of degree 2 in G_2 are $\{b, d\}$

Define a function $f: V(G_1) \to V(G_2)$ as

$f(1) = a, f(2) = b, f(3) = d$ and $f(4) = c$.

f is clearly one-one and onto.

Further,

$\{1,2\} \in E(G_1)$ and $\{f(1), f(2)\} = \{a, b\} \in E(G_2)$

$\{2,3\} \in E(G_1)$ and $\{f(2), f(3)\} = \{b, d\} \in E(G_2)$

$\{3,4\} \in E(G_1)$ and $\{f(3), f(4)\} = \{d, c\} \in E(G_2)$

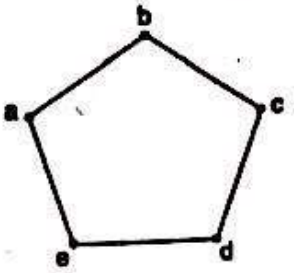
Hence f preserves adjacency of the vertices.

Therefore, G_1 and G_2 are isomorphic.

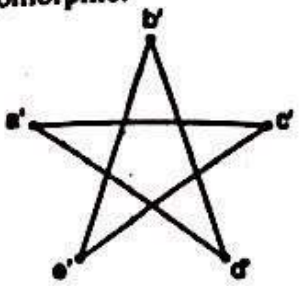
Scanned with CamScanner

Example 20. Show that the given pair of graphs are isomorphic.

G:



H:



Solution: Here $V(G) = \{a, b, c, d, e\}$, $V(H) = \{a', b', c', d', e'\}$, so $|V(G)| = |V(H)|$

Also, $|E(G)| = |E(H)|$

The vertices of degree 2 in G are $\{a, b, c, d, e\}$ and

The vertices of degree 2 in H are $\{a', b', c', d', e'\}$

Define a function $f: V(G) \to V(H)$ as

$f(a) = a', f(b) = c', f(c) = e'$ $f(d) = b'$ and $f(e) = d'$

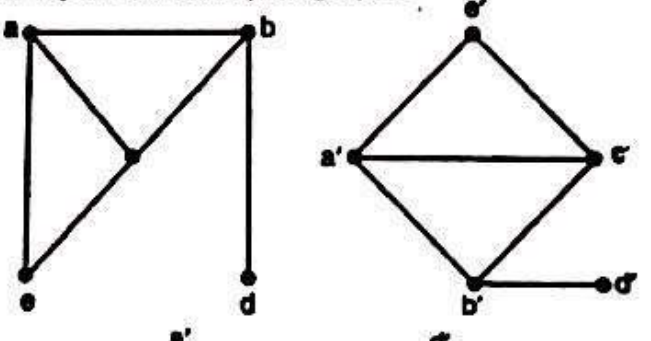
f is clearly one-one and onto.

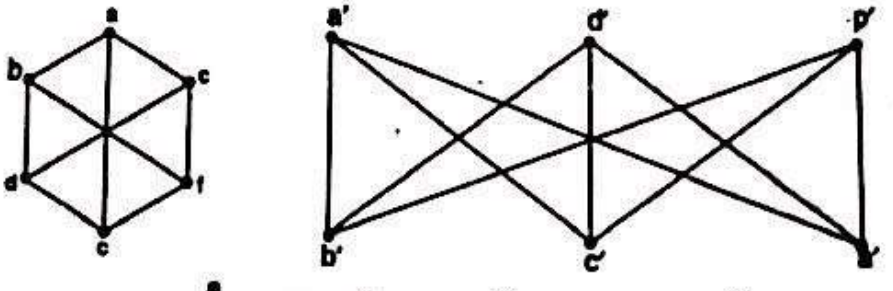
Further,

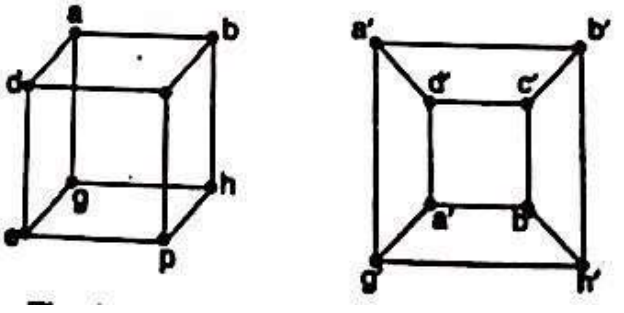
$\{a, b\} \in E(G)$	and	$\{f(a), f(b)\} = \{a', c'\} \in E(H)$
$\{a, e\} \in E(G)$	and	$\{f(a), f(e)\} = \{d', d'\} \in E(H)$
$\{b, c\} \in E(G)$	and	$\{f(b), f(c)\} = \{c', e'\} \in E(H)$
$\{c, d\} \in E(G)$	and	$\{f(c), f(d)\} = \{e', b'\} \in E(H)$
$\{d, e\} \in E(G)$	and	$\{f(d), f(e)\} = \{b', d'\} \in E(H)$

Hence f preserves adjacency of the vertices. Therefore, G and H are isomorphic.

We hereby give some examples of isomorphic graphs.







Scanned with CamScanner

A property shared by isomorphic graphs is called an **isomorphism invariant.** We may list some of the following invariants.

If $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$ are isomorphic graphs, then G_1 and G_2 have the

(*i*) same number of vertices, *i. e.*, $|V_1| = |V_2|$

(*ii*) same number of edges, *i.e.*, $|E_1| = |E_2|$

(*iii*) same degree sequences *i.e.*, if the degree of a vertex v_i in G_1 is m, then the degree of the vertex $f(v_i)$ in G_2 must also be m.

(*iv*) If $\{v, v\}$ is a loop in G_1 then $\{f(v), f(v)\}$ is also a loop in G_2.

If any of these quantities differ in two graphs, they can not be isomorphic. However, these conditions are by no means sufficient. For instance, the two graphs shown in Fig. 13.26 satisfy all three conditions, yet they are not isomorphic. That the graphs in Fig. 13.26 (*a*) and (*b*) are not isomorphic can be shown as follows : if the graph in Fig. 13.26 (*a*) are to be isomorphic to the one in (*b*), vertex x must correspond to y, because there are no other vertices of degree three. Now in (*b*) there is only one pendant vertex w, adjacent to y, while in (*a*) there are two pendant vertices u and v, adjacent to x.

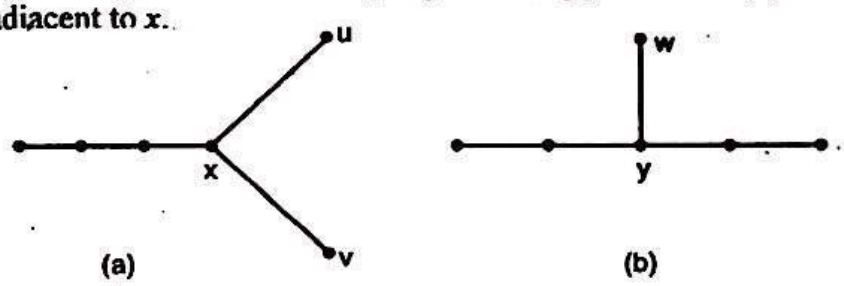
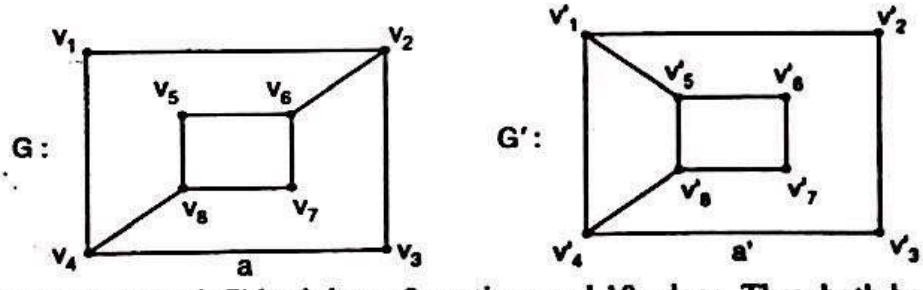


Fig. 13.26. Two graphs that are not isomorphic.

Example 21. Determine whether the following graphs are isomorphic.



Solution. The graphs G and G' both have 8 vertices and 10 edges. They both have 4 vertices each of degree 3 and 4 vertices each of degree 2.

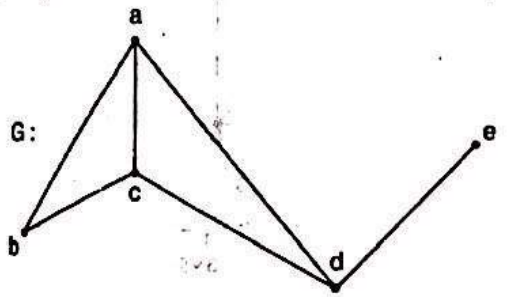
Now consider deg $(v_1) = 2$ in G. Then v_1 must correspond to either v_2', v_3', v_6', v_7', since these are vertices of degree 2 in G.

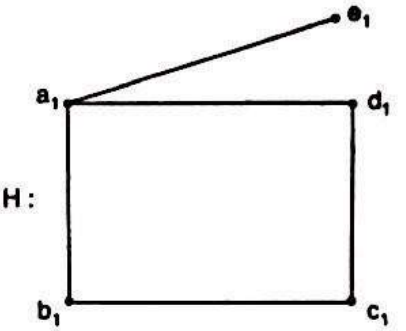
However, each of these vertices in G' is adjacent to another vertex of degree 2 in G' viz. v_2' is adjacent to v'_3, v'_6 is adjacent to v'_7 but v_1 is adjoint to v_2 and v_4 in G which are of degree 3. Thus the preservation of adjacency of the vertices is not maintained.

$\therefore$ G and G' are not isomorphic graphs.

Scanned with CamScanner

Example 22. Examine G and H for isomorphism.





Solution. The graph G has 5 vertices and 6 edges. The graph H has 5 vertices and 5 edges. So $|E(G)| \neq |E(H)|$ *i.e.* the number of edges in G is not equal the number of edges in H.
$\therefore$ G and H are not isomorphic.

Scanned with CamScanner

8. How many vertices and how many edges do the following graphs have ?

(*a*) K_n (*b*) C_n (*c*) W_n (*d*) $K_{m,n}$ (*d*) Q_n

9. Consider the following directed graph *G*:

$V(G) = \{a,b,c,d,e,f,g\}$

$E(G) = \{(a,a),(b,e),(a,e),(e,b),(g,c),(a,e),(d,f),(d,b),(g,g)\}$

(*i*) Indentify any loops or parallel edges

(*ii*) Are there any sources in G ?

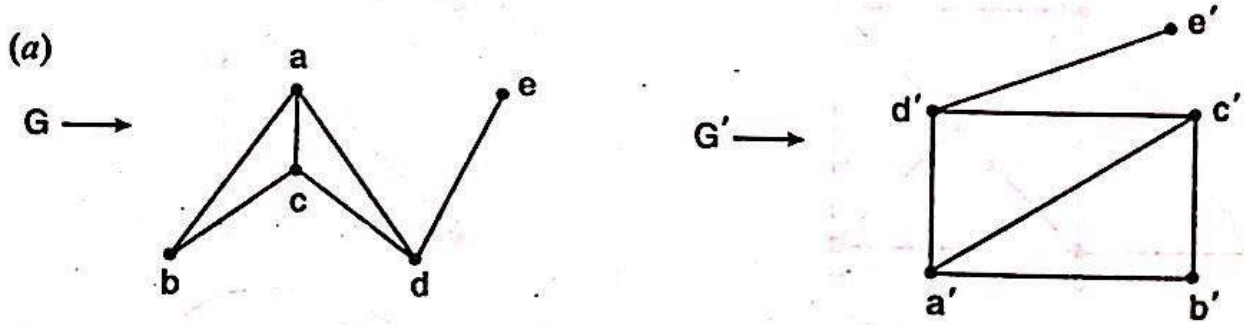
(*iii*) Are there any sinks in G ?

(*iv*) Find the subgraph H of G determined by the vertex set

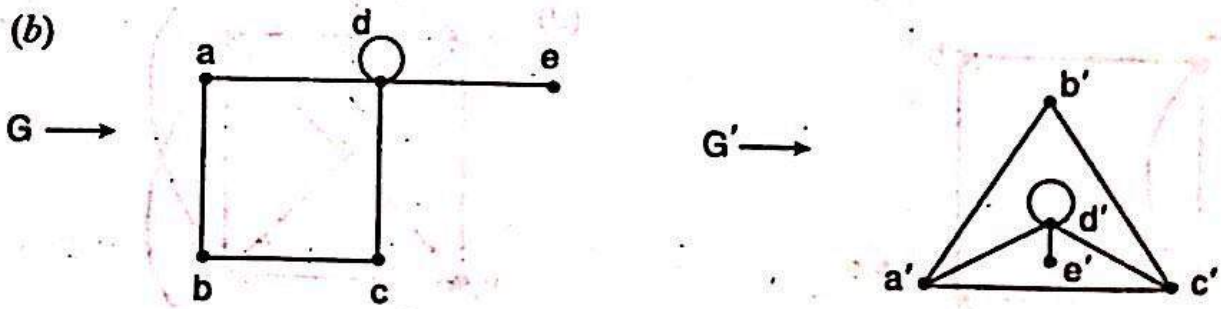
$V' = \{a,b,c,d\}$

10. Show that the following graphs are isomorphic.

(*a*)

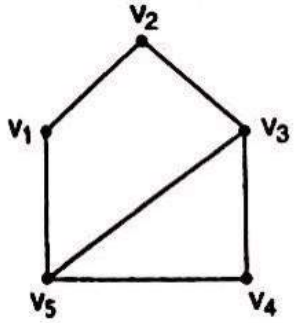


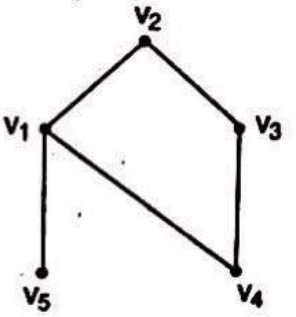
(*b*)



Scanned with CamScanner

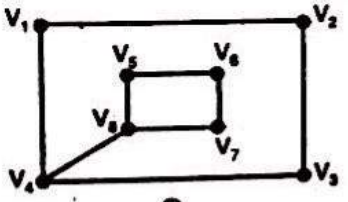
11. Show that graphs are not isomorphic.



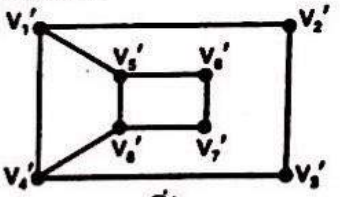


12. Determine whether the following graphs are isomorphic.

(*a*)

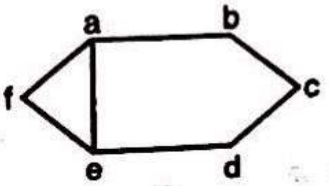


G_1

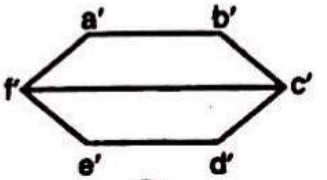


G'_1

(*b*)



G_2



G'_2

Scanned with CamScanner

13.11. Representation of Graphs

Although a diagrammatic representation of a graph is very convenient for a visual study but this is only possible when the number of nodes and edges is reasonably small. Two types of representation are given below.

Matrix Representation. The matrix are commonly used to represent graphs for computer processing. The advantages of representing the graph in matrix form lies on the fact that many results of matrix algebra can be readily applied to study the structural properties of graphs from an algebraic point of view. There are number of matrices which one can associate with any graph. We shall discuss adjacency matrix and the incidence matrix.

Adjacency Matrix

(*a*) Representation of Undirected Graph

The adjacency matrix of a graph G with n vertices and no parallel edges is an n by n matrix $A = [a_{ij}]$ whose elements are given by

$a_{ij} = 1$, if there is an edge between ith and jth vertices, and

$= 0$, if there is no edge between them.

Note that for a given graph, the adjacency matrix is based on ordering chosen for the vertices. Hence, there are as many as $n!$ different adjacency matrices for a graph with n vertices, since there are $n!$ different ordering of n vertices. However, any two such adjacency matrices are closely related in that one can be obtained from the other by simply interchanging rows and colums.

There are a number of observations that one can make about the adjacency matrix A of a graph G.

(*i*) A is symmetric *i.e.* $a_{ij} = a_{ji}$ for all i and j.

(*ii*) The entries along the principal diagonal of A all 0's if and only if the graph has no self loops. A self loop at the vertex corresponds to $a_{ii} = 1$.

(*iii*) If the graph is simple (no self loop, no parallel edges), the degree of vertex equals the number of 1's in the corresponding row or column of A.

(*iv*) The (i, j) entry of A^m is the number of paths of length (no. of occurence of edges) m from vertex v_i to vertex v_j.

(*v*) If G be a graph with n vertices $v_1, v_2 \ldots\ldots\ldots\ldots v_n$ and let A denote the adjacency matrix of G with respect to this listing of the vertices. Let B be the matrix.

$$B = A + A^2 + A^3 + \ldots\ldots\ldots\ldots + A^n \ (n > 1)$$

Scanned with CamScanner

Then G is a connected graph iff B has no zero entries.

This result can be used to check the connectedness of a graph by using its adjacency matrix.

Adjacency can also be used to represent undirected graphs with loops and multiple edges. A loop at the vertex v_i must have the element a_{ii} equal to 1 in the the adjacency matrix. When multiple edges are present, the adjacenmcy matrix is no longer a zero - one matrix, since the (i, j) th entry equals the number of edges that are associated between v_i and v_j. All undirected graphs, including multigraphs and pseudographs, have symmetric adjacency matrices.

(b) Representation of Directed Graph

The adjacency matrix of a diagraph D, with n vertices is the matrix $A = [a_{ij}]_{n \times n}$ in which

$$a_{ij} = 1 \text{ if arc } (v_i, v_j) \text{ is in D}$$
$$= 0 \text{ otherwise.}$$

One can make a number of observations about the adjacency matrix of a diagraph.

(*i*) A is not neccessary symmetric, since there may not be an edge from v_i to v_j when there is an edge from v_j to v_i

(*ii*) The sum of any column of j of A is equal to the number of arcs directed towards v_j.

(*iii*) The sum of entries in row i is equal to the number of arcs directed away from vertex v_i (out degree of vertex v_i)

(*iv*) The (i, j) entry of A^m is equal to the number of path of length m from vertex v_i to vertex v_j

(*v*) The diagonal elements of A. A^T show that out degree of the vertices. The diagonal entries of A^T. A shows the in degree of the vertices.

The adjacency matrices can also be used to represent directed multigraphs. Again such matrices are not zero – one matries when there are multiple edges in the same direction connecting two vertices. In the adjacency matrix for a directed multigraph. a_{ij} equals the number of edges that are associated to (v_i, v_j).

Scanned with CamScanner

Example 38 : Use adjacency matrix to represent the graphs shown in Fig. 13.51.

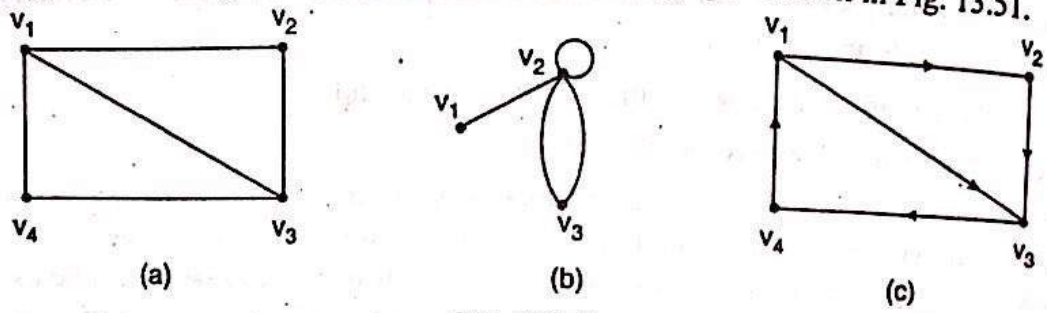


Fig. 13.51

Solution. We order the vertices in Fig 13.51 (a) as v_1, v_2, v_3 and v_4. Since there are four vertices, , the adjacency matrix representing the graph will be a square matrix of order four. The required adjacency matrix A is

$$A = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

We order the vertices in Fig. 13.51 (b) as v_1,v_2 and v_3 . The adjacency matrix representing the graph with loop and multiple edges is given by

Scanned with CamScanner

$$A = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \end{array} \begin{array}{c} \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 2 \\ 0 & 2 & 0 \end{bmatrix} \end{array}$$

Taking the order of the vertices in Fig.13.51 (c) as v_1, v_2, v_3 and v_4. The adjacency matrix epresenting the digraph is given by

$$A = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Example 39 : Draw the undirected graph represented by adjacency matrix A given by

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

Solution. Since the given matrix is a square of order 5, the graph G has five vertices v_1, v_2, v_3, and v_5. Draw an edge from v_i to v_j where $a_{ij} = 1$. The required graph is drawn in Fig. 13.52.

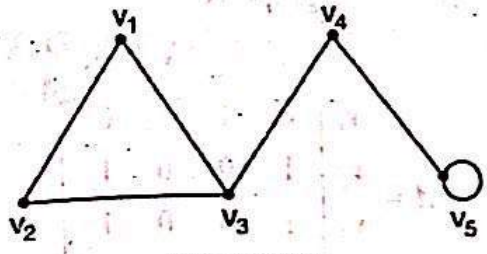


Fig. 13.52

Scanned with CamScanner

Incidence Matrix

(a) Representation of Undirected Graph

Consider a undirected graph G = (V, E) which has n vertices and m edges all labelled. The incidence matrix $I(G) = [b_{ij}]$, is then $n \times m$ matrix, where

$$b_{ij} = 1 \quad \text{when edge } e_j \text{ is incident with } v_i$$
$$= 0 \quad \text{otherwise}$$

A graph and its incidence matrix are shown in examples. We can make a number of observations about the incidence matrix of G.

(i) Each column of B comprises exactly two unit entries.

(ii) A row with all 0 entries corresponds to an isolated vertex.

(iii) A row with a single unit entry corresponds to a pendant vertex.

(iv) The number of unit entries in row i of B is equal to the degree of the corresponding vertex v_i

(v) The permutation of any two rows (any two columns) of I(G) corresponds to a relabelling of the vertices (edges) of G.

(vi) Two graphs are isomorphic if and only if their corresponding incidence matrices differ only by a permutation of rows and columns.

(vii) If G is connected with n vertices then the rank of I(G) is $n - 1$.

Incidence matrices can also be used to represent multiple edges and loops. Multiple edges are represented in the incidence matrix using columns with idential entries. Since these edges are incident with the same pair of vertices. Loops are represented using a column with exactly one entry equal to 1, corresponding to the vertex that is incident with this loop.

(b) Representation of Directed Graph

The incidence matrix $I(D) = [b_{ij}]$ of digraph D with n vertices and m edges is the $n \times m$ matrix in which.

$$b_{ij} = 1 \text{ if arc } j \text{ is directed away from a vertex } v_i$$
$$= -1 \text{ if arc } j \text{ is directed towards vertex } v_i$$
$$= 0 \text{ otherwise.}$$

Example 44 : Find the incidence matrix to represent the graph shown in Fig.13.56.

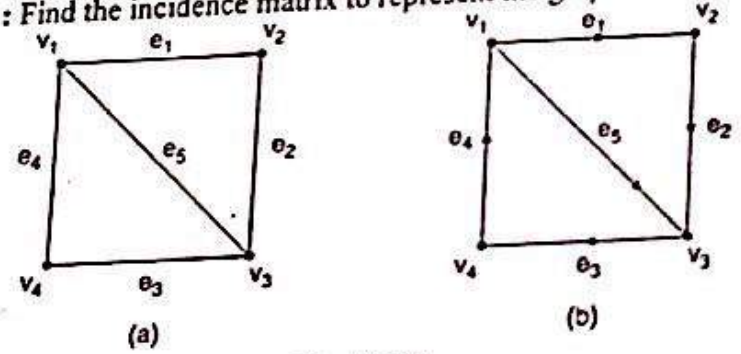


Fig. 13.56

Solution. The incidence matrix of Fig.(a) is obtained by entering for row v and column e is 1 if e is incident on v and 0 otherwise. The incidence matrix is.

$$I(G) = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{ccccc} e_1 & e_2 & e_3 & e_4 & e_5 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix} \end{array}$$

Scanned with CamScanner

The incidence matrix of the digraph of Fig.(b) is

$$I(D) = \begin{bmatrix} 1 & 0 & 0 & -1 & 1 \\ -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & -1 \\ 0 & 0 & -1 & 1 & 0 \end{bmatrix}$$

Example 45. Draw the graph whose incidence matrix is

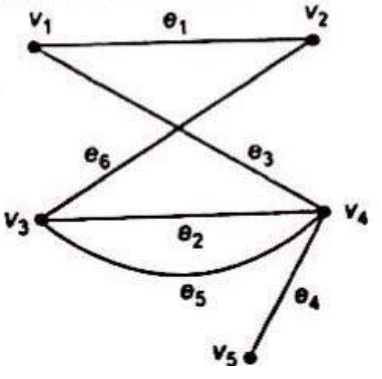
$$\begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Solution. Since the given matrix has 5 rows and 6 columns, its corresponding graph has 5 vertices and 6 edges. We re-write the incidence matrix as follows:

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

Since first column contains two 1 placed at 2nd and 3rd row, vertices v_2 and v_3 are connected by e_1. Similarly, the vertices v_1 and v_4 are connected by e_2, and the vertices v_2 and v_4 are connected by e_3 and so on.

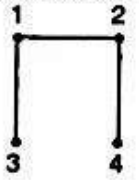
Therefore, the graph of the given incidence matrix is

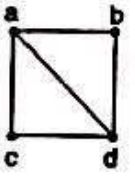


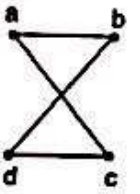
Scanned with CamScanner

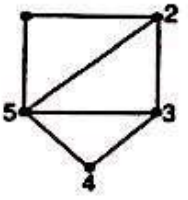
Problem Set

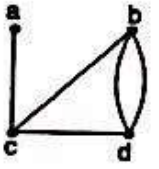
1. Find the adjacency matrix of each of the following graphs:

(a)


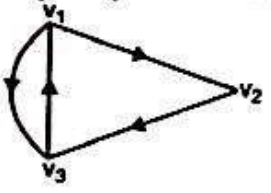
(b)


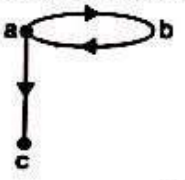
(c)


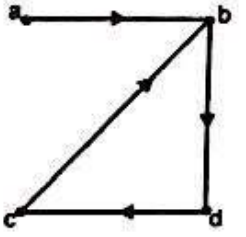
(d)


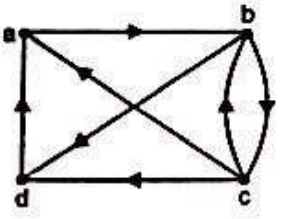
(e)


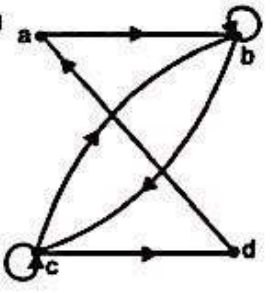
2. Find the adjacency matrix of each of the following digraphs:

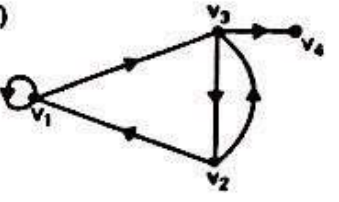
(a)


(b)


(c)


(d)


(e)


(f)


3. Draw the graph/digraph represented by the given adjacency matrix.

(a) $\begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$ (b) $\begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$ (c) $\begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix}$

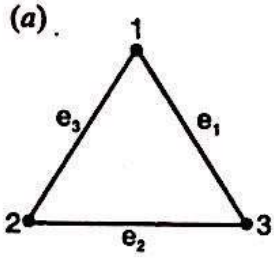
(d) $\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$ (e) $\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$ (f) $\begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}$ (g) $\begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$

(h) $\begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}$

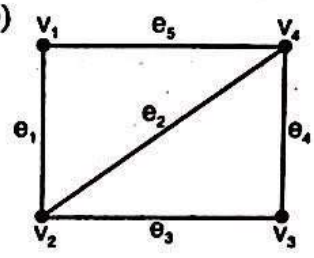
Scanned with CamScanner

4. Find the incidence matrix of each of the following graphs or diagraphs.

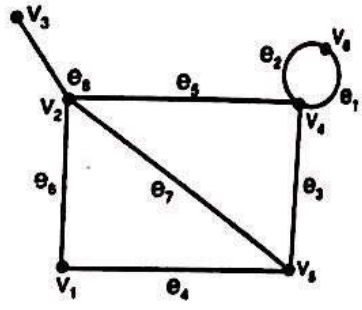
(a)



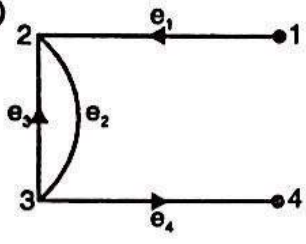
(b)



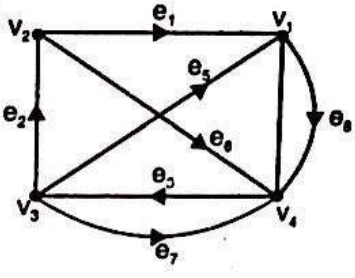
(c)



(d)



(e)



5. Draw the graphs of the following incidence matrices.

(a)
$$\begin{array}{c} \\ a \\ b \\ c \\ d \end{array}\begin{bmatrix} e_1 & e_2 & e_3 & e_4 & e_5 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

(b)
$$\begin{array}{c} \\ a \\ b \\ c \\ d \end{array}\begin{bmatrix} e_1 & e_2 & e_3 & e_4 & e_5 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix}$$

(c)
$$\begin{array}{c} \\ a \\ b \\ c \end{array}\begin{bmatrix} e_1 & e_2 & e_3 & e_4 & e_5 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

(d)
$$\begin{array}{c} \\ a \\ b \\ c \\ d \end{array}\begin{bmatrix} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \\ 1 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 & 1 & 0 \\ 0 & 1 & -1 & 0 & 0 & -1 \\ -1 & -1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

(e)
$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

(f)
$$\begin{array}{c} \\ a \\ b \\ c \\ d \end{array}\begin{bmatrix} e_1 & e_2 & e_3 & e_4 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

Scanned with CamScanner

Unit-2

Paths, Circuits, Cycles and Connectivity:

(1) Walk:

A walk in a graph G is a finite alternating sequence

$$v_0 - e_1 - v_1 - e_2 - v_2 - e_3 \ldots - e_n - v_n.$$

of vertices and edges of the graph. such that each edge e_i in the sequence joins vertices v_{i-1} and v_i. $(1 \le i \le n)$.

(i) The end vertices v_0 and v_n are the end or terminal vertices of the walk.

(ii) The vertices $v_1, v_2, \ldots, v_{n-1}$ are called its internal vertices.

(iii) The integer n, the no. of edges in the walk is called the length of the walk.

Note:- A walk may repeat both vertices and edges.

(2) Open Walk:

A walk is called open when the terminal vertices are distinct.

(3) Closed Walk:

For the same end terminal vertices, a walk is termed as closed.

(4) Trail:

A walk is called a trail if all its edges are distinct.

Scanned with CamScanner

Note: A trail is open or closed depends on whether its end vertices are distinct or not.

(5) Circuit :
A closed trail is called a circuit.

(6.) Path :
A walk is called a path if all its vertices and edges are distinct.

Note: Every path is a trail.

(7) Cycle :
A path in which only repeated vertex is the first vertex is called a cycle to describe such a closed path.

Scanned with CamScanner

	Repeated Edge	*Repeated Vertex*	*Starts and Ends at same points ?*
Walk (open)	allowed	allowed	no
Walk (closed)	allowed	allowed	yes
Trail	no	allowed	no
Circuit	no	allowed	yes
Path	no	no	no
Cycle	no	first and last only	yes

For example, in the graph given in Fig. 13.29.

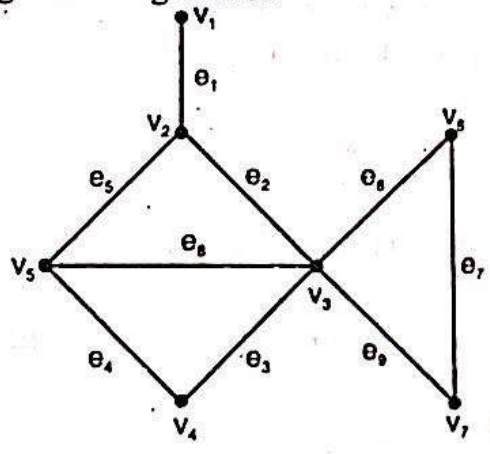


Fig. 13.29

(*i*) The sequence $v_1 - e_1 - v_2 - e_5 - v_5 - e_8 - v_3 - e_3 - v_4 - e_4 - v_5 - e_5 - v_2 - e_2 - v_3 - e_6 - v_6$ walk of length 8. It contains repeated vertices v_2, v_3 and v_5 and repeated edge e_5.

(*ii*) The sequence $v_1 - e_1 - v_2 - e_5 - v_5 - e_3 - v_3 - e_3 - v_4 - e_4 - v_5$ is a trail. It contains repeated vertex v_5 but does not contain repeated edge.

(*iii*) The sequence $v_1 - e_1 - v_2 - e_5 - v_5 - e_8 - v_3 - e_3 - v_4$ is a path. It does not contain repeated vertex and repeated edge.

(*iv*) The sequence $v_2 - e_2 - v_3 - e_3 - v_4 - e_4 - v_5 - e_5 - v_2$ is a cycle. It does not conatain repeated vertex and repeated edge except the first and last vertex.

Scanned with CamScanner

Reachability

A vertex v in a simple graph G is said to be reachable from the vertex u of G if there exists a path from u to v. The set of vertices which are reachable from a given vertex v is called the reachable set of v and is denoted by R(v).

For any subset U of the vertex set V, the reachable set of U is the set of all vertices which are reachable from any vertex set of S and this set is denoted by R(S). For example in the graph given on the next page.

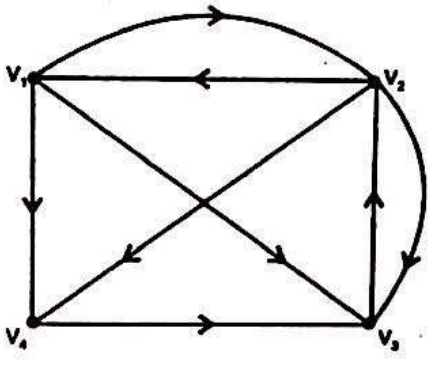


Fig. 13.30

$R(v_1) = \{v_2, v_3, v_4\}$, $R(v_2) = \{v_1, v_3, v_4\}$,
and $R(\{v_1, v_2\}) = \{v_3, v_4\}$.

Scanned with CamScanner

Connectedness in Undirected Graphs

A graph is connected if it is possible to travel from any vertex to any other vertex along a sequence of adjacent edges of the graph. Thus a graph G is **connected** if given any vertices v and w in G, there is a path from v to w. *i.e.*, if any pair of vertices are reachable from one another. A graph that is not connected is called **disconnected.** A disconnected graph is the union of two or more connected subgraphs each pair of which has no vertex in common. These disjoint connected subgraphs are called the **connected components** of the graph. A connected component of a graph is a connected subgraph of largest possible size. Thus a graph H is a connected component or simply component of a graph G if and only if

1. H is a subgraph of G;
2. H is connected;
3. No connected subgraph of G has H as a subgraph and contains vertices or edges that are not in H.

The graph G and a connected component of the graph is shown in Fig. 13.31 and Fig. 13.32.

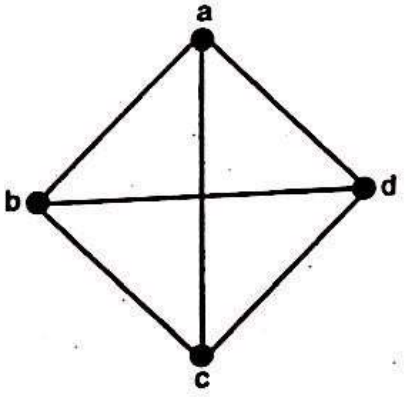


Fig. 13.31

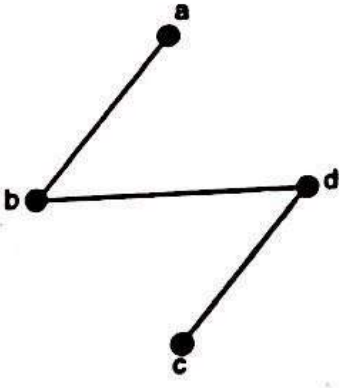
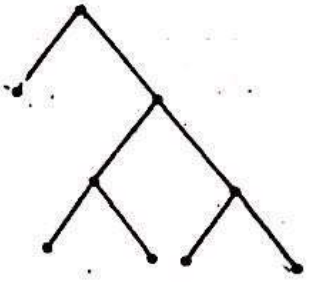
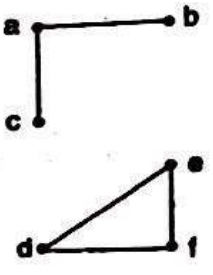


Fig. 13.32. Connected component of G

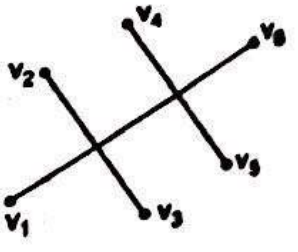
Example 24. Which of the graphs below are connected ?



(a)



(b)



(c)

Scanned with CamScanner

Solution. The graph shown in 13.33 (*a*) is a connected graph since for every pair of distinc vertices there is a path between them.

The graph shown in Fig. 13.33 (*b*) is not connected since there is no path in the graph betwee vertices *b* and *d*.

The graph shown in Fig. 13.33 (*c*) is not connected. In drawing a graph two edges may cros at a point which is not a vertex. The graph can be redrawn as

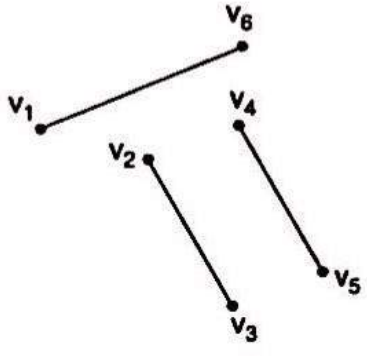


Fig. 13.34

Scanned with CamScanner

Connectedness in Directed Graphs

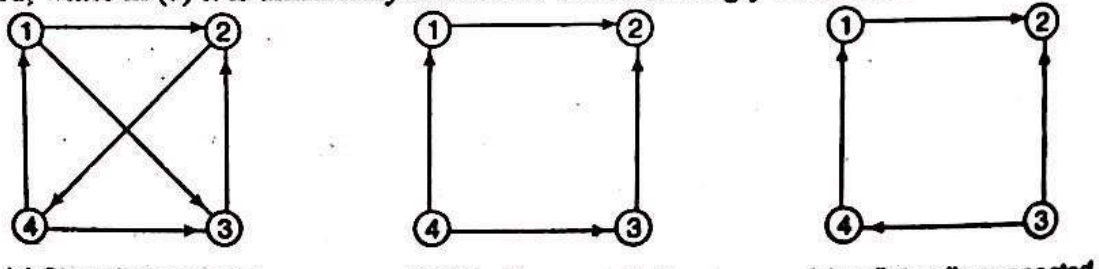
The definition of connectedness in undirected graphs can not be applied to directed graphs without some further modification because in a directed graph if a vertex u is reachable from another vertex v, the vertex v may not be reachable from u. To overcome this difficulty we define the connectedness in a diagraph as follows.

Connected or weakly connected: A directed graph is called connected or weakly connected if it is connected as an undirected graph in which each directed edge is converted to an undirected graph.

Unilaterally connected: A simple directed graph is said to be unilaterally connected if for any pair of vertices of the graph at least one of the vertices of the pair is reachable from other vertex.

Strongly connected: A directed graph is called strongly connected if for any pair of vertices of the graph both the vertices of the pair are reachable from one another.

For the digraphs in Fig. 13.37 the digraph in (*a*) is strongly connected, in (*b*) it is weakly connected, while in (*c*) it is unilateraly connected but not strongly connected.



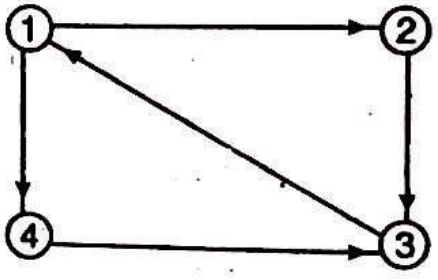
(a) Strongly connected (b) Weakly connected (c) unilaterally connected

Fig. 13.37. Connectivity in digraphs

Note that a unilaterally connected digraph is weakly connected but a weakly connected digraph is not necessarily unilaterally connected. A strongly connected digraph is both unilaterally and weakly connected.

Scanned with CamScanner

Example 26. Is the directed graph given below strongly connected ?



Solution. The possible pairs of vertices and the forward and the backward paths between then are shown below for the given graph

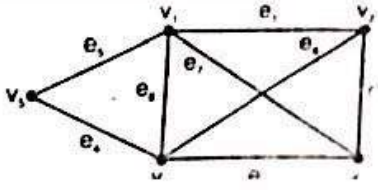
Pairs of vertices	*Forward path*	*Backward path*
(1,2)	1 – 2	2 – 3 – 1
(1,3)	1 – 2 – 3	3 – 1
(1,4)	1 – 4	4 – 3 – 1
(2,3)	2 – 3	3 – 1 – 2
(2,4)	2 – 3 – 1 – 4	4 – 3 – 1 – 2
(3,4)	3 – 1 – 4	4 – 3

Therefore, we see that between every pair of distinct vertices of the given graph there exists a forward as well as backward path, and hence it is strongly connected.

Scanned with CamScanner

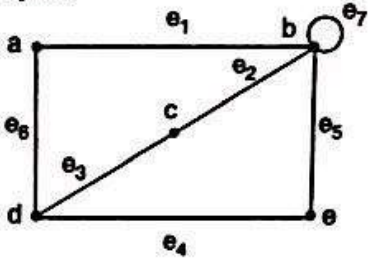
Problem Set

1. Consider the graph given below. Determine which of the following sequences are paths, simple paths, circuit, and cycle.



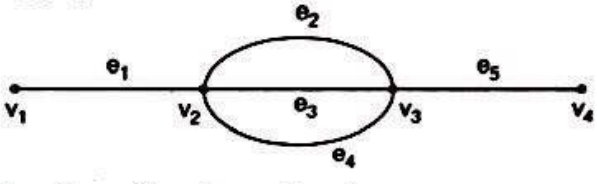
   (a) $v_1\, e_1\, v_2\, e_6\, v_4\, e_3\, v_3\, e_2\, v_2$
   (b) $v_1\, e_1\, v_2\, e_2\, v_3\, e_3\, v_4\, e_4\, v_5$
   (c) $v_1\, e_8\, v_4\, e_3\, v_3\, e_7\, v_1\, e_8\, v_4$
   (d) $v_5\, e_5\, v_1\, e_8\, v_4\, e_3\, v_3\, e_2\, v_2\, e_6\, v_4\, e_4\, v_5$
   (e) $v_2\, e_2\, v_3\, e_3\, v_4\, e_4\, e_5\, v_1\, e_1\, v_2$

2. Consider the graph given below. Determine which of the following sequences are paths, simple paths, cycle and simple cycle.



   (a) $b\, e_7\, b$
   (b) $d\, e_3\, c\, e_2\, b\, e_5\, e\, e_4\, d$
   (c) $a\, e_6\, d\, e_3\, c\, e_2\, b\, e_5\, e$

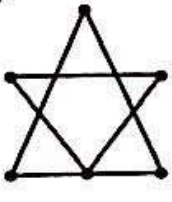
3. Consider the following graph.



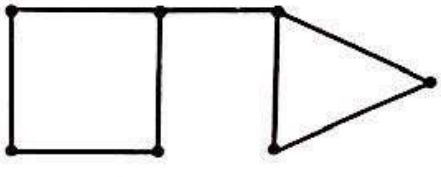
   (a) How many simple paths are there from v_1 to v_4 ?
   (b) How many trails are there from v_1 to v_4 ?
   (c) How many paths are there from v_1 to v_4 ?

4. Which of the graphs below are connected ?

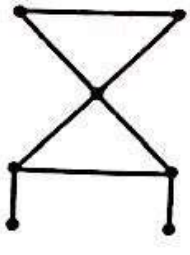
(a)



(b)

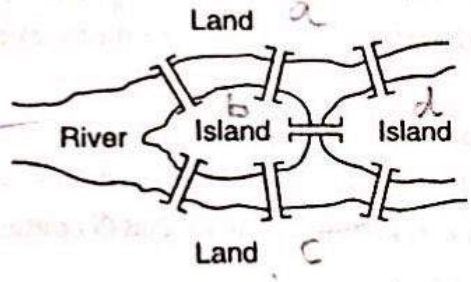


(c)

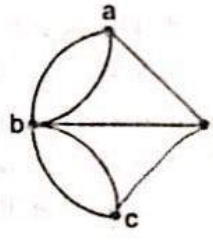


Scanned with CamScanner

One of the oldest problem involving graphs is the Konigsberg bridge probelm. There were two islands linked to each other and to the banks of the Pregel River (earlier known as Konigsberg) by seven bridges shown in Fig. 13.38 (*a*). The problem was to begin at any of the four land areas to walk across each bridge once and to return the starting point. Euler drew a graph like Fig. 13.38 (*b*) for the problem in which *a* and *c* represent the two river banks; *b* and *d* the two islands. The arcs joining them represent the seven bridges.



(a)



(b)

Fig. 13.38. Konigsberg graph

It is clear that the problem of walking each of the seven bridges exactly once and returning to the starting point is equivalent to finding a circuit in graph (*b*) that traverses each of the edge exactly once. Euler discovered a very simple criterion for determining whether such a circuit exists in a graph

Scanned with CamScanner

Eulerian Graph

A circuit in a connected graph is an Euler circuit if it contains every edge of the graph exactly once. A connected graph with an Euler circuit is called an Euler graph or Eulerian graph.

If there is an open trail from a to b in G and this trail traverses each edge in G exactly once, then the trail is called an Euler trail.

The existence of Euler circuit and trail depends on the degree of vertices.

The next theorem provides necessary and sufficient condition for characterising Euler graph.

Theorem 13.13. A nonempty connected graph G is Eulerian if and only if its vertices are all of even degree.

Theorem 13.14. A connected graph contains an Eulerian trail, but *not* an Eulerian circuit, if and only if it has exactly two vertices of odd degree.

Scanned with CamScanner

Example 28(*a*). Let G be a graph of Fig. 13.39. Verify that G has an Eulerian circuit,

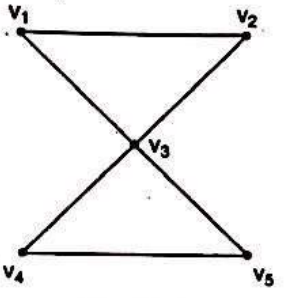


Fig. 13.39

Solution. We observe that G is connected and all the vertices are having even degree deg $(v_1) = \deg(v_2) = \deg(v_4) = \deg(v_5) = 2$.

Thus G has a Eulerian circuit. By inspection, we find the Eulerian circuit.

$$v_1 - v_3 - v_5 - v_4 - v_3 - v_2 - v_1$$

Example 28 (*b*). Show that the graphs in Fig. 13.40 contain no Eulerian circuit.

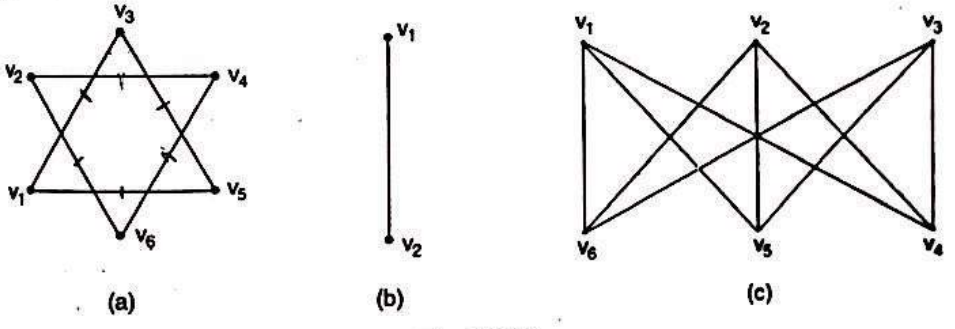


Fig. 13.40

Solution. The graph shown in Fig. 13.40 (*a*) does not contain Eulerian circuit since it is not connected.

The graph shown in Fig. 13.40 (*b*) is connected but vertices v_1 and v_2 are of degree 1. Hence it does not contain Eulerian circuit.

All the vertices of the graph shown in Fig. 13.40 (*c*) are of degree 3, hence it does not contai Eulerian circuit.

Scanned with CamScanner

Example 29. Show that the graph shown in Fig. 13.41 has no Eulerian circuit but has a Eulerian trail.

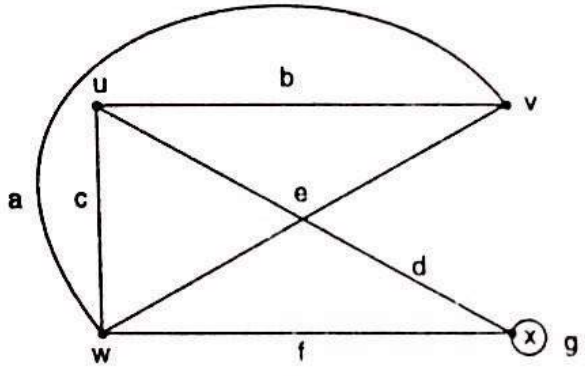


Fig. 13.41

Solution. Here deg (u) = deg (v) = 3 and deg (w) = 4, deg (x) = 4. Since u and v have only two vertices of odd degree, the graph shown in Fig. 13.41 does not contain Eulerian circuit, but the path $b-a-c-d-g-f-e$ is an Eulerian trail.

Scanned with CamScanner

Hamiltonian Graphs

Hamiltonian graphs are named after Sir William Hamilton, an Irish mathematician who introduced the problems of finding a circuit in which all vetices of a graph appear exactly once.

A circuit in a graph G that contains each vertex in G exactly once, except for the starting and ending vertex that appears twice is known as **Hamiltonian cycle.**

A graph G is called a **Hamiltonian cycle** if it contains a Hamiltonian cycle.

A Hamiltonian path is a simple path that contains all vertices of G where the end points may be distinct.

Example 33. Which of the graphs given in Fig. 13.43 is Hamiltonian cycle. Give the cycles on the graphs that contain them.

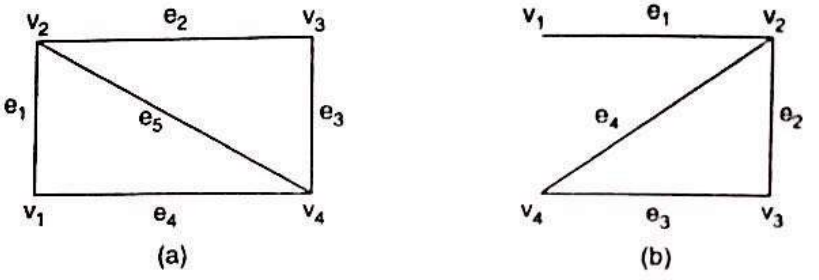


Fig. 13.43

Solution. The graph shown in Fig. 13.43 (a) has Hamilitonian cycle given by $v_1 e_1 v_2 e_2 v_3 e_3 v_4 e_4 v_1$. Note that all vertices appear in this a cycle but not all edges. The edge e_5 is not used in the cycle.

The graph shown in Fig.13.43 (b) does not contain Hamiltonian cycle since every cycle containing every vertex must contain the e_1 twice. But the graph does have a Hamiltonian path $v_1 - v_2 - v_3 - v_4$.

Scanned with CamScanner

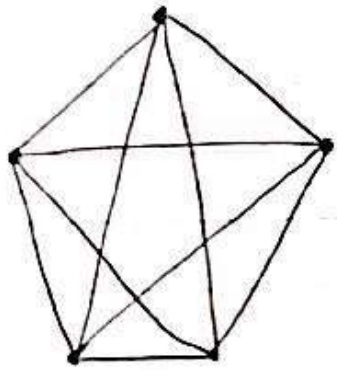
Hamiltonian Graphs

Theorems:

(1) A simple connected graph G with $n \geq 3$ vertices is Hamiltonian if $\deg(v) \geq \frac{n}{2}$ for every vertex v in G.

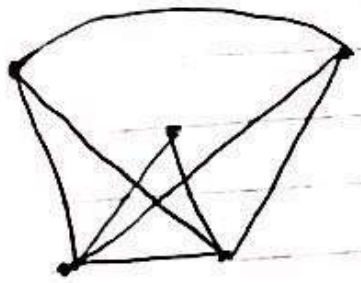
(2) A simple connected graph with n vertices and m edges is Hamiltonian if $m \geq [(n-1)(n-2)]/2 + 2$.

(a)



satisfies Theorem 1.

(b)



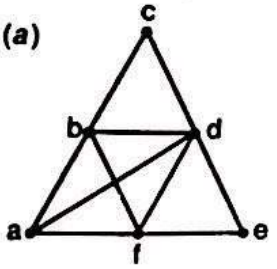
satisfies Theorem 2.

Scanned with CamScanner

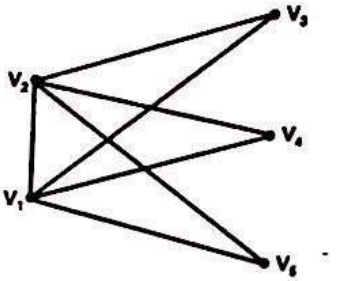
Problem Set

1. Determine which of the following graphs contain an Eulerian circuit. If it does, then find an Eulerian circuit for the graph.

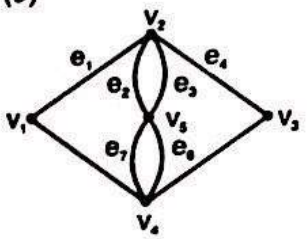
(a)



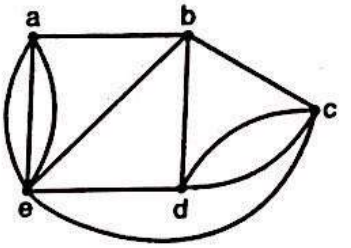
(b)



(c)

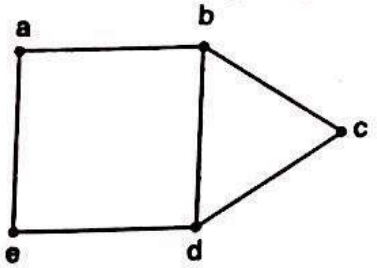


(d)

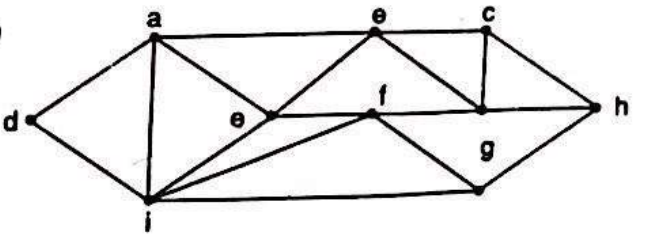


2. Determine whether the given graph has a Hamilton circuit. If it does, find such a circuit.

(a)



(b)



Scanned with CamScanner

Seating Problem: Nine members of a new club meet each day for lunch at a round table. They decide to sit such that every member has different neighbors at each lunch. How many days can this arrangement last?

This situation can be represented by a graph with nine vertices such that each vertex represents a member, and an edge joining two vertices represents the relationship of sitting next to each other. Figure 1-9 shows two possible

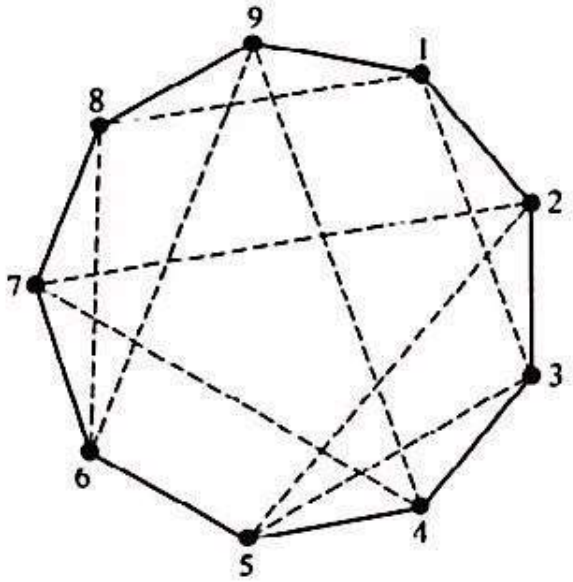


Fig. 1-9 Arrangements at a dinner table.

seating arrangements—these are 1 2 3 4 5 6 7 8 9 1 (solid lines), and 1 3 5 2 7 4 9 6 8 1 (dashed lines). It can be shown by graph-theoretic considerations that there are only two more arrangements possible. They are 1 5 7 3 9 2 8 4 6 1 and 1 7 9 5 8 3 6 2 4 1. In general it can be shown that for n people the number of such possible arrangements is

$$\frac{n-1}{2}, \qquad \text{if } n \text{ is odd,}$$

and

$$\frac{n-2}{2}, \qquad \text{if } n \text{ is even.}$$

Scanned with CamScanner

Travelling Salesman Problem

Suppose a salesman territory includes several cities with highways connecting certain pair of these cities. He is required to visit each city personally exactly once. Graph theory can be used to solve this transportation system. The system can be represented by a graph G whose vertices correspond to the cities and such that two vertices are joined by an edge if and only if a highway connects the corresponding cities.

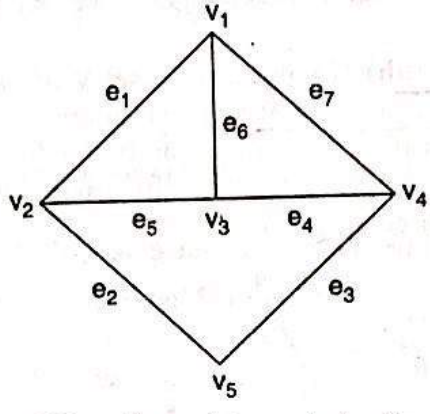


Fig. 13.4 Travelling salesman's territory graph

Starting at vertex v_1, salesman can visit each vertex by taking the edges e_1, e_2 e_3, e_4 and e_6 and back to v_1.

Scanned with CamScanner

Utilities Problem

An old problem concerns with three houses H_1, H_2, H_3, each to be connected to each of the three utilities – Water (W), Gas (G) and Electricity (E). Is it possible to connect each utility with each of the three houses without any two connections crossing each other ?

We can represent the connection of the three houses to the utilities by the graph of Fig. 13.3. Here we have diagram of six vertices, three of which represent the houses, denoted by H_1, H_2, H_3 the other three represent the utilities, denoted by G, W, E. An edge joins two vertices if and only if one vertex denotes a house and other vertex a utility. We shall see that it is not possible to draw this graph without edges crossing over. Thus the answer to this problem is no.

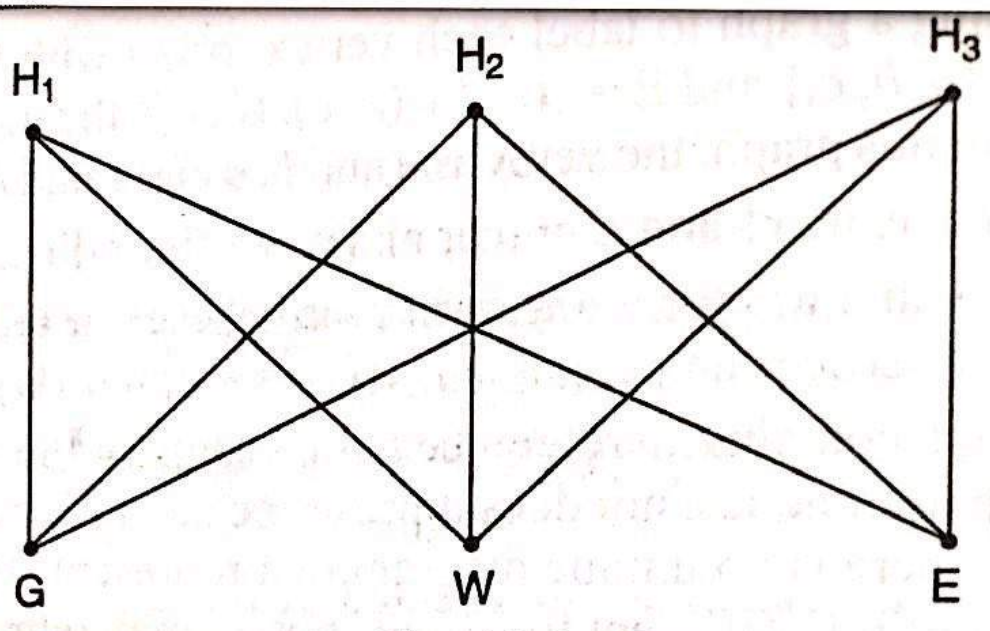


Fig. 13.3

Scanned with CamScanner

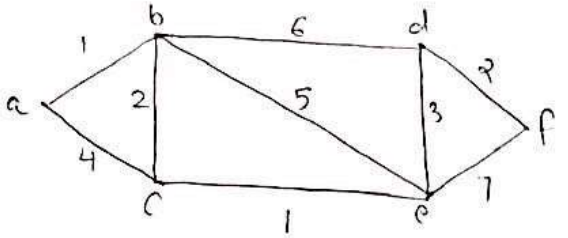
Weighted Graphs :

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

(Here, the number may represent mileage cost, computer time or some other quantity).

The length of a path in a weighted graph is the sum of the weights of the edges of this path and the shortest path between the two vertices is the minimum length of the path.

example :



Shortest Path Problem :

The shortest path problem is the problem of finding a path between two vertices in a graph such that the sum of the weights of its constituent edges is minimized.

Scanned with CamScanner

Fleury's Algorithm:

To find Euler path or Euler circuit in a Connected Graph:

(I) Let G be an Eulerian connected graph with each vertex of <u>even degree</u>

Step1 Select any vertex u as the starting vertex.

Step2 Select an edge $e = (u, v)$.
If there are many such edges, select one that is not a bridge. (ie. removing that edge does not make the graph disconnected).
Remove that edge.

Note: If e is a bridge (select only if there is no alternative) then remove the vertex u.

Now from vertex v proceed further.

Step3 Repeat Ser 2 until all the edges exhausted.

In this we get <u>Euler circuit</u>.
(In Euler circuit all vertices are of even degree).

II. To find Euler path:
(We get Euler path if the Graph G has exactly two vertices of odd degree).

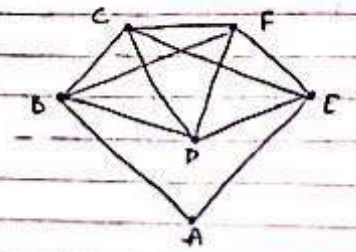
Step1 Select only those two vertices which have odd degree.

Step2 Same

Step3 Same.

Scanned with CamScanner

Ex.1.



Sol^n. Firstly, check degree.

deg (A) = 2 deg (D) = 4
deg (B) = 4 deg (E) = 4
deg (C) = 4 deg (F) = 4

Step1. Start with vertex A.

Step2. We can remove AB or AE
Lets remove AB.
Then we can remove BC, BF or BD
Lets remove BC.
Now we can remove CF, CE or CD
Lets remove CF.

From F, we can remove FB, FD or FE
We will remove FE.

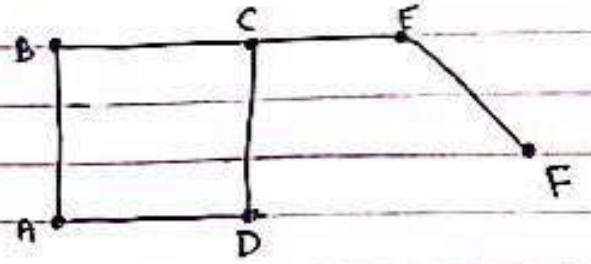
Now from E we can remove EC, ED, EA
We can't remove EA otherwise the graph will be disconnected
So, lets remove ED.

Now we remove DB
Then remove BF (only 1 option)
Then remove FD (")
Then remove DC (")
Then remove CE (")
Then remove EA (").

∴ Euler circuit :
A-B-C-F-E-D-B-F-D-C-E-A.

Scanned with CamScanner

Ex 2



Sol-1 Firstly, Check Degree.

deg (A) = 2 deg (D) = 2
deg (B) = 2 deg (E) = 2
deg (C) = 3 deg (F) = 1

Since two vertices C and F are of odd degree, we can find Euler path.

Step 1. Start with vertex C or F.
lets start from C.

Step 2. We can remove CE, CD, or CB.
But CE is a bridge so we can't remove that.

Lets remove CB.
Then remove BA (only 1 option)
Then remove AD (only 1 option)
Then remove DC (only 1 option)
Then remove CE (only 1 option)
Then remove EF (only 1 option)

Now all the edges have been exhausted.
CB, BA, AD, DC, CE, EF.

Scanned with CamScanner

Dijkstra's Algorithm

The algorithm relies on a series of iterations It involves assigning labels to vertices. Let G = (V, E) be a connected weighted graph. Let a and z be any two vertices where a be the starting point and z be the terminal point. Let L (v) denotes the labels at vertex v. At any point some vertices have temporary labels and the rest have permanent labels. It begins with by labelling starting vertex a, say, by 0 and other vertices with $\propto$. We next label all neighbours v of a by L (v), is the weight of the edge from a to v. Let u be the vertex among those v for which L (u) is minimum. Now find those neighbours of w of u and, for those w not already premanently labelled assign the label L (w) = L (u) + w (e), w (e) being the weight of the edge from u to w, while for those w already labelled L (w) change the label to L (u) + w (e) if this is smaller. Each iteration of the algorithm changes the status of one label from temporary to permanent, thus we may terminate the algorithm when z receives a permanent label.

Dijkstra's Algorithm

Input: A connected weighted graph G.

Output: L (z), the length of shortest distance from a to z

Step 1: Initially set the starting vertex a permanently with 0 i.e. L(a) = 0 and set L(v) = ∞ for all vertices $v \neq a$.

T = {vertices having temporary labels of G}

Step 2: Let u be a vertex in T for which L (u) is minimum and hence the permanent label of u.

Step 3: If $u = z$, stop.

Step 4: For every edge $e = (u, v)$, incident with u, if $v \in$ T, change L (v) to min (L (v), L (u) + w (e)).

Step 5: Change T to T $-$ {u} and go to step 2.

Note 1. If the weight of the edge is not defined, then we assume the weight = 1.

2. There may exit more than one shortest path between two vertices in a graph.

3. The algorithm does not actually gives the shortest path, it gives only the shortest distance.

Scanned with CamScanner

Example 36. Apply Dijkstra's algorithm to the graph given below and find the shortest path from a to f

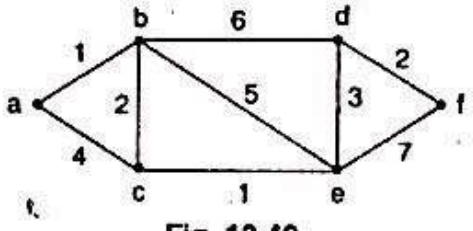


Fig. 13.49

Solution. The intial labelling is given by

Vertex V	a	b	c	d	e	f
L (v)	0	∞	∞	∞	∞	∞
T	{a,	b,	c,	d,	e,	f}

Iteration 1: $u = a$ has $L(u) = 0$. T becomes $T - \{a\}$. There are two edges incident with a *i.e.* ab and ac where b and $C \in T$.

$$L(b) = \min\{\text{old } L(b), L(a) + w(ab)\}$$
$$= \min\{\infty, 0 + 1\} = 1$$
$$L(c) = \min\{\text{old } L(c), L(a) + w(ac)\}$$
$$= \min\{\infty, 0 + 4\} = 4$$

Hence minimum label is $L(b) = 1$

Vertex V	a	b	c	d	e	f
L (v)	0	1	4	∞	∞	∞
T	{	b,	c,	d,	e,	f}

Iteration 2: $u = b$, the permanent label of b is 1. T becomes $T - \{b\}$ there are three edges incident with b *i. e.* bc,bd and be where $c, d, e \in T$.

$$L(c) = \min\{\text{old } L(c), L(b) + w(bc)\}$$
$$= \min\{4, 1 + 2\} = 3$$
$$L(d) = \min\{\text{old } L(d), L(b) + w(bd)\}$$
$$= \min\{\infty, 1 + 6\} = 7$$
$$L(e) = \min\{\text{old } L(e), L(b) + w(be)\}$$
$$= \min\{\infty, 1 + 5\} = 6$$

Vertex V	a	b	c	d	e	f
L(v)	0	1	3	7	6	∞
T	{		c,	d,	e,	f}

Thus minimum label is $L(c) = 3$

Itreation 3 : $u = c$, the permanent label of c is 3, T becomes $T - \{c\}$. There is one edge incident with c *i.e.* ce where $e \in T$.

$$L(e) = \min\{\text{old } L(e), L(c) + w(ce)\}$$
$$= \min\{6, 3 + 1\} = 4$$

Thus minimum label is $L(e) = 4$

Vertex V	a	b	c	d	e	f
L(v)	0	1	3	7	4	∞
T	{			d,	e,	f}

Scanned with CamScanner

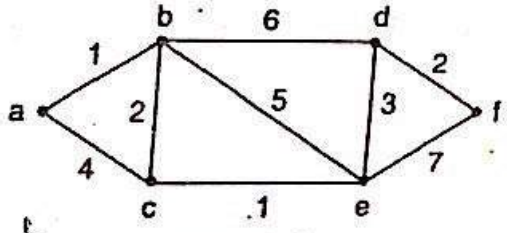


Fig. 13.49

Iteration 4 : $u = e$, the permanent label of e is 4, T becomes T – $\{e\}$. There are two edges incident with e *i.e.* ed and ef where $d, f \in$ T

$$L(d) = \min\{\text{old } L(d), L(e) + w(ed)\}$$
$$= \min\{7, 4+3\} = 7$$
$$L(f) = \min\{\text{old } L(f), L(e) + w(ef)\}$$
$$= \min\{\alpha, 4+7\} = 11$$

Vertex V	a	b	c	d	e	f
L(v)	0	1	3	7	4	11
T	{			d,		f}

Thus minimum label is L (d) = 7.

Iteration 5 : $u = d$. The permanent label of d is 7. T becomes $T - \{d\}$. There is one edge incident with d *i.e.* df where $f \in$ T

$$L(f) = \min\{\text{old } L(f), L(d) + w(df)\}$$
$$= \min\{11, 7+2\} = 9$$

Vertex V	a	b	c	d	e	f
L(v)	0	1	3	7	4	9
T	{					f}

The minimum label is L (f) = 9.

Since $u = f$, the only choice, iteration stops. Thus the shortest distance between a and f is 9.

Scanned with CamScanner

Representation of Graphs

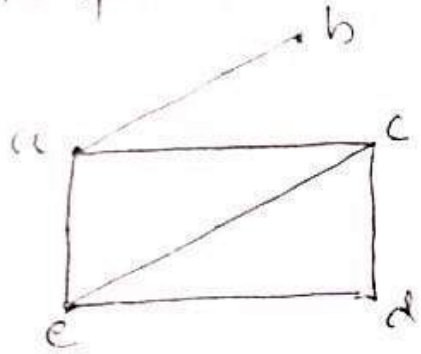
Adjacency List :-

It is a way to represent a graph without multiple edges.

It specifies to each vertex of a graph.

Example :

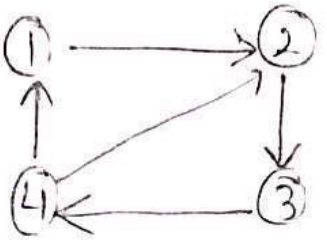
(1)



vertex		Adjacent vertices
a	→	b → c → e
b	→	a
c	→	a → d → e
d	→	c → e
e	→	a → c → d

Scanned with CamScanner

2.



$\boxed{1} \rightarrow \boxed{2}$

$\boxed{2} \rightarrow \boxed{3}$

$\boxed{3} \rightarrow \boxed{4}$

$\boxed{4} \rightarrow \boxed{1} \rightarrow \boxed{2}$

Scanned with CamScanner

BFS

Shortest Path in a Graph Without Weights

The length of a path in a graph without weights denotes the number of edges in a path and the shortest path is the path between two vertices in u and v that uses the least number of edges. One of the algorithms to find the shortest path between two vertices is known as **Breadth First Search** (B F S) algorithm . The algorithm is presented in terms of undirected graphs. Although the procedure works also for directed graphs. Let G be a graph and let s, t be two specified vertices of G. The general idea behind a breadth - fast search beginning at a starting vertex s is as follows. First we process the starting vertex s. Then we process all the neighbours of s. Then we process all the neighbours of neighbours of s and so on. Naturally we need to keep track of the neighbours of a vertex, and we need to guarantee that no vertex is processesd twice. The algorithm involves assigning labels to vertices.

Algorithm : The Breadth First Search Algorithm (BFS)

Step 1: Label vertex s with 0, set $i = 0$.

Step 2: Find all unlabelled vertices in G which are adjacent to vertices labelled i. If there are no such vertices then i is not connected to S.

If there are such vertices, label them $i + 1$.

Step 3: If t is labelled,go to step 4 . If not, increase i to $i + 1$ and go to step 2.

Step 4: The length of a shortest path from s to t is $i + 1$. Stop.

Once the length of the shortest path is found from the previous algorithm, we use the Backtracking algorithm to find the actual shortest path from s to t. This algorithm uses the label λ (v) which are generated in the BFS algorithm.

Algorithm: The back-tracking Algorithm for a Shortest Path

Step 1: Set $i = \lambda$ (t) and assign $v_i = t$

Step 2: Find a vertex u adjacent to v_i and with $\lambda(u) = i - 1$. Assign $v_{i-1} = u$.

Step 3: If $i = 1$ stop.

If not, decrease, i to $i - 1$ and go to step 2.

In general, there may be many shortest paths from s to t and the previous algorithm finds just one of them.

Scanned with CamScanner

Example 35. Find the shortest path from vertex s to t and its length from the graph given below.

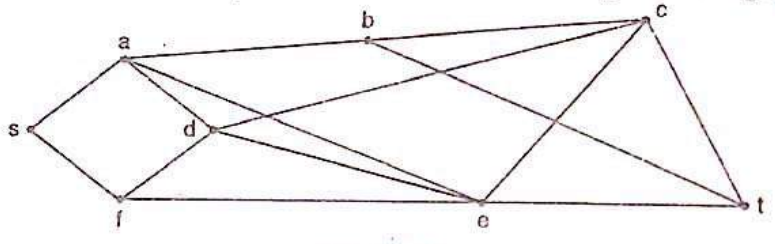


Fig. 13.47

Solution : Using BFS, label s as 0. Then a and f adjacent vertices of s are labelled $0 + 1 = 1$. Then b, d, e (adjacent vertices of a and f) are labelled $1 + 1 = 2$. Then c and t are labelled $2 + 1 = 3$. Since t is labelled 3, the length of a shortest path from s to t is 3.

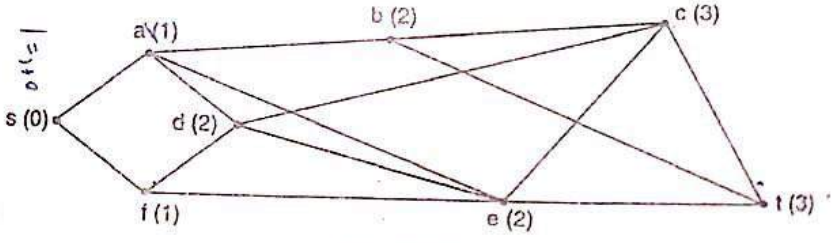


Fig. 13.48

Now, using second algorithm, since $\lambda(t) = 3$. We start with $i = 3$ and $v_i = t$.

We choose e (or b) adjacent to $v_3 = t$, with $\lambda(e) = 2$, and assign $v_2 = e$.

Next, we choose f adjacent to $v_2 = e$ with $\lambda(f) = 1$ and assign $v_1 = f$

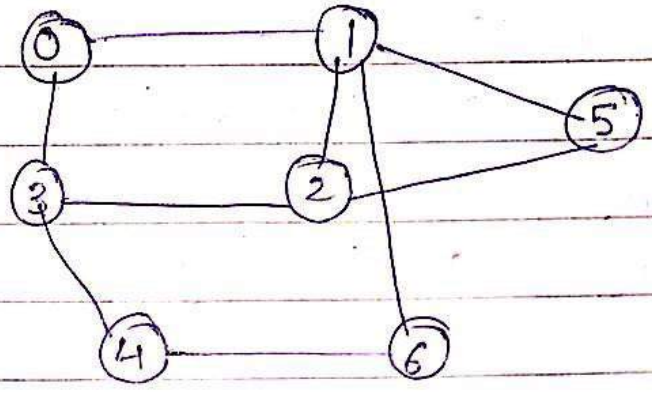
Finally, we take s adjacent to f with $\lambda(s) = 0$ and assign $v_0 = s$.

This gives the shortest path $v_0, v_1, v_2, v_3 = sfet$ from s to t.

Note : There could be several shortest path from s to t, to be precise, there are 3 *i.e.*, *saet*, *sabt* and *sfet*.

Scanned with CamScanner

(Undirected)
Example



(FIFO) Queue

0	1	3	2	5	6	4

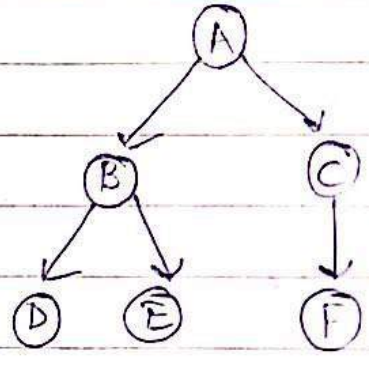
Result 0 1 3 2 5 6 4

Scanned with CamScanner

BFS in directed Graphs:

It starts at ~~the~~ some arbitrary node of a graph and explores the neighbour nodes first, before moving to the next level neighbours.

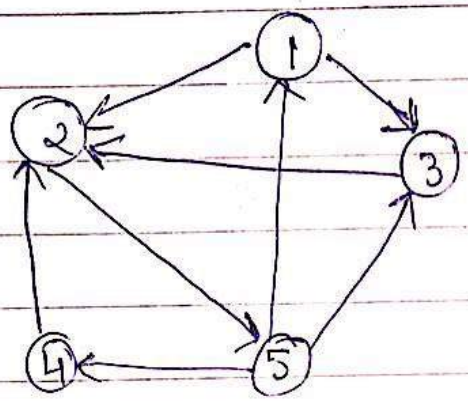
Example:



A B C D F E

Scanned with CamScanner

BFS (Directed Graph)



Lets Start with 5.

5 1 3 4 2.

5			

1	3	4	

3	4	2	

4	2		

2			

Scanned with CamScanner

DFS is a systematic way of visiting the nodes of either a directed or an undirected graph.

DFS visits the vertices of a graph in the following manner:

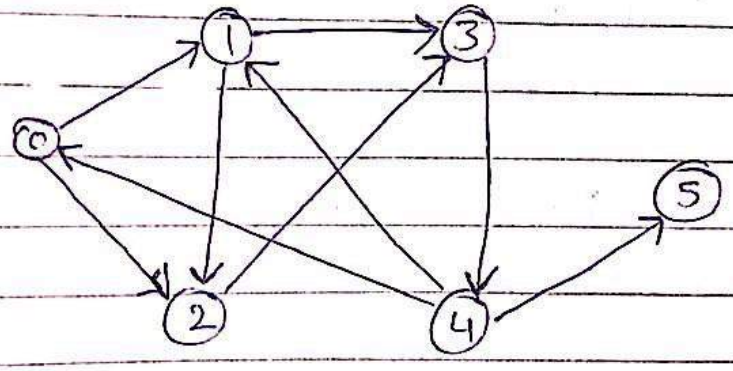
1. It selects a starting vertex v.
2. Then it chooses an incident edge (v,w) and searches recursively deeper in the graph whenever possible.
3. It does not examine all the incident edges one by one at the same time, but on the other hand, it goes deeper in the graph till no other such path exists.
4. When all edges incident to v have been explored, the algorithm backtracks to explore edges leaving the vertex from which v was discovered.

NOTE:- 1. We can pick any arbitrary node to start with.

2. We don't want to repeat visited nodes.

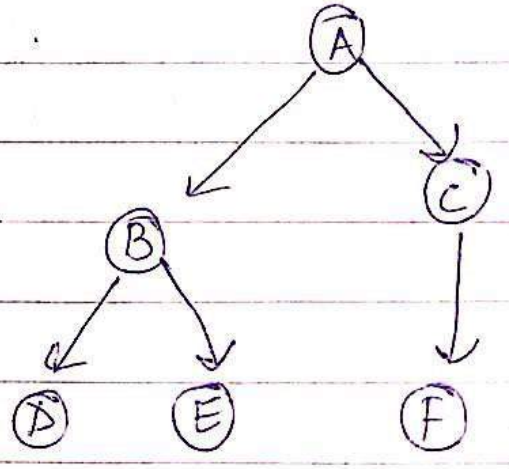
Scanned with CamScanner

Example:



0 1 3 4 5 2

Example:



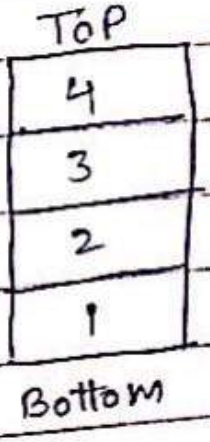
A B D E C F

Scanned with CamScanner

DFS :-

Stack is a linear data structure which follows a particular order in which the operations are performed. The order may be LIFO (Last in first out) or FILO (First In Last Out).

Example: Plates stacked over one another in the canteen. The plate which is at the top is the first one to be removed i.e. the plate which has been placed at the bottommost position remain in the stack for the longest period of time. It follows LIFO or FILO.

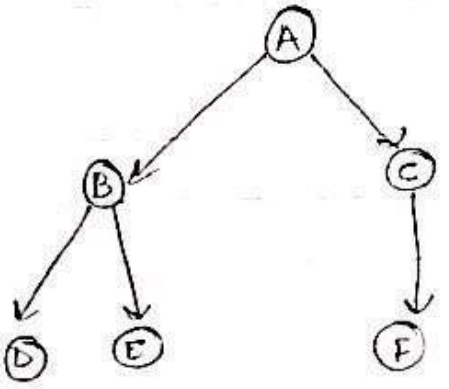


Push - Add item to top

Pop - Remove item from top

Scanned with CamScanner

Example:



Sol^n. (1) Start initially with the vertex. A.
Push A in the stack.

(2) Now see the neighbours of A, i.e B or C
Let us choose B.
A will be popped out and Push B in stack

Path – A

(3) Look for the neighbours of B i.e D and E
Pop B out, and push D and E

Path – AB

(4) Now D will be popped out first then E
Path – ABDE.

(5) Now Backtrack to A and see neighbours of A which are still not visited. i.e C
Push C in the stack

Scanned with CamScanner

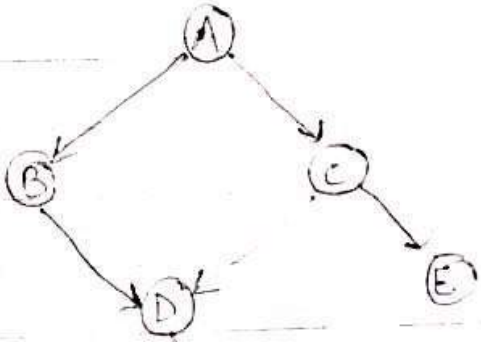
(6) Now C will popped out. and F will be pushed in.

Path - ABDEC

(7) F will be popped out.

Path - ABDECF.

Example:



Path - A B D F G C E

Scanned with CamScanner

FLOYD- WARSHALL ALGORITHM:-

Floyd Warshall algorithm is a shortest-path algorithm for graphs.

This algorithm computes the shortest distances between every pair of vertices in the input graph.

It can be used for finding shortest-paths in a weighted graph with positive or negative edge weights. (but with no negative cycles).

Note: A negative cycle is a cycle whose edges sum to a negative value.

Scanned with CamScanner

Formula :-

$d_{ij}^{(K)}$ = weight of shortest path from vertex i to j for which all intermediate node (vertices) are in $\{1, 2, \ldots K\}$.

$$d_{ij}^{(K)} = \begin{cases} w_{ij}, & \text{if } K = 0 \\ \min\left(d_{ij}^{(K-1)}, d_{iK}^{(K-1)} + d_{Kj}^{(K-1)}\right), & \text{if } K \geq 1 \end{cases}$$

We find $V \times V$ matrix.

Note : Order of the matrix is given by the number of vertices).

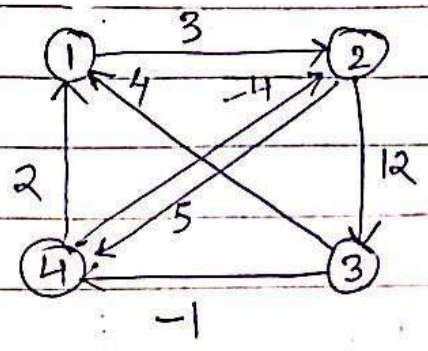
$$D^{(m)} = d_{ij}^{(m)}$$

and $$d_{ij}^{(0)} = \begin{cases} 0, & \text{if } i = j \\ w_{ij}, & \text{if } i \neq j \\ \infty, & \text{otherwise.} \end{cases}$$

Here w_{ij} is the weight of the edge from vertex i to j.

Scanned with CamScanner

Example:



I $D^0 =$

	1	2	3	4
1	0	3	∞	∞
2	∞	0	12	5
3	4	∞	0	-1
4	2	-4	∞	0

For $k=0$ $\quad d_{ij}^{(0)} = \{ w_{ij}$ if $k=0$

$$d_{ij}^{0} = \begin{cases} 0 & \text{if } i=j \\ w_{ij} & \text{if } i \neq j \\ \infty & \text{otherwise} \end{cases}$$

Scanned with CamScanner

$\therefore \; d_{11}^{(0)} = 0$ {$\because$ since there is no edge from ① to ① hence weight = 0}.

Similarly $d_{22}^{(0)} = 0$

$d_{33}^{(0)} = 0$

$d_{44}^{(0)} = 0$

$d_{12}^{(0)} = w_{12} = 3$ {weight from vertex ① to ② }

$d_{13}^{(0)} = \infty$ {No edge from ① to ③ vertex}

$d_{14}^{(0)} = \infty$ {No edge from ① to ④ vertex}

II ⇒ for $k = 1$.

$D^1 =$

	1	2	3	4
1	0	3	∞	∞
2	∞	0	12	5
3	4	7	0	-1
4	2	-4	∞	0

$$d_{11}^{(1)} = \min\{d_{11}^{(0)}, d_{11}^{(0)} + d_{11}^{(0)}\}$$
$$= \min\{0, 0\}$$
$$= 0$$

$\therefore \; d_{22}^{(1)} = 0, \; d_{33}^{(1)} = 0, \; d_{44}^{(1)} = 0.$

Now $d_{12}^{(1)} = \min\{d_{12}^{(0)}, d_{11}^{(0)} + d_{12}^{(0)}\}$
$$= \min\{3, 0+3\}$$
$$= 3$$

Scanned with CamScanner

For $k=2$

III Similarly with find. D^2.

$$D^2 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 3 & 15 & 8 \\ 2 & \infty & 0 & 12 & 5 \\ 3 & 4 & 7 & 0 & -1 \\ 4 & 2 & -4 & 8 & 0 \end{array}$$

IV For $K=3$

$$D^3 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 3 & 15 & 8 \\ 2 & 16 & 0 & 12 & 5 \\ 3 & 4 & 7 & 0 & -1 \\ 4 & 2 & -4 & 8 & 0 \end{array}$$

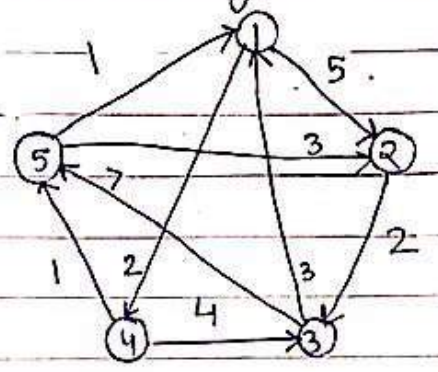
V for $k=4$.

$$D^4 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 3 & 15 & 8 \\ 2 & 7 & 0 & 12 & 5 \\ 3 & 1 & -5 & 0 & -1 \\ 4 & 2 & -4 & 8 & 0 \end{array}$$

Hence, we have find the shortest path between each pair of vertices.

Scanned with CamScanner

Example: Floyd Warshall's Algorithm



Step 1.

$$D^{(0)} = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 0 & 5 & \infty & 2 & \infty \\ 2 & \infty & 0 & 2 & \infty & \infty \\ 3 & 3 & \infty & 0 & \infty & 7 \\ 4 & \infty & \infty & 4 & 0 & 1 \\ 5 & 1 & 3 & \infty & \infty & 0 \end{array}$$

For $K=0$ $\quad d_{ij}^{(0)} = \{ w_{ij}, \text{ if } K=0 \}$.

& $$d_{ij}^{(0)} = \begin{cases} 0, & \text{if } i=j \\ w_{ij}, & \text{if } i \neq j \\ \infty, & \text{otherwise} \end{cases}$$

$\therefore \quad d_{11}^{(0)} = 0$

$d_{22}^{(0)} = d_{33}^{(0)} = d_{44}^{(0)} = d_{55}^{(0)} = 0.$

$d_{12}^{(0)} = 5.$, $d_{13}^{(0)} = \infty$ (as there is no edge from 1 to 3)

$d_{14}^{(0)} = 2$,

$d_{23}^{(0)} = 2$,

etc.

Scanned with CamScanner

Step 2.

$$D^{(1)} = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 0 & 5 & \infty & 2 & \infty \\ 2 & \infty & 0 & 2 & \infty & \infty \\ 3 & 3 & 8 & 0 & 5 & 7 \\ 4 & \infty & \infty & 4 & 0 & 1 \\ 5 & 1 & 3 & \infty & 3 & 0 \end{array}$$

For $k=1$:

$$d_{ij}^{(k)} = \min\{(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)})\}$$

$$d_{11}^{(1)} = \min\{d_{11}^{(0)}, d_{11}^{(0)} + d_{11}^{(0)}\}$$
$$= \min\{0, 0\}$$
$$= 0.$$

$$d_{12}^{(1)} = \min\{d_{12}^{(0)}, d_{11}^{(0)} + d_{12}^{(0)}\}$$
$$= \min\{5, 0+5\}$$
$$= 5.$$

$$d_{13}^{(1)} = \min\{d_{13}^{(0)}, d_{11}^{(0)} + d_{13}^{(0)}\}$$
$$= \min\{\infty, 0+\infty\}$$
$$= \infty \quad \text{and so on.}$$

Step 3.

$K=2$

$$D^{(2)} = \begin{bmatrix} 0 & 5 & 7 & 2 & \infty \\ \infty & 0 & 2 & \infty & \infty \\ 3 & 8 & 0 & 5 & 7 \\ \infty & \infty & 4 & 0 & 1 \\ 1 & 3 & 5 & 3 & 0 \end{bmatrix}$$

Scanned with CamScanner

Step 4. $k = 3$.

$$D^{(3)} = \begin{bmatrix} 0 & 5 & 7 & 2 & 14 \\ 5 & 0 & 2 & 7 & 9 \\ 3 & 8 & 0 & 5 & 7 \\ 7 & 12 & 4 & 0 & 1 \\ 1 & 3 & 5 & 3 & 0 \end{bmatrix}$$

Step 5. $K = 4$.

$$D^{(4)} = \begin{bmatrix} 0 & 5 & 6 & 2 & 3 \\ 5 & 0 & 2 & 7 & 8 \\ 3 & 8 & 0 & 5 & 6 \\ 7 & 12 & 4 & 0 & 1 \\ 1 & 3 & 5 & 3 & 0 \end{bmatrix}$$

Step 6. $K = 5$.

$$D^{(5)} = \begin{bmatrix} 0 & 5 & 6 & 2 & 3 \\ 5 & 0 & 2 & 7 & 8 \\ 3 & 8 & 0 & 5 & 6 \\ 2 & 4 & 4 & 0 & 1 \\ 1 & 3 & 5 & 3 & 0 \end{bmatrix}.$$

Scanned with CamScanner

Therefore,

$$B = A + A^2 + A^3 + A^4 = \begin{bmatrix} 3 & 1 & 3 & 1 & 4 \\ 1 & 3 & 1 & 3 & 4 \\ 3 & 1 & 7 & 5 & 4 \\ 1 & 3 & 5 & 7 & 4 \\ 4 & 4 & 4 & 4 & 8 \end{bmatrix}.$$

Since B has no zero entry off the main diagonal, the graph is connected.

Definition 8.63

Suppose a graph G has n vertices $v_1, v_2, \ldots, v_n$ and t edges $e_1, e_2, \ldots, e_t$. The **incidence matrix** $B(G)$ of G is the $n \times t$ matrix $B(G) = (b_{ij})$, where

$$b_{ij} = \text{the number of times that the vertex } v_i \text{ is incident with the edge } e_j,$$

that is,

$$b_{ij} = \begin{cases} 0 & \text{if } v_i \text{ is not end of } e_j \\ 1 & \text{if } v_i \text{ is an end of the non-loop } e_j \\ 2 & \text{if } v_i \text{ is an end of the loop } e_j \end{cases}$$

8.9 PLANAR GRAPHS

Definition 8.64

A graph which can be drawn in the plane so that its edges do not cross is said to be **planar.**
For example, the graph shown in Figure 8.107 is planar:

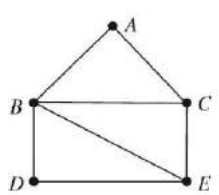


Figure 8.107

Also the complete graph K_4 shown below (Figure 8.108) is planar.

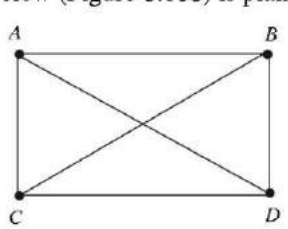


Figure 8.108

In fact, it can be redrawn as shown below in Figure 8.109 so that no edges cross.

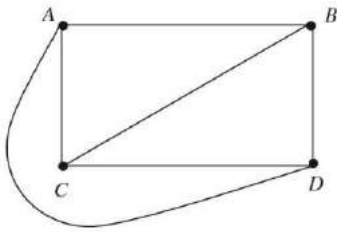


Figure 8.109

But the map K_5 is not planar because in this case, the edges cross each others (see Figure 8.110).

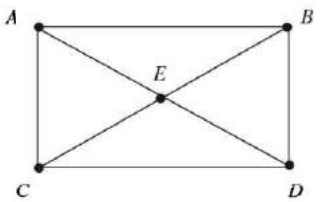


Figure 8.110

Definition 8.65

An area of the plane that is bounded by edges of the planar graph and is not further subdivided into subareas is called a **region** or **face** of the planar graph.

A face is characterized by the cycle that forms its boundary.

Definition 8.66

A region is said to be **finite** if its area is finite and **infinite** if its area is infinite. Clearly a planar graph has exactly one infinite region.

For example, consider the graphs shown in Figure 8.111.

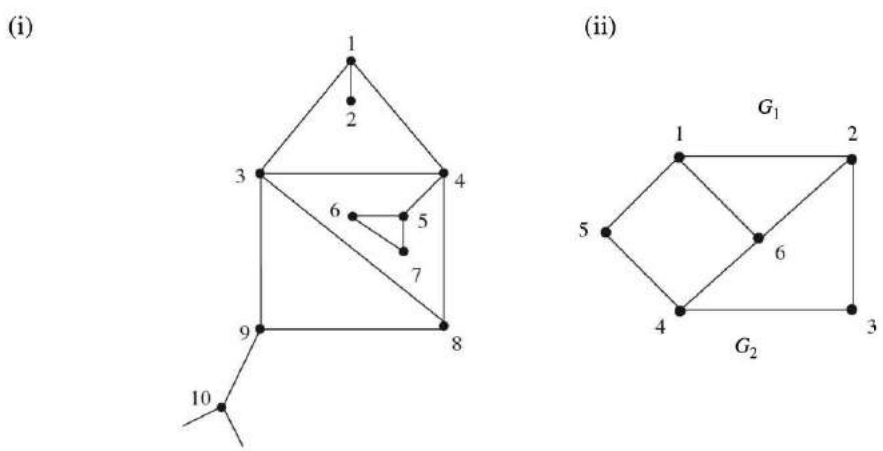


Figure 8.111

We note that in the graph G_1, there are five regions **A, B, C, D, E** as shown below:

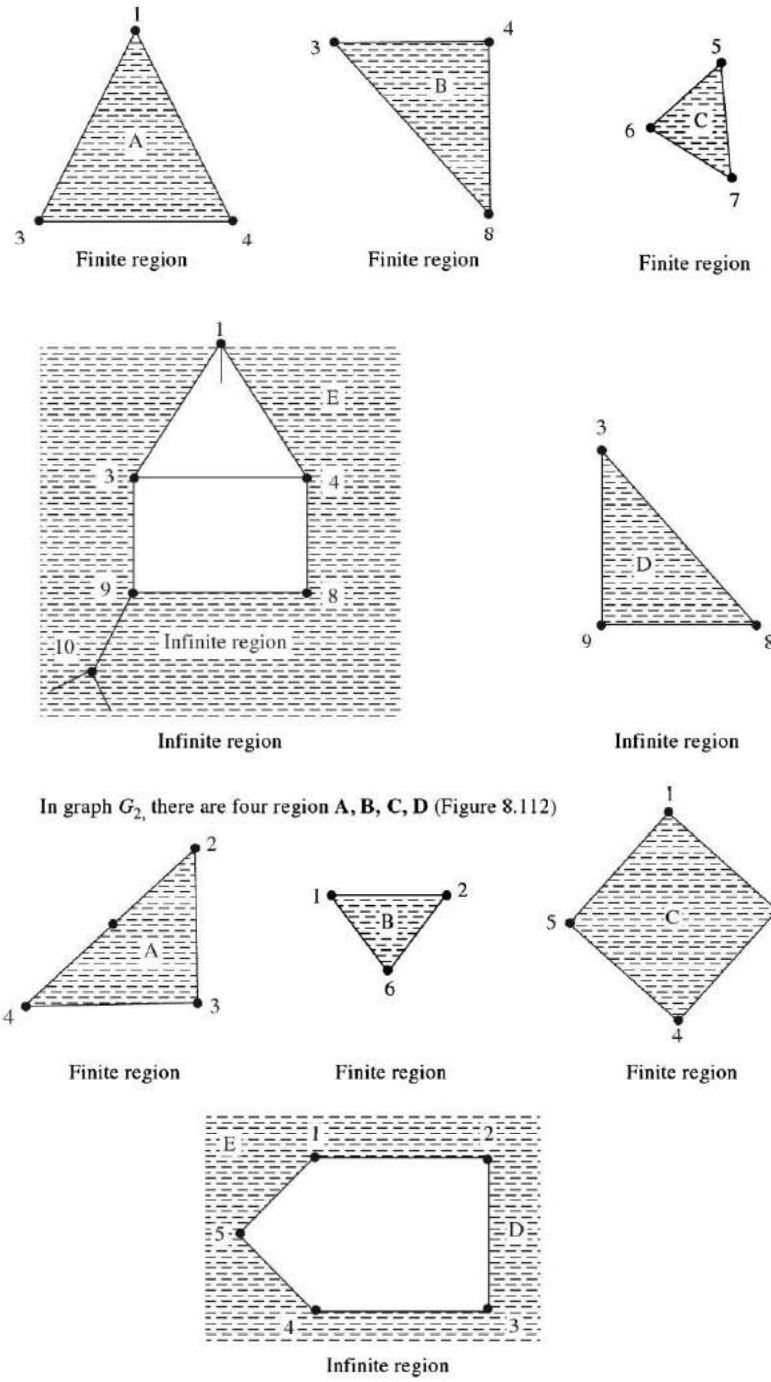


Figure 8.112

Definition 8.67

Let f be a face (region) in a planar graph. The length of the cycle (or closed walk) which borders f is called the **degree of the region** f. It is denoted by $\deg(f)$.

In a planar graph we note that **each edge either borders two regions or is contained in a region and will occur twice in any walk along the border of the region.** Thus we have the following:

Theorem 8.15

The sum of the degrees of the regions of a map is equal to twice the number of edges.

For example, in the graph G_2, discussed above, we have

$$\deg(A)=4, \quad \deg(B)=3, \quad \deg(C)=4, \quad \deg(d)=5.$$

The sum of degrees of all regions $=4+3+4+5=16$.

$$\text{Number of edges in } G_2=8.$$

Hence the sum of degrees of regions is twice the number of edges.

Theorem 8.16 (Euler's Formula for Connected Planar Graphs)

If G is a connected planar graph with e edges, v vertices and r regions, then

$$v-e+r=2.$$

Proof. We shall use induction on the number of edges. Suppose that $e=0$. Then the graph G consists of a single vertex, say P. Thus, G is as shown below:

$$\bullet\ P$$

and we have $e=0$, $v=1$, $r=1$. Thus $1-0+1=2$ and the formula holds in this case.

Suppose that $e=1$. Then the graph G is one of the two graphs shown below:

$e=1, v=2, r=1$ $\qquad$ $e=1, v=1, r=2$

We see that, in either case, the formula holds.

Suppose that the formula holds for connected planar graph with n edges. We shall prove that this holds for graph with $n+1$ edges. So, let G be the graph with $n+1$ edges. Suppose first that G contains no cycles. Choose a vertex v_1 and trace a path starting at v_1. Ultimately, we will reach a vertex a with degree 1, that we cannot leave (see Figure 8.113).

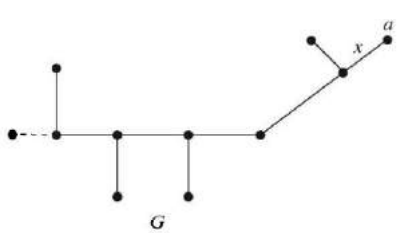


Figure 8.113

We delete "a" and the edge x incident on "a" from the graph G. The resulting graph G' (Figure 8.114) has n edges and so by induction hypothesis, the formula holds for G'. Since G has one more edge than G', one more vertex than G' and the same number of faces as G', it follows that the formula $v-e+r=2$ holds also for G.

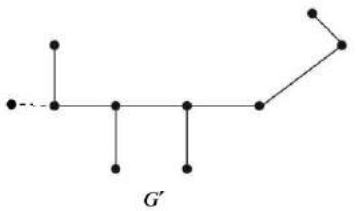


Figure 8.114

Now suppose that G contains a cycle. Let x be an edge in a cycle as shown in Figure 8.115.

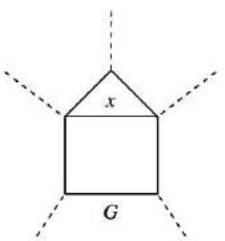


Figure 8.115

Now the edge x is part of a boundary for two faces. We delete the edge x but no vertices to obtain the graph G' as shown in Figure 8.116.

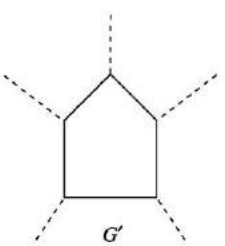


Figure 8.116

Thus G' has n edges and so by induction hypothesis the formula holds. Since G has one more face (region) than G', one more edge than G' and the same number of vertices as G', it follows that the formula $v-e+r=2$ also holds for G. Hence, by mathematical induction, the theorem is true.

Remark 8.8 Planarity of a graph is not affected if

(i) An edge is divided into two edges by the insertion of new vertex of degree 2 (Figure 8.117).

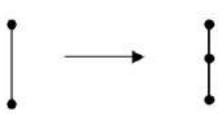


Figure 8.117

(ii) Two edges that are incident with a vertex of degree 2 are combined as a single edge by the removal of that vertex (Figure 8.118).

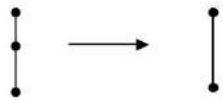


Figure 8.118

Definition 8.68

Two graphs G_1 and G_2 are said to be **isomorphic to within vertices of degree 2 (or homeomorphic)** if they are isomorphic or if they can be transformed into isomorphic graphs by repeated insertion and/or removal of vertices of degree 2.

Definition 8.69

The repeated insertion/removal of vertices of degree 2 is called **sequence of series reduction**.

For example, the graphs shown in Figure 8.119 are isomorphic to within vertices of degree 2.

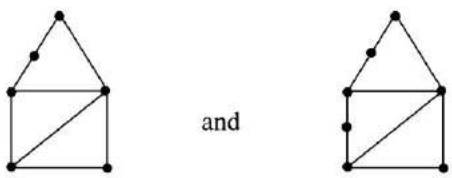


Figure 8.119

If we define a relation R on the set of graphs by $G_1 \; R \; G_2$ if G_1 and G_2 are homeomorphic, then R is an equivalence relation. Each equivalence class consists of a set of mutually homeomorphic graphs.

EXAMPLE 8.46

Show that the graph $K_{3,3}$, given in Figure 8.120 below, is not planar.

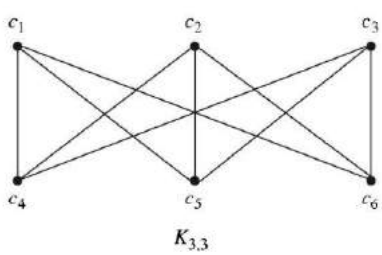


Figure 8.120

A problem based on this example can be stated as "Three cities c_1, c_2 and c_3 are to be directly connected by express ways to each of three cities c_4, c_5 and c_6. Can this road system be designed so that the express ways do not cross?" This example shows that it cannot be done.

Solution.

Suppose that $K_{3,3}$ is planar. Since every cycle in $K_{3,3}$ has at least four edges, each face (region) is bounded by at least four edges. Thus the number of edges that bound regions is at least $4r$. Also, in a planar graph each edge belongs to at most two bounding cycles. Therefore,

$$2e \geq 4r \text{ (sums of degrees of region is equal to twice the number of edges)}$$

But, by Euler's formula for planar graph,

$$r = e - v + 2.$$

Hence,

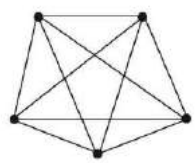
$$2e \geq 4(e - v + 2). \tag{1}$$

In case of $K_{3,3}$ we have $e=9$, $v=6$ and so (1) yields

$$18 \geq 4\,(9-6+2) = 20,$$

which is a contradiction. Therefore $K_{3,3}$ is not planar.

Remark 8.9 By a argument similar to the above example, we can show that the graph K_5 (Figure 8.121) is not planar.



(Non-planar graph K_5)

Figure 8.121

We observe that if a graph contains $K_{3,3}$ or K_5 as a subgraph, then it cannot be planar.

The following theorem, which we state without proof, gives necessary and sufficient condition for a graph to be planar.

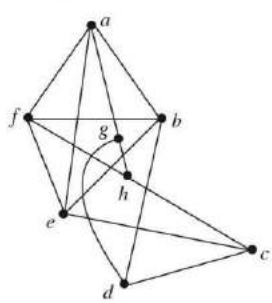
Kuratowski's Theorem 8.17

A graph G is planar if and only if G does not contain a **subgraph** homeomorphic to $K_{3,3}$ or K_5.

The complete graph K_5 and the complete bipartite graph $K_{3,3}$ are called the **Kuratowski graphs.**

EXAMPLE 8.47

Using Kuratowski's theorem show that the graph G, shown in Figure 8.122, is not planar.



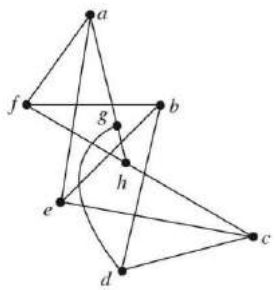
G

Figure 8.122

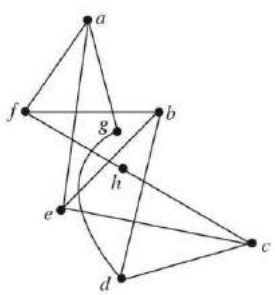
Solution.

Let us try to find $K_{3,3}$ in the graph G. We know that in $K_{3,3}$, each vertex has degree 3. But we note that in G, the degree of a, b, f and e each is 4. So we eliminate the edges (a, b) and (f, e) so that all vertices have degree 3. If we eliminate one more edge, we will obtain two vertices of degree 2 and we can then carry out series reduction. The resulting graph will have nine edges

Also we know that $K_{3,3}$ **has nine edges**. So this approach seems promising. Using trial and error, we find that the edge (g, h) should be removed. Then g and h have degree 2.

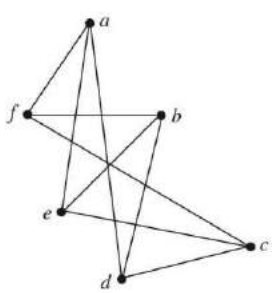


(Graph obtained by deleting edges (a, b) and (f, e)).



(Graph obtained by eliminating the edge (g, h).

Performing series reduction now, we obtain an isomorphic copy of $K_{3,3}$ (Figure 8.123).



(Isomorphic copy of $K_{3,3}$, obtained by series reduction)

Figure 8.123

Hence, by Kurtowski's theorem, the given graph G is not planar.

EXAMPLE 8.48

If the graph given in Figure 8.124 is planar, redraw it so that no edges cross, otherwise find a subgraph homeomorphic to either K_5 or $K_{3,3}$.

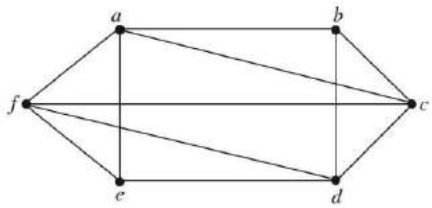


Figure 8.124

Solution.

The given graph can be redrawn as shown in the Figure 8.125.

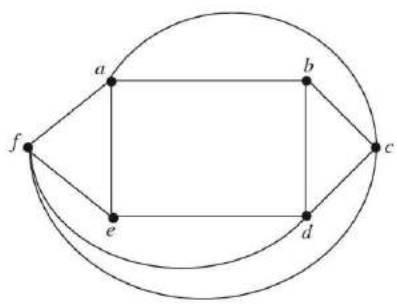


Figure 8.125

We then observe that no edges cross. Hence the given graph is planar.

Theorem 8.18

Let G be a connected planar graph with v vertices, e edges, where $v \geq 3$. Then $e \leq 3v-6$ or $6 \leq 3v - e$.

(Note that the theorem is not true for K_1, where $v=1$ and $e=0$ and is not true for K_2, where $v=2$ and $e=1$).

Proof. Let r be the number of regions in a planar representation of the graph G. The sum of the degrees of the regions is equal to $2e$. But each region has degree greater than or equal to 3. Hence $2\,e \geq 3\,r$, that is, $r \leq \frac{2e}{3}$. But, by Euler's formula,

$$v-e+r=2.$$

So,

$$2=v-e+r \leq v-e+\frac{2e}{3}=v-\frac{e}{3}.$$

Hence,

$$6 \leq 3\,v-e \quad \text{or} \quad e \leq 3v-6$$

Remark 8.10 Using above theorem, let us consider planarity of K_5. Here $v=5$, $e=10$. Suppose K_5 is planar, then by the above theorem,

$$6 \leq 3(5)-10=5,$$

which is impossible. Hence K_5 is non-planar.

Definition 8.70

Any graph homeomorphic either to K_5 or to $K_{3,3}$ is called **Kuratowski subgraph**.

EXAMPLE 8.49

Find Kuratowski subgraph of the following graph (Figure 8.126):

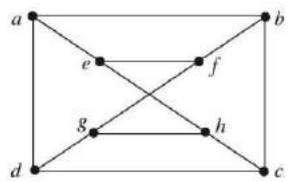


Figure 8.126

Solution.

In the given graph,

Number of vertices=8

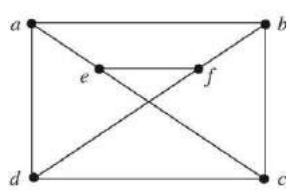
Number of edges=12.

Firstly, we remove the edge (g, h). Then $\deg(g)=2$, $\deg(h)=2$ and so the vertices g and h can be removed. Then, we have

Number of vertices=6

Number of edges=9

and the graph reduces to



,

which is homeomorphic to $K_{3,3}$ (Kuratowski graph) shown in Figure 8.127.

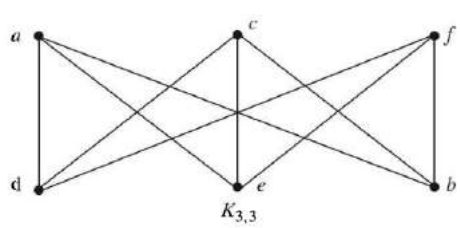


Figure 8.127

Thus the given graph is non-planar by Kuratowski theorem.

EXAMPLE 8.50

Find a subgraph homeomorphic to either K_5 or $K_{3,3}$ in the graph given below (Figure 8.128):

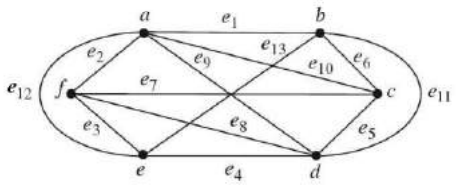


Figure 8.128

Solution.

Here,

Number of vertices = 6

Number of edges = 13

$$\deg(a)=5, \quad \deg(f)=4, \quad \deg(c)=4, \quad \deg(d)=5$$

But, in $K_{3,3}$, the degree of each vertex is 3. We delete the edges (a, c), (f, d), (a, e), (b, d), then degree of each vertex a, f, c and d becomes 3. As such, the subgraph becomes as given below (Figure 8.129), which has six vertices each with degree 3 and has 9 edges.

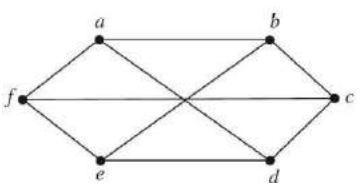


Figure 8.129

This subgraph is homeomorphic to $K_{3,3}$ as shown in Figure 8.130.

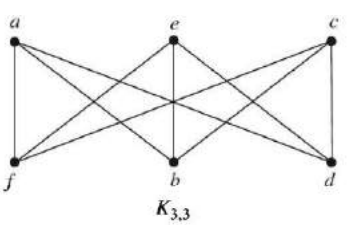


Figure 8.130

This also shows, by Kuratowski theorem, that the given graph is not planar.

8.10 COLOURING OF GRAPH

Definition 8.71

Let G be a graph. The assignment of colours to the vertices of G, one colour to each vertex, so that the adjacent vertices are assigned different colours is called **vertex colouring** or **colouring of the graph** G.

Definition 8.72

A graph G is **n-colourable** if there exists a colouring of G which uses n colours.

Definition 8.73

The minimum number of colours required to paint (colour) a graph G is called the **chromatic number of** G and is denoted by $\chi(G)$.

EXAMPLE 8.51

Find the chromatic number for the graph shown in the Figure 8.131.

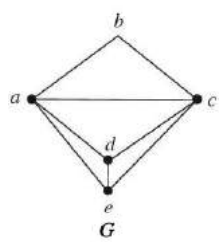


Figure 8.131

Solution.

The triangle $a\ b\ c$ needs three colours. Suppose that we assign colours c_1, c_2, c_3 to a, b and c respectively. Since d is adjacent to a and c, d will have different colour than c_1 and c_3. So we paint d by c_2. Then e must be painted with a colour different from those of a, d and c, that is, we cannot colour e with c_1, c_2 or c_3. Hence, we have to give e a fourth colour c_4. Hence

$$\chi(G)=4.$$

Welsh–Powell algorithm to determine upper bound to the chromatic number of a given graph.

The input is a given graph G.

1. Order the vertices of G according to decreasing degree.
2. Assign the first colour, say c_1, to the first vertex and then, in sequential order, assign c_1 to each vertex, which is not adjacent to a previous vertex assigned c_1.
3. Repeat Step 2 with a second colour c_2 and the subsequence of the remaining non-painted vertices.
4. Repeat Step 3 with a third colour c_3, then a fourth colour c_4 and so on until all vertices are coloured.
5. Exit.

EXAMPLE 8.52

Use Welsh-Powell algorithm to determine an upper bound to the chromatic number of the "Wheel" graph shown in Figure 8.132.

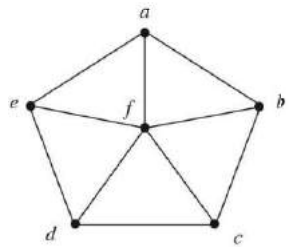


Figure 8.132

Solution.

We note that

$$\deg(f)=5,$$
$$\deg(a)=\deg(b)=\deg(c)=\deg(d)=\deg(e)=3.$$

Step 1. Ordering the vertices according to decreasing degree yields

$$f, a, b, c, d, e$$

Step 2. Paint f with colour c_1.
Step 3. Paint a, d with colour c_2.
Step 4. Paint b, e with colour c_3.
Step 5. Paint c with colour c_4.

Hence $\chi(G)\leq 4$. Also since there is a triangle in the given graph so we have $\chi(G)\geq 3$. Thus $3\leq\chi(G)\leq 4$. But we do not yet know exactly what the chromatic number is. We try to build a 3-colouring of G.

Let us start colouring the triangle abf with the colours c_1, c_2, c_3, respectively. Since c is adjacent to the vertices b and f of colour c_2 and c_3, respectively, c is forced to be coloured c_1 and then d is forced to be c_2. However, now the adjacent vertices a and e cannot both have colour c_1. Thus the graph cannot be 3-coloured. But using a fourth colour c_4 for e gives us 4-colouring of G. Hence, $\chi(G)=4$.

EXAMPLE 8.53

Use Welsh-Powell algorithm to colour the graph shown in Figure 8.133.

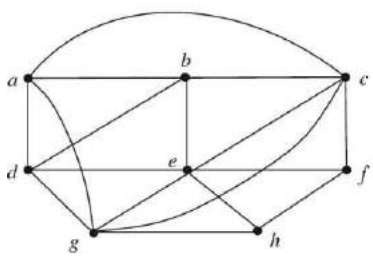


Figure 8.133

Solution.

Ordering the vertices according to decreasing degrees, we get the sequence e, c, g, a, b, d, f, h.

Use the colour c_1 to colour (paint) e and a.
Use the colour c_2 to paint c, d and h.
Use third colour c_3 to paint vertices g, b and f.

Thus all the vertices are painted such that no adjacent vertices get the same colour. Hence $\chi(G)=3$.

Some rules for colouring a graph

1. $\chi(G) \leq |V|$, where $|V|$ is the number of vertices of G.
2. A triangle always requires three colours, that is, $\chi(K_3)=3$. Similarly $\chi(K_n)=n$, where K_n is the complete graph of n vertices.
3. If some subgraph of G requires k-colours, then $\chi(G) \geq k$.
4. If $\deg(v)=n$, then at most n colours are required to colour the vertices adjacent to v.
5. $\chi(G)=\max\{\chi(C)$: C is a connected component of $G\}$.
6. Every n-colourable graph has at least n vertices v such that $\deg(v) \geq n-1$.
7. $\chi(G) \leq 1+\Delta(G)$, where $\Delta(G)$ is the largest degree of any vertex of G.
8. The following statements are equivalent:
   (i) A graph G is 2-colourable
   (ii) G is bipartite
   (iii) Every cycle of G has even length
9. If $\delta(G)$ is the smallest degree of any vertex of G, then

$$\chi(G) \geq \frac{|v|}{|v|-\delta(G)}.$$

EXAMPLE 8.54

Find chromatic number of the graphs shown in Figure 8.134.

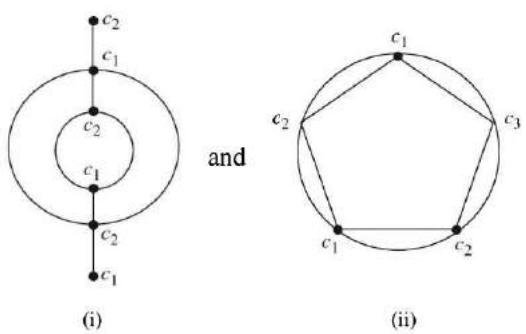


Figure 8.134

Solution.

The graph (i) is 2-colourable whereas the graph (ii) is 3-colourable.

Theorem 8.19

Bipartite graph is 2-colourable unless G is edgeless.

Proof. A two colouring is obtained by assigning one colour to every vertex in one of the bipartition parts and another colour to every vertex in the other partition part as shown in Figure 8.135.

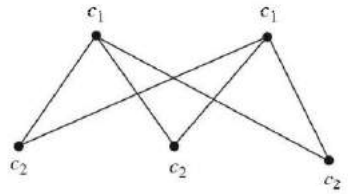


Figure 8.135

Corollary 8.3

Even cycle graphs C_{2n} have $\chi(C_{2n})=2$.

Proof. An even cycle graph is bipartite and therefore, by Theorem 8.19, its chromatic number is 2.

For example consider the graph shown in Figure 8.136.

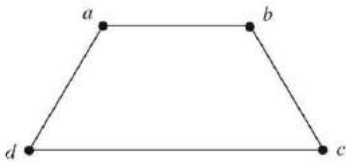


Figure 8.136

It is equivalent to the following bipartite graph shown in Figure 8.137.

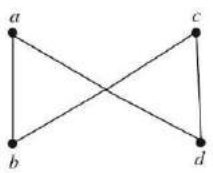


Figure 8.137

Therefore, it is 2-colourable.

Proposition 8.2

Odd cycle graph C_{2n+1} has $\chi(C_{2n+1})=3$.

Proof. Let $v_1, v_2, \ldots, v_{2n}, v_{2n+1}$ be vertices of a cycle graph c_{2n+1}. If two colours were sufficient, then they would have to alternate around the cycle. Thus, the odd subscripted would have to be one colour and the even subscripted vertices have to be second colour but vertex v_{2n+1} is adjacent to v_1, and so according to this scheme two adjacent vertices v_1 and v_{2n+1} have the same colour. This is a contradiction and so C_{2n+1} is not 2-colourable but 3-colourable.

It follows, therefore, that "The chromatic number of a cycle is either two or three, depending on whether its length is even or odd."

Definition 8.74

The **join** $G+H$ of the graphs G and H is obtained from the graph $G \cup H$ by adding an edge between each vertex of H.

For example, if we have a graph as shown below (Figure 8.138):

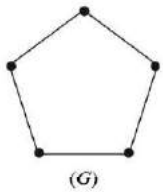


Figure 8.138

and a graph

$$\bullet\ K_1,$$

then the join $G+K_1$ of the graphs G and K_1 is the graph as shown below (Figure 8.139):

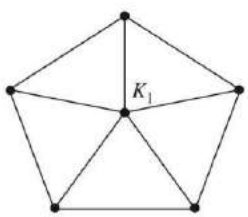


Figure 8.139

Regarding the chromatic number of the join $G+H$ of the graphs G and H we have the following:

Proposition 8.3

The join of a graph G and H has chromatic number

$$\chi(G+H)=\chi(G)+\chi(H).$$

Definition 8.75

The n vertex wheel graph W_n is called an **odd-order wheel** if n is odd and an **even-order wheel** if n is even.

Proposition 8.4

Odd-order wheel graph has

$$\chi(W_{2m+1})=3, \quad \text{for all } m\geq 1.$$

Proof. Using the fact that the wheel graph W_{2m+1} is the join $C_{2m}+K_1$, it follows that

$$\chi(W_{2m+1})=\chi(C_{2m})+\chi(K_1)=2+1=3.$$

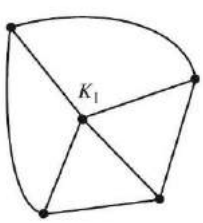


Figure 8.140 (W_5)

Proposition 8.5

Even-order wheel graph has $\chi(W_{2m})=4$.

Proof. Using the fact that $W_{2m}=c_{2m-1}+K_1$, it follows that

$$X(W_{2m})=\chi(c_{2m-1})+\chi(K_1)=3+1=4.$$

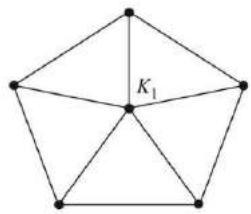


Figure 8.141 (W_6)

EXAMPLE 8.55

Find chromatic number of the graph shown in Figure 8.142:

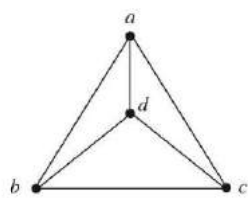


Figure 8.142

Solution.

Here degree of each vertex is 3. Therefore, $\chi(G)\leq 1+3=4$. Since it has a triangle, $\chi(G)\geq 3$. Thus, $3\leq\chi(G)\leq 4$.

We first colour the triangle *abd* with colours c_1, c_2, c_3, respectively. Since c is adjacent to each of a, b and d, therefore a different colour c_4 have to be given to it. Hence the graph is 4-colourable and so $\chi(G)=4$.

Note: If we apply rule 9, then $\chi \geq \dfrac{4}{(4-3)} = 4$. Hence $\chi(G) = 4$.

Theorem 8.20 (Four-Colour Theorem of Appel and Haken)

Any planar graph is 4-colourable.

(Proof of the theorem is out of the scope of this book).

8.11 DIRECTED GRAPHS

Definition 8.76

A **directed graph** or **digraph** consists of two finite sets:

(i) A set V of vertices (or nodes or points).

(ii) A set E of directed edges (or arcs), where each edge is associated with an ordered pair (v, w) of vertices called its endpoints. If edge e is associated with the ordered pair (v, w), then e is said to be **directed edge from** v **to** w.

The directed edges are indicated by arrows.

We say that edge $e=(v, w)$ is incident from v and is incident into w.

The vertex v is called **initial vertex** and the vertex w is called the **terminal vertex** of the directed edge (v, w).

Definition 8.77

Let G be a directed graph. The **outdegree of a vertex v of G** is the number of edges beginning at v. It is denoted by outdeg(v).

Definition 8.78

Let G be a directed graph. The **indegree of a vertex v of G** is the number of edges ending at v. It is denoted by indeg(v).

EXAMPLE 8.56

Consider the directed graph shown below (Figure 8.143):

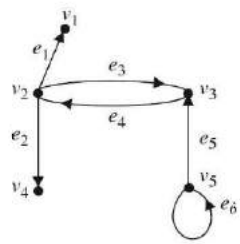


Figure 8.143

Here edge e_1 is (v_2, v_1) whereas e_6 is denoted by (v_5, v_5) and is called a loop. The indegree of v_2 is 1, outdegree of v_2 is 3.

Definition 8.79

A vertex with 0 indegree is called a **source**, whereas a vertex with 0 outdegree is called a **sink**. For instance, in the above example, v_1 is a sink.

Definition 8.80

If the edges and/or vertices of a directed graph G are labelled with some type of data, then G is called a **labelled directed graph.**

Definition 8.81

Let G be a **directed graph** with ordered vertices $v_1, v_2, \ldots, v_n$. **The adjacency matrix of G is the matrix** $A=(a_{ij})$ over the set of non-negative integers such that

$$a_{ij} = \text{the number of arrows from } v_i \text{ to } v_j,\ i, j = 1, 2, \ldots, n.$$

EXAMPLE 8.57

Find the adjacency matrices for the graphs given below (Figure 8.144):

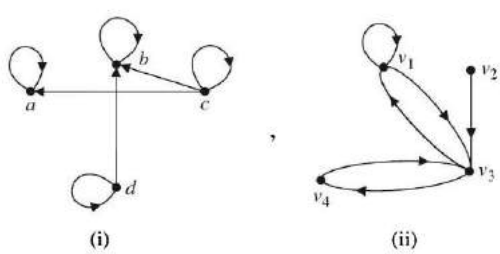


Figure 8.144

Solution.

(i) The edges in the directed graph are (a, a), (b, b), (c, c), (d, d), (c, a), (c, b) and (d, b). Therefore the adjacency matrix $A=(a_{ij})$ is

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

(ii) The edges in the graph in (ii) are (v_2, v_3), (v_1, v_1), (v_1, v_3), (v_3, v_1), (v_3, v_4), (v_4, v_3). Hence the adjacency matrix is

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

EXAMPLE 8.58

Find the directed graph represented by the adjacency matrix:

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

Solution.

We observe that $a_{12}=1, a_{23}=1, a_{34}=1, a_{35}=1, a_{41}=1, a_{42}=1$. Hence the digraph is as shown in Figure 8.145.

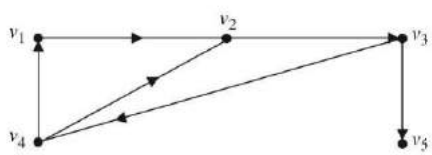


Figure 8.145

Definition 8.82

In a directed graph, if there is no more than one directed edge in a particular direction between a pair of vertices, then it is called **simple directed graph**.

For example, the graph shown in Figure 8.146 is a simple directed graph.

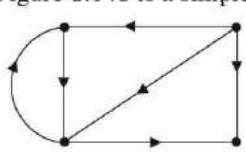


Figure 8.146

A directed graph which is not simple is called **directed multi-graph**.

Definition 8.83

A simple digraph is said to be **strongly connected** if for each pair of vertices v, w, there are path, from v to w and w to v.

For example, the graphs show in Figure 8.147 are strongly connected.

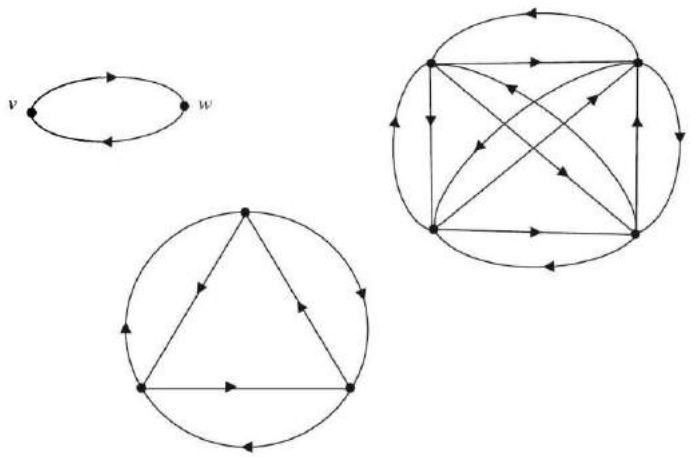


Figure 8.147

Definition 8.84

Let G be a simple digraph with n vertices. Then a $n \times n$ matrix $P=(p_{ij})$ such that

$$p_{ij} = \begin{cases} 1 & \text{if there is a path from } v_i \text{ to } v_j \\ 0 & \text{otherwise} \end{cases}$$

is called **path matrix** or **reachability matrix** of G.

The path matrix can be given in term of adjacency matrix:

$$P = A \vee A^{(2)} \vee A^{(3)} \vee \cdots \vee A^{(n)}$$

and

$$A^{(2)} = A \odot A, \quad A^{(r)} = A^{(r-1)} \odot A,$$

where $\vee$ represents Boolean addition and $\odot$ denotes the Boolean matrix multiplication.

Warshall's algorithm for finding path matrix from adjacency matrix

We discuss this algorithm with the help of the following example.

EXAMPLE 8.59

Find the adjacency matrix and the path matrix for the digraph shown in Figure 8.148.

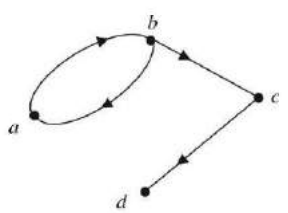


Figure 8.148

Solution.

The adjacency matrix of this digraph is:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

Then the path matrix $P=(p_{ij})$ is found as follows: We take

$$P_0 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} = A \text{ itself.}$$

Now we find P_1 by consulting column 1 and row 1. We note that P_0 has 1 in location **2** of column 1 and location **2** of row 1. Thus the element p_{22} is replaced by 1 in P_0. Thus we have

$$P_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

To find P_2 we consult column 2 and row 2 of P_1. We note that P_1 has 1 in locations 1 and 2 of column 2 and locations 1, 2 and 3 of row 2. Thus, to obtain P_2 we should put 1 in positions $p_{11}, p_{12}, p_{13}, p_{21}, p_{22}$ and p_{23} of matrix P_1. We thus have

$$P_2 = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

We now consult column 3 and row 3 of P_2. The matrix P_2 has 1 in locations 1 and 2 of column 3 and 1 in location 4 of row 3. Thus we shall obtain P_3 by putting 1 in positions p_{14}, p_{24}. Thus

$$P_3 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

Finally, we consult column 4 and row 4 of P_3. We have 1 in location 1, 2, 3 of column 4 and there is no 1 in row 4. Hence $P_4=P_3$. Thus the path matrix is

$$P = (p_{ij}) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

Remark 8.11 Given an adjacency matrix of a graph G, we first draw digraph of G and then apply definition of path matrix. We will obtain path matrix.

Now, we state a result without proof.

Theorem 8.21

Let A be the adjacency matrix of a graph G with n vertices and let $B_n = A + A^2 + A^3 + \cdots + A^n$. Then the path matrix P and B_n have the same non-zero entries.

Let G be a strongly connected directed graph. Then for any pair of vertices v and w in G, there is a path from v to w and from w to v. Accordingly, G is **strongly connected if and only if the path matrix P of G has no zero entries**.

Thus, in view of the above theorem, we have:

Theorem 8.22

Let A be adjacency matrix of a graph G with n vertices and let $B_n = A + A^2 + \ldots + A^n$. Then G is strongly connected if and only if B_n has no zero entries.

EXAMPLE 8.60

A directed graph has the following adjacency matrix. Check whether it is strongly connected.

$$A(G) = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}.$$

Solution.

To find the path matrix, we first consider column 1 and row 1 of

$$P_0 = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} = A(G).$$

In column 1, the entry 1 is located at 2 and 3 positions and 1 is at locations 2 and 3 in row 1. Thus put 1 at $p_{22}, p_{23}, p_{32}, p_{33}$ and get

$$P_1 = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

We note that 1 is located in column 2 of P_1 at position 1, 2, 3, and 1 is located at position 1, 2, 3 of row 2. Thus put 1 at $p_{11}, p_{12}, p_{13}, p_{21}, p_{22}, p_{23}, p_{31}, p_{32}, p_{33}$, and get

$$P_2 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

The next step will not change the position. Hence,

$$P = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad \text{(all non-zero entries).}$$

Therefore, G is strongly connected.

The graph of G is as shown below (Figure 8.149):

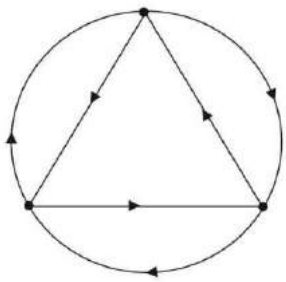


Figure 8.149

Remark 8.12 In the above example, we also note that

$$B_3 = A + A^2 + A^3$$
$$= \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} + \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} + \begin{bmatrix} 2 & 3 & 3 \\ 3 & 2 & 3 \\ 3 & 3 & 2 \end{bmatrix} = \begin{bmatrix} 4 & 5 & 5 \\ 5 & 4 & 5 \\ 5 & 5 & 4 \end{bmatrix} \quad \text{(all non-zero entries)}$$

showing that G is strongly connected.

8.12 TREES

Definition 8.85

A graph is said to be a **tree** if it is a connected acyclic graph.

A **trivial tree** is a graph that consists of a single vertex. An **empty tree** is a tree that does not have any vertices or edges.

For example, the graphs shown in Figure 8.150 are all trees.

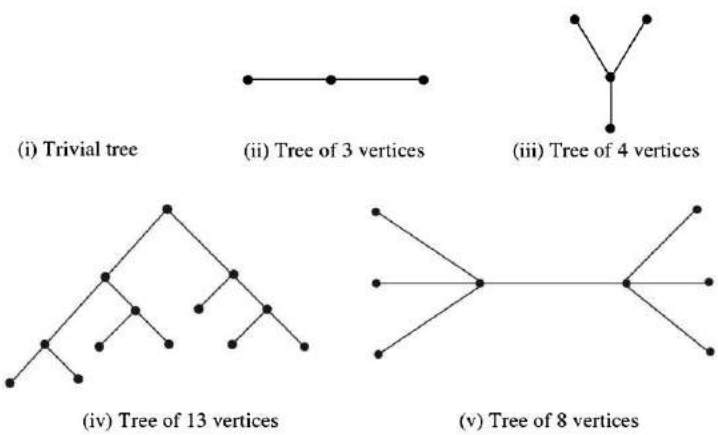


Figure 8.150

But the graphs shown in Figure 8.151 are not trees:

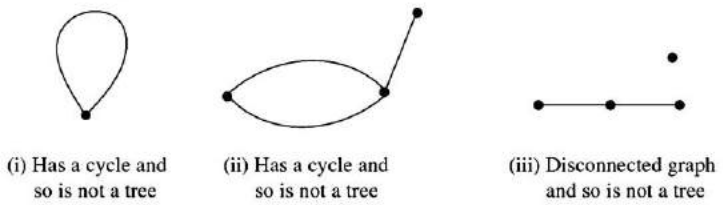


Figure 8.151

Definition 8.86

A collection of disjoint trees is called a **forest**.

Thus a graph is a forest if and only if it is circuit free.

Definition 8.87

A vertex of degree 1 in a tree is called a **leaf** or a **terminal node** or a **terminal vertex**.

Definition 8.88

A vertex of degree greater than 1 in a tree is called a **branch node** or **internal node** or **internal vertex**.

Consider the tree shown in Figure 8.152.

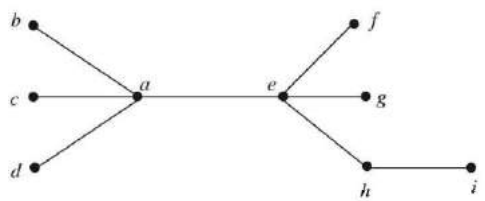


Figure 8.152

In this tree, the vertices b, c, d, f, g, i are leaves whereas the vertices a, e, h are branch nodes.

8.12.1 Characterization of Trees

We have the following interesting characterization of trees:

Lemma 8.1

A tree that has more than one vertex has at least one vertex of degree 1.

Proof. Let T be a particular but arbitrary chosen tree having more than one vertex (Figure 8.153).

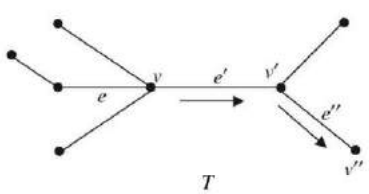


Figure 8.153

1. Choose a vertex v of T. Since T is connected and has at least two vertices, v is not isolated and there is an edge e incident on v.
2. If $\deg(v) > 1$, there is an edge $e' \neq e$ because there are, in such a case, at least two edges incident on v. Let v' be the vertex at the other end of e'. This is possible because e' is not a loop by the definition of a tree.
3. If $\deg(v') > 1$, then there are at least two edges incident on v'. Let e'' be the other edge different from e' and v'' be the vertex at other end of e''. This is again possible because T is acyclic.
4. If $\deg(v'') > 1$, repeat the above process. Since the number of vertices of a tree is finite and T is circuit free, the process must terminate and we shall arrive at a vertex of degree 1.

Remark 8.13 In the proof of the Lemma 8.1, after finding a vertex of degree 1, if we return to v and move along a path outward from v starting with e, we shall reach to a vertex of degree 1 again. Thus it follows that **"Any tree that has more than one vertex has at least two vertices of degree 1".**

Lemma 8.2

There is a unique path between every two vertices in a tree.

Proof. Suppose on the contrary that there are more than one path between any two vertices in a given tree T. Then T has a cycle which contradicts the definition of a tree because T is acyclic. Hence the lemma is proved.

Lemma 8.3

The number of vertices is one more than the number of edges in a tree.

Equivalently, we may state this lemma as

"For any positive integer n, a tree with n vertices has $n-1$ edges".

Proof. We shall prove the lemma by mathematical induction.

Let T be a tree with **one** vertex. Then T has no edges, that is, T has 0 edge. But $0 = 1-1$. Hence the lemma is true for $n = 1$.

Suppose that the lemma is true for $k > 1$. We shall show that it is then true also for $k+1$. Since the lemma is true for k, the tree has k vertices and $k-1$ edges. Let T be a tree with $k+1$ vertices. Since k is +ve, $k+1 \geq 2$ and so T has more than one vertex. Hence, by Lemma 8.1, T has a vertex v of degree 1. Also there is another vertex w and so there is an edge e connecting v and w. Define a subgraph T' of T so that

$$V(T') = V(T) - \{v\},$$

$$E(T') = E(T) - \{e\}.$$

Then number of vertices in $T' = (k+1) - 1 = k$ and since T is circuit free and T' has been obtained on removing one edge and one vertex, it follows that T' is acyclic. Also T' is connected. Hence T' is a tree having k vertices and therefore by induction hypothesis, the number of edges in T' is $k-1$.

But then

$$\text{Number of edges in } T = \text{number of edges in } T' + 1$$

$$= k - 1 + 1 = k.$$

Thus the Lemma is true for tree having $k+1$ vertices. Hence the lemma is true by mathematical induction.

Corollary 8.4

Let $C(G)$ denote the number of components of a graph. Then a forest G on n vertices has $n-C(G)$ edges.

Proof. Apply Lemma 8.3 to each component of the forest G.

Corollary 8.5

Any graph G on n vertices has at least $n-C(G)$ edges.

Proof. If G has cycle-edges, remove them one at a time until the resulting graph G^* is acyclic. Then G^* has $n-C(G^*)$ edges by Corollary 8.4. Since we have removed only circuit, $C(G^*)=C(G)$. Thus G^* has $n-C(G)$ edges. Hence G has at least $n-C(G)$ edges.

Lemma 8.4

A graph in which there is a unique path between every pair of vertices is a tree
(This lemma is converse of Lemma 8.2).

Proof. Since there is a path between every pair of points, therefore the graph is connected. Since a path between every pair of points is unique, there does not exist any circuit because existence of circuit implies existence of distinct paths between pair of vertices. Thus the graph is connected and acyclic and so is a tree.

Lemma 8.5 (Converse of Lemma 8.3)

A connected graph G with $e=v-1$ is a tree

First Proof. The given graph is connected and $e=v-1$. To prove that G is a tree, it is sufficient to show that G is acyclic. Suppose on the contrary that G has a cycle. Let m be the number of vertices in this cycle. Also, we know that **number of edges in a cycle is equal to number of vertices in that cycle**. Therefore number of edges in the present case is m. Since the graph is connected, every vertex of the graph which is not in cycle must be connected to the vertices in the cycle (see Figure 8.154).

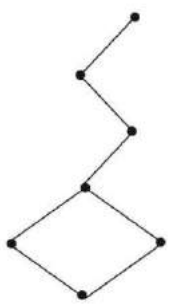


Figure 8.154

Now each edge of the graph that is not in the cycle can connect only one vertex to the vertices in the cycle. There are $v-m$ vertices that are not in the cycle. So the graph must contain at least $v-m$ edges that are not in the cycle. Thus we have $e \geq v-m+m=v$, which is a contradiction to our hypothesis. Hence there is no cycle and so the graph is a tree.

Second Proof. We shall show that a connected graph with v vertices and $v-1$ edges is a tree. It is sufficient to show that G is acyclic. Suppose on the contrary that G is not circuit free and has a nontrivial circuit C. If we remove one edge of C from the graph G, we obtain a graph G' which is connected (see Figure 8.155).

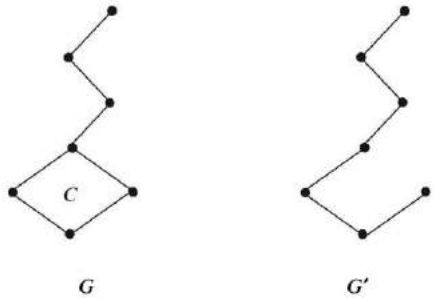


Figure 8.155

If G' still has a nontrivial circuit, we repeat the above process and remove one edge of that circuit obtaining a new connected graph. Continuing this process, we obtain a connected graph G^* which is circuit free. Hence G^* is a tree. Since no vertex has been removed, the tree G^* has v vertices. Therefore, by Lemma 8.3, G^* has $v-1$ edges. But at least one edge of G has been removed to form G^*. This means that G^* has not more than $v-1-1=v-2$ edges. Thus, we arrive at a contradiction. Hence our supposition is wrong and G has no cycle. Therefore G is connected and cycle free and so is a tree.

Lemma 8.6

A graph G with $e=v-1$, that has no circuit is a tree.

Proof. It is sufficient to show that G is connected. Suppose G is not connected and let G', G'', ... be the connected components of G. Since each of G', G'', ... is connected and has no cycle, they all are tree. Therefore, by Lemma 8.3,

$$\begin{aligned} e' &= v'-1, \\ e'' &= v''-1, \\ &\text{———} \\ &\text{———}, \end{aligned}$$

where e', e'', ... are the number of edges and v', v'', ... are the number of vertices in G', G'', ..., respectively. We have, on adding

$$e'+e''+\cdots=(v'-1)+(v''-1)+\cdots$$

Since

$$\begin{aligned} e &= e'+e''+\cdots, \\ v &= v'+v''+\cdots, \end{aligned}$$

we have

$$e<v-1,$$

which contradicts our hypotheses. Hence G is connected. So G is connected and acyclic and is therefore a tree.

EXAMPLE 8.61

Construct a graph that has six vertices and five edges but is not a tree.

Solution.

We have

$$\begin{aligned} \text{Number of vertices} &= 6 \\ \text{Number of edges} &= 5 \end{aligned}$$

So the condition $e=v-1$ is satisfied. Therefore, to construct graph with six vertices and five edges that is not a tree, we should keep in mind that the graph should not be connected. The graph shown in Figure 8.156 has six vertices and five edges but is not connected. Hence it is not a tree.

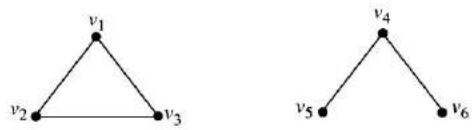


Figure 8.156

Definition 8.89

A directed graph is said to be a **directed tree** if it becomes a tree when the direction of edges are ignored.

For example, the graph shown in Figure 8.157 is a directed tree.

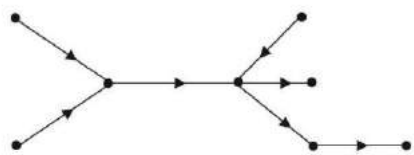


Figure 8.157

Definition 8.90

A directed tree is called a **rooted tree** if there is exactly one vertex whose incoming degree is 0 and the incoming degrees of all other vertices are 1.

The vertex with incoming degree 0 is called the **root** of the rooted tree.

A tree T with root v_0 will be denoted by (T, v_0).

Definition 8.91

In a rooted tree, a vertex, whose outgoing degree is 0 is called a **leaf** or **terminal node**, whereas a vertex whose outgoing degree is non-zero is called a **branch node** or an **internal node**.

Definition 8.92

Let u be a branch node in a rooted tree. Then a vertex v is said to be **child** (**son** or **offspring**) of u if there is an edge from u to v. In this case, u is called **parent** (**father**) of v.

Definition 8.93

Two vertices in a rooted tree are said to be **siblings** (**brothers**) if they are both children of same parent.

Definition 8.94

A vertex v is said to be a **descendent** of a vertex u if there is a unique directed path from u to v.

In this case, u is called the **ancestor** of v.

Definition 8.95

The **level** (or **path length**) of a vertex u in a rooted tree is the number of edges along the unique path between u and the root.

Definition 8.96

The **height** of a rooted tree is the maximum level to any vertex of the tree.

As an example of these terms, consider the rooted tree shown in Figure 8.158:

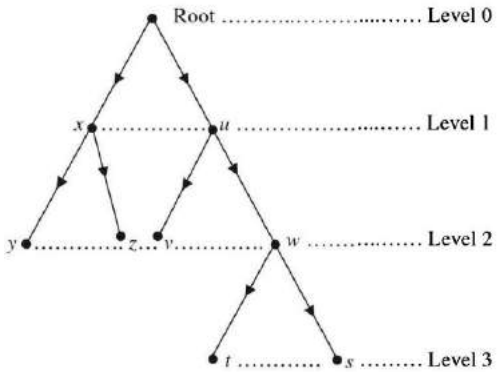


Figure 8.158

Here y is a child of x; x is the parent of y and z. Thus y and z are siblings. The descendents of u are v, w, t and s. Levels of vertices are shown in the figure. The height of this rooted tree is 3.

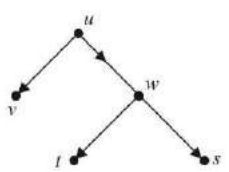
Definition 8.97

Let u be a branch node in the tree $T=(V, E)$. Then the subgraph $T'=(V', E')$ of T such that the vertices set V' contains u and all of its descendents and E' contains all the edges in all directed paths emerging from u is called a **subtree** with u as the root.

Definition 8.98

Let u be a branch node. By a subtree of u, we mean a subtree that has child of u as root.

In the above example, we note that the Figure 8.159 is a subtree of T,



,

Figure 8.159

whereas the Figure 8.160 is a subtree of the branch node u.

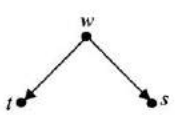


Figure 8.160

EXAMPLE 8.62

Let

$$V=\{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8\}$$

and let

$$E=\{(v_2, v_1), (v_2, v_3), (v_4, v_2), (v_4, v_5), (v_4, v_6), (v_6, v_7), (v_5, v_8)\}.$$

Show that (V, E) is rooted tree. Identify the root of this tree.

Solution.

We note that

Incoming degree of $v_1=1$,
Incoming degree of $v_2=1$,
Incoming degree of $v_3=1$,
Incoming degree of $v_4=0$,
Incoming degree of $v_5=1$,
Incoming degree of $v_6=1$,
Incoming degree of $v_7=1$,
Incoming degree of $v_8=1$.

Since incoming degree of the vertex v_4 is 0, it follows that v_4 is root.

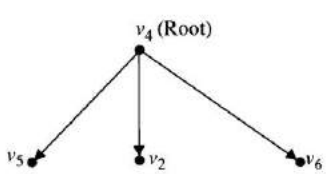
Further,

Outgoing degree of $v_1=0$,
Outgoing degree of $v_3=0$,
Outgoing degree of $v_7=0$,
Outgoing degree of $v_8=0$.

Therefore, v_1, v_3, v_7, v_8 are leaves. Also,

Outgoing degree of $v_2=2$,
Outgoing degree of $v_4=3$,
Outgoing degree of $v_5=1$,
Outgoing degree of $v_6=1$.

Now the root v_4 is connected to v_2, v_5 and v_6. So, we have the following diagram:



Now v_2 is connected to v_1 and v_3, v_5 is connected to v_8, v_6 is connected to v_7. Thus, we have the graph shown in Figure 8.161.

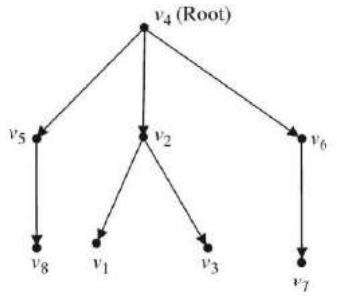


Figure 8.161

We thus have a connected acyclic graph and so (V, E) is a rooted tree with root v_4.

EXAMPLE 8.63

Let F be the set of all female descendent of a lady v_0. We define a relation T on F as follows: if $v_1, v_2, \in F$, then $v_1\ T\ v_2$ if v_1 is mother of v_2. Show that the relation T on F is a rooted tree with root v_0. Is this relation an equivalence relation?

Solution.

The graph of T is cycle free, because descendents cannot be parents of their parents. The graph is also connected. Hence T is rooted tree. We further note that

(i) Since v_1 is not mother of itself, $(v_1, v_1) \notin T$.

(ii) If $(v_1, v_2) \in T$, then v_1 is mother of v_2. But then v_2 cannot be mother of v_1. Hence $(u_1, v_2) \in T$ does not imply $(v_2, v_1) \in T$.

(iii) If $(v_1, v_2) \in T$, $(v_2, v_3) \in T$, then

$$v_1 \text{ is mother of } v_2 \quad \text{and} \quad v_2 \text{ is mother of } v_3.$$

This does not imply that v_1 is mother of v_3, i.e., $(v_1, v_3) \notin T$. Thus T is irreflexive, asymmetric and is not transitive. Hence T is not an equivalence relation.

Consider the rooted trees T and T' shown in the Figure 8.162, where T is family tree of a man who has two sons, with the elder son having no child and younger having three children.

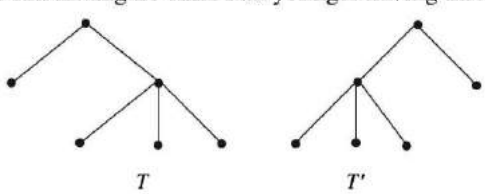


Figure 8.162

Although T'(as a graph) is isomorphic to T, it could be the family tree of another man whose elder son has three children and whose younger son has no child.

The above discussion motivates the following definitions:

Definition 8.99

A rooted tree in which the edges incident from each branch node are labelled with integers 1, 2, 3, … is called an **ordered tree**.

8.13 ISOMORPHISM OF TREES

Definition 8.100

Two ordered trees are said to be **isomorphic** if (i) there exists a one-to-one correspondence between their vertices and edges and that preserves the incident relation (ii) labels of the corresponding edges match.

In view of this definition, the ordered trees shown in Figure 8.163 are not isomorphic.

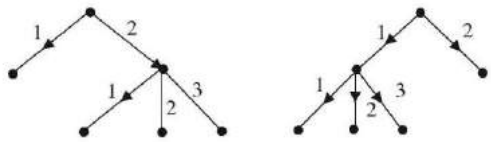


Figure 8.163

EXAMPLE 8.64

Show that the tree T_1 and T_2 shown in the diagram below are isomorphic.

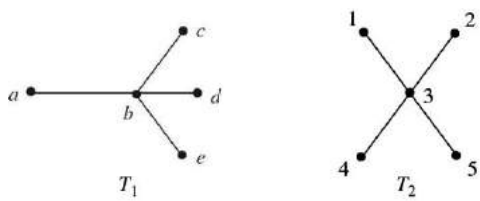


Figure 8.164

Solution.

We observe that in the tree T_1,

$$\deg(b)=4.$$

In the tree T_2,

$$\deg(3)=4.$$

Further, $\deg(a)=\deg(1)=1, \deg(c)=\deg(2), \deg(d)=\deg(4)=\deg(e)=1=\deg(5)$ and $\deg(b)=\deg(3)=4$. Thus we may define a function f from the vertices of T_1 to the vertices of T_2 by

$$f(a)=1, \quad f(b)=3, \quad f(c)=2, \quad f(d)=4, \quad f(e)=5.$$

This is a one-to-one and onto function. Also, adjacency relation is preserved because if v_i and v_j are adjacent vertices in T_1, then $f(v_i)$ and $f(v_j)$ are adjacent vertices in T_2. Hence T_1 is isomorphic to T_2.

EXAMPLE 8.65

Show that the tree T_1 and T_2, shown in the Figure 8.165 are isomorphic

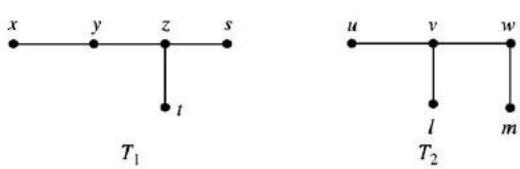


Figure 8.165

Solution.

Let f be a function defined by

$$f(z)=v, \quad f(y)=w, \quad f(x)=m,$$
$$f(s)=u, \quad f(t)=l.$$

Then f is an one-one, onto mapping which preserves adjacency. Hence, T_1 and T_2 are isomorphic.

Remark 8.14 There are three non-isomorphic trees (Figure 8.166) with five vertices:

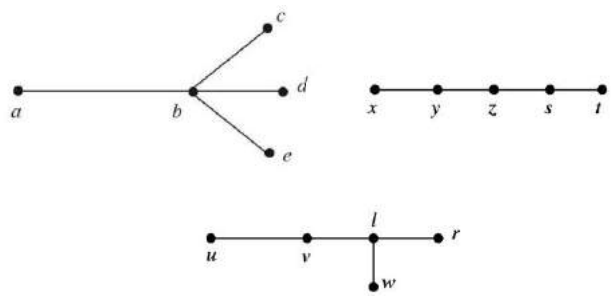


Figure 8.166

Definition 8.101

Let T_1 and T_2 be rooted trees with roots r_1 and r_2, respectively. Then T_1 and T_2 are **isomorphic** if there exists a one-one, onto function f from the vertex set of T_1 to the vertex set of T_2 such that

(i) Vertices v_i and v_j are adjacent in T_1 if and only if the vertices $f(v_i)$ and $f(v_j)$ are adjacent in T_2,

(ii) $f(r_1)=r_2$.

The function f is then called an **isomorphism**.

EXAMPLE 8.66

Show that the trees T_1 and T_2 shown in Figure 8.167 are isomorphic.

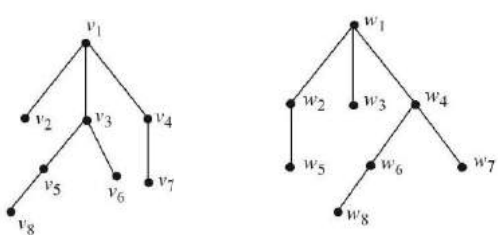


Figure 8.167

Solution.

We observe that T_1 and T_2 are rooted trees.
Define f: (vertex set of T_1) $\rightarrow$ (vertex set of T_2) by

$$f(v_1)=w_1, \quad f(v_2)=w_3, \quad f(v_3)=w_4$$
$$f(v_4)=w_2, \quad f(v_5)=w_6, \quad f(v_6)=w_7$$
$$f(v_7)=w_5, \quad f(v_8)=w_8.$$

Then f is one-to-one and adjacency relation is preserved. Hence f is an isomorphism and so the rooted trees T_1 and T_2 are isomorphic

EXAMPLE 8.67

Show that the rooted trees shown in Figure 8.168 are not isomorphic:

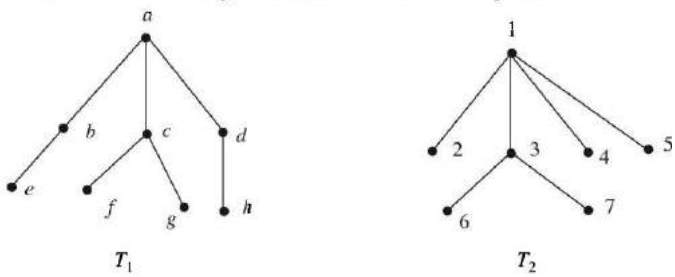


Figure 8.168

Solution.

We observe that the degree of root in T_1 is 3, whereas the degree of root in T_2 is 4. Hence T_1 is not isomorphic to T_2.

Definition 8.102

An ordered tree in which every branch node has at most n offspring is called a **n-ary tree** (or **n-tree**).

Definition 8.103

An n-ary tree is said to be **fully n-ary tree** (**complete n-ary tree** or **regular n-ary tree**) if every branch node has exactly n offspring.

Definition 8.104

An ordered tree in which every branch node has at most two offspring is called a **binary tree** (or **2-tree**).

Definition 8.105

A binary tree in which every branch node (internal vertex) has exactly two offspring is called a **fully binary tree**.

For example, the tree given in Figure 8.169 is a binary tree,

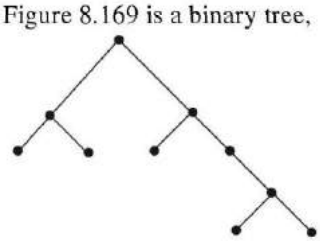


Figure 8.169

whereas the tree shown in Figure 8.170 is a fully binary tree.

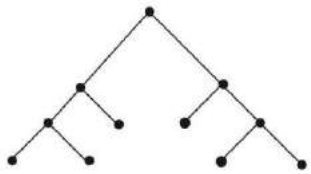


Figure 8.170

Definition 8.106

Let T_1 and T_2 be binary trees with roots r_1 and r_2, respectively. Then T_1 and T_2 are **isomorphic** if there is a one-one, onto function f from the vertex set of T_1 to the vertex set of T_2 satisfying

(i) Vertices v_i and v_j are adjacent in T_1 if and only if the vertices $f(v_i)$ and $f(v_j)$ are adjacent in T_2
(ii) $f(r_1)=r_2$
(iii) v is a left child of w in T_1 if and only if $f(v)$ is a left child of $f(w)$ in T_2
(iv) v is a right child of w in T_1 if and only if $f(v)$ is a right child of $f(w)$ in T_2

The function f is then called an **isomorphism** between binary trees T_1 and T_2.

EXAMPLE 8.68

Show that the trees given in Figure 8.171 are isomorphic.

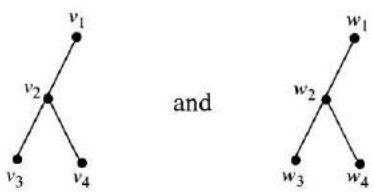


Figure 8.171

Solution.

Define f by $f(v_i)=w_i$, $i=1, 2, 3, 4$. Then f satisfies all the properties for isomorphism. Hence T_1 and T_2 are isomorphic.

EXAMPLE 8.69

Show that the trees given in Figure 8.172 are not isomorphic.

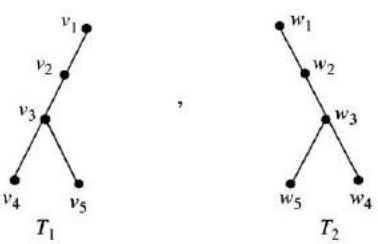


Figure 8.172

Solution.

Since the root v_1 in T_1 has a left child but the root w_1 in T_2 has no left child, the binary trees are not isomorphic.

Definition 8.107

Let v be a branch node of a binary tree T. The **left subtree** of v is the binary tree whose root is the left child of v, whose vertices consists of the left child of v and all its descendents and whose edges consists of all those edges of T that connects the vertices of the left subtree together.

The **right subtree** can be defined analogously.

For example, the left subtree and the right subtree of v in the tree (Figure 8.173)

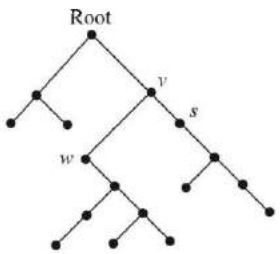


Figure 8.173

are, respectively the thees shown in Figure 8.174.

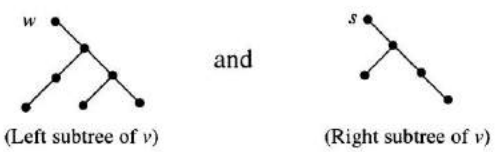


Figure 8.174

8.14 REPRESENTATION OF ALGEBRAIC EXPRESSIONS BY BINARY TREES

Binary trees are used in computer science to represent algebraic expressions involving parentheses. For example, the binary trees (Figure 8.175).

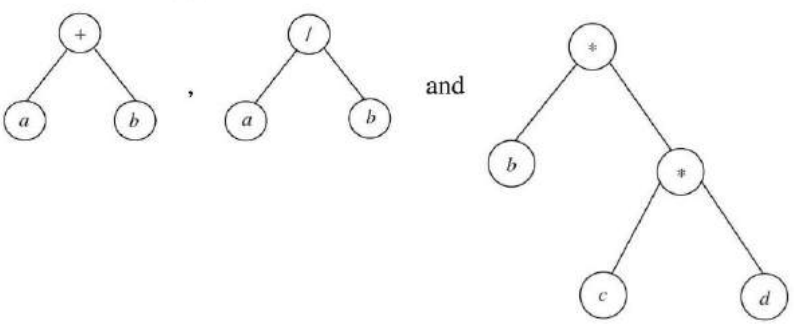


Figure 8.175

represent the expressions, $a+b$, a/b and $b * (c * d)$, respectively.

Thus, the **central operator acts as root of the tree.**

EXAMPLE 8.70

Draw a binary tree to represent

(i) $(2-(3\times x))+((x-3)-(2+x))$

(ii) $ab-(c/(d+e))$.

Solution.

(i) In this expression + is the central operator. Therefore the root of tree is + . The binary tree is shown in Figure 8.176.

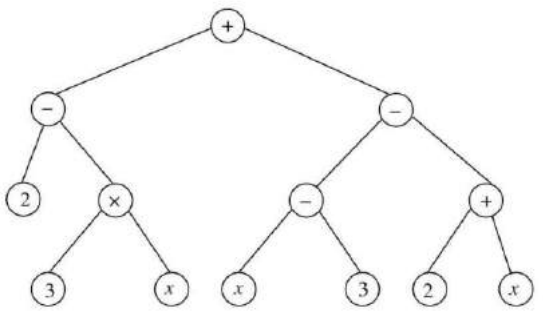


Figure 8.176

(ii) Here the central operator is −. Therefore it is the root of the tree. We have the following binary tree (Figure 8.177) to represent this expression:

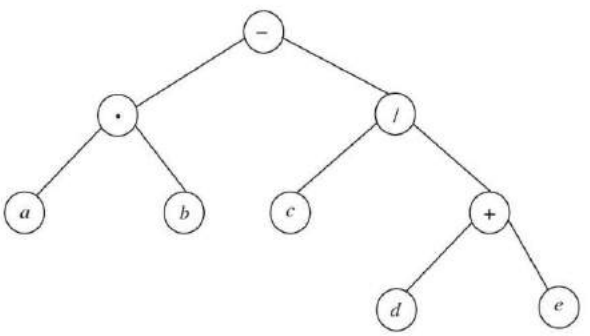


Figure 8.177

EXAMPLE 8.71

Consider a single elimination tournament with eight players in a wrestling game. How many rounds will be there to declare the champion and how many games are to be played?

Solution.

In this example, the leaves of the tree shown in Figure 8.178 will represent player, the branch nodes will represent winners in the first round and the second round. Then the tree showing the tournament shall be as given below: The graph of single elimination tournament is a full binary tree. The champion shall be at the root.

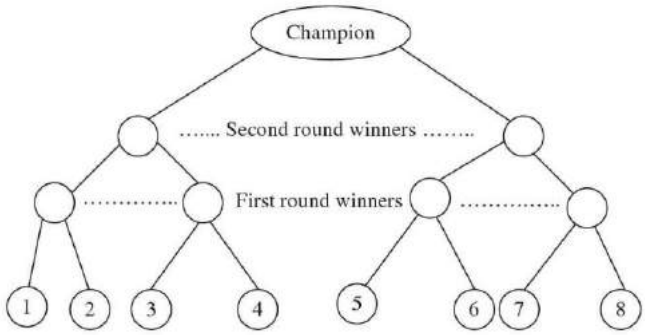


Figure 8.178

We note that there shall be three rounds to declare the champion. Let

i=Number of games played=number of internal vertices

p=Number of leaves=number of players.

Then, we see that

$$i=p-1=7.$$

In general, for complete n-ary tree, we will have

$$(n-1)\,i=p-1.$$

To derive this formula we first prove the following result:

Theorem 8.23

If T is a full binary tree with i internal vertices, then T has $i+1$ terminal vertices (leaves) and $2i+1$ total vertices.

Proof. The vertices of T consist of the vertices that are children (of some parent) and the vertices that are not children (of any parent). There is a non-child (the root). Since there are i internal vertices, each having two children, there are $2i$ children. Thus the total number of vertices of T is $2i+1$ and the number of terminal vertices is

$$(2i+1)-i=i+1.$$

This completes the proof.

In the context of above example, we have

$$\text{Number of leaves}=p=i+1$$
$$\Rightarrow i=p-1.$$

Remark 8.15 In case of full n-ary tree, if i denotes the number of branch nodes, then total number of vertices of T is $ni+1$ and the number of terminal vertices is

$$ni+1-i=i(n-1)+1.$$

If p is the number of terminal vertices, then

$$p=i(n-1)+1,$$

which yields

$$(n-1)\,i=p-1.$$

EXAMPLE 8.72

Find the minimum number of extension cords, each having four outlets, required to connect 22 bulbs to a single electric outlet.

Solution.

Clearly, the graph of the problem is a regular quaternary tree with 22 leaves.
Let i denote the internal vertices and p denote the number of leaves, then using the expression $(n-1)i=p-1$, we have

$$(4-1)i = 22-1 \quad \text{or} \quad i = \frac{21}{3} = 7.$$

Thus seven extension cords, as shown in Figure 8.179, are required.

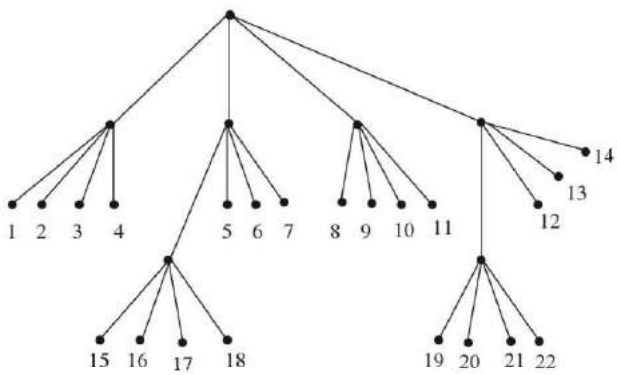


Figure 8.179

EXAMPLE 8.73

A computing machine has been given instruction which computes the sum of three numbers. How many times the addition instruction will be executed to perform the sum of 11 numbers?

Solution.

In this problem, we will have a regular ternary tree with 11 leaves. Thus $n=3$, $p=11$ and so the relation $(n-1)i=p-1$ yields

$$i = \frac{p-1}{n-1} = \frac{10}{2} = 5.$$

Theorem 8.24

If a binary tree of height h has p leaves, then

$$\log_2 p \le h \quad \text{or} \quad p \le 2^h.$$

Proof. It is sufficient to prove that

$$p \le 2^h, \tag{1}$$

because then taking logarithm to the base 2 of both sides, we will get

$$\log_2 p \le h.$$

We shall prove the inequality (1) by induction on h.

If $h=0$, then the binary tree consists of a single vertex and $p=1$ and so (1) is satisfied.

Now suppose that (1) holds for a binary tree whose height is less than h. We shall prove that it holds for a tree of height h. So let T be a tree with height $h>0$ with p leaves. We consider the following two cases:

(a) Suppose that the root of T has only one child (Figure 8.180).

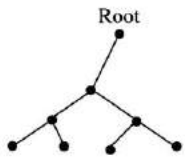


Figure 8.180

Eliminating the root and the edge incident on the root, we get a tree T' of height $h-1$ in which number of leaves is same as in T. Using induction hypothesis on T', we get $p \le 2^{h-1}$. Since $2^{h-1} < 2^h$, we have $p \le 2^h$ and the inequality holds in this case.

(b) Suppose the root of T has offspring's v and w (Figure 8.181).

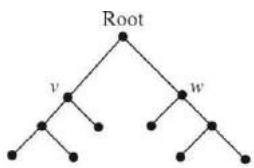


Figure 8.181

Let T_1 be the subtree rooted at v whose height is h_1 and has p_1 leaves. Similarly, let T_2 be the subtree rooted at w whose height is h_2 and has p_2 leaves. Using mathematical induction, to these trees, we have $p_1 \le 2^{h_1}$ and $p_2 \le 2^{h_2}$. But the leaves of T consist of the leaves of T_1 and T_2. Hence,

$$p = p_1 + p_2 \le 2^{h_1} + 2^{h_2}$$

$$\le 2^{h-1} + 2^{h-1} = 2^{h-1}(1+1) = 2^h.$$

Hence, by mathematical induction, the result holds.

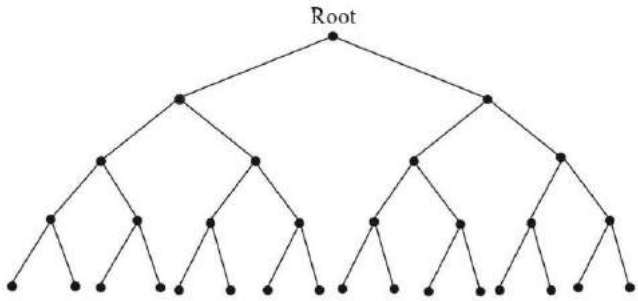
It can be proved on the same lines that an **n-ary tree of height h has atmost n^h leaves.**

EXAMPLE 8.74

Find the number of terminal vertices in a regular binary tree of height 4.

Solution.

Let p be the number of terminal vertices. We are given that height (h) of the tree is 4. So, we must have $p \le 2^4 = 16$. Since the tree is regular, we have $p = 16$.



(A regular binary tree of height 4 with 16 leaves)

Figure 8.182

EXAMPLE 8.75

Does there exist a full binary tree with 12 internal vertices and 15 leaves?

Solution.

We know that if i is the number of branch nodes in a full binary tree, then the number of leaves is $i+1$. Therefore for a tree with 12 branch nodes, the number of leaves should be 13 and not 15. Hence such tree does not exist.

EXAMPLE 8.76

Is there a binary tree with height 6 and 65 leaves?

Solution.

No, since

$$p \leq 2^h = 2^6 = 64.$$

We now state some results without proof on n-ary tree.

Theorem 8.25

The number b_n of different binary trees on n vertices is given by

$$b_n = \frac{1}{n+1}\binom{2n}{n}.$$

Theorem 8.26

A regular n-ary tree of height h has **at least** $n+(n-1)(h-1)$ leaves.

Theorem 8.27

If in a rooted regular n-ary tree,

I = Sum of the path length of all branch nodes

E = Sum of the path length of all leaves

i = Number of branch nodes

then

$$E=(n-1)I+n\,i$$

Thus, in particular, for a full binary tree,

$$E=I+2i.$$

EXAMPLE 8.77

How many binary trees are possible on three vertices?

Solution.

We know that the number of different trees on n vertices is given by

$$b_n=\frac{1}{n+1}\binom{2n}{n}.$$

So we have

$$b_n=\frac{1}{3+1}\binom{6}{3}=\frac{1}{4}\left(\frac{6\times5\times4\times3\times2\times1}{3\times2\times1\times3\times2\times1}\right)=5\,\text{trees}$$

and those are shown in Figure 8.183.

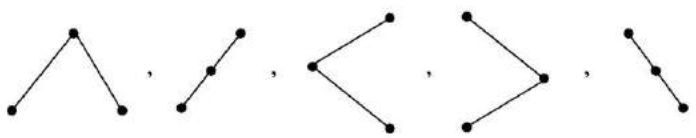


Figure 8.183

8.15 SPANNING TREE OF A GRAPH

Definition 8.108

Let G be a graph, then a subgraph of G which is a tree is called **tree of the graph**.

Definition 8.109

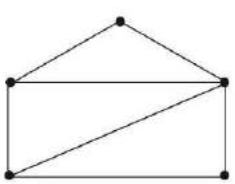
A **spanning tree** for a graph G is a subgraph of G that contains every vertex of G and is a tree.

Thus,

"A **spanning tree** for a graph G is a spanning subgraph of G which is a tree".

EXAMPLE 8.78

Determine a tree and a spanning tree for the connected graph given below (Figure 8.184):



G

Figure 8.184

Solution.

The given graph G contains circuits and we know that removal of the circuits gives a tree. So, we note that the Figure 8.185 is a tree.

Figure 8.185

And the Figure 8.186 is a spanning tree of the graph G.

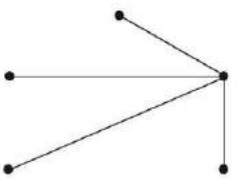


Figure 8.186

EXAMPLE 8.79

Find all spanning trees for the graph G shown below (Figure 8.187).

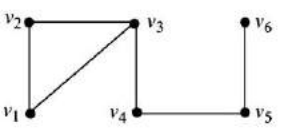


Figure 8.187

Solution.

The given graph G has a circuit $v_1\, v_2\, v_3\, v_1$. We know that removal of any edge of the circuit gives a tree. So the spanning trees of G are shown in Figure 8.188.

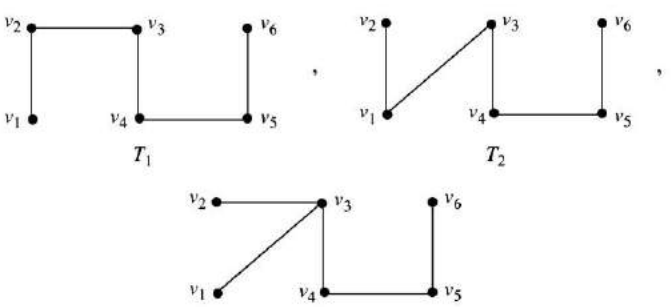


Figure 8.188

Remark 8.16 We know that a tree with n vertices has exactly $n-1$ edges. Therefore if G is a connected graph with n vertices and m edges, a spanning tree of G must have $n-1$ edges. Hence the number of edges that must be removed before a spanning tree is obtained must be

$$m-(n-1)=m-n+1.$$

For illustration, in Example 8.79, $n=6$, $m=6$, so, we had to remove one edge to obtain a spanning tree.

Definition 8.110

A **branch** of a tree is an edge of the graph that is in the tree.

Definition 8.111

A **chord** (or a **link**) of a tree is an edge of the graph that is not in the tree.

It follows from the above remark that the number of chords in a tree is equal to $m-n+1$, where n is the number of vertices and m is the number of edges in the graph related to the tree.

Definition 8.112

The set of the chords of a tree is called the **complement of the tree**.

EXAMPLE 8.80

Consider the graph discussed in the above example. We note that the edge (v_2, v_3) is a branch of the tree T_1, whereas (v_1, v_3) is a chord of the tree T_1.

Theorem 8.28

A graph G has a spanning tree if and only if G is connected.

Proof. Suppose first that a graph G has a spanning tree T. If v and w are vertices of G, then they are also vertices in T and since T is a tree there is a path from v to w in T. This path is also a path in G. Thus every two vertices are connected in G. Hence G is connected.

Conversely, suppose that G is connected. If G is acyclic, then G is its own spanning tree and we are done. So suppose that G contains a cycle C_1. If we remove an edge from the cycle, the subgraph of G so obtained is also connected. If it is acyclic, then it is a spanning tree and we are done. If not, it has at least one circuit, say C_2. Removing one edge from C_2, we get a subgraph of G which is connected. Continuing in this way, we obtain a connected circuit free subgraph T of G. Since T contains all vertices of G, it is a spanning tree of G.

Theorem 8.29 (Cayley's Formula)

The number of spanning trees of the complete graph K_n, $n \geq 2$ is n^{n-2}.

(Proof of this formula is out of scope of this book)

EXAMPLE 8.81

Find all the spanning trees of K_4 (Figure 8.189).

Solution.

According to Cayley's formula, K_4 has $4^{4-2}=4^2=16$ different spanning trees.

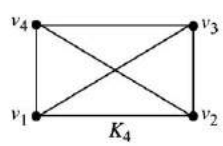


Figure 8.189

Here $n=4$, so the number of edges in any tree should be $n-1=4-1=3$. But here number of edges is equal to 6. So to get a tree, we have to remove three edges of K_4. The 16 spanning trees so obtained are shown in Figure 8.190.

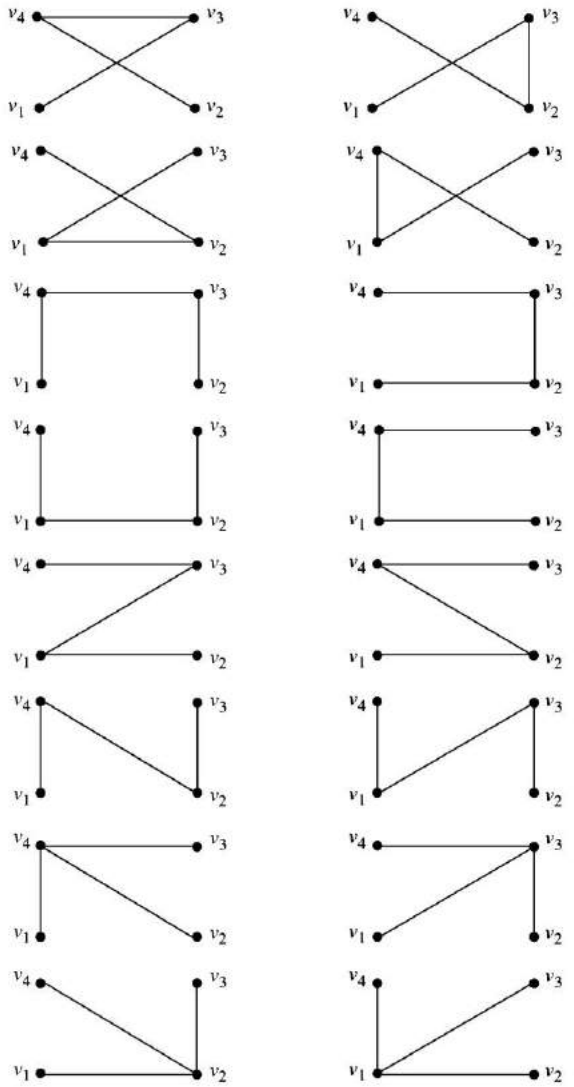


Figure 8.190

8.16 SHORTEST PATH PROBLEM

Let s and t be two vertices of a connected weighted graph G. Shortest path problem is to find a **path from s to t whose total edge weight is minimum.** We now discuss algorithm due to $E \cdot W$. Dijkstra which efficiently solves the shortest path problem. The idea is to grow a Disjkstra tree, starting at the vertex s, by adding, at each iteration, a frontier edge, whose non-tree endpoint is as close as possible to s. The algorithm involves assigning labels to vertices.

For each tree vertex x, let dist $[x]$ denote the distance from vertex s to x and for each edge e in the given weighted graph G, let $w(e)$ be its edge-weight.

After each iteration, the vertices in the Dijkstra tree (the labelled vertices) are those to which the shortest paths from s have been found.

Priority of the Frontier Edges: Let e be a frontier edge and let its P value be given by

$$P(e) = \text{dist } [x] + w(e),$$

where x is the labelled endpoint of e and $w(e)$ is the edge-weight of e. Then,

(i) The edge with the smallest P value is given the **highest priority**.

(ii) The P value of this highest priority edge e gives the distant from the vertex s to the unlabelled endpoint of e.

We are now in a position to describe Dijkstra's shortest path algorithm.

8.16.1 Dijkstra's Shortest Path Algorithm

Input: A connected weighted graph G with non-negative edge-weights and a vertex s of G.

Output: A spanning tree T of G, rooted at the vertex s, whose path from s to each vertex v is a shortest path from s to v in G and a vertex labelling giving the distance from s to each vertex.

Initialize the Dijkstra tree T as vertex s.
Initialize the set of frontier edges for the tree T as empty.

$$\text{dist: } [s] = 0.$$

Write label 0 on vertex s.
While Dijkstra tree T does not yet span G.
For each frontier edge e for T,
Let x be the labelled endpoint of edge e.
Let y be the unlabelled endpoint of edge e.
Set $P(e) = \text{dist } [x] + w(e)$
Let e be a frontier edge for T that has smallest P value
Let x be the labelled endpoint of edge e
Let y be the unlabelled endpoint of edge e
Add edge e (and vertex y) to tree T

$$\text{dist } [y]: P(e)$$

Write label dist $[y]$ on vertex y.

Return Dijkstra tree T and its vertex labels.

EXAMPLE 8.82

Apply Dijkstra algorithm to find shortest path from s to each other vertex in the graph given in Figure 8.191.

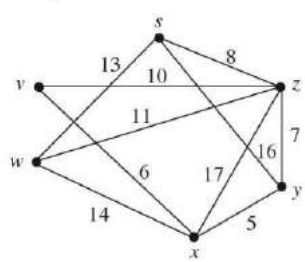


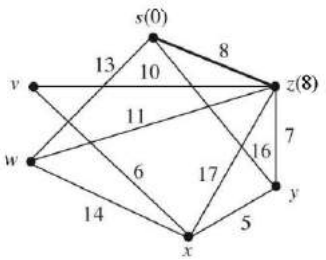
Figure 8.191

If t is the labelled endpoint of edge e, then P values are given by

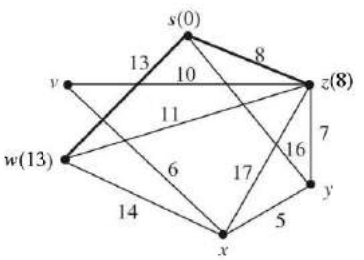
$$P(e) = \text{dist } [t] + w(e),$$

where dist $[t]$ = distance from s to t and $w(e)$ is the edge weight of edge e. For each vertex v, dist $[v]$ appears in the parenthesis. Iteration tree at the end of each iteration is **drawn in dark line**

Iteration 1



Iteration 2



dist $[s]=0$ $\quad P(sw)=13$ (minimum)

dist $[z]=8$ $\quad P(zy)=8+7=15$

$P(sy)=16$

$P(zv)=8+10=18$

$P(zw)=8+11=19$

$P(zx)=8+17=25$

dist $[s]=0$ $\quad P(zy)=8+7=15$ (minimum)

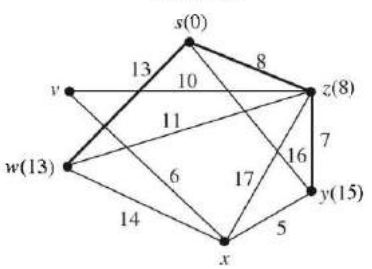
dist $[z]=8$ $\quad P(zx)=8+17=25$

dist $[w]=13$ $\quad P(zv)=8+10=18$

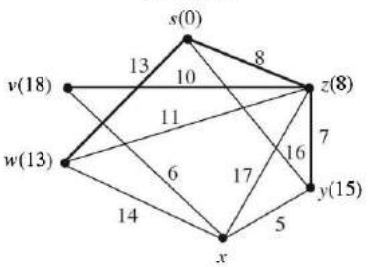
$P(sy)=16$

$P(wx)=13+14=27$

Iteration 3



Iteration 4



dist $[s]=0$ $\quad P(zv)=18$ (minimum)

dist $[z]=8$ $\quad P(zx)=8+17=25$

dist $[w]=13$ $\quad P(wx)=13+14=27$

dist $[y]=15$ $\quad P(yx)=15+5=20$

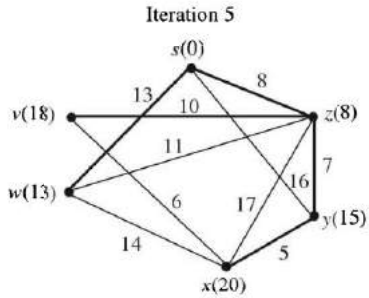
Dist $[s]=0$ $\quad P(yx)=20$ (minimum)

Dist $[z]=8$ $\quad P(zx)=8+17=25$

Dist $[w]=13$ $\quad P(vx)=18+6=24$

Dist $[y]=15$ $\quad P(wx)=13+14=27$

Dist $[v]=18$



dist $[s]=0$

dist $[z]=8$

dist $[w]=13$

dist $[y]=15$

dist $[v]=18$

dist $[x]=20$,

which are the required shortest paths from s to any other point. The Dijkstra tree is shown in dark lines in Figure 8.192.

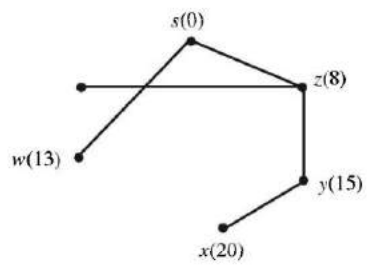


Figure 8.192

EXAMPLE 8.83

Apply Dijkstra algorithm to find the shortest path from s to t in the graph given below (Figure 8.193).

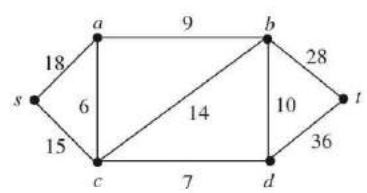


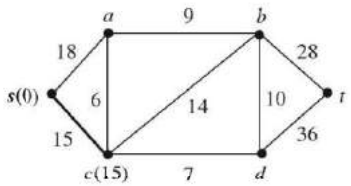
Figure 8.193

Solution.

Let x be the labelled endpoint of the edge e, then P values are given by $P(e)=\text{dist}\,[x]+w(e)$, where dist $[x]$ denotes the distance of x from s and $w(e)$ is the edge weight of e.

For each vertex v, dist $[v]$ appears in the bracket. Iteration tree at the end of each iteration is shown by dark lines:

Iteration 1



dist $[s]=0$ $P(s\ a)=18$ (minimum)

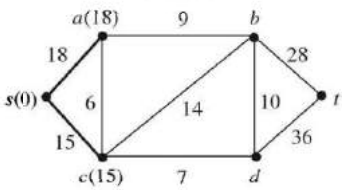
dist $[c]=15$ $P(c\ a)=15+6=21$

$P(c\ b)=15+14=29$

$P(c\ d)=15+7=22$

$P(c\ t)=\infty$

Iteration 2



dist $[s]=0$ $P(a\ b)=18+9=27$

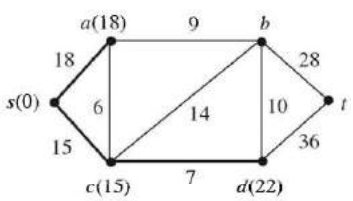
dist $[c]=15$ $P(c\ b)=15+14=29$

dist $[a]=18$ $P(c\ d)=15+7=22$ (minimum)

$P(c\ t)=\infty$

$P(a\ t)=\infty$

Iteration 3



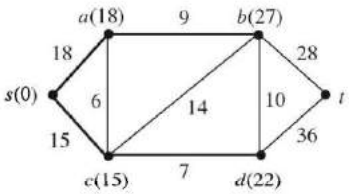
dist $[s]=0$ $P(c\ b)=15+14=29$

dist $[c]=15$ $P(a\ b)=18+9=27$ (minimum)

dist $[a]=18$ $P(d\ b)=22+10=32$

dist $[d]=22$ $P(d\ t)=22+36=58$

Iteration 4



dist $[s]=0$ $P(d\ t)=22+36=58$

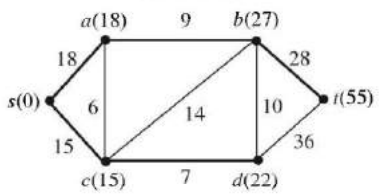
dist $[c]=15$ $P(b\ t)=27+28=55$ (minimum)

dist $[a]=18$

dist $[d]=22$

dist $[b]=27$

Iteration 5



dist $[s]=0$

dist $[c]=15$

dist $[a]=18$

dist $[s]=0$

dist $[c]=15$

dist $[a]=18$

dist $[d]=22$

dist $[b]=27$

dist $[t]=55$

Hence, **the length of the shortest path** (s, a, b, t) **from** s **to** t **is 55** and the Dijkstra's tree is shown in Figure 8.194.

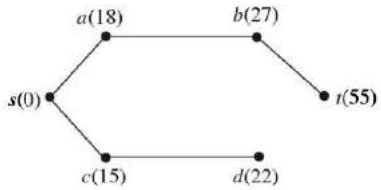


Figure 8.194

EXAMPLE 8.84

Find a shortest path from s to t and its length for the graph given below:

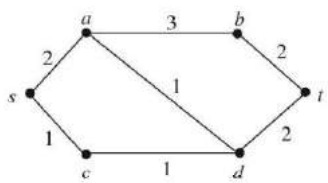


Figure 8.195

Solution.

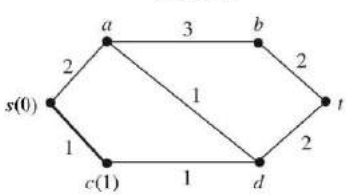
Let x be the labelled endpoint of edge e, then P values are given by

$$P(c) = \text{dist}\,[x] + w(e),$$

where dist $[x]$ denotes the distance from s to x and $w(e)$ is the weight of the edge e.

For each vertex v, dist $[v]$ appears in the bracket. Iteration tree at the end of each iteration is shown in dark lines.

Iteration 1



dist $[s]=0$ $\quad P(c\ d)=2$

dist $[c]=1$ $\quad P(s\ a)=2$ (minimum)

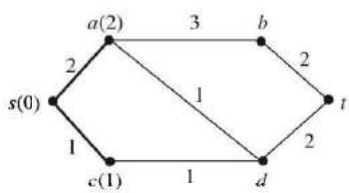
$P(s\ b)=\infty$

$P(s\ t)=\infty$

$P(d\ t)=\infty$

$P(b\ t)=\infty$

Iteration 2



dist $[s]=0$ $\quad P(a\ b)=2+3=5$

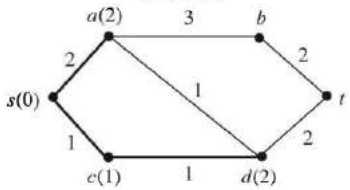
dist $[c]=1$ $\quad P(c\ d)=1+1=2$ (minimum)

dist $[a]=2$ $\quad P(d\ t)=\infty$

$P(b\ t)=\infty$

$P(a\ d)=3$

Iteration 3



dist $[s]=0$ $P(d\ t)=2$ (minimum)

dist $[c]=1$ $P(a\ b)=5$

dist $[a]=2$ $P(b\ t)=\infty$

dist $[d]=2$

dist $[s]=0$

dist $[c]=1$

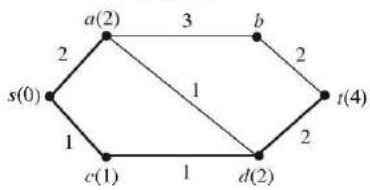
dist $[a]=2$

dist $[d]=2$

dist $[t]=4$

dist $[b]=5$

Iteration 4



dist $[s]=0$ $P(a\ b)=5$ (minimum)

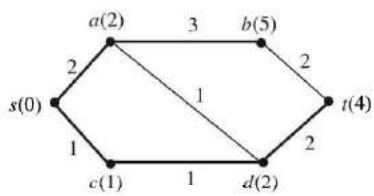
dist $[c]=1$ $P(b\ t)=\infty$

dist $[a]=2$

dist $[d]=2$

dist $[t]=4$

Iteration 5



Thus, the Dijkstra tree is

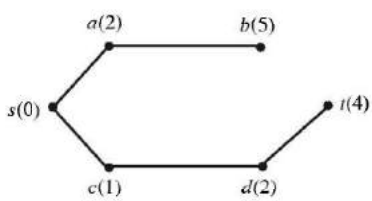


Figure 8.196

Thus the shortest path is *scdt* and its length is 4.

8.16.2 Shortest Path if All Edges Have Length 1

If all edges in a connected graph G have length 1, then a **shortest path** $v_1 \to v_k$ is the path that has the smallest number of edges among all paths $v_1 \to v_k$ in the given graph G.

Moore's Breadth First Search Algorithm

This method of finding shortest path in a connected graph G from a vertex s to a vertex t is used when all edges have length 1.

Input: Connected graph $G=(V, E)$, in which one vertex is denoted by s and one by t and each edge (v_i, v_j) has length 1.

Initially all vertices are unlabelled.

Output: A shortest path $s \to t$ in $G=(V, E)$.

1. Label s with 0.
2. Set $v_i=0$.
3. Find all unlabelled vertices **adjacent to a vertex labelled** v_i.
4. Label the vertices just found with v_{i+1}.
5. If vertex t is labelled, then "**back tracking**" gives the shortest path. If k is label of t(i.e., $t=v_k$), then

Output: $v_k, v_{k-1}, \ldots, v_1, 0$.

Else increase i by 1. Go to Step 3.
End Moore.

Remark 8.17 **There could be several shortest paths from s to t.**

EXAMPLE 8.85

Use BFS algorithm to find shortest path from s to t in the connected graph G given in Figure 8.197.

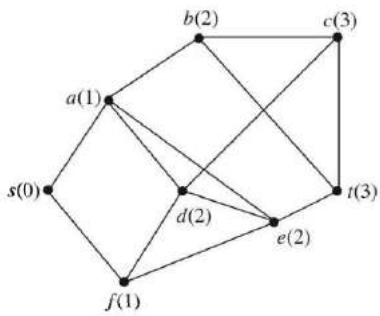


Figure 8.197

Solution.

Label s with 0 and then label the adjacent vertices with 1. Thus two vertices have been labelled by 1. Now label the adjacent vertices of all vertices labelled by 1 with label 2. Thus three vertices have been labelled with 2. Label the vertices adjacent to these vertices (labelled by 2) with 3. Thus two vertices have been labelled with 3. We have reached t. Now backtracking yields the following three shortest paths:

(i) $t(3), e(2), f(1), s(0)$, that is, $s\ f\ e\ t$,
(ii) $t(3), b(2), a(1), s(0)$, that is $s\ a\ b\ t$,
(iii) $t(3), e(2), a(1), s(0)$, that is, $s\ a\ e\ t$.

Thus there are three possible shortest paths of length 3.

EXAMPLE 8.86

Find a shortest path $s \to t$ in the graph given in Figure 8.198 (using Moore's BFS algorithm).

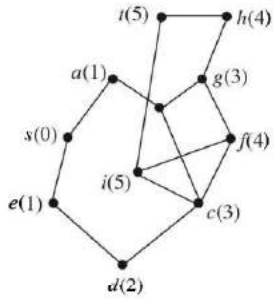


Figure 8.198

Solution.

Label s with 0 and then label the adjacent vertices with 1. Thus two vertices have been marked as 1. Label the adjacent vertices of al vertices labelled as 1 with 2. Thus two vertices have been marked as 2. Now label the adjacent vertices of all vertices labelled as 2 with 3. Thus two vertices have been marked as 3. Label the adjacent vertices of all vertices labelled as 3 with 4. Thus three vertices have been marked as 4. Now there is only one vertex t to be labelled. Label it with 5. We have reached at t.

Now backtracking gives the following shortest paths

$t\,h\,g\,b\,a\,s$	or	$s\,a\,b\,g\,h\,t$	with length 5,
$t\,i\,c\,d\,e\,s$	or	$s\,e\,b\,c\,i\,t$	of length 5,
$t\,i\,c\,b\,a\,s$	or	$s\,a\,b\,c\,i\,t$	of length 5.

Thus, there are three possible shortest paths of length 5.

8.17 MINIMAL SPANNING TREE

Definition 8.113

Let G be a weighted graph. A spanning tree of G with minimum weight is called **minimal spanning tree of** G.

We discuss two algorithms to find a minimal spanning tree for a weighted graph G.

8.17.1 Prim Algorithm

Prim algorithm builds a minimal spanning tree T by expanding outward in connected links from some vertex. In this algorithm, one edge and one vertex are added at each step. The edge added is the one of least weight that connects the vertices already in T with those not in T.

Input: A connected weighted graph G with n vertices.

Output: The set of edges E in a minimal spanning tree.

1. Choose a vertex v_1 of G. Let $V=\{v_1\}$ and $E=\{\ \}$.
2. Choose a nearest neighbour v_i of V that is adjacent to v_j, $v_j \in V$ and for which the edge (v_i, v_j) does not form a cycle with members of E. Add v_i to V and add (v_i, v_j) to E.
3. Repeat Step 2 till number of edges in T is $n-1$. Then V contains all n vertices of G and E contains the edges of a minimal spanning tree for G.

Definition 8.114

A **greedy algorithm** is an algorithm that optimizes the choice at each iteration without regard to previous choices.

For example, **Prim algorithm is a greedy algorithm.**

EXAMPLE 8.87

Find a minimal spanning tree for the graph shown in Figure 8.199.

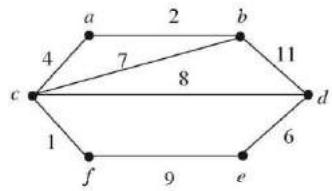


Figure 8.199

Solution.

We shall use **Prim algorithm to find the required minimal spanning tree.** We note that number of vertices in this connected weighted graph is 6. Therefore the tree will have five edges.

We start with any vertex, say c. The nearest neighbour of c is f and (c, f) does not form a cycle. Therefore (c, f) is the first edge selected.

Now we consider the set of vertices $V=\{c, f\}$. The vertex a is nearest neighbour to $V=\{c, f\}$ and the edge (c, a) does not form a cycle with the member of set of edges selected so far. Thus,

$$E=\{(c, f), (c, a)\} \quad \text{and} \quad V=\{c, f, a\}.$$

The vertex b is now nearest neighbour to $V=\{c, f, a\}$ and the edge (a, b) do not form a cycle with the member of $E=\{(c, f), (c, a)\}$. Thus,

$$E=\{(c, f), (c, a), (a, b)\} \quad \text{and} \quad V=\{c, f, a, b\}.$$

Now the edge (b, c) cannot be selected because it forms a cycle with the members of E. We note that d is the nearest point to $V=\{c, f, a, b\}$ and (c, d) is the edge which does not form a cycle with members of $E=\{(c, f), (c, a), (a, b)\}$. Thus we get

$$E=\{(c, f), (c, a), (a, b)\}, (c, d)\}, \quad V=\{c, f, a, b, d\}.$$

The nearest vertex to V is now e and (d, e) is the corresponding edge. Thus,

$$E=\{(c, f), (c, a), (a, b), (c, d), (d, e)\}, \quad V=\{c, f, a, b, d, e\}$$

Since number of edges in the Prim tree is 5, the process is complete. The minimal spanning tree is shown in Figure 8.200:

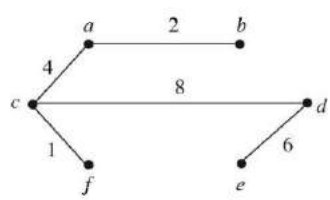


Figure 8.200

The length of the tree is $1+4+2+8+6=21$

EXAMPLE 8.88

Build the lowest-cost road system that will connect the cities in the graph given in Figure 8.201.

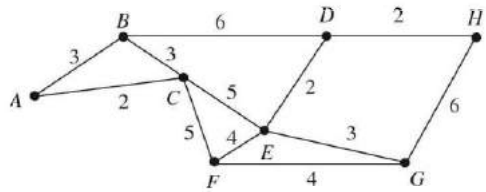


Figure 8.201

Solution.

We shall use Prim algorithm to build this system. Here the number of vertices (Cities) is 8. So, the minimal spanning tree would have seven edges.

We start from any vertex, say A and have at the initial stage:

$$V=\{A\}, \quad E=\{\ \}.$$

The nearest neighbour of $V=\{A\}$ is C and the corresponding edge is (A, C). So we have

$$E=\{(A, C)\}, \quad V=\{A, C\}.$$

Now B is the nearest neighbour of $V=\{A, C\}$ and edge (C, B) does not form cycle with the edges in $E=\{(A, C)\}$. Here the edge (A, B) can also be chosen. Thus, choosing arbitrarily one out of (A, B) and (C, B), we have

$$E=\{(A, C), (C, B)\}, \quad V=\{A, C, B\}.$$

Now, E and F are both nearest neighbour to V and (C, E) or (C, F) do not form cycle with the members of $E=\{(A, C), (C, B)\}$. So we may choose any of E or F. Let us choose F and the corresponding edge (C, F). Then,

$$E=\{(A, C), (C, B), (C, F)\}, \quad V=\{A, C, B, F\}.$$

Now E and G are nearest neighbours of $V=\{A, C, B, F\}$ and both (F, E) and (F, G) do not form cycle with the members of $E=\{(A, C), (C, B), (C, F)\}$. Therefore we may pick up any of these vertices. We pick arbitrarily E.

Then,

$$E=\{(A, C), (C, B), (C, F), (F, E)\}, \quad V=\{A, C, B, F, E\}.$$

Now D is the nearest neighbour of $V=\{A, C, B, F, E\}$ and (E, D) is the corresponding edge to be added to the tree. So, we have

$$E=\{(A, C), (C, B), (C, F), (F, E), (E, D)\}, \quad V=\{A, C, B, F, E, D\}.$$

Now H is the nearest neighbour and (D, H) is the corresponding edge and we have

$$E=\{(A, C), (C, B), (C, F), (F, E), (E, D), (D, H)\},$$
$$V=\{A, C, B, F, E, D, H\}.$$

Now G is the nearest neighbour and (E, G) is the corresponding edge. So, we have

$$E=\{(A, C), (C, B), (C, F), (F, E), (E, D), (D, H), (E, G)\},$$
$$V=\{A, C, B, F, E, D, H, G\}$$

Since the number of edges in the Prim tree is 7, the process is complete. The minimal spanning tree is shown in Figure 8.202:

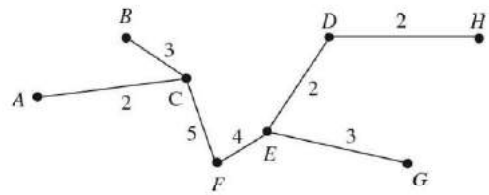


Figure 8.202

The total length of the roads is

$$2+3+5+4+2+2+3=21.$$

EXAMPLE 8.89

Using Prim algorithm, find the minimal spanning tree of the following graph (Figure 8.203):

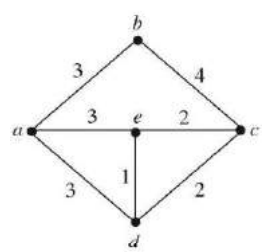


Figure 8.203

Solution.

Pick up the vertex a. Then

$$E=\{\ \}\quad\text{and}\quad V=\{a\}.$$

The nearest neighbour of V is b, d or e and the corresponding edges are (a, b) or (a, d) or (a, e). We choose arbitrarily (a, b) and have

$$E=\{(a, b)\},\quad V=\{a, b\}.$$

Now d is the nearest neighbour of $V=\{a, b\}$ and the corresponding edge (a, d) does not form cycle with (a, b). Thus we get

$$E=\{(a, b), (a, d)\},\quad V=\{a, b, d\}.$$

Now e is the nearest neighbour of $\{a, b, d\}$ and (d, e) does not form cycle with $\{(a, b), (a, d)\}$.

Hence,

$$E=\{(a, b), (a, d), (d, e)\},\quad V=\{a, b, d, e\}.$$

Now c is the nearest neighbour of $V=\{a, b, d, e\}$ and the corresponding edges are (e, c), (d, c). Thus we have, choosing (e, c),

$$E=\{(a, b), (a, d), (d, e), (e, c)\},\quad V=\{a, b, d, e, c\}$$

Total weight$=3+3+1+2=9$.

(If we choose (d, c), then total weight is $3+3+1+2=9$).

The minimal tree is shown in Figure 8.204.

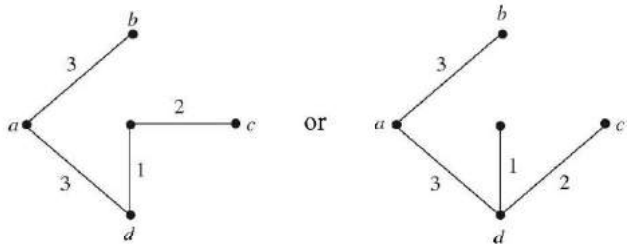


Figure 8.204

8.17.2 Kruskal's Algorithm

In Kruskal's algorithm, the edges of a weighted graph are examined one by one in order of increasing weight. At each stage, an edge with least weight out of edge-set remaining at that stage is added provided this additional edge does not create a circuit with the members of existing edge set at that stage. After $n-1$ edges have been added, these edges together with the n vertices of the connected weighted graph form a minimal tree.

Algorithm

Input: A connected weighted graph G with n vertices and the set $E=\{e_1, e_2, \ldots, e_k\}$ of weighted edges of G.

Output: The set of edges in a minimal spanning tree T for G.

Step 1. Initialize T to have all vertices of G and no edges.

Step 2. Choose an edge e_1 in E of least weight. Let

$$E^*=\{e_1\}, \quad E=E-\{e_1\}.$$

Step 3. Select an edge e_i in E of least weight that does not form circuit with members of E^*. Replace

$$E^* \text{ by } E^* \cup \{e_i\} \quad \text{and} \quad E \text{ with } E-\{e_i\}.$$

Step 4. Repeat Step 3 until number of edges in E^* is equal to $n-1$.

EXAMPLE 8.90

Use Kruskal's algorithm to determine a minimal spanning tree for the connected weighted graph G shown in Figure 8.205.

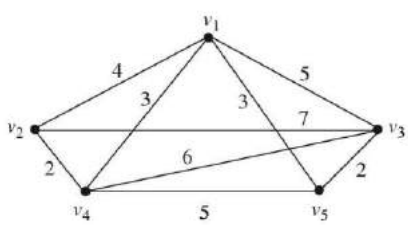


Figure 8.205

Solution.

The given weighted graph has five vertices. The minimal spanning tree would, therefore, have four edges.

Let

$$E=\{(v_1, v_2), (v_1, v_4), (v_1, v_5), (v_2, v_3), (v_1, v_3), (v_2, v_4), (v_4, v_5), (v_5, v_3), (v_3, v_4)\}.$$

The edges (v_2, v_4) and (v_3, v_5) have minimum weight. We choose arbitrarily one of these, say (v_2, v_4). Thus

$$E^*=\{(v_2, v_4)\},$$
$$E=E-\{(v_2, v_4)\}.$$

The edge (v_3, v_5) has minimum weight, so we pick it up. We have thus

$$E^*=\{(v_2, v_4), (v_3, v_5)\},$$
$$E=E-\{(v_2, v_4), (v_3, v_5)\}.$$

The edges (v_1, v_4) and (v_1, v_5) have minimum weight in the remaining edge set. We pick (v_1, v_4) say, as it does not form a cycle with E^*. Thus

$$E^*=\{(v_2, v_4), (v_3, v_5), (v_1, v_4),$$
$$E=E-\{(v_2, v_4), (v_3, v_5), (v_1, v_4)\}.$$

Now the edge (v_1, v_5) has minimum weight in $E-\{(v_1, v_4), (v_3, v_5), (v_1, v_4)\}$ and it does not form a cycle with E^*. So, we have

$$E^*=\{(v_2, v_4), (v_3, v_5), (v_1, v_4), (v_1, v_5)\}$$

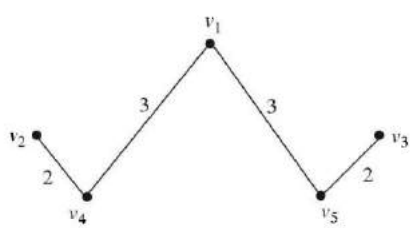
and

$$E=E-\{(v_2, v_4), (v_3, v_5), (v_1, v_4), (v_1, v_5)\}.$$

Thus all the four edges have been selected. The minimal tree has the edges.

$$(v_2, v_4), (v_3, v_5), (v_1, v_4), (v_1, v_5)$$

and is shown in the Figure 8.206.



(Minimal spanning tree)

Figure 8.206

EXAMPLE 8.91

Using Kruskal's algorithm, find the minimal spanning tree for the graph given in the Figure 8.207.

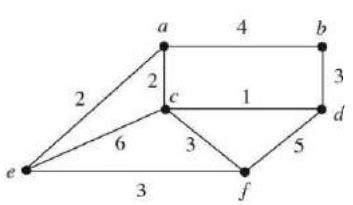


Figure 8.207

Solution.

The given graph has six vertices, therefore the minimal spanning tree would have $6-1=5$ edges

Let

$$E=\{(a, b), (a, c), (b, d), (c, f), (d, f), (c, d), (a, e), (e, f), (e, c)\}.$$

Choose the edge (c, d) first because it has minimum weight. Therefore,

$$E^*=\{(c, d)\}, \quad E=E-\{(c, d)\}.$$

Now choose the edge (a, c) because it has minimum weight. So,

$$E^*=\{(c, d), (a, c), \quad E=E-\{(c, d), (a, c)\}.$$

Now choose the edge (a, e), since it has also minimum weight in $E-\{(c, d),(a, c)\}$ and it does not form cycle with E^*. So

$$E^*=\{(c, d), (a, c), (a, e)\}, \quad E=E-\{(c, d), (a, c), (a, e)\}.$$

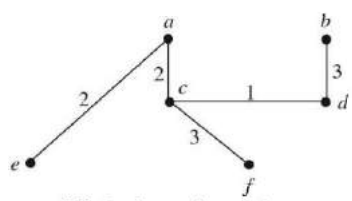
Now choose (b, d) since it has minimum weight and it does not form circuit with E^*. Thus

$$E^*=\{(c, d), (a, c), (a, e), (b, d)\},$$
$$E=E-\{(c, d), (a, c), (a, e), (b, d)\}.$$

Now choose (c, f) or (e, f) because they have equal and minimal weight. Let us choose (c, f). Then

$$E^*=\{(c, d), (a, c), (a, e), (b, d), (c, f)\},$$
$$E=E-\{(c, d), (a, c), (a, e), (b, d), (c, f)\}.$$

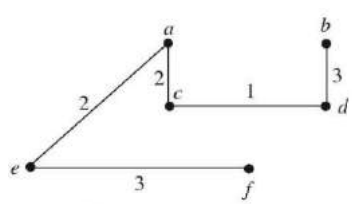
We have thus obtained all the five required edges. The minimal spanning tree is shown in the Figure 8.208.



(Minimal spanning tree)

Figure 8.208

Remark 8.18 In the above example, if we had chosen (e, f) in place of (c, f) in the last step, then the minimal spanning tree would have been as shown in the Figure 8.209.



(Minimal spanning tree)

Figure 8.209

EXAMPLE 8.92

Use Kruskal's algorithm to find a shortest spanning tree for the graph G given in the Figure 8.210.

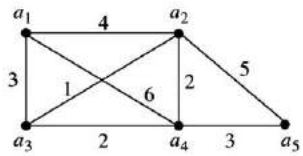


Figure 8.210

Solution.

The given graph has five vertices. So the required spanning tree would have $5-1=4$ edges. Let

$$E=\{(a_1, a_2), (a_1, a_3), (a_3, a_4), (a_2, a_4), (a_2, a_5), (a_4, a_5), (a_2, a_3), (a_1, a_4)\}.$$

Choose the edge (a_2, a_3) since it has minimum weight. Then

$$\begin{aligned} E^* &= \{(a_2, a_3)\}, \\ E &= E-\{a_2, a_3\} \\ &= \{(a_1, a_2), (a_1, a_3), (a_3, a_4), (a_2, a_4), (a_2, a_5), (a_4, a_5), (a_1, a_4)\}. \end{aligned}$$

Now choose the edge (a_2, a_4) or (a_3, a_4) having minimum weight in E. Let us take (a_2, a_4), say. Then

$$E^*=\{(a_2, a_3), (a_2, a_4)\}$$

and

$$\begin{aligned} E &= E-\{(a_2, a_3), (a_2, a_4)\} \\ &= \{(a_1, a_2), (a_1, a_3), (a_3, a_4), (a_2, a_5), (a_4, a_5), (a_1, a_4)\}. \end{aligned}$$

We cannot choose (a_3, a_4) now because it will form circuit with already chosen edges (a_2, a_3), (a_2, a_4). Therefore we can choose (a_1, a_3) or (a_4, a_5). Suppose we choose (a_1, a_3). Then

$$\begin{aligned} E^* &= \{(a_2, a_3), (a_2, a_4), (a_1, a_3)\}, \\ E &= E-\{(a_2, a_3), (a_2, a_4), (a_1, a_3)\} \\ &= \{(a_1, a_2), (a_3, a_4), (a_2, a_5), (a_4, a_5), (a_1, a_4)\}. \end{aligned}$$

Now choose (a_4, a_5) as it has minimal weight in E. Then

$$E^*=\{(a_2, a_3), (a_2, a_4), (a_1, a_3), (a_4, a_5)\}$$

and

$$\begin{aligned} E &= E-\{(a_2, a_3), (a_2, a_4), (a_1, a_3), (a_4, a_5)\} \\ &= \{(a_1, a_2), (a_3, a_4), (a_2, a_5), (a_1, a_4)\}. \end{aligned}$$

Thus we have obtained four edges and therefore the process is complete.

The Kruskal minimal spanning tree is shown in the Figure 8.211.

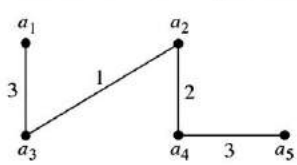


Figure 8.211

Its length is $3+1+2+3=9$.

8.18 CUT SETS

Let G be a connected graph. We know that the distance between two vertices v_1 and v_2, denoted by $d(v_1, v_2)$, is the **length of the shortest path**.

Definition 8.115

The **diameter** of a connected graph G, denoted by diam (G), is the maximum distance between any two vertices in G.

For example, in graph G shown in the Figure 8.212, we have

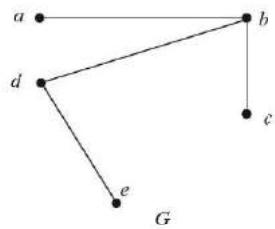


Figure 8.212

$d(a, e)=3$, $d(a, c)=2$, $d(b, e)=2$ and diam $(G)=3$.

Definition 8.116

A vertex v in a connected graph G is called a **cut point** if $G-v$ is disconnected, where $G-v$ is the graph obtained from G by deleting v and all edges containing v.

For example, in the above graph, d is a cut point.

Definition 8.117

An edge e of a connected graph G is called a **bridge** (or **cut edge**) if $G-e$ is disconnected, where $G-e$ is the graph obtained by deleting the edge e.

For example, consider the graph G shown in the Figure 8.213.

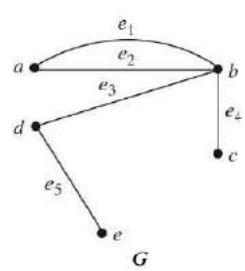


Figure 8.213

We observe that $G-e_3$ is disconnected. Hence the edge e_3 is a bridge.

Definition 8.118

A minimal set C of edges in a connected graph G is said to be a **cut set** (or **minimal edge-cut**) if the subgraph $G-C$ has more connected components than G has.

For example, in the above graph, if we delete the edge $(b, d)=e_3$, the resulting subgraph $G-e_3$ has two connected components shown in the Figure 8.214.

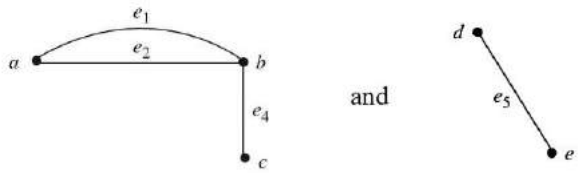


Figure 8.214

So, in this example, the cut set consists of single edge $(b, d)=e_3$, which is called cut edge or bridge.

EXAMPLE 8.93

Find a cut set for the graph given below in the Figure 8.215.

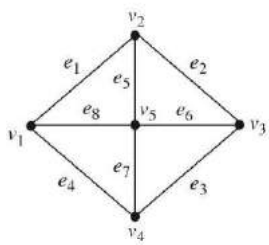


Figure 8.215

Solution.

The given graph is connected. It is sufficient to reduce the graph into two connected components. To do so we have to remove the edges e_1, e_4, e_5, e_6, e_7. The two connected components are shown in the Figure 8.216.

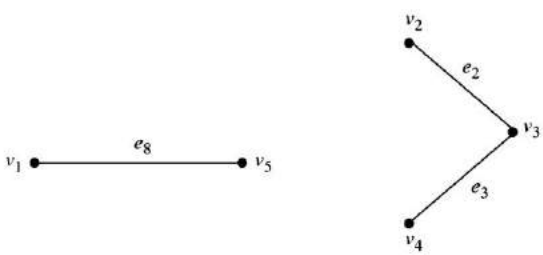


Figure 8.216

But, if we remove any proper subset of $\{e_1, e_4, e_5, e_6, e_7\}$, then there is no increase in connected components of G. Hence $\{e_1, e_4, e_5, e_6, e_7\}$ is a cut set.

EXAMPLE 8.94

Find a cut set for the graph given in the Figure 8.217.

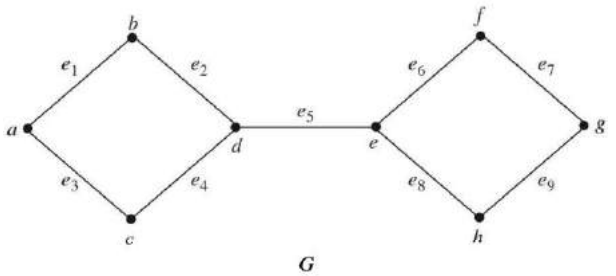


Figure 8.217

The given graph G is connected. Clearly $G-e_5$ has two connected components give in the Figure 8.218.

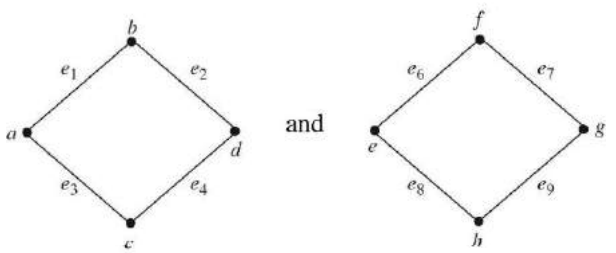


Figure 8.218

Hence a cut set for this graph is $\{e_5\}$. Some other cut sets for G are $\{e_1, e_3\}$, $\{e_3, e_2\}$, $\{e_6, e_9\}$, $\{e_6, e_8\}$

EXAMPLE 8.95

Find a cut set for the graph represented by the Figure 8.219.

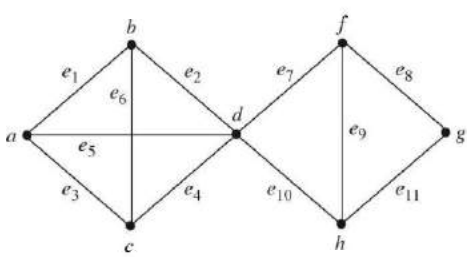


Figure 8.219

Solution.

The given graph is a connected graph. We note that removal of the edges e_7 and e_{10} creates two connected components of G shown in Figure 8.220.

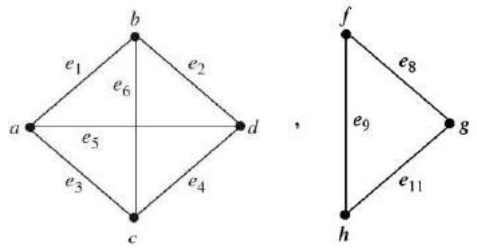


Figure 8.220

Hence the set $\{e_7, e_{10}\}$ is a cut set for the given graph G.

EXAMPLE 8.96

Find a cut set (minimal edge cut) for the following graph (Figure 8.221).

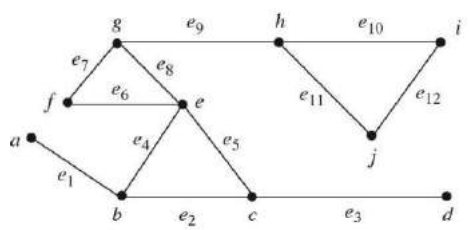
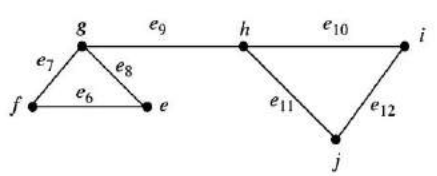


Figure 8.221

Solution.

We note that $\{e_4, e_5\}$ is a cut set (minimal edge cut) for the given graph because removal of e_4 and e_5 gives rise to two components shown in the Figure 8.222.



and

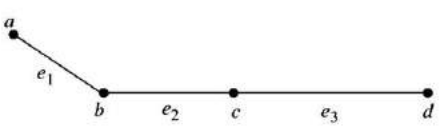


Figure 8.222

Similarly, $\{e_9\}$ forms a cut set (bridge) for this graph.

Theorem 8.30

Let G be a connected graph with n vertices. Then G is a tree if and only if every edge of G is a bridge (cut edge).

(This theorem asserts that every edge in a tree is a bridge).

Proof. Let G be a tree. Then it is connected and has $n-1$ edges (proved already). Let e be an arbitrary edge of G. Since $G-e$ has $n-2$ edges, and also we know that a graph G with n vertices has at least $n-C(G)$ edges, it follows that $n-2 \geq n-C(G-e)$. Thus, $G-e$ has at least two components. Thus removal of the edge e created more components than in the graph G. Hence e is a cut edge. This proves that every edge in a tree is a bridge.

Conversely, suppose that G is connected and every edge of G is a bridge. We have to show that G is a tree. To prove it, we have only to show that G is circuit-free. Suppose on the contrary that there exists a cycle between two points x and y in G (Figure 8.223). Then any edge on this cycle is not a cut edge which contradicts the fact that every edge of G is a cut edge. Hence G has no cycle. Thus G is connected and acyclic and so is a tree.

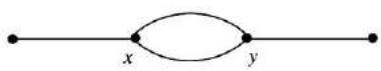


Figure 8.223

8.18.1 Relation Between Spanning Trees, Circuits and Cut Sets

A spanning tree contains a unique path between any two vertices in the graph. Therefore, addition of a chord to the spanning tree yields a subgraph that contains exactly one circuit. For example, consider the graph G shown below in the Figure 8.224.

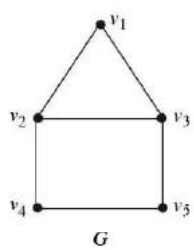


Figure 8.224

For this graph, the Figure 8.225 is a spanning tree.

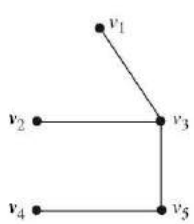


Figure 8.225 (Spanning tree)

The chords of this tree are (v_1, v_2) and (v_2, v_4). If we add (v_1, v_2) to this spanning tree, we get a circuit $v_1 v_2 v_3 v_1$. Similarly addition of (v_2, v_4) gives one more circuit $v_2 v_3 v_5 v_4 v_2$. If there are v vertices and e edges in a graph, then there are $e-v+1$ chords in a spanning tree. Therefore, if we add all the chords to the spanning tree, there will be $e-v+1$ circuits in the graph.

Definition 8.119

Let v be the number of vertices and e be the number of edges in a graph G. Then the set of $e-v+1$ circuits obtained by adding $e-v+1$ chords to a spanning tree of G is called the **fundamental system of circuits relative to the spanning tree**.

A circuit in the fundamental system is called a **fundamental circuit**.

For example, $\{v_1, v_2, v_3, v_1\}$ is the fundamental circuit corresponding to the chord (v_1, v_2).

On the other hand, since each branch of a tree is cut edge, removal of any branch from a spanning tree breaks the spanning tree into two trees. For example, if we remove (v_1, v_3) from the above figured spanning tree, the resulting components are shown in the Figure 8.226.

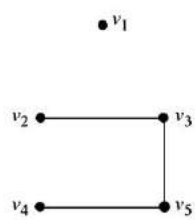


Figure 8.226

Thus, **to every branch in a spanning tree, there is a corresponding cut set.** But, in a spanning tree, there are $v-1$ branches. Therefore, **there are $v-1$ cut sets corresponding to $v-1$ branches.**

Definition 8.120

The set of $v-1$ cut sets corresponding to $v-1$ branches in a spanning tree of a graph with v vertices is called the **fundamental system of cut sets relative to the spanning tree.**

A cut set in the fundamental system of cut sets is called a **fundamental cut set.**

For example, the fundamental cut sets in the spanning tree (figured above) are

$$\{(v_1, v_2), (v_1, v_3)\}, \{(v_1, v_3), (v_2, v_3), (v_3, v_4)\},$$
$$\{(v_3, v_5), (v_4, v_5)\}, \{v_2, v_4), (v_4, v_5)\}.$$

Theorem 8.31

A circuit and the complement of any spanning tree must have at least one edge in common.

Proof. We recall that the set of all chords of a tree is called the complement of the tree. Suppose on the contrary that a circuit has no common edge with the complement of a spanning tree. This means the circuit is wholly contained in the spanning tree. This contradicts the fact that a tree is acyclic (circuit- free). Hence a circuit has at least one edge in common with complement of a spanning tree.

Theorem 8.32

A cut set and any spanning tree must have at least one edge in common.

Proof. Suppose on the contrary that there is a cut set which does not have a common edge with a spanning tree. Then removal of cut set has no effect on the tree, that is, the cut set will not separate the graph into two components. But this contradicts the definition of a cut set. Hence the result.

Theorem 8.33

Every circuit has an even number of edges in common with every cut set.

Proof. We know that a cut set divides the vertices of the graph into two subsets each being set of vertices in one of the two components. Therefore a path connecting two vertices in one subset must traverse the edges in the cut set an even number of times. Since a circuit is a path from some vertex to itself, it has an even number of edges in common with every cut set.

8.19 TREE SEARCHING

Let T be a binary tree of height $h \geq 1$ and root v. Since $h \geq 1$, v has at least one child : v_L and/or v_R. Now v_L and v_R are the roots of the left and right subtrees of v called T_L and T_R, respectively.

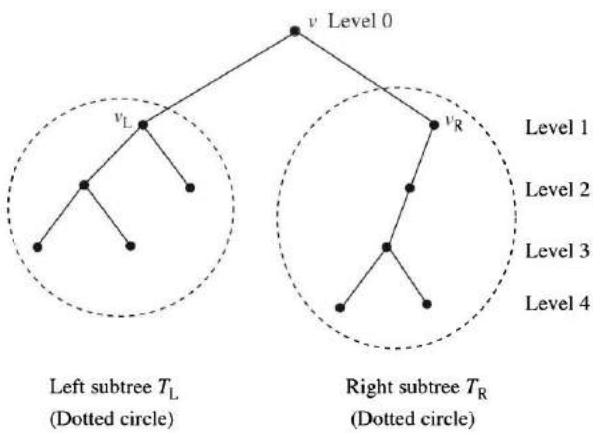


Figure 8.227

Definition 8.121

Performing appropriate tasks at a vertex is called **visiting the vertex**.

Definition 8.122

The process of visiting each vertex of a tree in some specified order is called **searching the tree** or **walking** or **traversing the tree**.

We now discuss methods of searching a tree.

1. Pre-order Search Method

Input: The root v of a binary tree.

Output: Vertices of a binary tree using preorder traversal

1. **Visit** v
2. If v_L (left child of v) exists, then apply the algorithm to $(T(v_L), v_L)$
3. If v_R (right child of v) exists, then apply this algorithm to $(T(v_R), v_R)$

End of algorithm pre-order.

In other words, pre-order search of a tree consists of the following steps:

Step 1. Visit the root.

Step 2. Search the left subtree if it exists.

Step 3. Search the right subtree if it exists.

EXAMPLE 8.97

Find the order of the vertices of the following tree T processed using pre-order traversal.

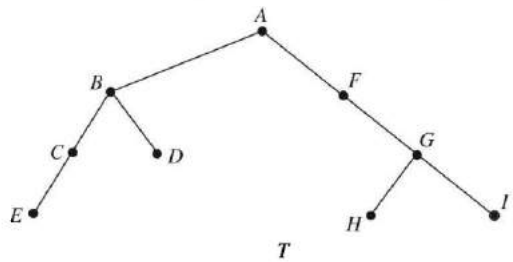


Figure 8.228

Solution.

The root of the given tree is A. It has two children B and F. The left subtree and right subtree are shown in Figure 8.229.

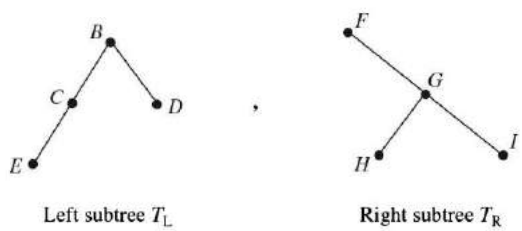


Figure 8.229

If we apply pre-order algorithm to the given tree T, we visit the root and print A. Using pre-order to left subtree, we visit the root B and print B and then search T_L printing C and E and then D. Up to this point, the search has yielded the string $A\ B\ C\ E\ D$. Thus the search of T_L is complete. We now go to subtree T_R. Using the same procedure, we get the string $F\ G\ H\ I$. Thus the complete search of the tree T is

A B C E D F G H I.

EXAMPLE 8.98

Find binary tree representation of the expression

$$(a-b)\times(c+(d\div e))$$

and represent the expression in string form using pre-order traversal.

Solution.

In the given expression, $\times$ is the central operator and therefore shall be the root of the binary tree. Then the operator $-$ acts as v_L and the operator $+$ acts as v_R. Thus the tree representation of the given expression is as shown in the Figure 8.230.

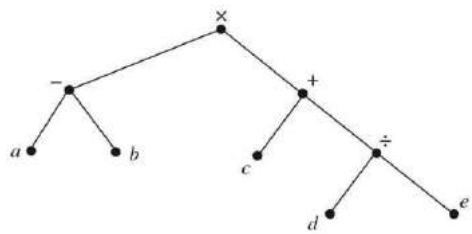


Figure 8.230

The result of the pre-order traversal to this binary tree is the string

$$\times - a\ b + c \div d\ e.$$

This form of the expression is called **prefix form** or **polish form** of the expression

$$(a-b)\times(c+(d\div e)).$$

In a polish form, the variables $a, b, c, \ldots$ are called **operands** and $-, +, \times, \div$ are called **operators**. We observe that, **in polish form, the operands follow the operator**.

EXAMPLE 8.99

Represent the expression $(A+B) * (C-D)$ as a binary tree and write the prefix form of the expression.

Solution.

Here * is the central operator. Further + and − operators are v_L and v_R. Hence the binary tree for the given experssion is as shown in the Figure 8.231.

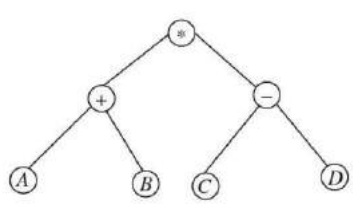


Figure 8.231

Using pre-order traversal, the prefix expression for it is

$$*+A\ B-C\ D.$$

Fig. (b) shows the maximum flow network. Note that the vertex a has a negative label. (b)

The maximal flow is 8.

13.15. MATCHING

Given a graph $G = (V, E)$, a matching M is G in a set of pairwise non-adjacent edges, that is no two edges in M have a vertex in common.

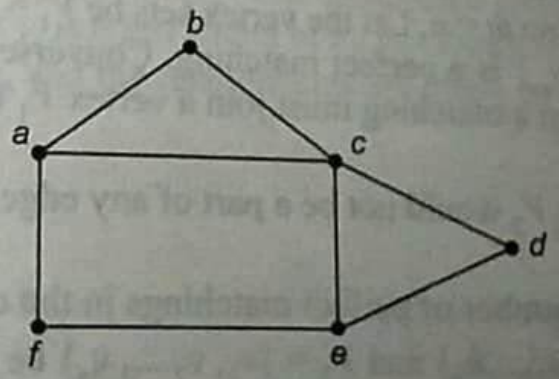
A vertex which is incident with an edge in the set M is said to be saturated (or M-saturated) otherwise the vertex is unsaturated (or M-unsaturated)

A matching M is **perfect**, if and only if, every vertex of G is saturated; that is, if and only if, every vertex is incident with precisely one edge of the matching. In some book, the term complete matching is used.

A **maximum matching** is a matching that contains the largest possible member of edges *i.e.*, if G has no matching M' with a greater number of edges than M has.

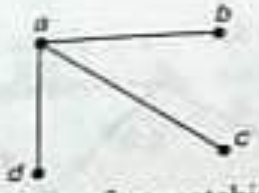
Note that every perfect matching is a maximum matching but the converse is not true.

In the graph below



$M_1 = \{ab, ce\}$ and $M_2 = \{ab, cd, ef\}$ are both matching. In M_1, the vertices a, b, c and e are saturated while f and d are unsaturated. The edge set M_2 is a perfect matching, because each of the vertices of G is an end vertex of some member of M_2. The matching $M_3 = \{ac, fe\}$ is a maximum matching but not a perfect matching.

Note that if G has a perfect matching, then G has an even number of vertices. The converse is not true. The graph shown below



has even number of vertices but it has no perfect matching.

Definition: Given a matching M is G, a path is called **an alternating path** in which the edges belong alternately between M and not in M *i.e.*, in $E-M$ (or E/M).

An alternating path whose origin and terminal vertices are both unsaturated in called an **augmenting path**.

In matching $M_4 = \{ab, fc, ed\}$, the path *fedc* is an alternating path and is the matching $M_5 = \{ac, fe\}$, the path *bacfed* is an **augmenting path**.

Maximum matching have been characterized by the following theorem.

Theorem 13.25 The matching M of a graph G is maximum if, and only if, G has no M-augmenting path.

Example 66. For what values of n does the complete graph K_n have perfect matching?

Solution. Let n is even and $V(K_n) = \{1, 2, ..., n\}$. Then $M = \{12, 34,....., (n-1)n\}$ is a perfect matching of K_n. Clearly if n is odd, the graph has no perfect matching. Thus K_n has a perfect matching if and only if n is even.

Example 67. Let a_n denotes the number of perfect matching in the complete graph K_{2n}. Find a recurrence relation for a_n and solve.

Solution. Let $V(K_{2n}) = \{v_1, v_2,....., v_{2n}\}$.

We can pair the vertex v_1 with any of the other $2n-1$ vertices, we are then confronted in the case when $n \geq 2$, with finding a perfect matching for the graph k_{n-2}. Hence

$$a_n = (2n-1)a_{n-1}, \qquad a_1 = 1$$

We find

$$\begin{aligned} a_n &= (2n-1)a_{n-1} = (2n-1)(2n-3)a_{n-2} \\ &= (2n-1)(2n-3)(2n-5)a_{n-3} \\ &= (2n-1)(2n-3)(2n-5)....(5)(3)(1) \\ &= \frac{(2n)(2n-1)(2n-2)(2n-3)....(4)(3)(2)(1)}{(2n)(2n-2)....(4)(2)} \\ &= \frac{(2n)!}{2^n n!} \end{aligned}$$

Example 68. Determine necessary and sufficient conditions for the complete bipartite graph $k_{m,n}$ to have a perfect matching.

Solution. First we assume $m = n$. Let the vertex sets be $V_1 = \{u_1, u_2,..., u_m\}$ and $V_2 = \{v_1, v_2,..., v_m\}$. Then $\{u_1v_1, u_2v_2,....., u_mv_m\}$ is a perfect matching. Conversely, let we have a perfect matching and $m \leq n$. Since, each edge in a matching must join a vertex V_1 to vertex a V_2, there can be at most m edges.

In $m < n$, some vertex is V_2 would not be a part of any edge in the matching which is a contradiction. Thus $m = n$.

Example 69. Find the number of perfect matchings in the complete bipartite graph $k_{n,n}$.

Solution. Let $V_1 = \{u_1, u_2,....u_n\}$ and $V_2 = \{v_1, v_2,....., v_n\}$ be a bipartition of $k_{n,n}$.

We see that any matching of $k_{n,n}$ that saturates vertex of V_1 is a perfect matching. The vertex u_1 can be saturated in n ways by choosing any of the edges $u_1v_1, u_1v_2,...u_1v_n$. Having saturated u_1, the vertex u_2 can be saturated is $n-1$ ways. In general having saturated $u_1, u_2,.....u_i$, the next vertex u_{i+1} can be saturated in $n-i$ ways. Thus the number of perfect matchings in $k_{n,n}$ is $n(n-1)(n-2)...3.2.1 = n!$.

Matchings in bipartite graphs

Matching problems are often concerned with bipartite graphs. The following problems motivate the study of matchings in bipartite graphs.

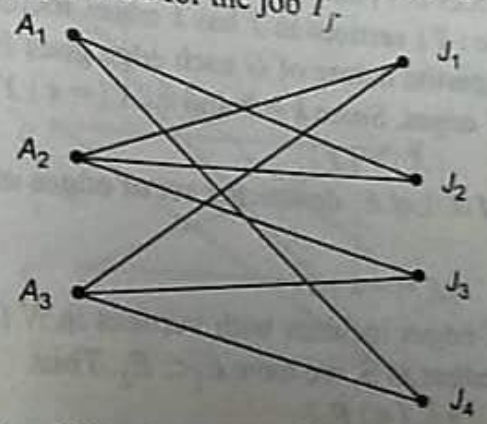
(i) Assignment Problem

Suppose three persons A_1, A_2 and A_3 apply for four jobs J_1, J_2, J_3 and J_4 in a company. Each person is qualified for one or more jobs. The following information is obtained from the applications of the candidates.

Applicant	Jobs
A_1	J_2, J_4
A_2	J_1, J_2, J_3
A_3	J_1, J_3, J_4

A question with obvious practical implication is the following. Is it possible to assign one job for each applicant for which he is qualified?

We construct a bipartite graph G with bipartition $V_1 = \{A_1, A_2, A_3\}$ and $V_2 = \{J_1, J_2, J_3, J_4\}$. An edge joins A_i to J_j if and only if A_i is qualified for the job T_j.



Notice that the graph in Fig. is a bipartite graph. Then our assignment problem translates into the graph problem: Is it possible to find a matching in G that saturates all the vertices in V_1.

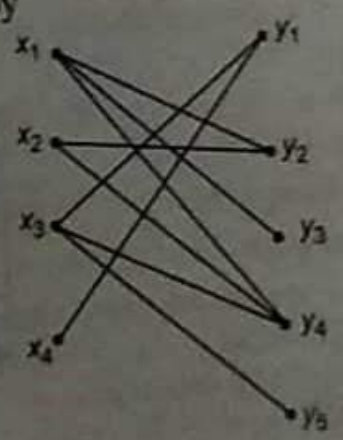
A particular solution to the above problem is for the match $M = \{A_1J_2, A_2J_1, A_3J_3\}$ A_1 is assigned job J_2, A_2 is assigned job J_1 and A_3 is assigned job J_3. Thus each applicant can be assigned one of the jobs for which he or she is qualfied.

(ii) The Marriage Problem

Let we have n boys and each boy has one or more girl friends. Under what conditions can we arrange marriage in such a way that each boy marries one of his girl friends? This problem is known as the marriage problem. (We assume no boy can marry more than one girl).

We construct a bipartite graph G with bipartition $V(G) = A \cup B$ where $A = \{x_1, x_2, \ldots, x_n\}$ represents the set of n boys and $B = \{y_1, y_2, \ldots, y_m\}$ represents their girl friends, m in all. An edge joins vertex x_i to vertex y_j if and only if y_j is a girl friend of x_i. For example, if there are four boys, x_1, x_2, x_3, x_4 and five girls y_1, y_2, y_3, y_4, y_5 and the relationships are given by

Boy	Girl friend
x_1	y_2, y_3, y_4
x_2	y_2, y_4
x_3	y_1, y_4, y_5
x_4	y_1



Note that the graph in Fig. is a bipartite graph. A particular solution to the above problem is for the match $M = \{x_1y_3, x_2y_2, x_3y_4, x_4y_1\}$, x_1 to marry y_3, x_2 to marry y_2, x_3 to marry y_4 and x_4 to marry y_1.

In general, a solution to the above matching problem is barpartite graph has been given by P. Hall.

For a subset $S \subseteq V$ is a graph G, $N(s)$ denotes the neighbor set of S, that is, the set of all vertices each of which is adjacent to at least one vertex is S.

Theorem 13.26 (Hall's marriage theorem) Let G be a bipartite graph with bipartition (X, Y). Then G has a matching that saturates all the vertices of X if, and only if,

$$|N(S)| \geq |S|$$

for every subset S of X.

The theorem is known as Hall's marriage theorem because originally elements in X were thought of as women and elements is Y were thought of as men. A matching between the two sets corresponded to marrying each woman to a man.

The following is an important consequence of Hall's marriage theorem.

Theorem 13.27 Let G be a K-regular bipartite graph with $K > 0$. Then G has a perfect matching.

Proof. Let G have bipartition $V = X \cup Y$. There are $|X|$ vertices in X and each of these vertices has k edges (all going to vertices is Y) incident with it. Thus there are $k \times |X|$ edges going from X to Y. Similarly since each of the $|Y|$ vertices in Y has k edges incident with it there are $k \times |Y|$ edges going from Y to X. By the bipartite nature of G each edge goes from X to Y (the same as from Y to X) and so G has $k|X| = k|Y$ edges. Since $k > 0$ and $k|X| = k|Y|$, we get

$$|X| = |Y|$$

Now let S be a subset of X. Let E_1 denote the set of edges incident with vertices in S. Since G is k-regular, we have

$$|E_1| = k|S|$$

Let E_2 denote the set of edges incident with vertices in $N(S)$. Then, since $N(S)$ is the set of vertices which are joined by edges to S, we have $E_1 \subset E_2$. Thus

$$|E_1| \leq |E_2|.$$

Again by the k-regularity of G we have

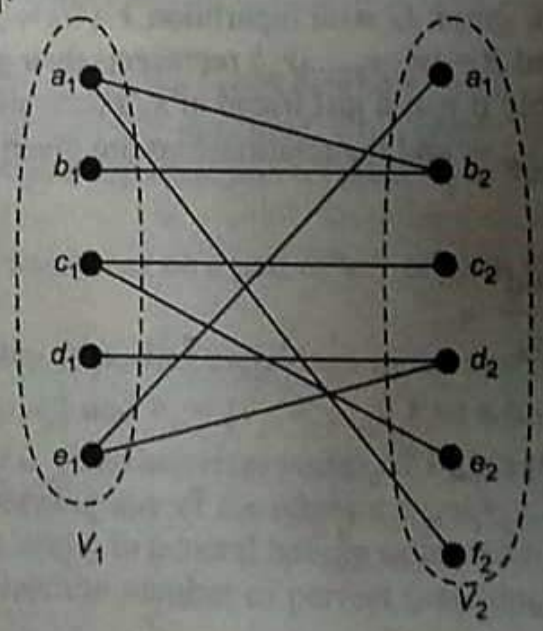
$$|E_2| = k|N(S)|.$$

From (1), (2) and (3), we get $k|N(S)| = |E_2| \geq |E_1| = k|S|$ and so

$$k|N(S)| \geq k|S|$$

Since $k > 0$ this $k|N(S)| \geq k|S|$. Since S was an arbitrary subset of X, it follows from Hall's mariage theorem that G contains a matching M which saturates every vertex in X. Since $|X| = |Y|$, the edges in the matching M also saturate every vertex in Y. Thus M is a perfect matching in G.

Example 70. Find Whether a complete matching of V_1 into V_2 exist for the following. What can you say from V_2 into V_1.



Solution. A complete matching exists from V_1 into V_2, which is $M_1 = \{a_1f_2, b_1b_2, c_1c_2, d_1d_2, e_1a_2\}$. This matching is not unique, because $M_2 = \{a_1f_2, b_1b_2, c_1c_2, d_1d_2, e_1a_2\}$ is also a complete matching from V_1 into V_2. Complete matching from V_2 into V_1 does not exist because cardinality of V_2 is more than the cordinality of V_1.

Example 71. Suppose that three boys a, b, c know four girls w, x, y, z as follows:

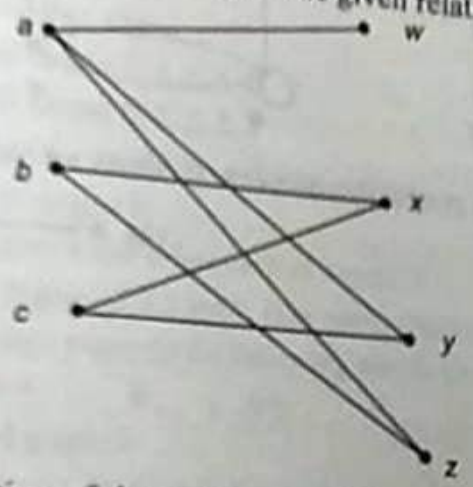
Boy	Girls know by boy			
a	w	y	z	
b	x	z		
c	x	y		

(*i*) Draw the bipartite graph corresponding to this table of relationships.

(*ii*) Find five different solutions of the corresponding marriage problem.

(*iii*) Check the marriage conditions for this problem.

Solution. (*i*) The bipartite graph corresponding to the given relationship is shown below.



(*ii*) The five different selection of the problem are $M_1 = \{aw, bx, cy\}$, $M_2 = \{aw, bz, cx\}$, $M_3 = \{aw, bz, cy\}$, $M_4 = \{ay, bz, cx\}$, $M_5 = \{az, bx, cy\}$

(*iii*)

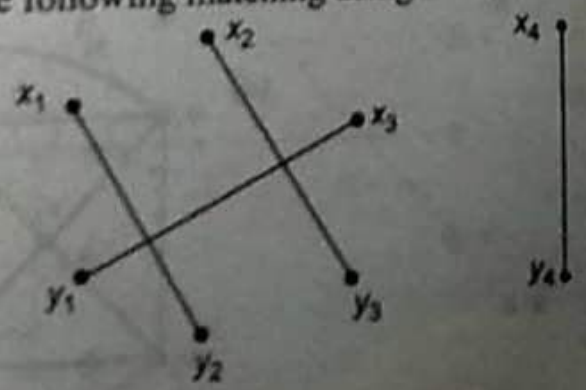
A	d	a	b	c	ab	ac	bc	abc
$\lvert A \rvert$	0	1	1	1	2	2	2	3
$\lvert d(A) \rvert$	0	3	2	2	4	4	3	4

Thus $\lvert A \rvert \le \lvert d(A) \rvert$ for each subset of $A = \{a, b, c\}$.

Example 72. Show that a tree cannot have more than one perfect matching.

Solution. Suppose M and M' are two perfect matchings in a tree such that there is an edge e_1 joining vertices v_1 and v_2 in $M - M'$. Edge e_1 cannot be a terminal edge since a terminal edge of a tree has to be an edge in every perfect matching. So there is a nonterminal vertex v_3 and edge e_2 joining v_2 and v_3 such that e_2 is in $M' - M$. If this process continues, we get a cycle that will terminate at v_1. But the graph is acyclic.

Example 73. Express the following matching using matrices.



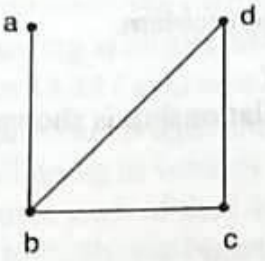
Solution. The matrix representation of the given matching is

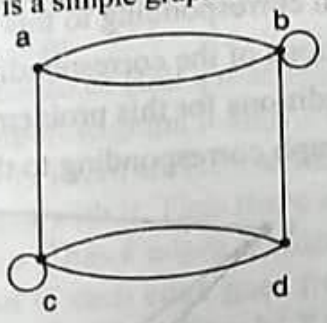
$$\begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{cccc} y_1 & y_2 & y_3 & y_4 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

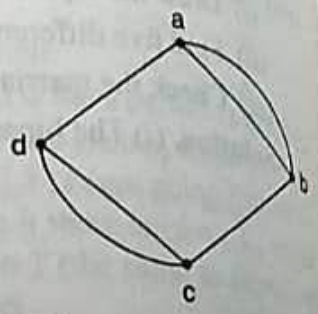
GRAPH THEORY PROBLEM SET

Problem Set 13.1

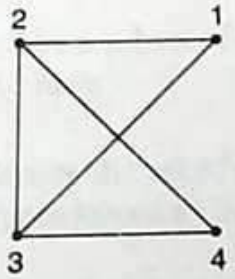
1. Determine whether the graphs shown is a simple graph a multigraph, a pseudograph.

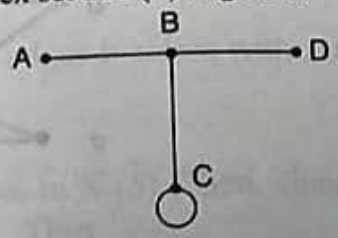
(a)


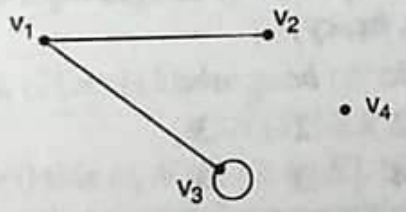
(b)


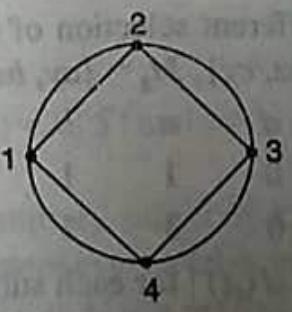
(c)


2. Consider the following graphs, determine the (*i*) vertex set and (*ii*) edge set.

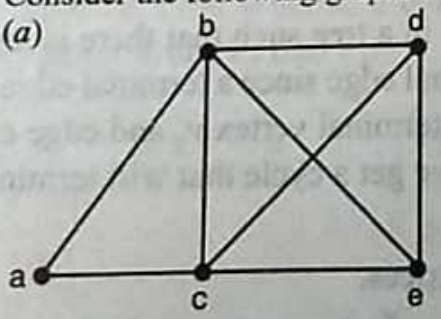
(a)


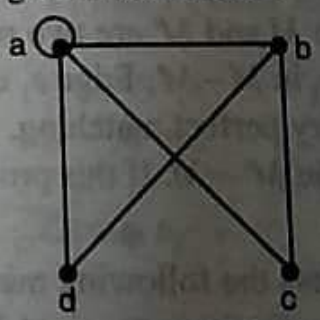
(b)


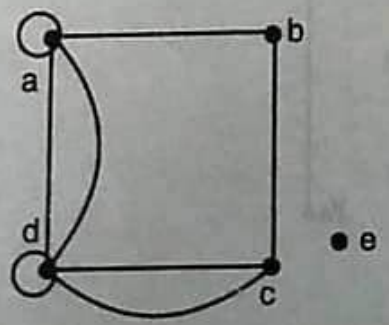
(c)


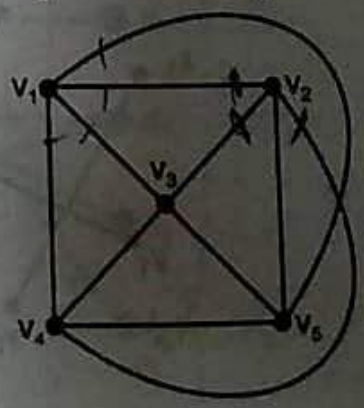
(d)


3. Consider the following graphs and determine the degree of each vertex.

(a)


(b)


(c)


(d)


AMOOR- V (2014 Course) CBCS : WINTER - 2016

Subject : Combinotrics & Graph Theory

Day : Tuesday
Date : 15/11/2016

32287

Time : 02.00 PM TO 05.00 PM
Max Marks : 100 Total Pages : 2

N.B.:

1) Attempt **ANY FOUR** from Section-I & **ANY TWO** from Section-II.
2) Use of non programmable scientific **CALCULATOR** is allowed.

SECTION – I

Q.1 Write axioms of probability. If P (A) = 0.35, P (B) = 0.75 and P(AUB) = 0.65, find **[15]**

i) $P(A \cap B)$ if A and B are independent events
ii) $P(A \cap B)$ if A and B are dependent events
iii) $P\overline{(A \cap B)}$ if A and B are dependent events.

Q.2 Find n in each of the following equations: **[15]**

i) P (n, 2) = 72
ii) P (n, 4) = 840
iii) 2P (n, 2) + 50 = P (2n, 2)

Q.3 Write a note on probability density function of continuous random variable. **[15]**

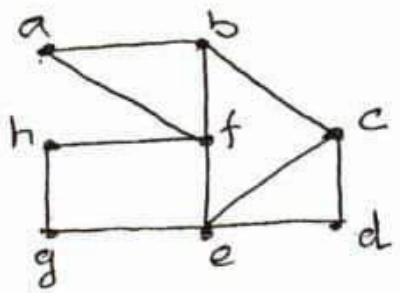
Five cards are numbered from 1 to 5. Two cards are drawn at random . If X denotes the sum of the numbers drawn,

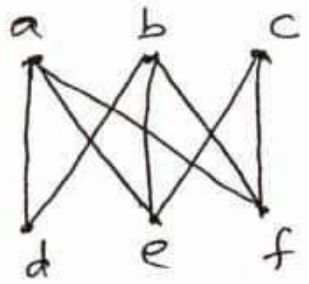
a) find the distribution of X
b) find mean.

Q.4 Represent the following graphs diagrammatically and explain in short: **[15]**

a) Multigraphs b) Sub graphs C) Isomorphic graphs
d) Complete graphs e) Planner graphs

Q.5 State whether the following graphs are Eulerian and or Hamiltonian with proper explanation. **[15]**





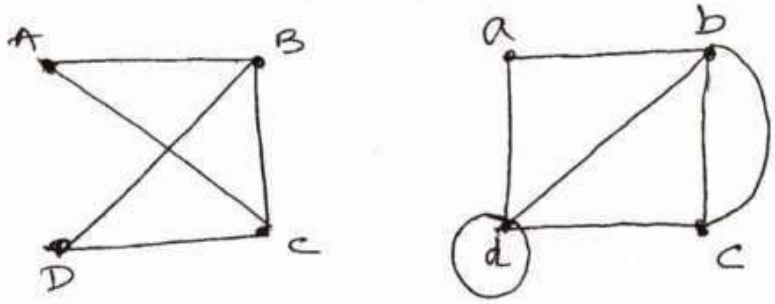
Q.6 Write short notes on: **[15]**

a) Colouring of graphs
b) Crossing of river problem
c) Inclusion - Exclusion Principle

P.T.O.

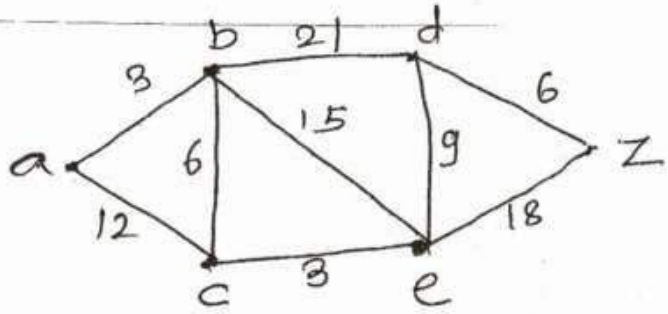
SECTION - II

Q.7 a) Find the adjacency matrix A of each of the following graph: **[10]**



b) Consider the following addresses which are in random order: **[10]**
1.1, 2.2.1, 2.2.2, 3.2, 2.2.1.1, 3.2.1.1, 3.1, 3.2.1.1, 1.1.1, 1.2.1, 3.1.1, 3.2.2.
a) Place the above addresses in lexicographic order.
b) Draw the corresponding ordered rooted tree.

Q.8 Write Dijkstra's algorithm. Compute the shortest distance between source 'a' and destination 'z' using Dijkstra's algorithm from the following graph. **[20]**



Q.9 a) An item is produced on three machines A, B and C. 50% of items are produced on machine A, 30% of items are produced on machine B and 20% of items are produced on machine C.The chances of an item to be produced as defective on these A, B, and C machines are 3%, 1.5% and 2% respectively. An item produced on any one of these machines is randomly selected and it is found to be defective. What is the probability that it is produced on machine B? **[10]**

b) In Pune ladies club, 60% of the members play tennis, 40% play golf and 20% play both tennis and golf. A member is chosen at random. **[10]**
a) Find the probability that she plays neither tennis nor golf.
b) Given that she plays tennis, find the probability that she plays golf.
c) Given that she plays golf, find the probability that she plays tennis.

* * * *

AMOOR- V (2014 Course) CBCS : SUMMER - 2017

Subject : Combinotrics & Graph Theory

Day : Tuesday
Date : 18/04/2017

35642

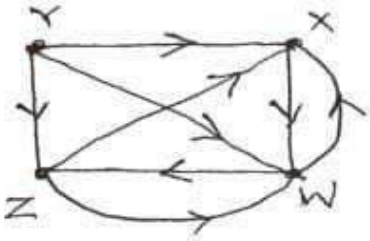
Time : 02.00 PM TO 05.00 PM
Max Marks : 100 Total Pages : 2

N.B.:

1) Attempt **ANY FOUR** from Section-I & **ANY TWO** from Section-II.
2) Use of non programmable scientific **CALCULATOR** is allowed.

SECTION – I

Q.1 From the following graph, find the number of paths of length 5 from Y to Z. **[15]**

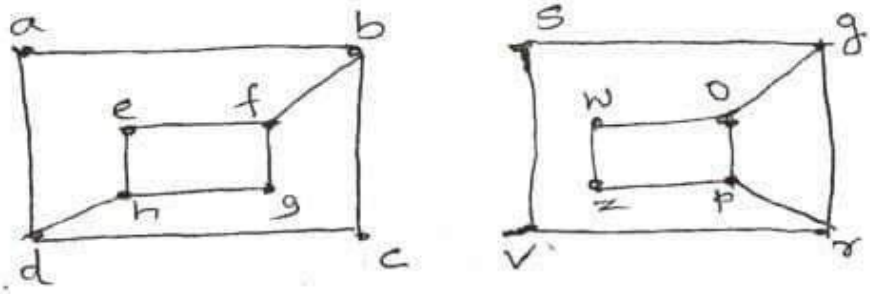


Q.2 A bag contains 6 white marbles and 4 red marbles. Find the number of ways of drawing 6 marbles from the bag so that **[15]**

i) they can be of any colour .
ii) there must be equal number of white marbles and red marbles.
iii) there must be at least 3 white marbles .

Q.3 A fair coin is tossed six times. Prepare probability distribution table which shows the number of heads appears on upper face. Also find expected number of heads. **[15]**

Q.4 State whether the following G and H graphs are isomorphic with proper explanation. **[15]**



Q.5 Two unbiased dice are thrown simultaneously and the total score is counted on the basis of addition of numbers that appear on the upper faces. What is the probability that **[15]**

i) the total score is multiple of 5?
ii) the total score is a perfect square?
iii) the total score is at least 10?

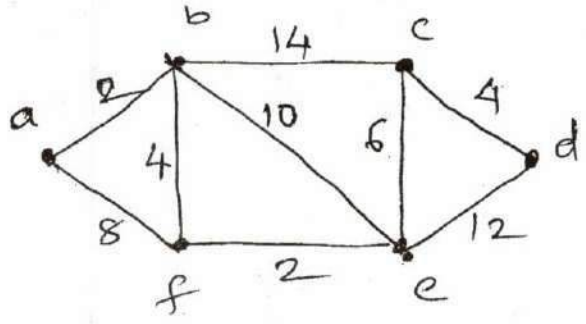
Q.6 Write short notes on: **[15]**

a) Warshall's algorithm
b) Probability density function of continuous random variable
c) Any five applications of pigeonhole principle

P.T.O.

SECTION - II

Q.7 Write Dijkstra's algorithm .Find the shortest path from 'a' to 'd' from the following graph by using Dijkstra's algorithm. **[20]**



Q.8 a) Explain the following applications of graph: **[10]**
a) Bridges of Konigsberg
b) Utilities problem
c) Seating arrangement problem
d) Traveling salesman problem.

b) If A and B are any two events such that P (A) = 1/3 , P (B) = 1/4 and P $(A \cup B)$ =1/2 , then answer the following: **[10]**
a) Find P (A/B) and P (B/A).
b) Are A and B independent?

Q.9 a) There are 10 batsmen, 7 bowlers, and 3 wicket keepers. The board has to select a team of 11 players to be sent for matches to be played abroad. In how many ways ,can the board select a team such that **[10]**
a) there are 2 wicket keepers and exactly 4 batsmen ?
b) there are at least 5 bowlers and 1 wicket keeper ?

b) Prove that C (2n, 2) = 2C (n,2) + n^2 **[10]**

* * * *

B.C.A. SEM–V (2014 COURSE) CBCS : SUMMER - 2018

SUBJECT: COMBINOTRICS & GRAPH THEORY

Day: **Friday**
Date: **04/05/2018**

S-2018-1713

Time: **02.00 PM TO 05.00 PM**
Max. Marks: 100

N.B.:

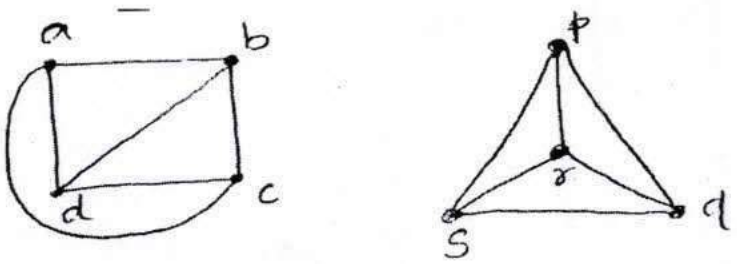
1) Attempt **ANY FOUR** from Section-I & **ANY TWO** from Section-II.
2) Use of non programmable scientific **CALCULATOR** is allowed.

SECTION – I

Q.1 Three cards are drawn from a well shuffled pack of 52 playing cards. Find **[15]**

i) the number of ways of drawing all the three cards as number cards.
ii) the number of ways of drawing 2 heart cards and 1 an ace card.
iii) the probability of getting all the three cards as picture cards.

Q.2 State whether the following graphs are isomorphic with explanation. **[15]**



Q.3 Two friends A and B appeared for an interview independently. Their chances of getting selection for the jobs are 3/5 and 1/5 respectively. Find the probability that **[15]**

i) both friends A and B are selected.
ii) only one of them is selected.
iii) none of them is selected.

Q.4 Two numbers from 1 to 3 are chosen at random with repetition allowed. If X denotes the sum of the numbers, then **[15]**

i) find the distribution of X.
ii) find the expection of X.

Q.5 Explain Hamiltonian and Eulerian graphs. Draw a graph **[15]**

a) which is Hamiltonian but non Eulerian .
b) which is Eulerian but non Hamiltonian.

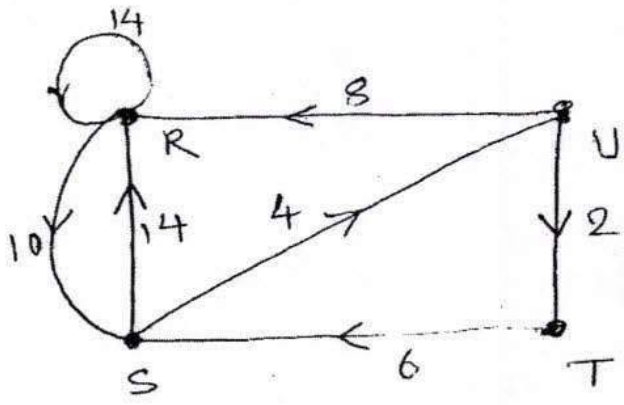
Q.6 Write short notes on: **[15]**

a) Sheep cabbage problem and its analysis by using graph.
b) Colouring of graphs .
c) Probability density function of continuous random variable.

P.T.O.

SECTION - II

Q.7 Write Warshall's algorithm to find the shortest path. Apply it to find the shortest path from T to R from the following graph. **[20]**



Q.8 **a)** An author has 4 typists typing the manuscript of his latest book. Typist A does 30% of typing, typist B does 25% of typing, typist C does 20% of typing and typist D does remaining typing work. Error occur on 5% of the pages typed by A, on 4% typed by B, on 3% typed by C and 2% typed by D. **[10]**

A page is chosen at random and found to contain errors. What is the probability that it was typed by typist A?

b) Write a note on Cumulative Probability Function with suitable example. **[10]**

Q.9 **a)** Explain the following graphs with neat diagram **[10]**

i) Multigraphs
ii) Isomorphic graphs
iii) Complete graphs
iv) Planer graphs
v) Homographic graphs

b) There are 4 professors, 3 clerks and 6 students for forming a committee of 5 people. In how many ways can the committee be formed so that **[10]**

i) there are at least 3 professors in the committee?
ii) there are at most 2 clerks in the committee.

* * * *

B.C.A. SEM–V (2014 Course) CBCS : SUMMER - 2019
SUBJECT : COMBINATORICS & GRAPH THEROY

Day : Wednesday
Date : 24/04/2019

S-2019-2078

Time : 02.00 PM TO 05.00 PM
Max. Marks : 100

N.B.:

1) Attempt **ANY FOUR** questions from Section – I. Each question carries **15** marks.
2) Attempt **ANY TWO** questions from Section – II. Each question carries **20** marks.
3) Answers to both the sections should be written in SAME answer books.

SECTION – I

Q.1 Three cards are drawn from a well shuffled pack of 52 playing cards. What is the probability that:

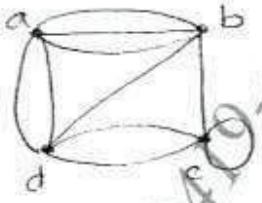
i) all the three cards drawn are red cards?
ii) all the three cards drawn bear number?
iii) all the three cards drawn belong to same suit?

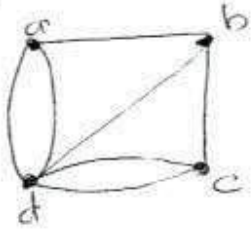
Q.2 How many permutations of the letter ABCDEFG contain

i) the string BCD?
ii) the string BA and GF?
iii) the strings CBA and BED?

Q.3 Represent the following graphs by using:

i) adjacency matrix ii) adjacency list





240419-e-mgt-delhi

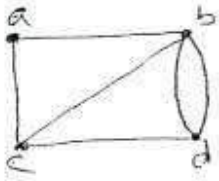
Q.4 Following table shows p.d.f. of discrete random variable X. Find:

i) E(X) ii) V(X) iii) Standard Deviation.

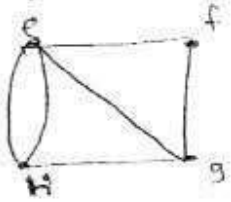
X = x	1	2	3	4	5	6
P (X = x)	0.14	0.18	0.24	0.20	0.18	0.08

Q.5 Explain : i) Utilities problem ii) Traveling salesman problem.

Q.6 Check whether the following graphs are isomorphic :



G



H.

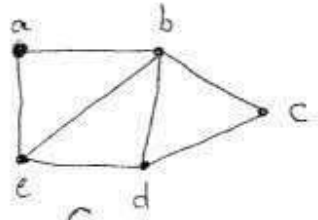
P.T.O.

Q.7 Write short notes on **ANY THREE** of the following:

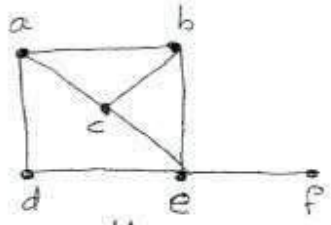
a) Binomial distribution
b) Planer graphs
c) Types of graphs
d) Coloring of graphs
e) Sheep cabbage problem

SECTION – II

Q.8 a) Determine whether the following graphs have:
i) Hamilton circuit ii) Hamilton path



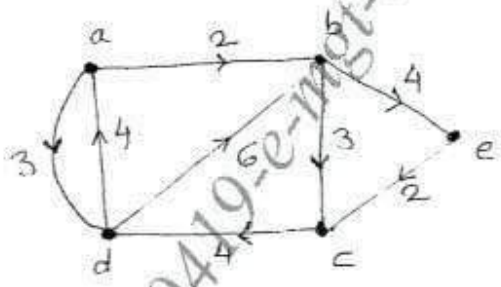
G



H

b) Explain : i) Complete graphs ii) Pigeonhole principle

Q.9 Explain Warshall's algorithm. Use Warshall's algorithm to find the shortest path from each vertex to each of the other vertex from the following graph:



240419-e-mst-delhi

Q.10 a) A product is manufactured on three machines A, B and C. 30% of total products are manufactured on machine A, 35% of total products are manufactured on Machine B and remaining products are manufactured on machine C. The probabilities that the manufactured product is defective on these machines are 2% , 3 % and 5 % respectively. A manufactured product is selected at random from the lot and is found to be defective. What is the probability that,
i) the product is manufactured on machine A?
ii) the product is manufactured on machine C?

b) When three fair coins are tossed simultaneously, what is the probability that
i) all three coins show same result?
ii) at least two coins show head?

* * * *

Bharati Vidyapeeth Deemed University,
Institute of Management and Research
(BVIMR), New Delhi
1st Internal Examination (September, 2018)

Course – BCA Semester - V

Subject–Combinatorics and Graph Theory Course Code -

Max. Marks: 40 Max. Time: 2 Hours

Instructions (if any):- (accounting, mathematics regarding use of Calculator, if required)

Q. 1 Attempt any five questions. [5 x 2]

a) Define Mutually Exclusive Events

b) Define Exhaustive Events

c) Define Conditional Probability

d) Define Complete Bipartite graph

e) Define Subgraphs

f) Define Combinations

g) Define Permutations

h) Define Graph

Q. 2 Attempt any two questions. [2 x 5]

a) Find the number of different 8-letter arrangements that can be made from the letters of the word "DAUGHTER" so that
   a. All vowels occur together
   b. All vowels do not occur together

b) A committee of 3 persons is to be constituted from a group of 2 men and 3 women. In how many ways can this be done? How many of these committees would consist of 1 man and 2 women?

c) What is the number of ways of choosing 4 cards from a pack of 52 playing cards? In how many of these:

   a. Four cards are of the same suit
   b. Four cards belong to four different suits
   c. Two are red cards and two are black cards
   d. Cards are of same colour

Q.3 Attempt any two questions.
[2 x 5]

a) Assume that each born child is equally likely to be a boy or a girl. If a family has two children, what is the conditional probability that both are girls given that (i) the youngest is a girl; (ii) at least one is a girl ?

b) Given three identical boxes I, II and III, each containing two coins. In box I, both coins are gold coins, in box II, both are silver coins and in box III, there is one gold and one silver coin. A person chooses a box at random and takes out a coin. If the coin is of gold, what is the probability that the other coin in the box is also of gold?

c) Let A and B be independent events with $P(A) = 0.3$ and $P(B) = 0.4$. Find
(i) $P(A \cap B)$ (ii) $P(A \cup B)$ (iii) $P(A|B)$ (iv) $P(B|A)$

Q.4 Attempt any one question.
[1 x 5]

a) In a hostel, 60% of the students read Hindi newspaper, 40% read English newspaper and 20% read both Hindi and English newspapers. A student is selected at random.
(i) Find the probability that she reads neither Hindi nor English newspapers ii. If she reads Hindi newspaper, find the probability that she reads English newspaper. iii. If she reads English newspaper, find the probability that she reads Hindi newspaper.

b) Find the probability distribution of number of doublets in three throws of a pair of dice. Find the expected value and standard deviation.

Bharati Vidyapeeth Deemed University,
Institute of Management and Research
(BVIMR), New Delhi
2nd Internal Examination (October, 2016)

Course – BCA | Semester - V

Subject–Combinatorics and Graph Theory | Course Code -

Max. Marks: 40 | Max. Time: 2 Hours

Instructions (if any):- (accounting, mathematics regarding use of Calculator, if required)

Q. 1 Attempt any five questions. Answer in 50 words. [5 x 2]

a) Define Bipartite Graph.
b) Define Walks, Paths and Circuits.
c) Define Weighted Graph.
d) Define Colouring of Graph.
e) Define Directed Graphs.
f) Define Hamiltonian Circuit.
g) Define Euler Circuit.
h) Define Subgraphs.

Q. 2 Attempt any two question. Answer in 200 words. [2 x 5]

a) Explain Fleury's Algorithm.
b) Explain all the conditions for Hamiltonian Circuit.
c) Explain the concept of Trees.

Q.3 Attempt any two questions. Answer in 200 words [2 x 5]

a) A graph G has the following adjacency matrix. Verify whether it is connected

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

b) Find the graph that has the following adjacency matrix

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

c) Find the adjacency matrix of the graph shown below

Q.4 Attempt any one. Answer in 600 words [10 x 1]

a) Explain Dijkstra's Shortest Path Algorithm with examples.

b) Discuss two algorithms to find a minimal spanning rue for a weighted graph 'G' with examples.

Internal Back Log Examination

Course – BCA Semester - V

Subject – Combinatorics and Graph Theory Course Code -

Max. Marks: 40 Max. Time: 2 Hours

Bharati Vidyapeeth Deemed University,
Institute of Management and Research
(BVIMR), New Delhi

Instructions (if any):- (accounting, mathematics regarding use of Calculator, if required)

Q. 1 Attempt any five questions. Answer in 50 words. [5 x 2]

a) Define Isomorphic graph.
b) Define complete graph.
c) Write a note on colouring of graph.
d) Define rooted tree.
e) Define sub graph.
f) Define permutations.
g) Define directed graph.
h) Define combinations.

Q. 2 Attempt any two question. Answer in 200 words. [2 x 5]

a) Explain eulerian graph and hamiltonion graph.
b) Explain bridges of konigsberge.
c) Explain farmer-wolf-goat-cabbage problem.

Q.3 Attempt any two questions. Answer in 200 words [2 x 5] a) Explain dijkstra's algorithm.

b) Prove that $C(2n,2) = 2C(n,2) + n^2$
c) Find "n" in each of the following equations:-
   i. $P(n,2) = 72$
   ii. $P(n,4) = 840$
   iii. $2P(n,2) + 50 = P(2n,2)$

Q.4 Attempt any one. Answer in 600 words [10 x 1]

a) In a hostel, 60% of the students read Hindi newspaper, 40% read English newspaper and 20% read both Hindi and English newspapers. A student is selected at random.

i. Find the probability that she reads neither Hindi nor English newspapers

ii. If she reads Hindi newspaper, find the probability that she reads English newspaper. iii. If she reads English newspaper, find the probability that she reads Hindi newspaper.

b) Find the probability distribution of number of doublets in three throws of a pair of dice. Find the expected value and standard deviation.

Checklist for Coursepacks

- ☐ Title page should be standardized bearing title of subject, course, course code, semester, year of batch (see sample attached)
  - Name of the instructors teaching the course
  - Name of course leader
- ☐ Forwarding by HOD bearing his/her signature for approval by Director
- ☐ Logo of BVIMR, name of the institution, address
- ☐ Warning "strictly for internal use" must be printed on the front title page.
- ☐ Table of content bearing
  - Serial no.
  - Contents
  - Page no.
- ☐ Copy of latest syllabus of course as specified by university
- ☐ Lesson plan bearing
  - Introduction to course
  - Course objectives
  - Learning outcomes
- ☐ List of topics/ modules with content
- ☐ Evaluation criteria
  - CES evaluation description
  - Recommended text books & reference books
  - Internet resources
  - Swayan courses
- ☐ Session plan bearing
  - Session number
  - Topic
  - Readings/case required
  - Pedagogy followed
  - Learning outcome
- ☐ Contact details of instructors along with profile
- ☐ Main body of course pack having reading material, exercises, case studies, pages for notes
- ☐ University question papers (preferably last five years including latest university paper)
- ☐ Internal question papers (internal-I-05 papers), (Internal-II-05 papers with latest last year papers)

Note: Include question paper of same subject of old syllabus if required to cover up five years papers.

Declaration by Faculty

I **Aanchal** and **Amit Grover**, **Designation** **Visiting Faculty** Teaching **Graph theory** subject in **BCA Afternoon and Morning,** course **Vth** sem have incorporated all the necessary pages section/quotations papers mentioned in this check list above.

Aanchal