SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF SCIENCE AND HUMANITIES

DEPARTMENT OF MATHEMATICS

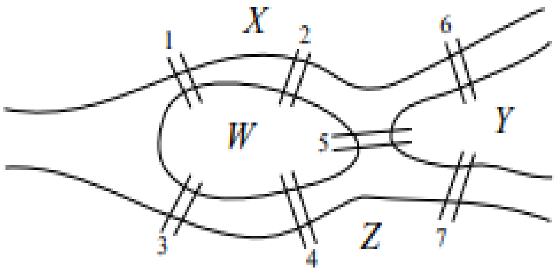
UNIT – I – Introduction to Graph Theory – SMT 1505

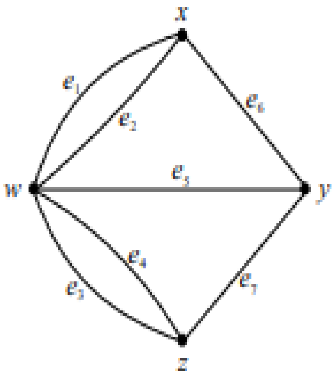
UNIT I

INTRODUCTION TO GRAPH THEORY

1 The Königsberg Bridge Problem

The city of Königsberg was located on the Pregel river in Prussia. The river divided the city into four separate landmasses, including the island of Kneiphopf. These four regions were linked by seven bridges as shown in the diagram. Residents of the city wondered if it were possible to leave home, cross each of the seven bridges exactly once, and return home. The Swiss mathematician Leonhard Euler (1707-1783) thought about this problem and the method he used to solve it is considered by many to be the birth of graph theory.





Exercise 1.1. See if you can find a round trip through the city crossing each bridge exactly once, or try to explain why such a trip is not possible.

The key to Euler's solution was in a very simple abstraction of the puzzle. Let us redraw our diagram of the city of Königsberg by representing each of the land masses as a vertex and representing each bridge as an edge connecting the vertices corresponding to the land masses. We now have a graph that encodes the necessary information. The problem reduces to finding a "closed walk" in the graph which traverses each edge exactly once, this is called an Eulerian circuit. Does such a circuit exist?

2 Fundamental Definitions

We will make the ideas of graphs and circuits from the Königsberg Bridge problem more precise by providing rigorous mathematical definitions.

A **graph** G is a triple consisting of a **vertex set** $V(G)$, an **edge set** $E(G)$, and a relation that associates with each edge, two vertices called its **endpoints** (not necessarily distinct).

Graphically, we represent a graph by drawing a point for each vertex and representing each edge by a curve joining its endpoints.

For our purposes all graphs will be **finite graphs**, i.e. graphs for which $V(G)$ and $E(G)$ are finite sets, unless specifically stated otherwise.

Note that in our definition, we do not exclude the possibility that the two endpoints of an edge are the same vertex. This is called a **loop**, for obvious reasons. Also, we may have **multiple edges**, which is when more than one edge shares the same set of endpoints, i.e. the edges of the graph are not uniquely determined by their endpoints.

A **simple graph** is a graph having no loops or multiple edges. In this case, each edge e in $E(G)$ can be specified by its endpoints u, v in $V(G)$. Sometimes we write $e = uv$.

When two vertices u, v in $V(G)$ are endpoints of an edge, we say u and v are **adjacent**.

A **path** is a simple graph whose vertices can be ordered so that two vertices are adjacent if and only if they are consecutive in the ordering. A path which begins at vertex u and ends at vertex v is called a u, v-path.

A **cycle** is a simple graph whose vertices can be cyclically ordered so that two vertices are adjacent if and only if they are consecutive in the cyclic ordering.

We usually think of paths and cycles as subgraphs within some larger graph.

1.1 WHAT IS A GRAPH ? DEFINITION

A graph G consists of a set of objects V = $\{v_1, v_2, v_3, ...\}$ called **vertices** (also called **points** or **nodes**) and other set E = $\{e_1, e_2, e_3, ...\}$ whose elements are called **edges** (also called **lines** or **arcs**).

The set V(G) is called the **vertex set** of G and E(G) is the **edge set.**

Usually the graph is denoted as **G = (V, E)**

Let G be a graph and $\{u, v\}$ an edge of G. Since $\{u, v\}$ is 2-element set, we may write $\{v, u\}$ instead of $\{u, v\}$. It is often more convenient to represent this edge by uv or vu.

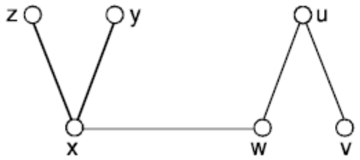
If $e = uv$ is an edge of a graph G, then we say that u and v are **adjacent** in G and that e joins u and v. (We may also say that each that of u and v is adjacent to or with the other).

For example :

A graph G is defined by the sets

$$V(G) = \{u, v, w, x, y, z\} \text{ and } E(G) = \{uv, uw, wx, xy, xz\}.$$

Now we have the following graph by considering these sets.



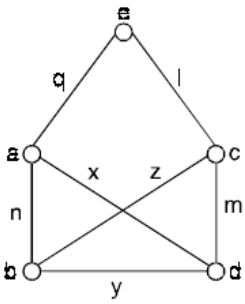
Every graph has a diagram associated with it. The vertex *u* and an edge *e* are **incident** with each other as are *v* and *e*. If two distinct edges say *e* and *f* are **incident** with a common vertex, then they are adjacent edges.

A graph with *p*-vertices and *q*-edges is called a **(*p, q*) graph.**

The (1, 0) graph is called **trivial graph.**

In the following figure the vertices *a* and *b* are adjacent but *a* and *c* are not. The edges *x* and *y* are adjacent but *x* and *z* are not.

Although the edges *x* and *z* intersect in the diagram, their intersection is not a vertex of the graph.



Examples :

(1) Let V = {1, 2, 3, 4} and E = {{1, 2}, {1, 3}, {3, 2}. {4, 4}}.
Then G(V, E) is a graph.

(2) Let V = {1, 2, 3, 4} and E = {{1, 5}, {2, 3}}.
Then G(V, E) is not a graph, as 5 is not in V.

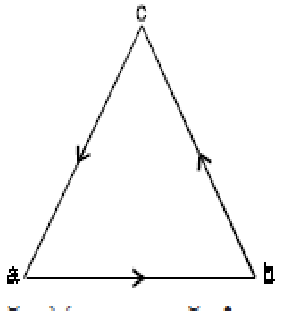
1.2.1. Directed graph

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.

Figure 1(a) is an example of a directed graph.



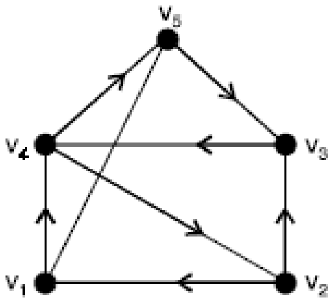
Suppose $e = (u, v)$ is a directed edge in a digraph, then (i) u is called the **initial vertex** of e and v is the terminal vertex of e

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

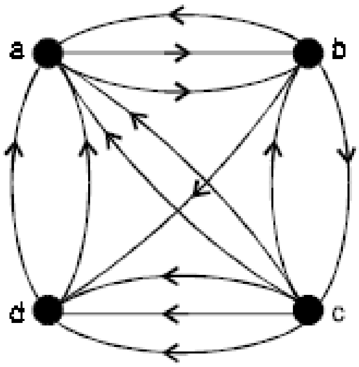
(iii) u is adjacent to v, and v is adjacent from u.

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

Example



(*i*)



(*ii*)

Solution. (*i*)	in-degree $v_1 = 2$,	out-degree $v_1 = 1$
	in-degree $v_2 = 2$,	out-degree $v_2 = 2$
	in-degree $v_3 = 2$,	out-degree $v_3 = 1$
	in-degree $v_4 = 2$,	out-degree $v_4 = 2$
	in-degree $v_5 = 0$,	out-degree $v_5 = 3$
(*ii*)	in-degree $a = 6$,	out-degree $a = 1$
	in-degree $b = 1$,	out-degree $b = 5$
	in-degree $c = 2$,	out-degree $c = 5$
	in-degree $d = 2$,	out-degree $d = 2$.

1.3 BASIC TERMINOLOGIES

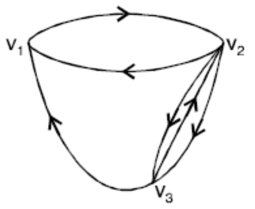
1.3.1 Loop : An edge of a graph that joins a node to itself is called **loop** or **self loop.**

i.e., a loop is an edge (v_i, v_j) where $v_i = v_f$.

1.3.2. Multigraph

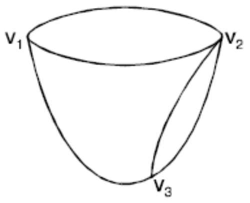
In a multigraph no loops are allowed but more than one edge can join two vertices, these edges are called **multiple edges** or parallel edges and a graph is called **multigraph.**

Two edges (v_i, v_j) and (v_f, v_r) are parallel edges if $v_i = v_r$ and v_j, v_f.



Directed multigraph

Fig. 2(*a*)



Un-directed multigraph

Fig. 2(*b*)

In Figure 1.2(*a*), there are two parallel edges associated with v_2 and v_3.

In Figure 1.2(*b*), there are two parallel edges joining nodes v_1 and v_2 and v_2 and v_3.

1.3.3. Pseudo graph

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

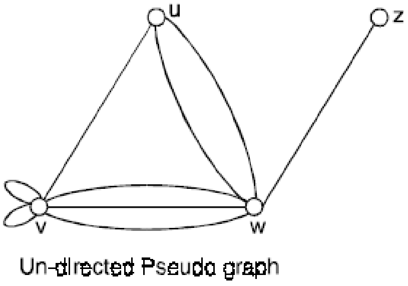


Fig. 3(*a*)

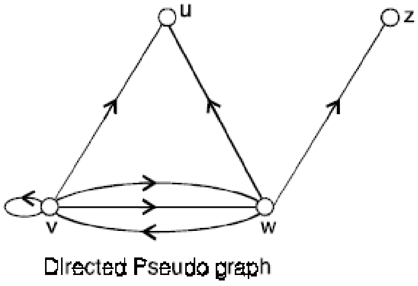


Fig. 3(*b*)

1.3.4. Simple graph

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.**

Figure 1.1(*a*) and (*b*) represents simple undirected and directed graph because the graphs do not contain loops and the edges are all distinct.

1.3.5. Finite and Infinite graphs

A graph with finite number of vertices as well as a finite number of edges is called a **finite graph.** Otherwise, it is an **infinite graph.**

1.4 DEGREE OF A VERTEX

The number of edges incident on a vertex v_i with **self-loops counted twice** (is called the **degree of a vertex** v_i and is denoted by $\mathbf{deg}_G(v_i)$ or **deg** v_i or $d(v_i)$.

The degrees of vertices in the graph G and H are shown in Figure 4(*a*) and 4(*b*).

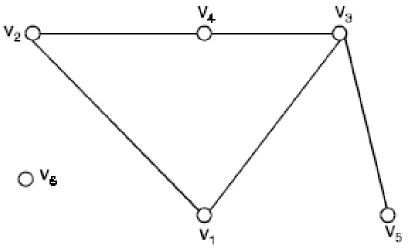


Fig. 4(*a*)

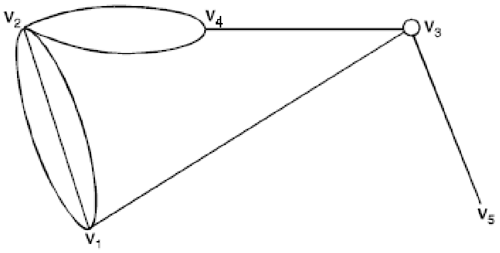


Fig. 4(*b*)

In G as shown in Figure 4(*a*),

$\deg_G(v_2) = 2 = \deg_G(v_4) = \deg_G(v_1)$, $\deg_G(v_3) = 3$ and $\deg_G(v_5) = 1$ and

In H as shown in Figure 4(*b*),

$\deg_H(v_2) = 5$, $\deg_H(v_4) = 3$, $\deg_H(v_3) = 5$, $\deg_H(v_1) = 4$ and $\deg_H(v_5) = 1$.

The degree of a vertex is some times also referred to as its **valency.**

1.5 ISOLATED AND PENDENT VERTICES

1.5.1. Isolated vertex

A vertex having **no incident edge** is called an **isolated vertex.**

In other words, isolated vertices are those with zero degree.

1.5.2. Pendent or end vertex

A vertex of **degree one,** is called a **pendent vertex** or an **end vertex.**

In the above Figure, v_5 is a pendent vertex.

1.5.3. In degree and out degree

In a graph G, the out degree of a vertex v_i of G, denoted by out $\deg_G(v_i)$ or $\deg_G^+(v_i)$, is the number of edges beginning at v_i and the in degree of v_i, denoted by in $\deg_G$ **(v_i)** or $\deg_G^{-1}$ **(v_i),** is the number of edges ending at v_i.

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.

1.6 THE HANDSHAKING THEOREM 1.1

If G = (v, E) be an undirected graph with e edges.

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

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

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.

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

$$\Rightarrow \quad 2e - \sum_{v_i \in U} \deg_G(v_1) = \sum_{v_i \in W} \deg_G(v_1) \qquad ...(1)$$

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

$$\text{Therefore, from (1) } \sum_{v_i \in W} \deg_G(v_i) \text{ is even}$$

$\therefore$ The no. of odd vertices in G is even.

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.

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

$$\Rightarrow \quad 2e - \sum_{v_i \in U} \deg_G (v_1) = \sum_{v_i \in W} \deg_G (v_1) \qquad \text{...(1)}$$

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

Therefore, from (1) $\sum_{v_i \in W} \deg_G (v_i)$ is even

$\therefore$ The no. of odd vertices in G is even.

Theorem 1.2. If V = $\{v_1, v_2, \ldots\ldots v_n\}$ is the vertex set of a non directed graph G,

$$\text{then } \sum_{i=1}^{n} \deg (v_i) = 2 \,|\, E \,|$$

If G is a directed graph, then $\sum_{i=1}^{n} \deg^+ (v_i) = \sum_{i=1}^{n} \deg^- (v_i) = |\, E \,|$

Proof : Since when the degrees are summed.

Each edge contributes a count of one to the degree of each of the two vertices on which the edge is incident.

Corollary (1) : In any non directed graph there is an even number of vertices of odd degree.

Proof : Let W be the set of vertices of odd degree and let U be the set of vertices of even degree.

$$\text{Then } \sum_{v \in V(G)} \deg (v) = \sum_{v \in W} \deg (v) + \sum_{v \in U} \deg (v) = 2 \,|\, E \,|$$

Certainly, $\sum_{v \in U} \deg (v)$ is even,

Hence $\sum_{v \in W} \deg (v)$ is even,

Implying that | W | is even.

Corollary (2) : If $k = \delta(G)$ is the minimum degree of all the vertices of a non directed graph G, then

$$k \,|\, V \,| \leq \sum_{v \in V(G)} \deg (v) = 2 \,|\, E \,|$$

In particular, if G is a k-regular graph, then

$$k \,|\, V \,| = \sum_{v \in V(G)} \deg (v) = 2 \,|\, E \,|.$$

Example 1 *Determine the number of edges in a graph with 6 vertices, 2 of degree 4 and 4 of degree 2. Draw two such graphs.*

Solution. Suppose the graph with 6 vertices has e number of edges. Therefore by Handshaking lemma

$$\sum_{i=1}^{6} \deg(v_i) = 2e$$

$$\Rightarrow \quad d(v_1) + d(v_2) + d(v_3) + d(v_4) + d(v_5) + d(v_6) = 2e$$

Now, given 2 vertices are of degree 4 and 4 vertices are of degree 2.

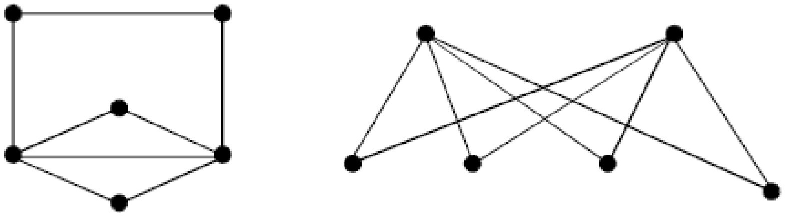
Hence the above equation,

$$(4 + 4) + (2 + 2 + 2 + 2) = 2e$$

$$\Rightarrow \qquad 16 = 2e \qquad \Rightarrow \qquad e = 8.$$

Hence the number of edges in a graph with 6 vertices with given condition is 8.

Two such graphs are shown below in Figure (11).



Example 2 *How many vertices are needed to construct a graph with 6 edges in which each vertex is of degree 2.*

Solution. Suppose these are P vertices in the graph with 6 degree. Also given the degree of each vertex is 2.

By handshaking lemma,

$$\sum_{i=1}^{P} \deg(v_i) = 2q = 2 \times 6$$

$$\Rightarrow \quad d(v_1) + d(v_2) + \ldots\ldots + d(v_n) = 12$$

$$\Rightarrow \quad 2 + 2 + \ldots\ldots + 2 = 12$$

$$\Rightarrow \quad 2P = 12 \qquad \Rightarrow \quad P = 6 \text{ vertices are needed.}$$

Example 3

It is possible to draw a simple graph with 4 vertices and 7 edges ? Justify.

Solution. In a simple graph with P-vertices, the maximum number of edges will be $\frac{P(P-1)}{2}$.

Hence a simple graph with 4 vertices will have at most $\frac{4 \times 3}{2}$ = 6 edges.

Therefore, the simple graph with 4 vertices cannot have 7 edges.

Hence such a graph does not exist.

1.7 TYPES OF GRAPHS

Some important types of graph are introduced here.

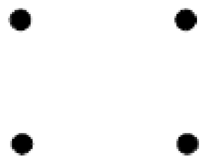
1.7.1. Null graph

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 Figure (13), Note that each vertex of a null graph is isolated.



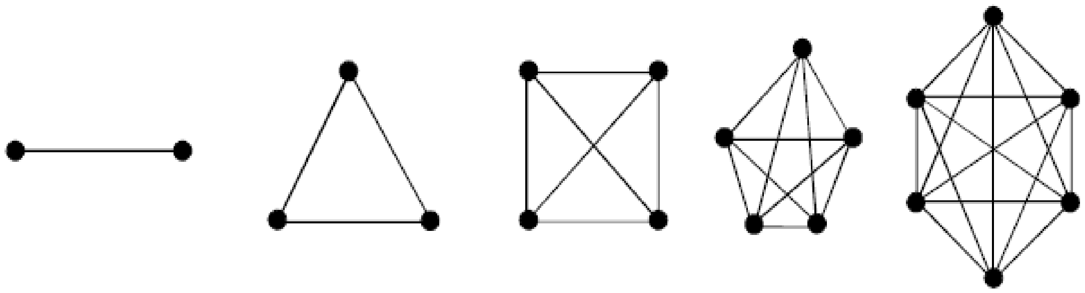
1.7.2. Complete graph

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 comple graph is usually denoted by $\mathbf{K}_n$. 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 show in Figure 14.



1.7.3. Regular graph

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 $\left(\frac{1}{2}\right)r\,n$ edges.

1.8 SUBGRAPH

A subgraph of G is a graph having all of its vertices and edges in G. If G_1 is a subgraph of G, then G is a super graph of G_1.

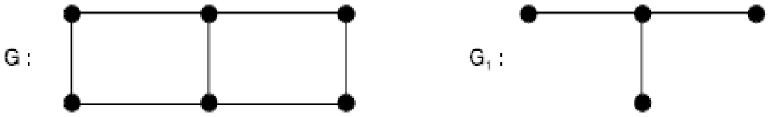


Fig. 19. G_1 is a subgraph of G.

In other words. If G and H are two graphs with vertex sets V(H), V(G) and edge sets E(H) and E(G) respectively such that $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$ then we call H as a subgraph of G or G as a supergraph of H.

1.8.1. Spanning subgraph

A spanning subgraph is a subgraph containing all the vertices of G.

In other words, if $V(H) \subset V(G)$ and $E(H) \subseteq E(G)$ then H is a proper subgraph of G and if V(H) = V(G) then we say that H is a spanning subgraph of G.

A spanning subgraph need not contain all the edges in G.

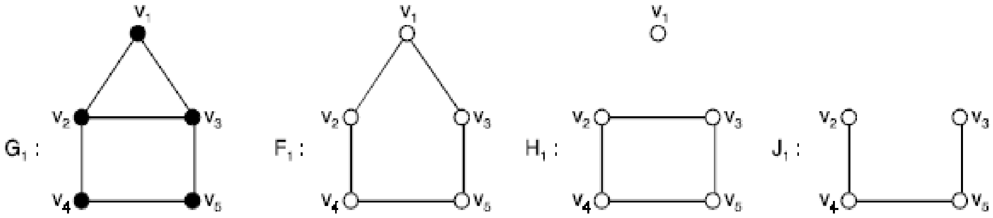


Fig. 20.

The graphs F_1 and H_1 of the above Fig. 20 are spanning subgraphs of G_1, but J_1 is not a spanning subgraph of G_1.

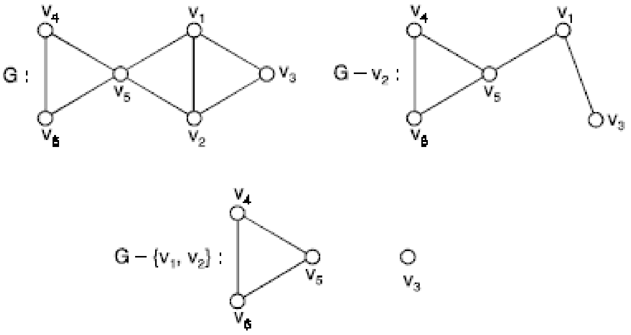
Since $V_1 \in V(G_1) - V(J_1)$. If E is a set of edges of a graph G, then G – E is a spanning subgraph of G obtained by deleting the edges in E from E(G).

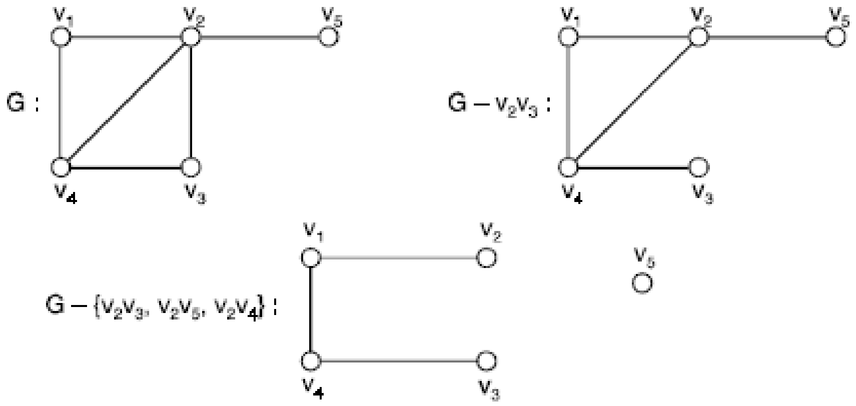
In fact, H is a spanning subgraph of G if and only if H = G – E, where E = E(G) – E(H). If *e* is an edge of a graph G, then we write G – *e* instead of G – {*e*}. For the graphs G_1, F_1 and H_1 of the Fig. 20, we have $F_1 = G_1 - v_2v_3$ and $H_1 = G_1 - \{v_1v_2, v_2v_3\}$.

1.8.2. Removal of a vertex and an edge

The removal of a vertex v_i from a graph G result in that subgraph $G - v_i$ of G containing of all vertices in G except v_i and all edges not incident with v_i. Thus $G - v_i$ is the maximal subgraph of G not containing v_i. On the otherhand, the removal of an edge x_j from G yields the spanning subgraph $G - x_j$ containing all edges of G except x_j.

Thus $G - x_j$ is the maximal subgraph of G not containing x_j.

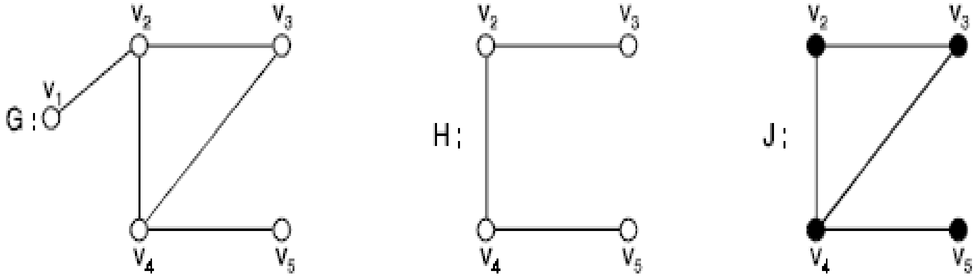




1.8.3. Induced subgraph

For any set S of vertices of G, the vertex induced subgraph or simply an induced subgraph <S> is the maximal subgraph of G with vertex set S. Thus two vertices of S are adjacent in <S> if and only if they are adjacent in G.

In other words, if G is a graph with vertex set V and U is a subset of V then the subgraph G(U) of G whose vertex set is U and whose edge set comprises exactly the edges of E which join vertices in U is termed as induced subgraph of G.



Here H is not an induced subgraph since $v_4v_1 \in E(G)$, but $v_4v_3 \notin E(H)$.

On the otherhand the graph J is an induced subgraph of G. Thus every induced subgraph of a graph G is obtained by deleting a subset of vertices from G.

Note : 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 subgraphs is equal to $(2^m - 1) \times 2^n$.

The number of spanning subgraphs is equal to 2^n.

1.9 Graph Isomorphism

Let $G_1 = (v_1, E_1)$ and $G_2 = (v_2, E_2)$ be two graphs. A function $f : v_1 \rightarrow v_1$ is called a graphs isomorphism if

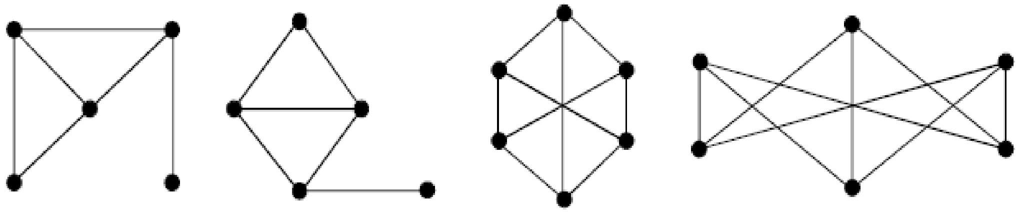
(*i*) f is one-to-one and onto.

(*ii*) for all $a, b \in v_1$, $\{a, b\} \in E_1$ if and only if $\{f(a), f(b)\} \in E_2$ when such a function exists, G_1 and G_2 are called isomorphic graphs and is written as $G_1 \cong G_2$.

In other words, two graphs G_1 and G_2 are said to be isomorphic to each other if there is a one-to-one correspondence between their vertices and between edges such that incidence relationship is preserve. Written as $G_1 \cong G_2$ or $G_1 = G_2$.

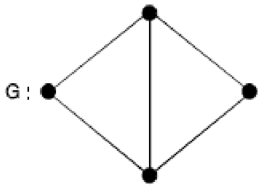
The necessary conditions for two graphs to be isomorphic are

1. Both must have the **same number of vertices**
2. Both must have the **same number of edges**
3. Both must have **equal number of vertices** with the **same degree.**
4. They must have the same degree sequence and same cycle vector $(c_1, ..., c_n)$, where c_i is the number of cycles of length i.

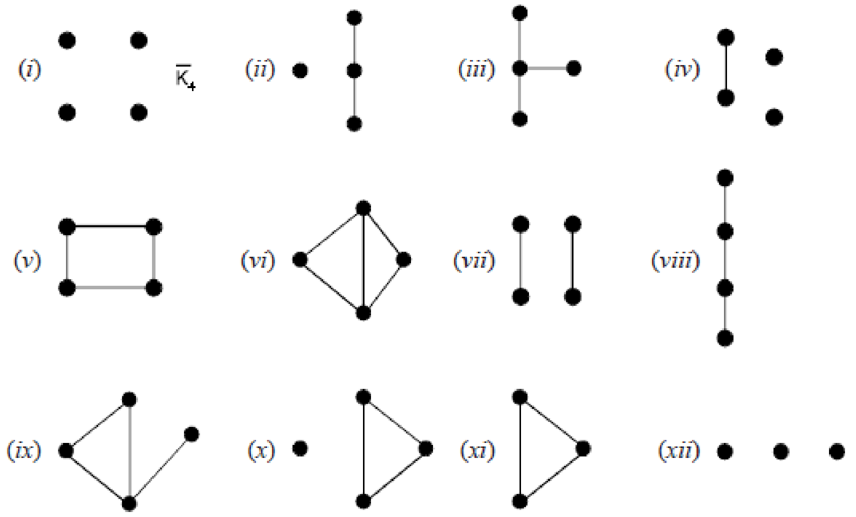


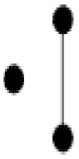
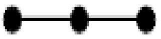
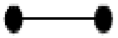
(*i*) (*ii*)

Example *Write down all possible non-isomorphic subgraphs of the following graphs G. How many of they are spanning subgraphs ?*



Solution. Its possible all (non-isomorphic) subgraphs are



(*xiii*)  (*xiv*)  (*xv*) (*xvi*)  (*xvii*)

of these graphs (*i*) to (*x*) are spanning subgraphs of G.

All the graphs except (*vi*) are proper subgraphs of G.

Theorem 1.3. *For any graph G with six points, G or $\overline{G}$ contains a triangle.*

Proof. Let v be a point of a graph G with six points. Since v is adjacent either in G or in $\overline{G}$ to the other five points of G.

We can assume without loss of generality that there are three points u_1, u_2, u_3 adjacent to v in G.

If any two of these points are adjacent, then they are two points of a triangle whose third point is v.

If no two of them are adjacent in G, then u_1, u_2 and u_3 are the points of a triangle in $\overline{G}$.

1.10 Walk, Path, Circuit

A walk is defined as a finite alternative sequence of vertices and edges, of the form

$$v_ie_j,\ v_{i+1}\ e_{j+1},\ v_{i+2},\ \ldots\ldots\ldots,\ e_kv_m$$

which begins and ends with vertices, such that

(*i*) each edge in the sequence is incident on the vertices preceding and following it in the sequence.

(*ii*) no edge appears more than once in the sequence, such a sequence is called a walk or a trial in G.

For example, in the graph shown in Figure 34, the sequences

$v_2e_4v_6e_5v_4e_3v_3$ and $v_1e_8v_2e_4v_6e_6v_5e_7v_5$ are walks.

Note that in the first of these, each vertex and each edge appears only once whereas in the second each edge appears only once but the vertex v_5 appears twice.

These walks may be denoted simply as $v_2v_6v_4v_3$ and $v_7v_2v_6v_5v_5$ respectively.

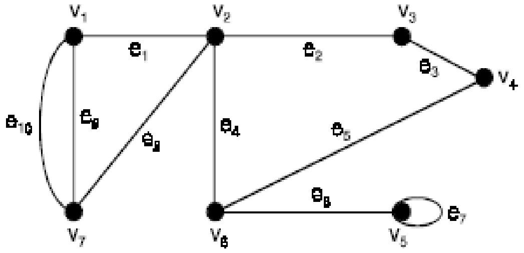


Fig. 34.

The vertex with which a walk begins is called the initial vertex and the vertex with which a walk ends is called the final vertex of the walk. The initial vertex and the final vertex are together called terminal vertices. Non-terminal vertices of a walk are called its internal vertices.

A walk having u as the initial vertex and v as the final vertex is called a walk from u to v or briefly a **$u - v$ walk.** A walk that begins and ends at the same vertex is called a **closed walk. In other words,** a closed walk is a walk in which the terminal vertices are coincident.

A walk that is not closed is called an **open walk.**

In other words, an open walk is a walk that begins and ends at two different vertices.

For example, in the graph shown in Figure 34.

$v_1e_9v_7e_8v_2e_1v_1$ is a closed walk and $v_5e_7v_5e_6v_6e_5v_4$ is an open walk.

In a walk, a vertex can appear more than once. An open walk in which no vertex appears more than once is called a **simple path** or a **path.**

For example, in the graph shown in Figure 34.

$v_6e_5v_4e_3v_3e_2v_2$ is a path whereas $v_5e_7v_5e_6v_6$ is an open walk but not a path.

A closed walk with atleast one edge in which no vertex except the terminal vertices appears more than once is called **a circuit** or a cycle.

For example, in the graph shown in Figure 34,

$v_1e_1v_2e_8v_7e_9v_1$ and $v_2e_4v_6e_5v_4e_3v_3e_2v_2$ are circuits.

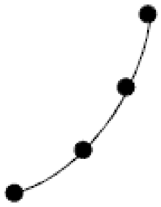
But $v_1e_9v_7e_8v_2e_4v_6e_5v_4e_3v_3e_2v_2e_1v_1$ is a closed walk but not a circuit.

Note : (*i*) In walks, path and circuit, no edge can appears more than once.

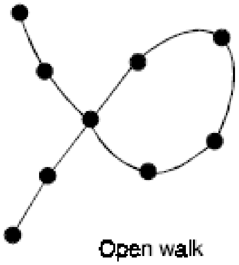
(*ii*) A vertex can appear more than once in a walk but not in a path.

(*iii*) A path is an open walk, but an open walk need not be a path.

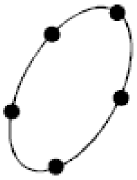
(*iv*) A circuit is a closed walk, but a closed walk need not be a circuit.



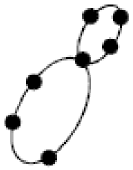
Open walk
which is a path



Open walk
which is not a path



Closed walk
which is a circuit



Closed walk
which is not a circuit.

The number of edges in a walk is called its length. Since paths and circuits are walks, it follows that the length of a path is the number of edges in the path and the length of a circuit is the number of edges in the circuit.

A circuit or cycle of length *k*, (with *k* edges) is called a *k*-circuit or a *k*-cycle. A *k*-circuit is called odd or even according as *k* is odd or even. A 3-cycle is called a triangle.

For example, in the graph shown in Figure 34,

The length of the open walk $v_6e_6v_5e_7v_5$ is 2

The length of the closed walk $v_1e_9v_7e_8v_2e_1v_1$ is 3

The length of the circuit $v_2e_4v_6e_5v_4e_3v_3e_2v_2$ is 4

The length of the path $v_6e_5v_4e_3v_3e_2v_2e_1v_1$ is 4

The circuit $v_1e_1v_2e_8v_7e_{10}v_1$ is a triangle.

Note : (*i*) A self-loop is a 1-cycle.

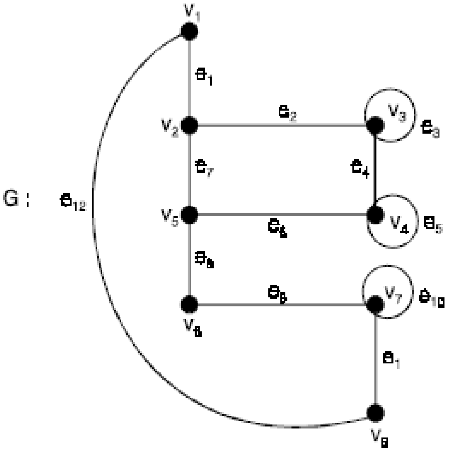
(*ii*) A pair of parallel edges form a cycle of length 2.

(*iii*) The edges in a 2-cycle are parallel edges.

Problem 1.76. *Write down all possible*

(*i*) *paths from v_1 to v_8* (*ii*) *Circuits of G and* (*iii*) *trails of length three.*

in G from v_3 to v_5 of the graph shown in Figure (35).



Solution.

(*i*) $P_1 : v_1e_{12}v_8,\ l(P_1) = 1$

$P_2 : v_1e_1v_2e_7v_5e_8v_6e_9v_7e_{11}v_8,\ l(P_2) = 5$

$P_3 : v_1e_1v_2e_2v_3e_4v_4e_6v_5e_8v_6e_9v_7e_{11}v_8,\ l(P_3) = 7$

These are the only possible paths from v_1 to v_8 in G.

(*ii*) $C_1 : v_1e_1v_2e_7v_5e_8v_6e_9v_7e_{11}v_8e_{12}v_1,\ l(C_1) = 6$

$C_2 : v_1e_1v_2e_2v_3e_4v_4e_6v_5e_8v_6e_9v_7e_{11}v_8e_{12}v_1,\ l(C_2) = 8$

$C_3 : v_2e_2v_3e_4v_4e_6v_5e_7v_2,\ l(C_3) = 4$

$C_4 : v_3e_3v_3,\ l(C_4) = 1$

$C_5 : v_4e_5v_4,\ l(C_5) = 1$

$C_6 : v_7e_7v_{10},\ l(C_6) = 1$

These are the only possible circuits of G.

$W_1 : v_3e_3v_3e_2v_2e_7v_5,\ l(W_1) = 3$

$W_2 : v_3e_3v_3e_4v_4e_6v_5,\ l(W_2) = 3$

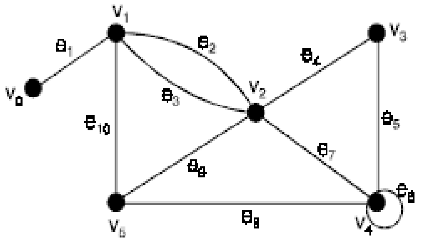
$W_3 : v_3e_4v_4e_5v_4e_6v_5,\ l(W_3) = 3.$

These are the only possible trails of length three from v_3 to v_5.

Problem 1.77. *In the graph below, determine whether the following are paths, simple paths, trails, circuits or simple circuits,*

(*i*) $v_0e_1v_1e_{10}v_5e_9v_2e_2v_1$ (*ii*) $v_4e_7v_2e_9v_5e_{10}v_1e_3v_2e_9v_5$

(*iii*) v_2 (*iv*) $v_5v_2v_3v_4v_4v_4v_5$.



Solution. (*i*) The sequence has a repeated vertex v_1 but does not have a repeated edge so it is a trail. It is not cycle or circuit.

(*ii*) The sequence has a repeated vertex v_2 and repeated edge e_9. Hence it is a path. It is not cycle or circuit.

(*iii*) It has no repeated edge, no repeated vertex, starts and ends at same vertex. Hence it is a simple circuit.

(*iv*) It is a circuit since it has no repeated edge, starts and ends at same vertex. It is not a simple circuit since vertex v_4 is repeated.

Theorem 1.14. *In a graph (directed or undirected) with n vertices, if there is a path from vertex u to vertex v then the path cannot be of length greater than (n – 1).*

Proof. Let $\pi : u, v_1, v_2, v_3, \ldots\ldots v_k, v$ be the sequence of vertices in a path u and v.

If there are m edges in the path then there are $(m + 1)$ vertices in the sequence.

If $m < n$, then the theorem is proved by default. Otherwise, if $m \geq n$ then there exists a vertex v_j in the path such that it appears more than once in the sequence

$$(u, v_1, \ldots\ldots, v_j \ldots\ldots, v_j, \ldots\ldots v_k, v).$$

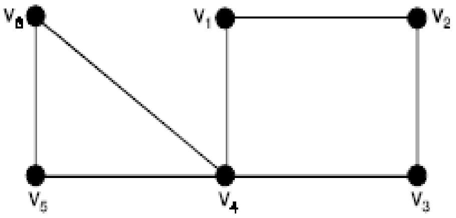
Deleting the sequence of vertices that leads back to the node v_j, all the cycles in the path can be removed.

The process when completed yields a path with all distinct nodes. Since there are n nodes in the graph, there cannot be more than n distinct nodes and hence $n - 1$ edges.

Problem 1.78. *For the graph shown in Figure, indicate the nature of the following sequences of vertices*

(*a*) $v_1v_2v_3v_2$ (*b*) $v_4v_1v_2v_3v_4v_5$ (*c*) $v_1v_2v_3v_4v_5$

(*d*) $v_1v_2v_3v_4v_1$ (*e*) $v_6v_5v_4v_3v_2v_1v_4v_6$



Solution. (*a*) Not a walk

(*b*) Open walk but not a path

(*c*) Open walk which is a path

(*d*) Closed walk which is a circuit

(*e*) Closed walk which is not a circuit.

Theorem 1.15. *Let $G = (V, E)$ be an undirected graph, with $a, b \in V$, $a \neq b$. If there exists a trail (in G) from a to b, then there is a path (in G) from a to b.*

Proof. Since there is an trail from a to b.

We select one of shortest length, say $\{a, x_1\}, \{x_1, x_2\}, \ldots\ldots, \{x_n, b\}$.

If this trail is not a path, we have the situation $\{a, x_1\}, \{x_1, x_2\}, \ldots\ldots, \{x_{k-1}, x_k\}, \{x_k, x_{k+1}\}, \{x_{k+1}, x_{k+2}\}, \ldots\ldots, \{x_{m-1}, x_m\}, (x_m, x_{m+1}\}, \ldots\ldots, \{x_n, b\}$,

where $k < m$ and $x_k = x_m$, possibly with $k = 0$ and $a(= x_0) = x_m$, or $m = n + 1$ and $x_k = b\ (= x_{n+1})$

But then we have a contradiction, because

$\{a,x_1\}, \{x_1, x_2\}, \ldots\ldots, \{x_{k-1}, x_k\}, \{x_m, x_{m+1}\}, \ldots\ldots, \{x_n, b\}$ is a shortest trail from a to b.

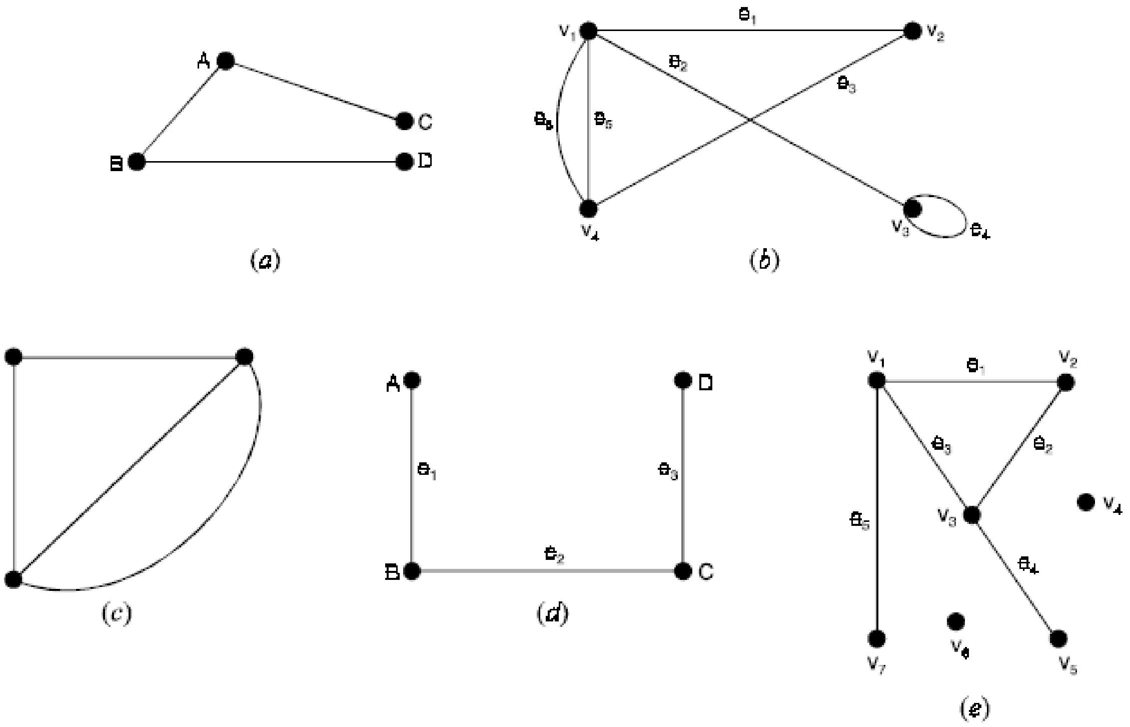
1.12 CONNECTED AND DISCONNECTED GRAPHS

A graph G is said to be a **connected** if every pair of vertices in G are connected. Otherwise, G is called a **disconnected** graph. Two vertices in G are said to be connected if there is at least one path from one vertex to the other.

In other words, a graph G is said to be connected if there is at least one path between every two vertices in G and disconnected if G has at least one pair of vertices between which there is no path.

A graph is **connected** if we can reach any vertex from any other vertex by travelling along the edges and disconnected otherwise.

For example, the graphs in Figure 30(*a, b, c, d, e*) are connected whereas the graphs in Figure 31(*a, b, c*) are disconnected.



(*a*) (*b*)

(*c*) (*d*) (*e*)

A complete graph is always connected, also, a null graph of more than one vertex is disconnected (see Fig. 32). All paths and circuits in a graph G are connected subgraphs of G.

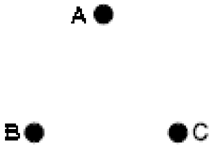


Fig. 32.

Every graph G consists of one or more connected graphs, each such connected graph is a subgraph of G and is called a component of G. A connected graph has only one component and a disconnected graph has two or more components.

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF SCIENCE AND HUMANITIES

DEPARTMENT OF MATHEMATICS

UNIT – II – EULERIAN AND HAMILTONIAN GRAPH – SMT 1505

UNIT II

EULERIAN AND HAMILTONIAN GRAPH

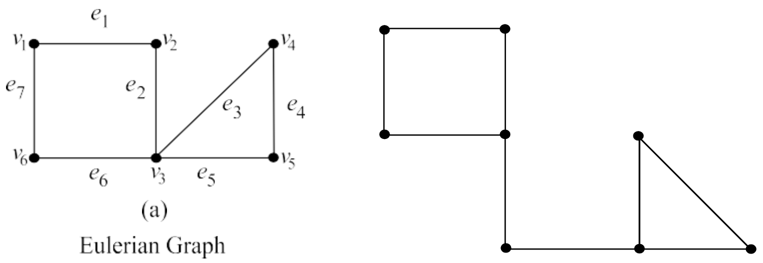
There are many games and puzzles which can be analysed by graph theoretic concepts. In fact, the two early discoveries which led to the existence of graphs arose from puz- zles, namely, the Konigsberg Bridge Problem and Hamiltonian Game, and these puzzles also resulted in the special types of graphs, now called Eulerian graphs and Hamiltonian graphs. Due to the rich structure of these graphs, they find wide use both in research and application.

2.1 Euler Graphs

A closed walk in a graph *G* containing all the edges of *G* is called an *Euler line* in *G*. A graph containing an Euler line is called an *Euler graph*.

We know that a walk is always connected. Since the Euler line (which is a walk) contains all the edges of the graph, an Euler graph is connected except for any isolated vertices the graph may contain. As isolated vertices do not contribute anything to the understanding of an Euler graph, it is assumed now onwards that Euler graphs do not have any isolated vertices and are thus connected.

Example Consider the graph shown in Figure 3.1. Clearly, v_1 e_1 v_2 e_2 v_3 e_3 v_4 e_4 v_5 e_5 v_3 v_6 e_7 v_1 in (a) is an Euler line, whereas the graph shown in (b) is non-Eulerian.



(a)
Eulerian Graph

(b)
Non-Eulerian Graph

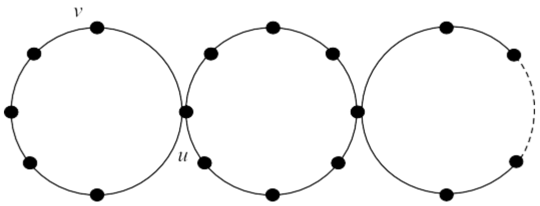
The following theorem due to Euler [74] characterises Eulerian graphs. Euler proved the necessity part and the sufficiency part was proved by Hierholzer [115].

Theorem 2.1 (Euler) A connected graph *G* is an Euler graph if and only if all vertices of *G* are of even degree.

Proof
Necessity Let G(V, E) be an Euler graph. Thus G contains an Euler line Z, which is a closed walk. Let this walk start and end at the vertex u ∈ V. Since each visit of Z to an

intermediate vertex v of Z contributes two to the degree of v and since Z traverses each edge exactly once, d(v) is even for every such vertex. Each intermediate visit to u contributes two to the degree of u, and also the initial and final edges of Z contribute one each to the degree of u. So the degree d(u) of u is also even.

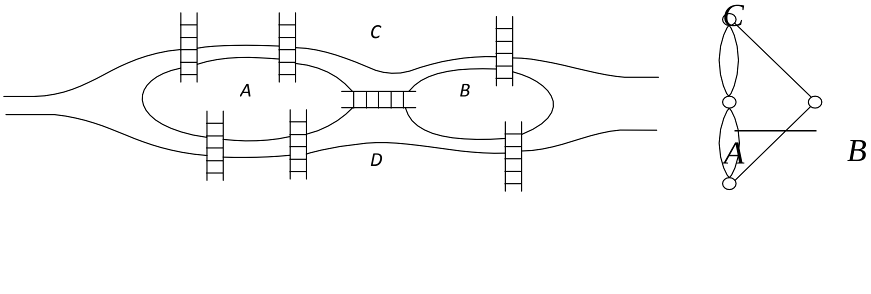
Second proof for sufficiency Assume that all vertices of *G* are of even degree. We con- struct a walk starting at an arbitrary vertex *v* and going through the edges of *G* such that no edge of *G* is traced more than once. The tracing is continued as far as possible. Since every vertex is of even degree, we exit from the vertex we enter and the tracing clearly cannot stop at any vertex but *v*. As *v* is also of even degree, we reach *v* when the tracing comes to an end. If this closed walk *Z* we just traced includes all the edges of *G*, then *G* is an Euler graph. If not, we remove from *G* all the edges in *Z* and obtain a subgraph Z^J of *G* formed by the remaining edges. Since both *G* and *Z* have all their vertices of even degree, the degrees of the vertices of Z^J are also even. Also, Z^J touches *Z* at least at one vertex say *u*, because *G* is connected. Starting from *u*, we again construct a new walk in Z^J. As all the vertices of Z^J are of even degree, therefore this walk in Z^J terminates at vertex *u*. This walk in Z^Jcombined with *Z* forms a new walk, which starts and ends at the vertex *v* and has more edges than *Z*. This process is repeated till we obtain a closed walk that traces all the edges of *G*. Hence *G* is an Euler graph (Fig. 3.2) ❑



Konigsberg Bridge Problem

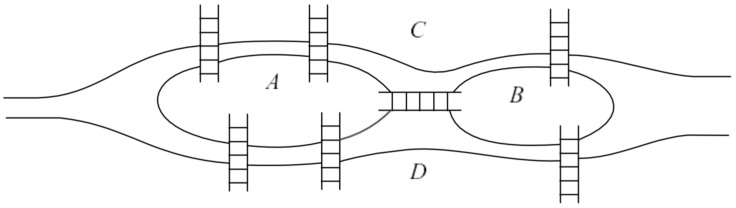
Two islands *A* and *B* formed by the Pregal river (now Pregolya) in Konigsberg (then the capital of east Prussia, but now renamed Kaliningrad and in west Soviet Russia) were connected to each other and to the banks *C* and *D* with seven bridges. The problem is to start at any of the four land areas, *A*, *B*, *C*, or *D*, walk over each of the seven bridges exactly once and return to the starting point.

Euler modeled the problem representing the four land areas by four vertices, and the seven bridges by seven edges joining these vertices. This is illustrated in Figure.

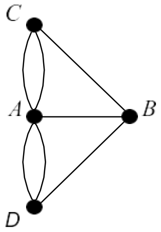


We see from the graph *G* of the Konigsberg bridges that not all its vertices are of even degree. Thus *G* is not an Euler graph, and implies that there is no closed walk in *G* con- taining all the edges of *G*. Hence it is not possible to walk over each of the seven bridges exactly once and return to the starting point.

Note Two additional bridges have been built since Euler’s day. The first has been built between land areas *C* and *D* and the second between the land areas *A* and *B*. Now in the graph of Konigsberg bridge problem with nine bridges, every vertex is of even degree and the graph is thus Eulerian. Hence it is now possible to walk over each of the nine bridges exactly once and return to the starting point .



Konigsberg Bridges



Underlying Graph

The following characterisation of Eulerian graphs is due to Veblen [254].

Theorem 2.2 A connected graph *G* is Eulerian if and only if its edge set can be decom- posed into cycles.
Proof Let G(V, E) be a connected graph and let G be decomposed into cycles. If k of these cycles are incident at a particular vertex v, then d(v) = 2k. Therefore the degree of every vertex of G is even and hence G is Eulerian. Conversely, let G be Eulerian. We show G can be decomposed into cycles. To prove this, we use induction on the number of edges. Since d(v) ≥ 2 for each v ∈ V, G has a cycle C. Then G−E(C) is possibly a disconnected graph, each of whose components C1, C2, ..., Ck is an even degree graph and hence Eulerian. By the induction hypothesis, each Ci is a disjoint union of cycles. These together with C provide a partition of E(G) into cycles.

Theorem 2.3 If *W* is a walk from vertex *u* to vertex *v*, then *W* contains an odd number of *u* − *v* paths.

Proof Let W be a walk which we consider as a graph in itself, and not as a subgraph of some other graph. Let u and v be initial and final vertices of the walk W. Clearly, d(u|W) and d(v|W) are odd, and d(w|W) is even, for every w ∈V(W)− {u, v}. We count the number of distinct u−v walks in W. These walks are the subgraphs of W. When we take a u − v walk by successively selecting the edges e_1, e_2, ..., e_s , initial vertex of e1being u and terminal vertex of es being v, for each edge there are an odd number of choices. The total number of such edges is the product of these odd numbers and is therefore odd. Now from these walks, we find the u−v paths. If a u−v walk W1 is not a path, then it contains one or more cycles. The traversal of these cycles in the two possible alternative directions (clockwise and anticlockwise) produces in all an even number of walks, all with the same edge set as W1. Omitting these even number of walks which are not paths from the total odd collection of u−v walks, gives an odd number of u−v paths. ❑

Toida [244] proved the necessity part and McKee [157] the sufficiency part of the next characterisation. The second proof of this result can be found in Fleischner [79], [80].

Theorem 2.4 A connected graph is Eulerian if and only if each of its edges lies on an odd number of cycles.

Proof

Necessity Let G be a connected Eulerian graph and let e = uv be any edge of G. Then G−e is a u−v walk W, and so G−e = W contains an odd number of u−v paths. Thus each of the odd number of u−v paths in W together with e gives a cycle in G containing e and these are the only such cycles. Therefore there are an odd number of cycles in G containing e.

Sufficiency Let G be a connected graph so that each of its edges lies on an odd number of cycles. Let v be any vertex of G and Ev = {e1, ..., ed } be the set of edges of G incident on v, then |Ev| = d(v) = d. For each i, 1 ≤ i ≤ d, let ki be the number of cycles of G containing ei . By hypothesis, each ki is odd. Let c(v) be the number of cycles of G containing v. Then clearly c(v) = 1 2 d ∑ i=1 ki implying that 2c(v) = d ∑ i=1 ki . Since 2c(v) is even and each ki is odd, d is even. Hence G is Eulerian.

Corollary 2.1 The number of edge−disjoint paths between any two vertices of an Euler graph is even.

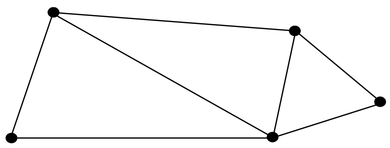
A consequence of Theorem 3.4 is the result of Bondy and Halberstam [37], which gives yet another characterisation of Eulerian graphs.

Corollary 2.2 A graph is Eulerian if and only if it has an odd number of cycle decompositions.

Proof In one direction, the proof is trivial. If G has an odd number of cycle decompositions, then it has at least one, and hence G is Eulerian. Conversely, assume that G is Eulerian. Let e ∈ E(G) and let C1, ..., Cr be the cycles containing e. By Theorem 3.4, r is odd. We proceed by induction on m = |E(G)|, with G being Eulerian. If G is just a cycle, then the result is true. Now assume that G is not a cycle. This means that for each i, 1 ≤ i ≤ r, by the induction assumption, Gi = G−E(Ci) has an odd number, say si , of cycle decompositions. (If Gi is disconnected, apply the induction assumption to each of the nontrivial components of Gi). The union of each of these cycle decompositions of Gi and Ci yields a cycle decomposition of G. Hence the number of cycle decompositions of G containing Ci is si , 1 ≤ i ≤ r. Let s(G) denote the number of cycle decompositions of G. Then s(G) ≡ r ∑ i=1 si ≡ r(mod 2) (since si ≡ 1(mod 2)) ≡ 1(mod 2).

Unicursal Graphs

An open walk that includes (or traces) all edges of a graph without retracing any edge is called a unicursal line or open Euler line. A connected graph that has a unicursal line is called a unicursal graph. Figure 3.6 shows a unicursal graph.



Unicursal graph

Clearly by adding an edge between the initial and final vertices of a unicursal line, we get an Euler line.

The following characterisation of unicursal graphs can be easily derived from Theorem 3.1.

Theorem 2.5 A connected graph is unicursal if and only if it has exactly two vertices of odd degree.

Proof Let G be a connected graph and let G be unicursal. Then G has a unicursal line, say from u to v, where u and v are vertices of G. Join u and v to a new vertex w of G to get a graph H. Then H has an Euler line and therefore each vertex of H is of even degree. Now, by deleting the vertex w, the degree of vertices u and v each get reduced by one, so that u and v are of odd degree.

Conversely, let u and v be the only vertices of G with odd degree. Join u and v to a new vertex w to get the graph H. So every vertex of H is of even degree and thus H is Eulerian.

Therefore, $G = H - w$ has a $u - v$ unicursal line so that G is unicursal. ❑

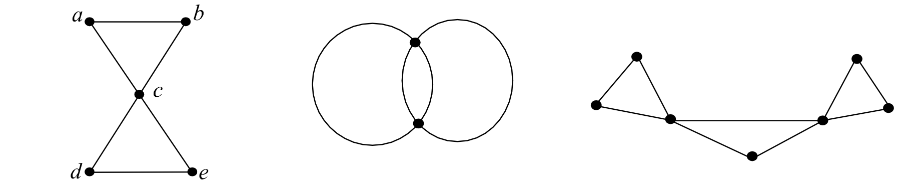
The following result is the generalisation of Theorem 3.5.

Theorem 2.6 In a connected graph G with exactly $2k$ odd vertices, there exists k edge disjoint subgraphs such that they together contain all edges of G and that each is a unicursal graph.

Proof Let G be a connected graph with exactly 2k odd vertices. Let these odd vertices be named v1, v2, ..., vk ; w1, w2, ..., wk in any arbitrary order. Add k edges to G between the vertex pairs (v1, w1), (v2, w2), ..., (vk , wk) to form a new graph H, so that every vertex of H is of even degree. Therefore H contains an Euler line Z. Now, if we remove from Z the k edges we just added (no two of these edges are incident on the same vertex), then Z is divided into k walks, each of which is a unicursal line. The first removal gives a single unicursal line, the second removal divides that into two unicursal lines, and each successive removal divides a unicursal line into two unicursal lines, until there are k of them. Hence the result.

Arbitrarily Traceable Graphs

An Eulerian graph G is said to be arbitrarily traceable (or randomly Eulerian) from a vertex v if every walk with initial vertex v can be extended to an Euler line of G. A graph is said to be arbitrarily traceable if it is arbitrarily traceable from every vertex (Fig. 3.7).



(a) Arbitrarily traceable graph from c

(b) Arbitrarily traceable graph from all vertices

(c) Euler graph, not arbitrarily traceable

The following characterisation of arbitrarily traceable graphs is due to Ore [174]. Such graphs were also characterised by Chartrand and White [56] .

Theorem 2.7 An Eulerian graph *G* is arbitrarily traceable from a vertex *v* if and only if every cycle of *G* passes through *v*.

Proof

Necessity Let the Eulerian graph G be arbitrarily traceable from a vertex v. Assume there is a cycle C not passing through v. Let H = G−E(C). Then every vertex of H has an even degree and the component of H containing v is Eulerian. This component of H can be traversed as an Euler line Z, starting and ending with v and contains all those edges of G which are incident at v. Clearly, this v−v walk cannot be extended to contain the edges of C also, contradicting that G contains v. Thus every cycle in G contains v. Sufficiency Let every cycle of the Eulerian graph G pass through the vertex v of G. We show that G is arbitrarily traceable from v. Assume, on the contrary, that G is not arbitrarily traceable from v. Then there is a v − v closed walk W of G containing all the edges of G incident with v and yet not containing all the edges of G. Let one such edge be incident at a vertex u on W. So every vertex of H = G − E(W) is of even degree and v is an isolated vertex of H and u is not. The component of H containing u is therefore Eulerian subgraph of G not passing through v, contradicting the assumption. Hence the result follows.

Corollary 2.3 Cycles are the only arbitrarily traceable graphs.

Sub-Eulerian Graphs

A graph *G* is said to be *sub-Eulerian* if it is a spanning subgraph of some Eulerian graph. The following characterisation of sub-Eulerian graphs is due to Boesch, Suffel and Tindell [28].

Theorem 2.8 A connected graph *G* is sub-Eulerian if and only if *G* is not spanned by a complete bipartite graph.

Proof

Necessity We prove that no spanning supergraph *H* of an odd complete bipartite graph *G* is Eulerian. Let $V_1 \cup V_2$ be the bipartition of the vertex set of *G*. Since degree of each vertex of *G* is odd, and *G* is complete bipartite, therefore $|V_1|$ and $|V_2|$ are odd. If H_1 is the induced subgraph of *H* on V_1, then at least one vertex, say *v*, of V_1 has even degree in H_1, since $|V_1|$

is odd. But then $d(v|H) = d(v|H) + |V_2|$, which is odd. Therefore *H* is not Eulerian.

Sufficiency Refer Boesch et. al., [28]. ❑

Super-Eulerian graphs

A non-Eulerian graph *G* is said to be *super-Eulerian* if it has a spanning Eulerian subgraph.

The following sufficient conditions for super-Eulerian graphs are due to Lesniak-Foster and Williams [148].

Theorem 2.9

If a graph G is such that n ≥ 6, δ ≥ 2 and d(u)+d(v) ≥ n−1, for every pair of non-adjacent vertices u and v, then G is super-Eulerian.

The following result is due to Balakrishnan and Paulraja [12].

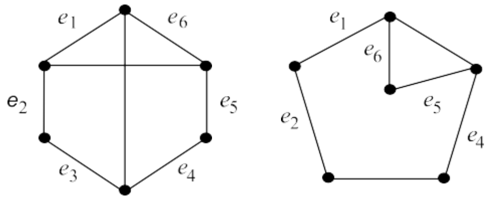
Theorem 2.10 If *G* is any connected graph and if each edge of *G* belongs to a triangle in *G*, then *G* has a spanning Eulerian subgraph.

Proof Since *G* has a triangle, *G* has a closed walk. Let *W* be the longest closed walk in *G*. Then *W* must be a spanning Eulerian subgraph of *G*. If not, there exists a vertex $v \notin W$ and *v* is adjacent to a vertex *u* of *W*. By hypothesis, *uv* belongs to a triangle, say *uvw*. If none of the edges of this triangle is in *W*, then $W \cup \{uv, vw, wu\}$ yields a closed walk longer than *W* (Fig. 3.8). If $uw \in W$, then $(W - uw) \cup \{uv, vw\}$ would be a closed walk longer than

W. This contradiction proves that *W* is a spanning closed walk in *G*. ❑

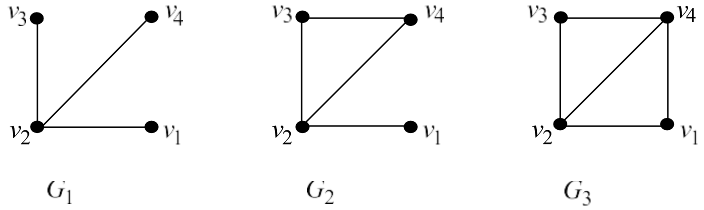
Hamiltonian Graphs

A cycle passing through all the vertices of a graph is called a *Hamiltonian cycle*. A graph containing a Hamiltonian cycle is called a *Hamiltonian graph*. A path passing through all the vertices of a graph is called a *Hamiltonian path* and a graph containing a Hamiltonian path is said to be *traceable*. Examples of Hamiltonian graphs are given in Figure .

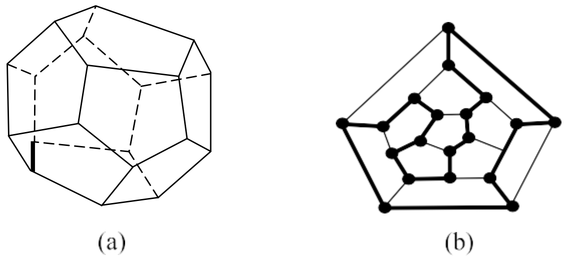


Hamiltonian Graphs

If the last edge of a Hamiltonian cycle is dropped, we get a Hamiltonian path. However, a non-Hamiltonian graph can have a Hamiltonian path, that is, Hamiltonian paths cannot always be used to form Hamiltonian cycles. For example, in Figure 3.10, G_1 has no Hamil- tonian path, and so no Hamiltonian cycle; G_2 has the Hamiltonian path $v_1v_2v_3v_4$, but has no Hamiltonian cycle, while G_3 has the Hamiltonian cycle $v_1v_2v_3v_4v_1$.



Hamiltonian graphs are named after Sir William Hamilton, an Irish Mathematician (1805−1865), who invented a puzzle, called the Icosian game, which he sold for 25 guineas to a game manufacturer in Dublin. The puzzle involved a dodecahedron on which each of the 20 vertices was labelled by the name of some capital city in the world. The aim of the game was to construct, using the edges of the dodecahedron a closed walk of all the cities which traversed each city exactly once, beginning and ending at the same city. In other words, one had essentially to form a Hamiltonian cycle in the graph corresponding to the dodecahedron. Figure 3.11 shows such a cycle.

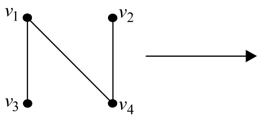


Dedecahedron and its graph shown with the Hamiltonian cycle

Clearly, the n-cycle C_n with n distinct vertices (and n edges) is Hamiltonian. Now, given any Hamiltonian graph G, the supergraph G^J (obtained by adding in new edges between non-adjacent vertices of G) is also Hamiltonian. This is because any Hamiltonian cycle in G is also a Hamiltonian cycle of G^J. For instance, K_n is a supergraph of an n-cycle and so K_n is Hamiltonian.

A multigraph or general graph is Hamiltonian if and only if its underlying graph is Hamiltonian, because if G is Hamiltonian, then any Hamiltonian cycle in G remains a Hamiltonian cycle in the underlying graph of G. Conversely, if the underlying graph of a graph G is Hamiltonian, then G is also Hamiltonian.

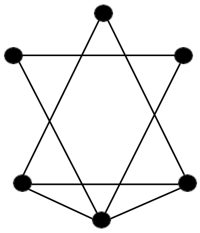
Let G be a graph with n vertices. Clearly, G is a subgraph of the complete graph K_n. From G, we construct step by step supergraphs of G to get K_n, by adding an edge at each step between two vertices that are not already adjacent (Fig. 3.12).



Now, let us start with a graph G which is not Hamiltonian. Since the final outcome of the procedure is the Hamiltonian graph K_n, we change from a non-Hamiltonian graph to a Hamiltonian graph at some stage of the procedure. For example, the non-Hamiltonian.

graph G_1 above is followed by the Hamiltonian graph G_2. Since supergraphs of Hamilto- nian graphs are Hamiltonian, once a Hamiltonian graph is reached in the procedure, all the subsequent supergraphs are Hamiltonian.

Definition: A simple graph G is called *maximal non-Hamiltonian* if it is not Hamiltonian and the addition of an edge between any two non-adjacent vertices of it forms a Hamilto- nian graph. For example, G_1 above is maximal non-Hamiltonian. Figure 3.13 shows a maximal non-Hamiltonian graph.



It follows from the above procedure that any non-Hamiltonian graph with n-vertices is a subgraph of a maximal non-Hamiltonian graph with n vertices.

The above procedure is used to prove the following sufficient conditions due to Dirac [68].

Theorem 2.11 (Dirac) If G is a graph with n vertices, where $n \geq 3$ and $d(v) \geq n/2$, for every vertex v of G, then G is Hamiltonian. Hamiltonian graph H in which $d(v) \geq n/2$, for every vertex of H.

Proof Assume that the result is not true. Then for some value $n \geq 3$, there is a non-graph K (i.e., with the same vertex set) of H, $d(v) \geq n/2$ for every vertex of K, since any

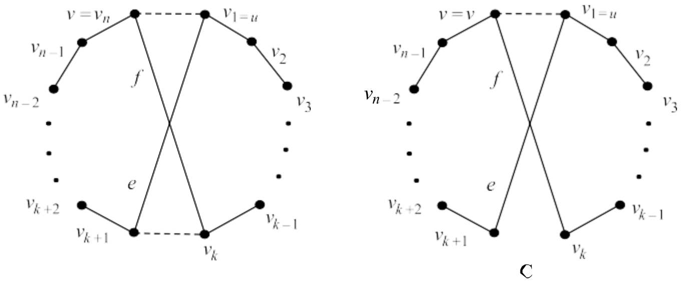
non-Hamiltonian graph G with n vertices and $d(v) \geq n/2$ for every v in G. Using this G, we proper supergraph of this form is obtained by adding more edges. Thus there is a maximal obtain a contradiction.

Clearly, G *f*= K_n, as K_n is Hamiltonian. Therefore there are non-adjacent vertices u and v in G. Let $G + uv$ be the supergraph of G by adding an edge between u and v. Since G is maximal non-Hamiltonian, $G + uv$ is Hamiltonian. Also, if C is a Hamiltonian cycle of $G + uv$, then C contains the edge uv, since otherwise C is a Hamiltonian cycle of G, which is not possible. Let this Hamiltonian cycle C be $u = v_1, v_2, \ldots, v_n = v, u$.

Now, let $S = \{v_i \in C$: there is an edge from u to v_{i+1} in $G\}$ and $T = \{v_j \in C$: there is an edge from v to v_j in $G\}$.

Then $v_n \notin T$, since otherwise there is an edge from v to $v_n = v$, that is a loop, which is impossible.

Also $v_n \not\in S$, (taking v_{n+1} as v_1), since otherwise we again get a loop from u to $v_1 = u$. Therefore, $v_n \in S \cup T$.



Let $|S|$, $|T|$ and $|S \cup T|$ be the number of elements in S, T and $S \cup T$ respectively. So

Therefore, $|S| = d(u)$. Similarly, $|T| = d(v)$. $|S \cup T| < n$. Also, for every edge incident with u, there corresponds one vertex v_i in S.

Now, if v_k is a vertex belonging to both S and T, there is an edge e joining u to v_{k+1} and an edge f joining v to v_k. This implies that $C^J = v_1, v_{k+1}, v_{k+2}, \ldots, v_n, v_k, v_{k-1}, \ldots, v_2, v_1$ is that there is no vertex v_k in $S \cap T$, so that $S \cap T = \Phi$. a Hamiltonian cycle in G, which is a contradiction as G is non-Hamiltonian. This shows

Thus $|S \cup T| = |S| + |T| - |S \cap T|$ gives $|S| + |T| = |S \cup T|$, so that $d(u) + d(v) < n$. This is a contradiction, because $d(u) \geq n/2$ for all u in G, and so $d(u) + d(v) \geq n/2 + n/2$ giving $d(u) + d(v) \geq n$. Hence the theorem follows. ❑

The following result is due to Ore [176].

vertices in G such that $d(u) + d(v) \geq n$. Let $G + uv$ denote the super graph of G obtained by joining u and v by an edge. Then G is Hamiltonian if and only if $G + uv$ is Hamiltonian.

Theorem 2.12 (Ore) Let G be a graph with n vertices and let u and v be non-adjacent

in G such that $d(u) + d(v) \geq n$. Let $G + uv$ be the super graph of G obtained by adding the ***Proof*** Let G be a graph with n vertices and suppose u and v are non-adjacent vertices edge uv. Let G be Hamiltonian. Then obviously $G + uv$ is Hamiltonian. Conversely, let $G + uv$ be Hamiltonian. We have to show that G is Hamiltonian. Then, as in Theorem 3.11, we get $d(u) + d(v) < n$, which contradicts the hypothesis that $d(u) + d(v) \geq n$. Hence G is Hamiltonian.

The following is the proof of Bondy [35] of Theorem 2.12, and this proof bears a close resemblance to the proof of Dirac's theorem given by Newman [170], but is more direct.

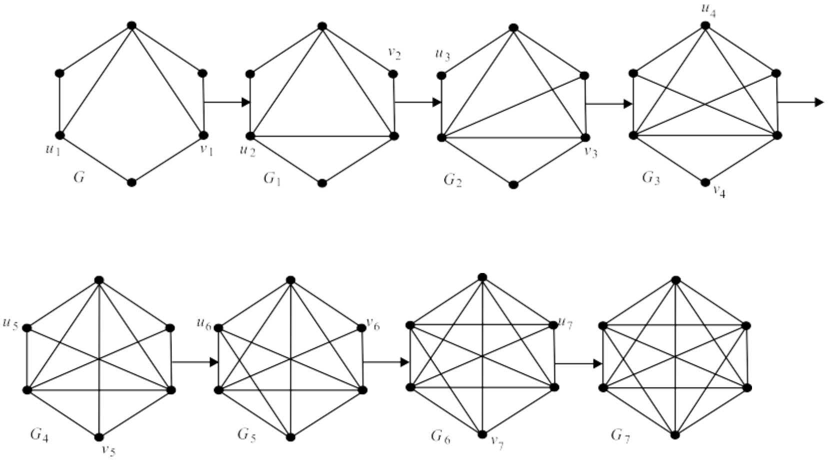
Proof (Bondy [35]) Consider the complete graph K on the vertex set of G in which the edges of G are coloured blue and the remaining edges of K are coloured red. Let C be a Hamiltonian cycle of K with as many blue edges as possible. We show that every edge of , in other words, that C is Hamiltonian cycle of G.

Suppose to the contrary, C has a red edge uu^- (where u^- is the successor of u on C). Consider the set S of vertices joined to u by blue edges (that is, the set of neighbours of u in G). The successor u^- of u on C must be joined by a blue edge to some vertex v^- of S^-, because if u^- is adjacent in C only to vertices $V - (S^- U \{u^-\})$, $d_G(u) + d_G(u^-) = |N_G(u)| + |N_G(u^-)| \leq |S| + (|V| - |S^-| - 1) = |V(G)| - 1$, contradicting the hypothesis that $d_G(u) + d_G(u^-) \geq |V(G)|$, u and u^- being non-adjacent in G. But now the cycle C obtained from C by exchanging the edges uu^- and vv^- has more blue edges than C, which is a contradiction. ❑

and v_1 in G such that $d(u_1) + d(v_1) \geq n$, join u_1 and v_1 by an edge to form the super graph G_1. Now, if there are two non-adjacent vertices u_2 and v_2 in G_1 such that $d(u_2) + d(v_2) \geq n$,

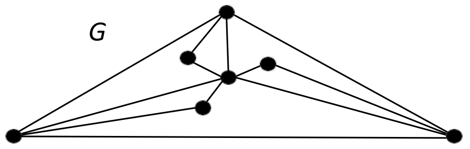
Definition: Let G be a graph with n vertices. If there are two non-adjacent vertices u_1 join u_2 and v_2 by an edge to form supergraph G_2. Continue in this way, recursively joining. The final supergraph thus obtained is called the *closure* of G and is denoted by $c(G)$. pairs of non-adjacent vertices whose degree sum is at least n until no such pair remains.

The example in Figure 3.15 illustrates the closure operation.



vertices u and v with $d(u) + d(v) \geq n$. Therefore the closure procedure can be carried out in We observe in this example that there are different choices of pairs of non-adjacent several different ways and each different way gives the same result.

In the graph shown in below Figure, $n = 7$ and $d(u) + d(v) < 7$, for any pair u, v of adjacent vertices. Therefore, $c(G) = G$.



The importance of $c(G)$ is given in the following result due to Bondy and Chvatal [36].

Theorem 2.13 A graph G is Hamiltonian if and only if its closure $c(G)$ is

Hamiltonian. ***Proof*** Let $c(G)$ be the closure of the graph G. Since $c(G)$ is a supergraph of G, therefore, if G is Hamiltonian, then $c(G)$ is also Hamiltonian.

Conversely, let $c(G)$ be Hamiltonian. Let $G, G_1, G_2, \ldots, G_{k-1}, G_k = c(G)$ be the sequence

of graphs obtained by performing the closure procedure on G. Since $c(G) = G_k$ is obtained from G_{k-1} by setting $G_k = G_{k-1} + uv$, where u, v is a pair of non adjacent vertices in G_{k-1} with $d(u)+ d(v) \geq n$, therefore it follows that G_{k-1} is Hamiltonian. Similarly G_{k-2}, so G_{k-3},

. .., G_1 and thus G is Hamiltonian. ❑

Corollary 2.4 Let G be a graph with n vertices with $n \geq 3$. If $c(G)$ is complete, then G is Hamiltonian.

There can be more than one Hamiltonian cycle in a given graph, but the interest lies in the edge-disjoint Hamiltonian cycles. The following result gives the number of edge-disjoint Hamiltonian cycles in a complete graph with odd number of vertices.

The next result involving degrees give the sufficient conditions for a graph to be Hamiltonian.

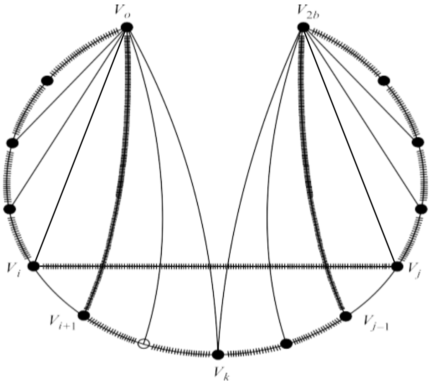
Theorem 2.14 (Nash-Williams) Every k-regular graph on $2k + 1$ vertices is Hamiltonian.

Proof Let G be a k-regular graph on $2k + 1$ vertices. Add a new vertex w and join it by an edge to each vertex of G. The resulting graph H on $2k + 2$ vertices has $\delta = k + 1$. Thus by Theorem 3.15 (A), H is Hamiltonian. Removing w from H, we get a Hamiltonian path,

say $v_0v_1 \ . \ . \ .v_{2k}$. then $v_{i-1}v_{2k} \in E$, since $d(v_0) = d(v_{2k}) = k$. Assume that G is not Hamiltonian, so that (a) if $v_0v_i \in E$, then $v_{i-1}v_{2k} \notin E$, (b) if $v_0v_i \notin E$,

The following cases arise.

Case (i) v_0 is adjacent to $v_1, v_2, \ldots, v_k$, and v_{2k} is adjacent to $v_k, v_{k+1}, , v_{2k-1}$. Thenthere is



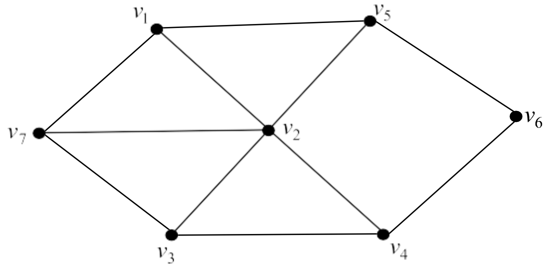
an i with $1 \leq i \leq k$ such that v_i is not adjacent to some v_j for $0 \leq j \leq k(j \not\equiv i)$. But $d(v_i) = k$. So v_i is adjacent to v_j for some j with $k + 1 \leq j \leq 2k - 1$. Then the cycle C given by $v_iv_{i-1} \ldots v_0v_{i+1} \ldots v_{j-1}v_{2k}v_{2k+1}\, v_j$ is a Hamiltonian cycle of G (Fig 3.18).

Case (ii) There is an i with $1 \leq i \leq 2k - 1$ such that $v_{i+1}v_0 \in E$, but $v_iv_0 \notin E$. Then by (b), $v_{i-1}v_{2k} \in E$. Thus G contains the $2k$-cycle $v_{i-1}v_{i-2} \ldots v_0v_{i+1}$. Renaming the $2k$-cycle C as $u_1u_2 \ldots u_{2k}$ and let u_0 be the vertex of G not on C. Then u_0 cannot be adjacent to two consecutive vertices on C and hence u_0 is adjacent to every second vertex on C, say $u_1, u_3, \ldots, u_{2k-1}$. Replacing u_{2i} by u_0, we obtain another maximum cycle C^J of G and hence u_{2i} must be adjacent to $u_1, u_3, \ldots, u_{2k-1}$. But then u_1 is adjacent to $u_0, u_2, \ldots, u_{2k}$, implying $d(u_1) \geq k + 1$. This is a contradiction and hence G is Hamiltonian. ❑

Pancyclic Graphs

Definition: A graph G of order $n(\geq 3)$ is *pancyclic* if G contains all cycles of lengths from 3 to n. G is called *vertex-pancyclic* if each vertex v of G belongs to a cycle of every length A, $3 \leq$ A $\leq n$.

Example Clearly, a vertex-pancyclic graph is pancyclic. However, the converse is not true. Figure displays a pancyclic graph that is not vertex-pancyclic.



The result of pancyclic graphs was initiated by Bondy [34], who showed that Ore's sufficient condition for a graph G to be Hamiltonian (Theorem 6.2.5) actually implies much more. Note that if $\delta \geq {}^{n}$, then $m \geq {}^{n2}$. The proof of the following result due to Thomassen can be found in Bollobas [29].

Exercises

Prove that the wheel W_n is Hamiltonian for every $n \geq 2$, and n-cube Q_n is Hamiltonian for each $n \geq 2$.

If G is a k-regular graph with $2k-1$ vertices, then prove that G is Hamiltonian.
Show that if a cubic graph G has a spanning closed walk, then G is Hamiltonian.

If $G = G(X, Y)$ is a bipartite Hamiltonian graph, then show that $|X| = |Y|$.
Prove that for each $n \geq 1$, the complete tripartite graph $K_{n,\ 2n,\ 3n}$ is Hamiltonian, but

$K_{n,\ 2n,\ 3n+1}$ is not Hamiltonian.

Prove that a graph G with $n \geq 3$ vertices is randomly traceable if and only if it is randomly Hamiltonian.

Find the closure of the graph given in Figure 3.2. Is it Hamiltonian?

Does there exist an Eulerian graph with

i. an even number of vertices and an odd number of edges,
ii. and odd number of vertices and an even number of edges.

Draw such a graph if it exists.

Characterise graphs which are both Eulerian and Hamiltonian.

Characterise graphs which possess Hamiltonian paths but not Hamiltonian cycles.

Characterise graphs which are unicursal but not Eulerian.

Give an example of a graph which is neither pancyclic nor bipartite, but whose *n*-closure is complete.

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF SCIENCE AND HUMANITIES

DEPARTMENT OF MATHEMATICS

UNIT – III – TREES – SMT 1505

UNIT III

TREES

INTRODUCTION

Kirchhoff developed the theory of trees in 1847, in order to solve the system of simultaneous linear equations which give the current in each branch and arround each circuit of an electric network.

In 1857, Cayley discovered the important class of graphs called **trees** by considering the changes of variables in the differential calculus. Later, he was engaged in enumerating the isomers of saturated **hydro carbons $C_n H_{2n+2}$** with a given number of n of carbon atoms as

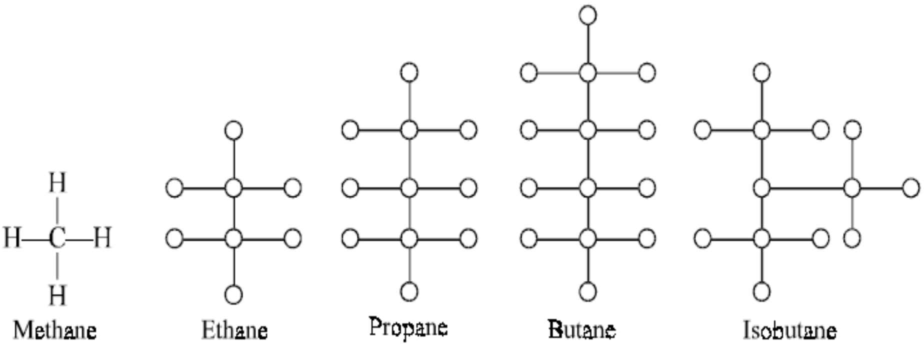


Fig. 3.1.

3.1 TREE

3.1.1. Acyclic graph

A graph is acyclic if it has no cycles.

3.1.2. Tree

A tree is a connected acyclic graph.

3.1.3. Forest

Any graph without cycles is a forest, thus the components of a forest are trees.

The tree with 2 points, 3 points and 4-points are shown below :

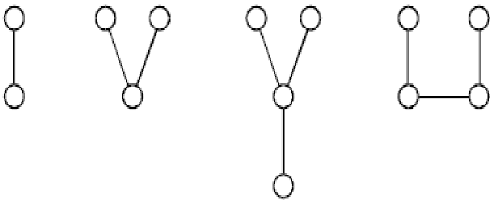


Fig. 3.2.

Note :

(1) Every edge of a tree is a bridge.

i.e., every block of G is acyclic.

Conversely, every edge of a connected graph G is a bridge, then G is a tree.

(2) Every vertex of G (tree) which is not an end vertex is neccessarily a cut-vertex.

(3) Every nontrivial tree G has at least two end vertices.

3.2 SPANNING TREE

A spanning tree is a spanning subgraph, that is a tree.

3.2.1. Branch of tree

An edge in a spanning tree T is called a branch of T.

3.2.2. Chord

An edge of G that is not in a given spanning tree is called a chord.

Note :

(1) The branches and chords are defined only with respect to a given spanning tree.

(2) An edge that is a branch of one spanning tree T_1 (in a graph G) may be chord, with respect to another spanning tree T_2.

3.3 ROOTED TREE

A rooted tree T with the vertex set V is the tree that can be defined recursively as follows :

T has a specially designated vertex $v_1 \in V$, called the **root** of T. The subgraph of T_1 consisting of the vertices $V - \{v\}$ is partitionable into subgraphs.

$T_1, T_2, \ldots\ldots, T_r$ each of which is itself a rooted tree. Each one of these *r*-rooted tree is called a **subtree of v_1.**

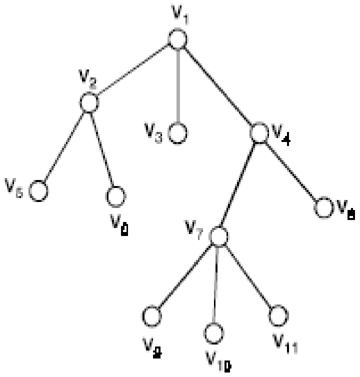


Fig. 3.3. A rooted tree.

3.3.1. Co tree

The cotree T* of a spanning tree T in a connected graph G is the spanning subgraph of G containing exactly those edges of G which are not in T. The edges of G which are not in T* are called its twigs.

For example :

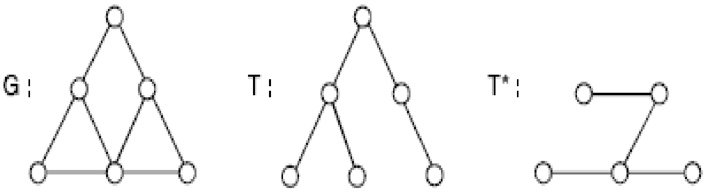


Fig. 3.4.

3.4 BINARY TREES

A binary tree is a rooted tree where each vertex v has atmost two subtrees ; if both subtrees are present, one is called a left subtree of v and the other right-subtree of v. If only one subtree is present, it can be designated either as the left subtree or right subtree of v.

In other words, a binary tree is a 2-ary tree in which each child is designated as a left child or right child.

In a binary tree e very vertex has two children or no children.

Properties : (Binary trees) :

(1) The number of vertices n in a complete binary tree is always odd. This is because there is exactly one vertex of even degree, and remaining $n - 1$ vertices are of odd degree. Since from theorem (*i.e.,* the number of vertices of odd degree is even), $n - 1$ is even. Hence n is odd.

(2) Let P be the number of end vertices in a binary tree T. Then $n - p - 1$ is the number of vertices of degree 3. The number of edges in T is

$$\frac{1}{2}[p+3(n-p-1)+2]=n-1 \qquad \text{or} \qquad p=\frac{n+1}{2} \qquad \ldots(1)$$

(3) A non end vertex in a binary tree is called an **internal vertex.** It follows from equation (1) that the number of internal vertices in a binary is one less than the number of end vertices.

(4) In a binary tree, a vertex v_i is said to be at **level** l_i if v_i is at a distance l_i from the root. Thus the root is at level O.

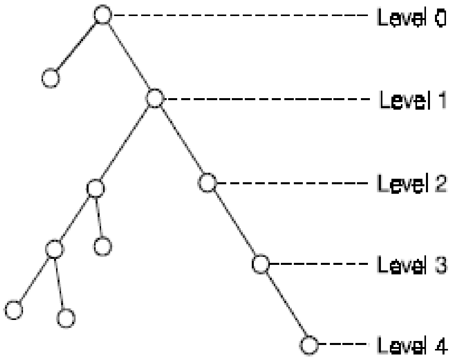


Fig. 3.5. 13-vertices, 4-level binary tree.

The maximum numbers of vertices possible in a k-level binary tree is $2^0+2^1+2^2+\ldots..+2^k \geq n$,

The maximum level, $l_{\max}$ of any vertex in a binary tree is called the **height** of the tree.

On the other hand, to construct a binary tree for a given n such that the farthest vertex is as for as possible from the root, we must have exactly two vertices at each level, except at the O level.

Hence max $l_{\max} = \dfrac{n-1}{2}$.

For example,

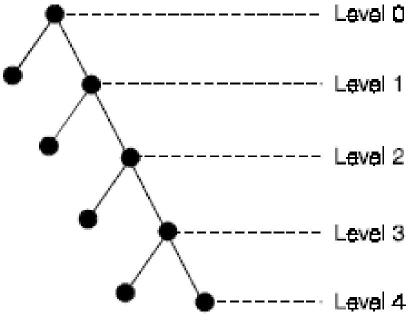


Fig. 3.6.

Max $l_{\max} = \dfrac{9-1}{2} = 4$

The minimum possible height of n-vertex binary tree is min $l_{\max} = [\log_2(n+1)-1]$

In analysis of algorithm, we are generally interested in computing the sum of the levels of all end vertices. This quantity, known as the **path length** (or external path length) of a tree.

3.4.1. Path length of a binary tree

It can be defined as the sum of the path lengths from the root to all end vertices.

For example,

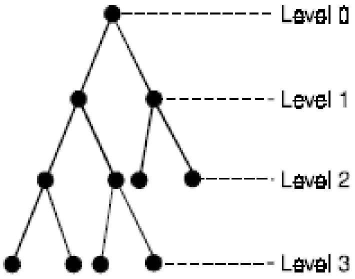


Fig. 3.7.

Here the sum is 2 + 2 + 3 + 3 + 3 + 3 = 16 is the path length of a given above binary tree.

The path length of the binary tree is often directly related to the executive time of an algorithm.

3.4.2. Binary tree representation of general trees

There is a straight forward technique for converting a general tree to a binary tree form. The algorithm has two easy steps :

Step 1 :

Insert edges connecting siblings and delete all of a parents edges to its children except to its left most off spring.

Step 2 :

Rotate the resulting diagram 45° to distinguish between left and right subtrees.

For example,

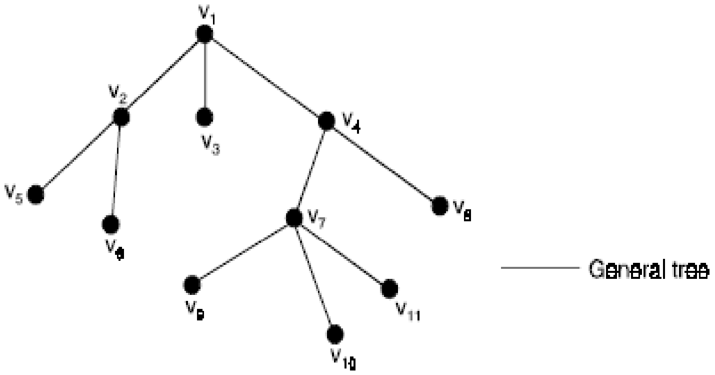


Fig. 3.8.

Here v_2, v_3 and v_4 are siblings to the parent v_1, now apply the steps given above we have a binary tree as shown here.

Here v_2, v_3 and v_4 are siblings to the parent v_1, now apply the steps given above we have a binary tree as shown here.

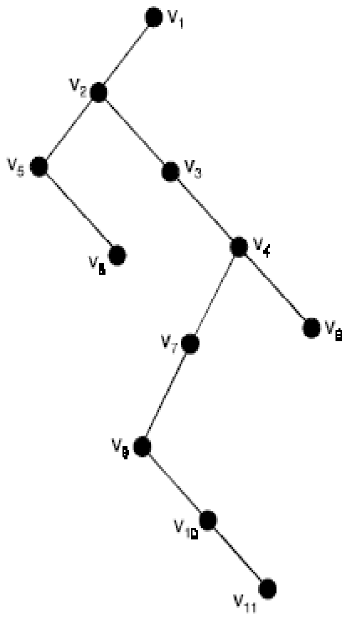


Fig. 3.9.

Theorem 3.1. *A(p, q) graph is a tree if and only if it is acyclic and $p = q + 1$ or $q = p - 1$.*

Proof. If G is a tree, then it is acyclic.

By definition to verify the equality $p = q + 1$.

We employ induction on p.

For $p = 1$, the result is trivial.

Assume, then that the equality $p = q + 1$ holds for all (p, q) trees with $p \geq 1$ vertices.

Let G_1 be a tree with $p + 1$ vertices.

Let v be an end-vertex of G_1.

The graph $G_2 = G_1 - v$ is a tree of order p, and so $p = |E(G_2)| + 1$.

Since G_1 has one more vertex and one more edge than that of G_2.

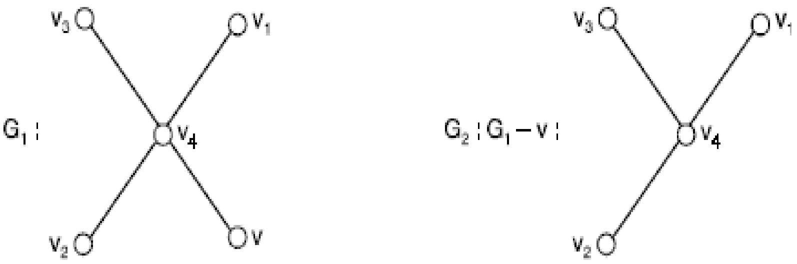


Fig. 3.10.

Fig. 3.10.

$$|V(G_1)| = p + 1 = (|E(G_2)| + 1) + 1$$

$$= |E(G_1)| + 1$$

$$\therefore \quad |V(G_1)| = |E(G_1)| + 1.$$

Conversely : Let G be an acyclic (p, q) graph with $p = q + 1$.

To show G is a tree, we need only verify that G is connected. Denote by G_1, G_2, ..., G_k, the components of G, where $k \geq 1$.

Furthermore, let G_i be a (p_i, q_i) graph.

Since each G_i is a tree, $p_i = q_i + 1$.

Hence $$p - 1 = q = \sum_{i=1}^{k} q_i$$

$$= \sum_{i=1}^{k} (p_i - 1) = p - k$$

$\Rightarrow \quad p - 1 = p - k \quad \Rightarrow \quad k = 1$ and G is connected.

Hence, (p, q) graph is a tree.

Hence the proof.

Corollary : A forest G of vertices p has $p - k$ edges where k is the number of components.

Theorem 3.2. *A (p, q) graph G is a tree if and only if G is connected and $p = q + 1$.*

Proof. Let G be a (p, q) tree.

By definition of G, it is connected and by theorem : *i.e.,* A(p, q) graph is a tree if and only if it is acyclic and $p = q + 1$), $p = q + 1$.

Conversely : We assume G is connected (p, q) graph with $p = q + 1$.

It is sufficient to show that G is acyclic.

If G contains a cycle C and e is an edge of C, then G − e is a connected graph with p vertices having $p - 2$ edges.

This is impossible by the definition (*i.e.*, A(p, q) graph has $q < p - 1$ then G is disconnected).

This contradicts our assumption.

Hence G is connected.

Theorem 3.3. *A complete n-ary tree with m internal nodes contains n × m + 1 nodes.*

Proof. Since there are m internal nodes, and each internal node has n descendents, there are $n \times m$ nodes in three other than root node.

Since there is one and only one root node in a tree, the total number of nodes in the tree will $n \times m + 1$.

Problem 3.1. *A tree has five vertices of degree 2, three vertices of degree 3 and four vertices of degree 4. How many vertices of degree 1 does it have ?*

Solution. Let x be the number of nodes of degree one.

Thus, total number of vertices

$$= 5 + 3 + 4 + x = 12 + x.$$

The total degree of the tree $= 5 \times 2 + 3 \times 3 + 4 \times 4 + x = 35 + x$

Therefore number of edges in the three is half of the total degree of the tree.

If G = (V, E) be the tree, then, we have

$$|\,V\,| = 12 + x \text{ and } |\,E\,| = \frac{35 + x}{2}$$

In any tree, $|\,E\,| = |\,V\,| - 1$.

Therefore, we have $\dfrac{35 + x}{2} = 12 + x - 1$

$\Rightarrow \quad 35 + x = 24 + 2x - 2$

$\Rightarrow \quad x = 13$

Thus, there are 13 nodes of degree one in the tree.

Problem 3.2. *A tree has 2n vertices of degree 1, 3n vertices of degree 2 and n vertices of degree 3. Determine the number of vertices and edges in the tree.*

Solution. It is given that total number of vertices in the tree is $2n + 3n + n = 6n$.

The total degree of the tree is $2n \times 1 + 3n \times 2 + n \times 3 = 11n$.

The number of edges in the tree will be half of $11n$.

If G = (V, E) be the tree then, we have

$$|\,V\,| = 6n \quad \text{and} \quad |\,E\,| = \frac{11n}{2}$$

In any tree, $|\,E\,| = |\,V\,| - 1$.

Therefore, we have

$$\frac{11n}{2} = 6n - 1$$

$\Rightarrow \quad 11n = 12n - 2$

$\Rightarrow \quad n = 2$

Thus, there are 6 × 2 = 12 nodes and 11 edges in the tree.

Theorem 3.4. *There are at the most n^h leaves in an n-ary tree of height h.*

Proof. Let us prove this theorem by mathematical induction on the height of the tree.

As basis step take $h = 0$, *i.e.,* tree consists of root node only.

Since $n^\circ = 1$, the basis step is true.

Now let us assume that the above statement is true for $h = k$.

i.e., an n-ary tree of height k has at the most n^k leaves.

If we add n nodes to each of the leaf node of n-ary tree of height k, the total number of leaf nodes will be at the most $n^h \times n = n^{h+1}$.

Hence inductive step is also true.

This proves that above statement is true for all $h \geq 0$.

Theorem 3.5. *In a complete n-ary tree with m internal nodes, the number of leaf node l is given by the formula*

$$l = \frac{(n-1)(x-1)}{n}.$$

where, x is the total number of nodes in the tree.

Proof. It is given that the tree has m internal nodes and it is complete n–ary, so total number of nodes

$$x = n \times m + 1.$$

Thus, we have $\quad m = \dfrac{(x-1)}{n}$

It is also given that l is the number of leaf nodes in the tree.

Thus, we have $\quad x = m + l + 1$

Substituting the value of m in this equation, we get

$$x = \left(\frac{x-1}{n}\right) + l + 1$$

or

$$l = \frac{(n-1)(x-1)}{n}$$

Theorem 3.6. *If T = (V, E) be a rooted tree with v_0 as its root then*

(*i*) *T is a acyclic*

(*ii*) *v_0 is the only root in T*

(*iii*) *Each node other than root in T has in degree 1 and v_0 has indegree zero.*

Proof. We prove the theorem by the method of contradiction.

(*i*) Let there is a cycle π in T that begins and end at a node v.

Since the in degree of root is zero, $v \neq v_0$.

Also by the definition of tree, there must be a path from v_0 to v, let it be p.

Then πp is also a path, distinct from p, from v_0 to v.

This contradicts the definition of a tree that there is unique path from root to every other node.

Hence T cannot have a cycle in it.

i.e., a tree is always acyclic.

(*ii*) Let v_1 is another root in T.

By the definition of a tree, every node is reachable from root.

This v_0 is reachable from v_1 and v_1 is reachable from v_0 and the paths are π_1 and π_2 respectively.

Then $\pi_1\pi_2$ combination of these two paths is a cycle from v_0 and v_0.

Since a tree is always acyclic, v_0 and v_1 cannot be different.

Thus, v_0 is a unique root.

(*iii*) Let w be any non-root node in T.

Thus, $\exists$ a path $\pi : v_0, v_1, \ldots\ldots, v_k w$ from v_0 to w in T.

Now let us suppose that indegree of w is two.

Then $\exists$ two nodes w_1 and w_2 in T such that edges (w_1, v_0) and (w_2, v_0) are in E.

Let π_1 and π_2 be paths from v_0 to w_1 and w_2 respectively.

Then $\pi_1 : v_0 v_1 \ldots\ldots v_k w_1 w$ and $\pi_2 : v_0 v_1 \ldots\ldots v_k w_2 w$ are two possible paths from v_0 to w.

This is in contradiction with the fact that there is unique path from root to every other nodes in a tree.

Thus indegree of w cannot be greater than 1.

Next, let indegree of $v_0 > 0$. Then $\exists$ a node v in T such that $(v, v_0) \in$ E.

Let π be a path from v_0 to v, thus $\pi(v, v_0)$ is a path from v_0 to v_0 that is a cycle.

This is again a contradiction with the fact that any tree is acyclic.

Thus indegree of root node v_0 cannot be greater than zero.

Problem 3.3. *Let $T = (V, E)$ be a rooted tree. Obviously E is a relation on set V. Show that*

(*i*) *E is irreflexive*

(*ii*) *E is asymmetric*

(*iii*) *If $(a, b) \in E$ and $(b, c) \in E$ then $(a, c) \notin E$, $\forall\, a, b, c \in V$.*

Solution. Since a tree is acyclic, there is no cycle of any length in a tree.

This implies that there is no loop in T.

Thus, $(v, v) \notin$ E $\forall\, a \in$ V.

Thus E is an irreflexive relation on V.

Let $(x, y) \in$ E. If $(y, x) \in$ E, then there will be cycle at node x as well as on node y.

Since no cycle is permissible in a tree, either pair (x, y) or (y, x) can be in E but never both.

This implies that presence of (x, y) excludes the presence of (y, x) in E and *vice versa*.

Thus E is a asymmetric relation on V.

Let $(a, c) \in$ E.

Thus presence of pairs (b, c) and (a, c) in E implies that c has indegree > 1.

Hence $(a, c) \notin$ E.

Problem 3.4. *Prove that a tree T is always separable.*

Solution. Let w be any internal node in T and node v is the parent of w.

By the definition of a tree, in degree of w is one.

If w is dropped from the tree T, the incoming edge from v to w is also removed.

Therefore all children of w will be unreachable from root and tree T will become disconnected.

See the forest of the Figure (3.11), which has been obtained after removal of node F from the tree of Figure (3.12).

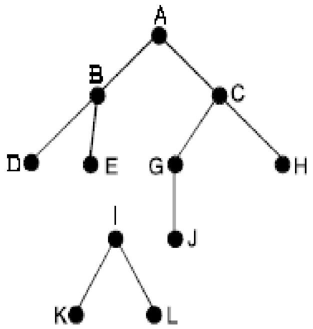


Fig. 3.11

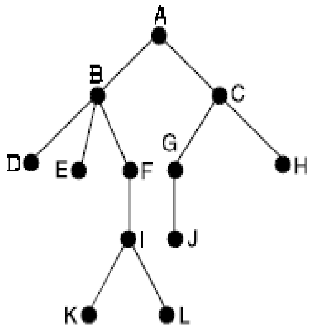


Fig. 3.12

Problem 3.5. *Let $A = \{v_1, v_2, v_3, v_4, v_5, v_6,v_7, v_8, v_9, v_{10}\}$ and let*

$T = \{(v_2, v_3), (v_2, v_1), (v_4, v_5), (v_4, v_6), (v_5, v_8), (v_6, v_7), (v_4, v_2), (v_7, v_9), (v_7, v_{10})\}$.

Show that T is a rooted tree and identify the root.

Solution. Since no paths begin at vertices v_1, v_3, v_8, v_9 and v_{10}, these vertices cannot be roots of a tree.

There are no paths from vertices v_6, v_7, v_2 and v_5 to vertex v_4, so we must eliminate these vertices as possible roots.

Thus, if T is a rooted tree, its root must be vertex v_4.

It is easy to show that there is a path from v_4 to every other vertex.

For example, the path v_4, v_6, v_7, v_9 leads from v_4 and v_9, since (v_4, v_6), (v_6, v_7) and (v_7, v_9) are all in T.

We draw the digraph of T, beginning with vertex v_4, and with edges shown downward.

The result is shown in Fig. (3.13). A quick inspection of this digraph shows that paths from vertex v_4 to every other vertex are unique, and there are no paths from v_4 and v_4.

Thus T is a tree with root v_4.

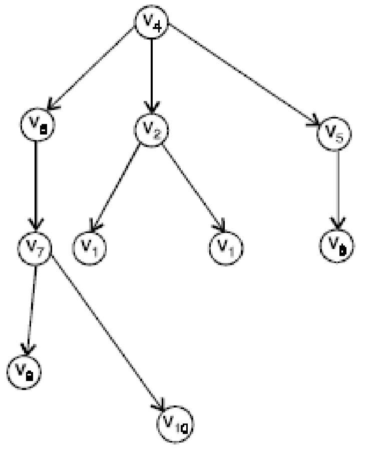


Fig. 3.13

Theorem 3.7. *There is one and only one path between every pair of vertices in a tree T.*

Proof. Since T is a connected graph, there must exist atleast one path between every pair of vertices in T.

Let there are two distinct paths between two vertices u and v of T.

But union of these two paths will contain a cycle and then T cannot be a tree.

Theorem 3.8. *If in a graph G there is one and only one path between every pair of vertices, G is a tree.*

Proof. Since there exists a path between every pair of vertices then G is connected.

A cycle in a graph (with two or more vertices) implies that there is atleast one pair of vertices u, v such that there are two distinct paths between u and v.

Since G has one and only one path between every pair of vertices, G can have no cycle.

Therefore, G is a tree.

Theorem 3.9. *A tree T with n vertices has n – 1 edges.*

Proof. The theorem is proved by induction on n, the number of vertices of T.

Basis of Inductive : When $n = 1$ then T has only one vertex. Since it has no cycles, T can not have any edge.

i.e., it has $e = 0 = n - 1$

Induction step : Suppose the theorem is true for $n = k \geq 2$ where k is some positive integer.

We use this to show that the result is true for $n = k + 1$.

Let T be a tree with $k + 1$ vertices and let uv be edge of T. Let uv be an edge of T. Then if we remove the edge uv from T we obtain the graph T – uv. Then the graph is disconnected since T – uv contains no (u, v) path.

If there were a path, say u, v_1, v_2 ...... v from u to v then when we added back the edge uv there would be a cycle $u, v_1, v_2,$ ...... v, u in T.

Thus, T – uv is disconnected. The removal of an edge from a graph can disconnected the graph into at most two components. So T – uv has two components, say, T_1 and T_2.

Since there were no cycles in T to begin with, both components are connected and are without cycles.

Thus, T_1 and T_2 are trees and each has fewer than n vertices.

This means that we can apply the induction hypothesis to T_1 and T_2 to give

$$e(T_1) = v(T_1) - 1$$
$$e(T_2) = v(T_2) - 1$$

But the construction of T_1 and T_2 by removal of a single edge from T gives that

$$e(T) = e(T_1) + e(T_2) + 1$$

and that $\quad v(T) = v(T_1) + v(T_2)$

it follows that

$$\begin{aligned} e(T) &= v(T_1) - 1 + v(T_2) - 1 + 1 \\ &= v(T) - 1 \\ &= k + 1 - 1 = k. \end{aligned}$$

Thus T has k edges, as required.

Hence by principle of mathematical induction the theorem is proved.

Theorem 3.10. *For any positive integer n, if G is a connected graph with n vertices and n – 1 edges, then G is a tree.*

Proof. Let n be a positive integer and suppose G is a particular but arbitrarily chosen graph that is connected and has n vertices and $n - 1$ edges.

We know that a tree is a connected graph without cycles. (We have proved in previous theorem that a tree has $n - 1$ edges).

We have to prove the converse that if G has no cycles and $n - 1$ edges, then G is connected.

We decompose G into k components, $c_1, c_2, \ldots\ldots c_k$.

Each component is connected and it has no cycles since G has no cycles.

Hence, each C_k is a tree.

Now $e_1 = n_1 - 1$ and $\sum_{i=1}^{k} e_i = \sum_{i=1}^{k} (n_i - 1) = n - k$

$\Rightarrow \quad e = n - k$

Then it follows that $k = 1$ or G has only one component.

Hence G is a tree.

Problem 3.6. *Consider the rooted tree in Figure (3.14).*

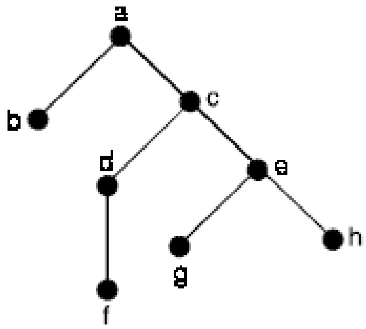


Fig. 3.14.

(*a*) *What is the root of T ?*

(*b*) *Find the leaves and the internal vertices of T.*

(*c*) *What are the levels of c and e.*

(*d*) *Find the children of c and e.*

(*e*) *Find the descendants of the vertices a and c.*

Solution. (*a*) Vertex *a* is distinguished as the only vertex located at the top of the tree.

Therefore *a* is the root.

(*b*) The leaves are those vertices that have no children. These *b*, *f*, *g* and *h*. The internal vertices are *c*, *d* and *e*.

(*c*) The levels of *c* and *e* are 1 and 2 respectively.

(*d*) The children of *c* are *d* and *e* and of *e* are *g* and *h*.

(*e*) The descendants of *a* are *b*, *c*, *d*, *e*, *f*, *g*, *h*.

The descendants of *c* are *d*, *e*, *f*, *g*, *h*.

Theorem 3.11. *A full m-ary tree with i internal vertex has $n = mi + 1$ vertices.*

Proof. Since the tree is a full *m*-ary, each internal vertex has *m* children and the number of internal vertex is *i*, the total number of vertex except the root is *mi*.

Therefore, the tree has $n = mi + 1$ vertices.

Since 1 is the number of leaves, we have $n = l + i$ using the two equalities $n = mi + 1$ and $n = 1 + i$, the following results can easily be deduced.

A full *m*-ary tree with

(*i*) *n* vertices has $i = \dfrac{(n-1)}{m}$ internal vertices and $l = \dfrac{[(m-1)(n+1)]}{m}$ leaves.

(*ii*) *i* internal vertices has $n = mi + 1$ vertices and $l = (m - 1)i + 1$ leaves.

(*ii*) *i* internal vertices has $n = mi + 1$ vertices and $l = (m - 1)i + 1$ leaves.

(*iii*) *l* leaves has $n = \dfrac{(ml-1)}{(m-1)}$ vertices and $i = \dfrac{(l-1)}{(m-1)}$ internal vertices.

Theorem 3.12. *There are at most m^h leaves in an m-ary tree of height h.*

Proof. We prove the theorem by mathematical induction.

Basis of Induction :

For $h = 1$, the tree consists of a root with no more than *m* children, each of which is a leaf.

Hence there are no more than $m^1 = m$ leaves in an *m*-ary of height 1.

Induction hypothesis :

We assume that the result is true for all *m*-ary trees of heights less than *h*.

Induction step :

Let T be an *m*-ary tree of height *h*. The leaves of T are the leaves of subtrees of T obtained by deleting the edges from the roots to each of the vertices of level 1.

Each of these subtrees has at most m^{h-1} leaves. Since there are at most *m* such subtrees, each with a maximum of m^{h-1} leaves, there are at most $m \,.\, m^{h-1} = m^h$.

Problem 3.7. *Find all spanning trees of the graph G shown in Figure 3.15.*

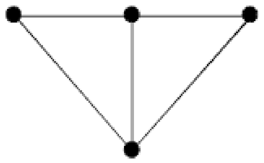


Fig. 3.15.

Solution. The graph G has four vertices and hence each spanning tree must have 4 – 1 = 3 edges.

Thus each tree can be obtained by deleting two of the five edges of G.

This can be done in 10 ways, except that two of the ways lead to disconnected graphs.

Thus there are eight spanning trees as shown in Figure (3.16).

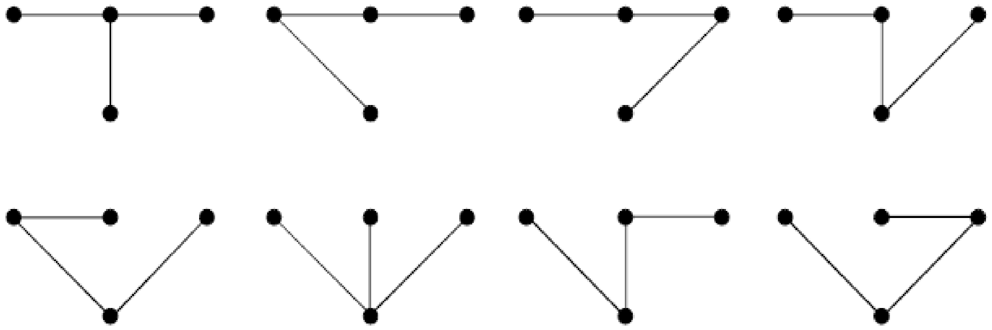


Fig. 3.16.

Problem 3.8. *Find all spanning trees for the graph G shown in Figure 3.17, by removing the edges in simple circuits.*

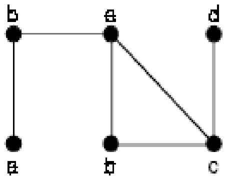


Fig. 3.17.

Solution. The graph G has one cycle *cbec* and removal of any edge of the cycle gives a tree.

There are three trees which contain all the vertices of G and hence spanning trees.

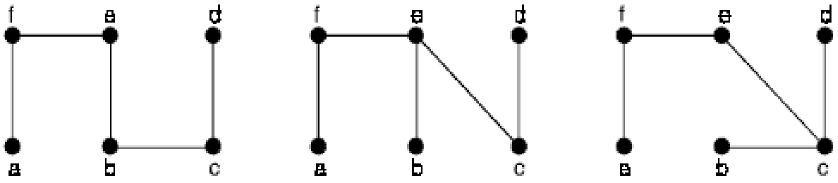


Fig. 3.18.

Theorem 3.13. *A simple graph G has a spanning tree if and only if G is connected.*

Proof. First, suppose that a simple graph G has a spanning tree T. T contains every vertex of G. Let a and b be vertices of G. Since a and b are also vertices of T and T is a tree, there is a path P between a and b.

Since T is subgraph, P also serves as path between a and b in G.

Hence G is connected.

Conversely, suppose that G is connected.

If G is not a tree, it must contain a simple circuit. Remove an edge from one of these simple circuits. The resulting subgraph has one fewer edge but still contains all the vertices of G and is connected.

If this subgraph is not a tree, it has a simple circuit, so as before, remove an edge that is in a simple circuit.

Repeat this process until no simple circuit remain.

This is possible because there are only a finite number of edges in the graph, the process terminates when no simple circuits remain.

Thus we eventually produce an acyclic subgraph T which is a tree.

The tree is a spanning tree since it contains every vertex of G.

$\sum_{i=1}^{p} d_i = 2q = 2(p-1) = 2p - 2$ which contradicts in equality (1).

Hence T contains atleast two end vertices.

Theorem 3.16. *If G is a tree and if any two non adjacent vertices of G are joined by an edge e, then G + e has exactly one cycle.*

Proof. Suppose G is a tree. Then there is exactly one path joining any two vertices of G.

If we add an edge of G, that edge together with unique path joining u and v forms a cycle.

Theorem 3.17. *A graph G is connected if and only if it contains a spanning tree.*

Proof. It is immediate that, if a graph contains a spanning tree, then it must be connected.

Conversely, if a connected graph does not contain any cycle then it is a tree.

For a connected graph containing one or more cycles, we can remove an edge from one of the cycles and still have a connected subgraph. Such removal of edges from cycles can be repeated until we have a spanning tree.

Theorem 3.18. *If u and v are distinct vertices of a tree T contains exactly one $u - v$ path.*

Proof. Suppose, to the contrary that T contains two $u - v$ paths say P and Q are different $u - v$, paths there must be a vertex x (*i.e.*, $x = u$) belonging to both P and Q such that the vertex immediately following x on Q. See Figure 3.19.

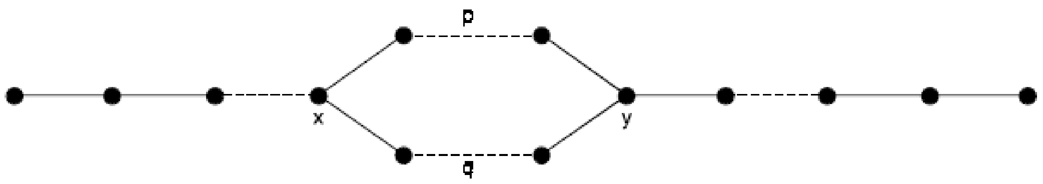


Fig. 3.19.

Let y be the first vertex of P following x that also belongs to Q (y could be v).

Then this produces to $x - y$ paths that have only x and y in common.

These two paths produces a cycle in T, which contradicts the fact that T is a tree.

Therefore, T has only one $u - v$ path.

Problem 3.9. *Construct two non-isomorphic trees having exactly 4 pendant vertices or vertices.*

Solution.

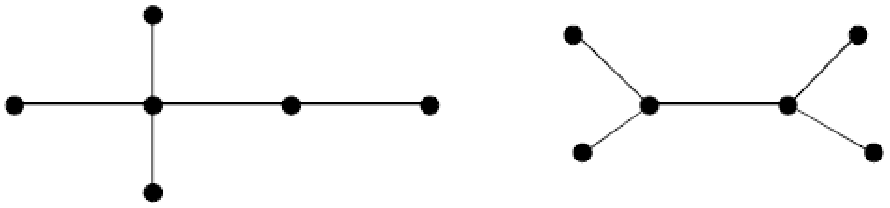


Fig. 3.20.

Problem 3.10. *Construct three distinct trees with exactly*

(*i*) *one central vertex* (*ii*) *two central vertices.*

Solution. (*i*) The following trees contain only one central vertex.

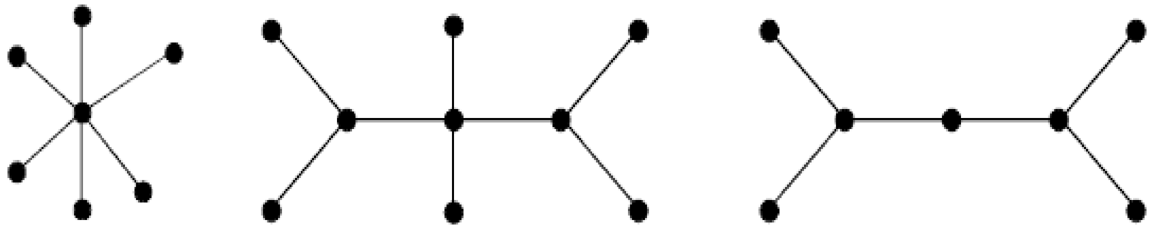


Fig. 3.21.

(*ii*) The following trees contain exactly two central vertices.

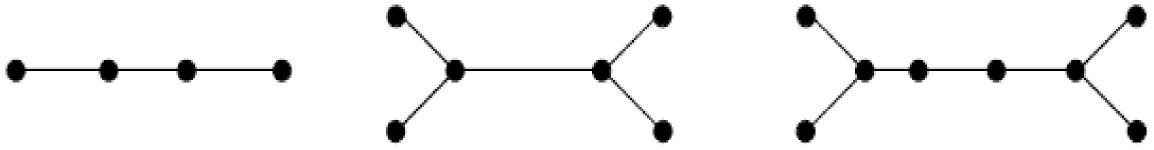


Fig. 3.22.

3. 5 ALGORITHMS FOR CONSTRUCTING SPANNING TREES

An algorithm for finding a spanning tree based on the proof of the theorem : A simple graph G has a spanning tree if and only if G is connected, would not be very efficient, it would involve the time- consuming process of finding cycles. Instead of constructing spanning trees by removing edges, spanning tree can be built up by successively adding edges. Two algorithms based on this principle for finding a spanning tree are Breath-first search (BFS) and Depth-first search (DFS).

3.5.1 BFS algorithm

In this algorithm a rooted tree will be constructed, and underlying undirected graph of this rooted

forms the spanning tree. The idea of BFS is to visit all vertices on a given level before going into the next
level.

Procedure :

(*i*) Arbitrarily choose a vertex and designate it as the root. Then add all edges incident to this vertex, such that the addition of edges does not produce any cycle.
(*ii*) The new vertices added at this stage become the vertices at level 1 in the spanning tree, arbitrarily order them.
(*iii*) Next, for each vertex at level 1, visited in order, add each edge incident to this vertex to the
tree as long as it does not produce any cycle.
(*iv*) Arbitrarily order the children of each vertex at level 1. This produces the vertices at level 2 in the tree.
(*v*) Continue the same procedure until all the vertices in the tree have been added.
(*vi*) The procedure ends, since there are only a finite number of edges in the graph.
(*vii*) A spanning tree is produced since we have produced a tree without cycle containing every
vertex of the graph.

3.5.2 DFS algorithm

An alternative to Breath-first search is Depth-first search which proceeds to successive levels in
a tree at the earliest possible opportunity.
DFS is also called back tracking.

Procedure :

(*i*) Arbitrarily choose a vertex from the vertices of the graph and designate it as the root.
(*ii*) Form a path starting at this vertex by successively adding edges as long as possible where each new edge is incident with the last vertex in the path without producing any cycle.
(*iii*) If the path goes through all vertices of the graph, the tree consisting of this path is a spanning tree.
Otherwise, move back to the next to last vertex in the path, and, if possible, form a new path starting at this vertex passing through vertices that were not already visited.

(iv) If this cannot be done, move back another vertex in the path, that is two vertices back in the
path, and repeat.
(*v*) Repeat this procedure, beginning at the last vertex visited, moving back up the path one vertex at a time, forming new paths that are as long as possible until no more edges can be added.
(*vi*) This process ends since the graph has a finite number of edges and is connected. A spanning

tree is produced.

Problem 3.48. *Use BFS algorithm to find a spanning tree of graph G of Fig. (3.55).*

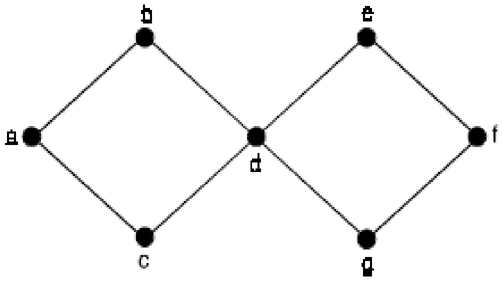


Fig. 3.55.

Solution. (*i*) Choose the vertex *a* to be the root.

(*ii*) Add edges incident with all vertices adjacent to *a*, so that edges {*a*, *b*}, {*a*, *c*} are added. The two vertices *b* and *c* are in level 1 in the tree.

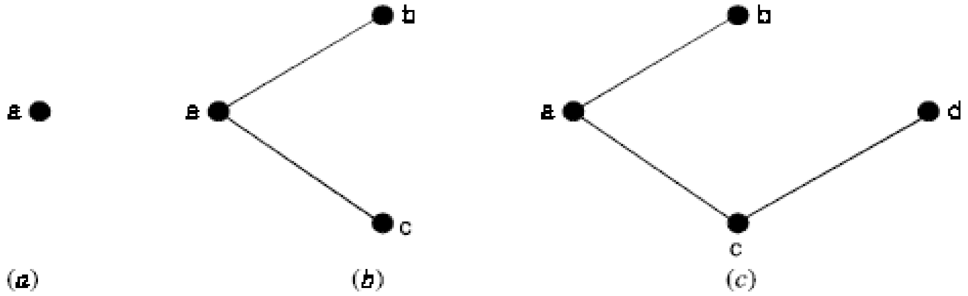
(*iii*) Add edges from these vertices at level 1 to adjacent vertices not already in the tree.

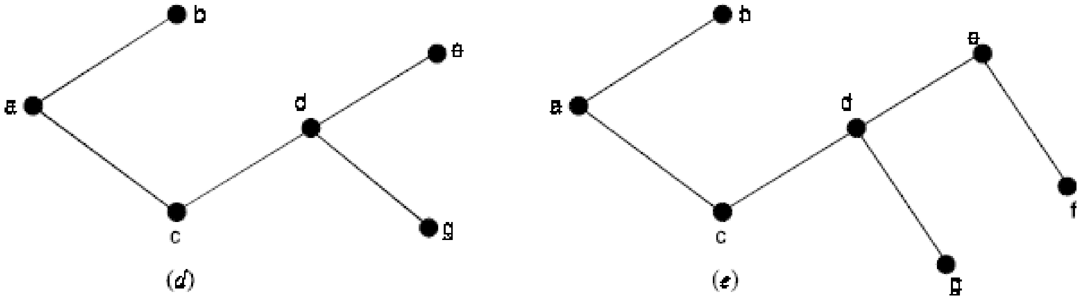
Hence the edge {*c*, *d*} is added. The vertex *d* is in level 2.

(*iv*) Add edge from *d* in level 2 to adjacent vertices not already in the tree. The edge {*d*, *e*} and {*d*, *g*} are added.

Hence *e* and *g* are in level 3.

(*v*) Add edge from *e* at level 3 to adjacent vertices not already in the tree and hence {*e*, *f* } is added. The steps of Breath first procedure are shown in Fig. (3.56).





(*d*) (*e*)

Problem 3.49. *Find a spanning tree of the graph of Fig. (3.57) using Depth-first search algorithm.*

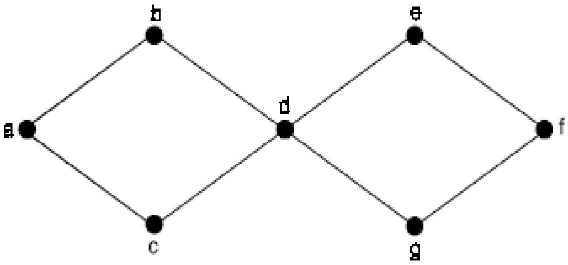


Fig. 3.57.

Solution. Choose the vertex *a*.

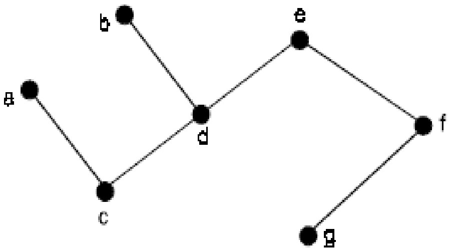
Form a path by successively adding edges incident with vertices not already in the path as long as possible.

This produces the path $a - c - d - e - f - g$.

Now back track of *f*. There is no path beginning at *f* containing vertices not already visited.

Similarly, after backtrack at *e*, there is no path. So move back track at *d* and form the path $d - b$.

This produces the required spanning tree which is shown in Fig. (3.58).



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited “A” Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

SCHOOL OF SCIENCE AND HUMANITIES

DEPARTMENT OF MATHEMATICS

UNIT – IV –OPTIMIZATION AND MATCHING – SMT 1505

UNIT IV

OPTIMIZATION AND MATCHING

Cut vertex cut set and bridge

Sometimes the removal of a vertex and all edges incident with it produces a subgraph with more connected components. A cut vertex of a connected graph G is a vertex whose removal increases the number of components. Clearly if v is a cut vertex of a connected graph G, G – v is disconnected.

A cut vertex is also called a **cut point.**

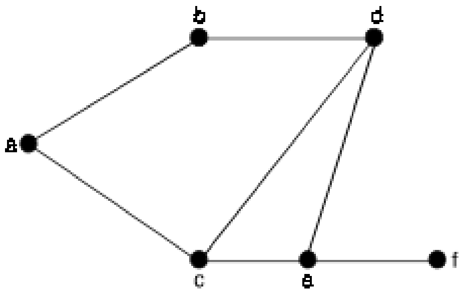
Analogously, an edge whose removal produces a graph with more connected components then the original graph is called a **cut edge** or **bridge.**

The set of all minimum number of edges of G whose removal disconnects a graph G is called a **cut set** of G. Thus a cut set S of a satisfy the following :

(*i*) S is a subset of the edge set E of G.

(*ii*) Removal of edges from a connected graph G disconnects G.

(*iii*) No proper subset of G satisfy the condition.



In the graph in Figure below, each of the sets $\{\{b, d\}, \{c, d\}, \{c, e\}\}$ and $\{\{e, f\}\}$ is a cut set. The edge $\{e, f\}$ is the only bridge. The singleton set consisting of a bridge is always a cut of set of G.

4.3.10. Connected or weakly connected

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

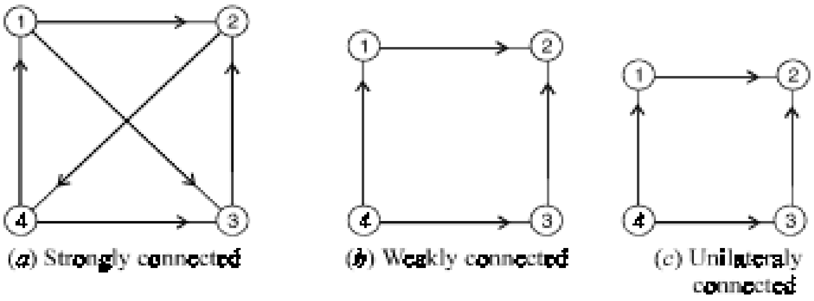
4.3.11. Unilaterally connected

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

4.3.12. 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 diagraphs is Fig. (4.61) the digraph in (*a*) is strongly connected, in a (*b*) it is weakly connected, while in (*c*) it is unilaterally connected but not strongly connected.



(*a*) Strongly connected (*b*) Weakly connected (*c*) Unilateraly connected

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

4.3.13. Connectivity

To study the measure of connectedness of a graph G we consider the minimum number of vertices and edges to be removed from the graph in order to disconnect it.

4.3.14. Edge connectivity

Let G be a connected graph. The edge connectivity of G is the minimum number of edges whose removal results in a disconnected or trivial graph. The edge connectivity of a connected graph G is denoted by $\lambda(G)$ or $E(G)$.

4.3.15. Vertex connectivity

Let G be a connected graph. The vertex connectivity of G is the minimum number of vertices whose removal results in a disconnected or a trivial graph. The vertex connectivity of a connected graph is denoted by $k(G)$ or $V(G)$

(*i*) If G is a disconnected graph, then $\lambda(G)$ or $E(G) = 0$.

(*ii*) Edge connectivity of a connected graph G with a bridge is 1.

(*iii*) The complete graph k_n cannot be disconnected by removing any number of vertices, but the removal of $n - 1$ vertices results in a trivial graph. Hence $k(k_n) = n - 1$.

(*iv*) The vertex connectivity of a graph of order atleast there is one if and only if it has a cut vertex.

(*v*) Vertex connectivity of a path is one and that of cycle C_n $(n \geq 4)$ is two.

Problem 4.26. *Find the (i) vertex sets of components (ii) cut-vertices and (iii) cut-edges of the graph given below.*

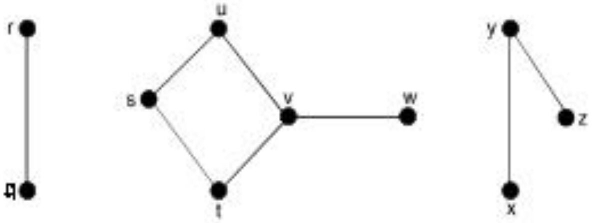


Fig. 4.62.

Solution. The graph has three components. The vertex set of the components are $\{q, r\}$, $\{s, t, u, v, w\}$ and $\{x, y, z\}$. The cut vertices of the graph are t and y.

Its cut-edges are qr, st, xy and yz.

Problem 4.27. *Is the directed graph given below strongly connected ?*

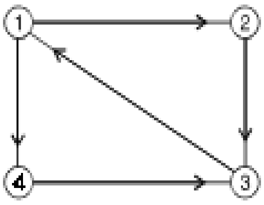


Fig. 4.63.

Solution. The possible pairs of vertices and the forward and the backward paths between them 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)	4–3	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.

Theorem 4.7. *Let v be a point a connected graph G. The following statements are equivalent*

(1) v is a cutpoint of G

(2) There exist points u and v distinct from v such that v is on every u–w path.

(3) There exists a partition of the set of points V–{v} into subsets U and W such that for any points $u \in U$ and $w \in W$, the point v is on every u–w path.

Proof. (1) implies (3)

Since v is a cutpoint of G, G–v is disconnected and has atleast two components. Form a partition of V–{v} by letting U consist of the points of one of these components and W the points of the others.

The any two points $u \in$ U and $w \in$ W lie in different components of G–v.

Therefore every u–w path in G contains v.

(3) implies (2)

This is immediate since (2) is a special case of (3).

(2) implies (1)

If v is on every path in G joining u and w, then there cannot be a path joining these points in G–v.

Thus G–v is disconnected, so v is a cutpoint of G.

Theorem 4.8. *Every non trivial connected graph has atleast two points which are not cutpoints.*

Proof. Let u and v be points at maximum distance in G, and assume v is a cut point.

Then there is a point w in a different component of G–v than u.

Hence v is in every path joining u and w, so $d(u, w) > d(u, v)$ which is impossible.

Therefore v and similarly u are not cut points of G.

Theorem 4.9. *Let x be a line of a connected graph G. The following statements are equivalent :*

(1) x is a bridge of G

(2) x is not on any cycle of G

(3) There exist points u and v of G such that the line x is on every path joining u and v.

(4) These exists a partition of V into subsets U and W such that for any points $u \in U$ and $w \in W$, the line x is on every path joining u and w.

Theorem 4.10. *A graph H is the block graph of some graph if and only if every block of H is complete.*

Proof. Let H = B(G), and assume there is a block H_i of H which is not complete.

Then there are two points in H_i which are non adjacent and lie on a shortest common cycle Z of length atleast 4.

But the union of the blocks of G corresponding to the points of H_i which lie on Z is then connected and has no cut point, so it is itself contained in a block, contradicting the maximality property of a block of a graph.

On the otherhand, let H be a given graph in which every block is complete.

From B(H), and then form a new graph G by adding to each point H_i of B(H) a number of end lines equal to the number of points of the block H_i which are not cut points of H. Then it is easy to see that B(G) is isomorphic to H.

Theorem 4.11. *Let G be a connected graph with atleast three points. The following statements are equivalent :*

(1) *G is a block*

(2) *Every two points of G lie on a common cycle*

(3) *Every point and line of G lie on a common cycle.*

(4) *Every two lines of G lie on a common cycle*

(5) *Given two points and one line of G, there is a path joining the points which contains the line.*

(6) *For every three distinct points of G, there is a path joining any two of them which contains the third.*

(7) *For every three distinct points of G, there is a path joining any two of them which does not contain the third.*

Proof. (1) implies (2)

Let u and v be distinct points of G and let U be the set of points different from u which lie on a cycle containing u.

Since G has atleast three points and no cutpoints, it has no bridges.

Therefore, every point adjacent to u is in U, so U is not empty.

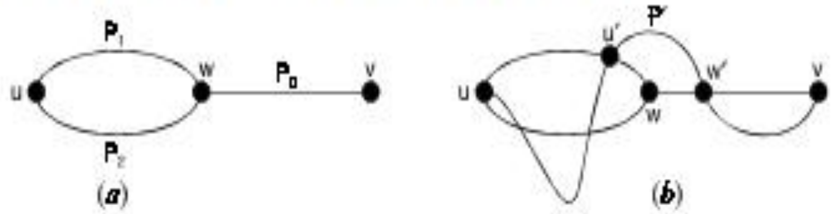


Fig. 4.64. Pahs in blocks.

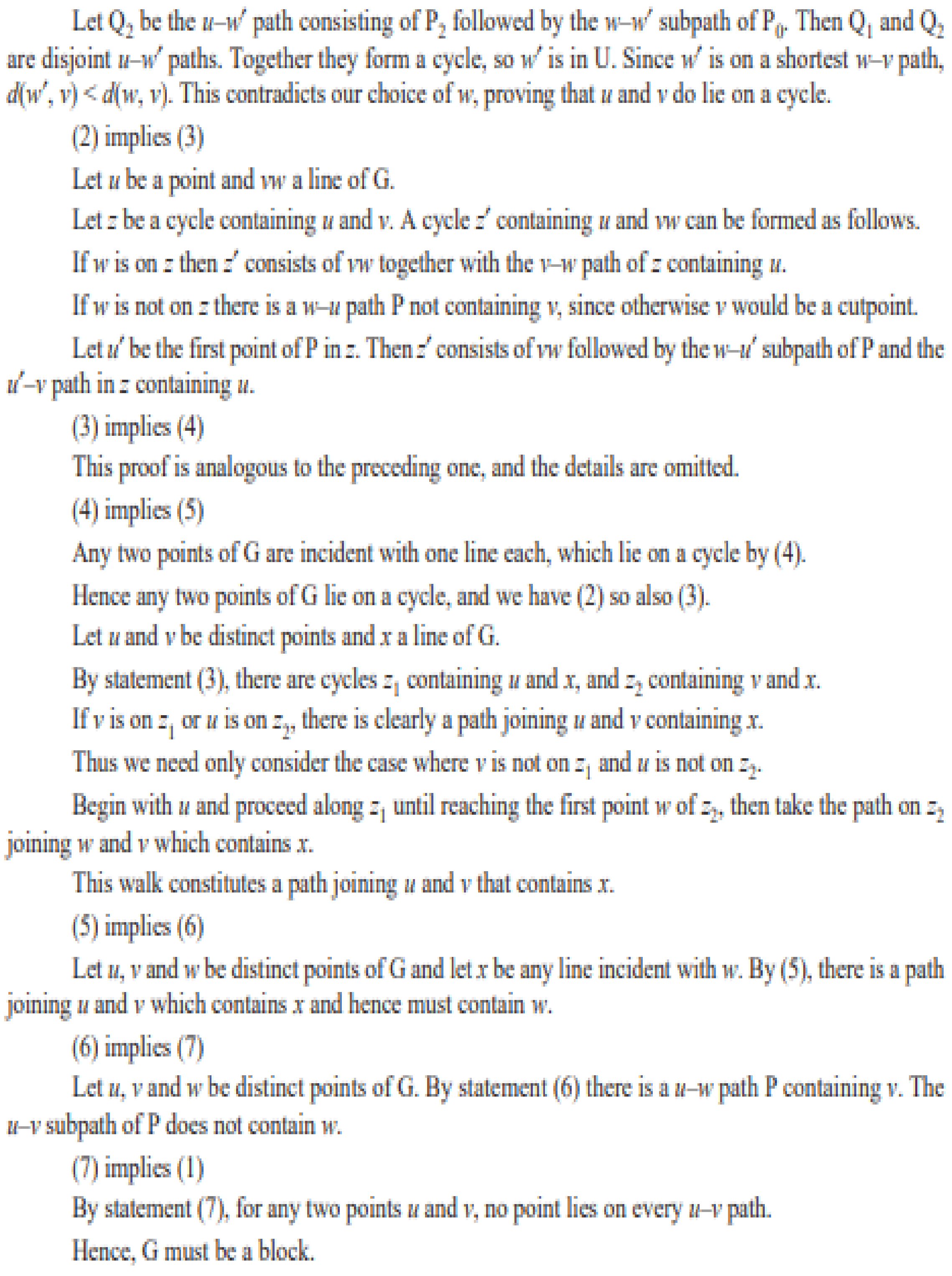
Suppose v is not in U. Let w be a point in U for which the distance $d(w, v)$ is minimum.

Let P_0 be a shortest w–v path, and let P_1 and P_2 be the two u–w paths of a cycle containing u and w (see Fig. 4.64(a)).

Since w is not a cutpoint, there is a u–v path P′ not containing w (see Fig. 4.64(b)).

Let w' be the point nearest u in P′ which is also in P_0 and let u be the last point of the u–w subpath of P′ in either P_1 or P_2. Without loss of generality, we assume u' is in P_1.

Let Q_1 be the u–w' path consisting of the u–u' subpath of P_1 and the u'–w' subpath of P′.



Let Q_2 be the u–w' path consisting of P_2 followed by the w–w' subpath of P_0. Then Q_1 and Q_2 are disjoint u–w' paths. Together they form a cycle, so w' is in U. Since w' is on a shortest w–v path, $d(w', v) < d(w, v)$. This contradicts our choice of w, proving that u and v do lie on a cycle.

(2) implies (3)

Let u be a point and vw a line of G.

Let z be a cycle containing u and v. A cycle z' containing u and vw can be formed as follows.

If w is on z then z' consists of vw together with the v–w path of z containing u.

If w is not on z there is a w–u path P not containing v, since otherwise v would be a cutpoint.

Let u' be the first point of P in z. Then z' consists of vw followed by the w–u' subpath of P and the u'–v path in z containing u.

(3) implies (4)

This proof is analogous to the preceding one, and the details are omitted.

(4) implies (5)

Any two points of G are incident with one line each, which lie on a cycle by (4).

Hence any two points of G lie on a cycle, and we have (2) so also (3).

Let u and v be distinct points and x a line of G.

By statement (3), there are cycles z_1 containing u and x, and z_2 containing v and x.

If v is on z_1 or u is on z_2, there is clearly a path joining u and v containing x.

Thus we need only consider the case where v is not on z_1 and u is not on z_2.

Begin with u and proceed along z_1 until reaching the first point w of z_2, then take the path on z_2 joining w and v which contains x.

This walk constitutes a path joining u and v that contains x.

(5) implies (6)

Let u, v and w be distinct points of G and let x be any line incident with w. By (5), there is a path joining u and v which contains x and hence must contain w.

(6) implies (7)

Let u, v and w be distinct points of G. By statement (6) there is a u–w path P containing v. The u–v subpath of P does not contain w.

(7) implies (1)

By statement (7), for any two points u and v, no point lies on every u–v path.

Hence, G must be a block.

MATCHING THEORY

A matching in a graph is a set of edges with the property that no vertex is incident with more than one edge in the set. A vertex which is incident with an edge in the set is said to be saturated. A matching is perfect if and only if every vertex is saturated, that is ; if and only if every vertex is incident with precisely one edge of the matching.

Let G = (V, E) be a bipartite graph with V partitioned as X ∪ Y. (each edge of E has the form $\{x, y\}$ with $x \in$ X and $y \in$ Y).

(*i*) A matching in G is a subset of E such that no two edges share a common vertex in X or Y.

(*ii*) A complete matching of X into Y is a matching in G such that every $x \in$ X is the end point of an edge.

Let G = (V, E) be bipartite with V partitioned as X ∪ Y. A maximal matching in G is one that matches as many vertices in X as possible with the vertices in Y.

Let G = (V, E) be a bipartite graph where V is partitioned as X ∪ Y. If A ⊆ X, then δ(A) = | A | – | R(A) | is called the deficiency of A. The deficiency of graph G, denoted δ(G), is given by δ(G) = max {δ(A)/A ⊆ X}.

For example, in the graph shown on the left in Fig. (4.85)

(*i*) the single edge *bc* is a matching which saturates *b* and *c*, but neither *a* nor *d* ;

(*ii*) the set {*bc*, *bd*} is not a matching because vertex *b* belongs to two edges ;

(*iii*) the set {*ab*, *cd*} is a perfect matching.

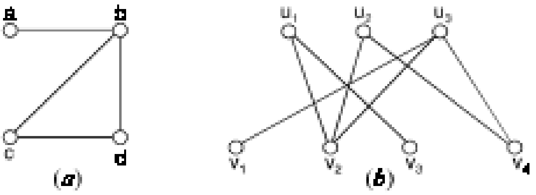


Fig. 4.85.

Edge set {*ab*, *cd*} is a perfect matching in the graph on the left. In the graph on the right, edge set $\{u_1, v_2, u_2v_4, u_3v_1\}$ is a matching which is not perfect.

Note that, if a matching is perfect, the vertices of the graph can be partitioned into two sets of equal size and the edges of the matching provide a one-to-one correspondence between these sets. In the graph on the left in Fig. (4.85), for instance, the edges of the perfect matching {*ab*, *cd*} establish a one-to-one correspondence between {*a*, *c*} and {*b*, *d*} : $a \to b, c \to d$.

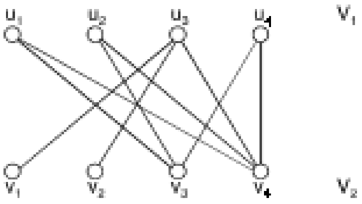
In the graph on the right of Fig. (4.85).

(*i*) the set of edges $\{u_1v_2, u_2v_4, u_3v_1\}$ is a matching which is not perfect but which saturates $v_1 = \{u_1, u_2, u_3\}$,

(*ii*) no matching can saturate $v_2 = \{v_1, v_2, v_3, v_4\}$ since such a matching would require four edges but then at least one u_i would be incident with more than one edge.

In the figure to the right, if X = $\{u_1, u_2, u_4\}$, then A(X) = $\{v_3, v_4\}$.

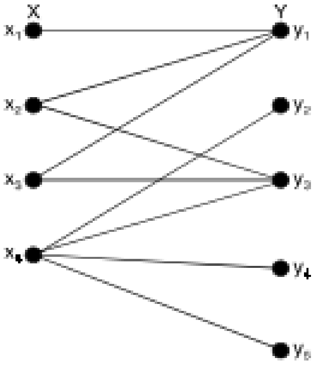
Since | X | $\nleq$ | A(X) |, the workers in X cannot all find jobs for which they are qualified. There is no matching in this graph which saturates V_1.



The bipartite graph shown in Fig. (4.87) has no complete matching. Any attempt to construct such a matching must include $\{x_1, y_1\}$ and either $\{x_2, y_3\}$ or $\{x_3, y_3\}$.

If $\{x_2, y_3\}$ is included, there is no match for x_3. Likewise, if $\{x_3, y_3\}$ is included, we are not able to match x_2.

If $A = \{x_1, x_2, x_3\} \subseteq X$, then $R(A) = \{y_1, y_3\}$. With $|A| = 3 > 2 = |R(A)|$, it follows that no complete matching can exist.



4.7 HALL'S MARRIAGE THEOREM (4.20)

If G is a bipartite graph with bipartition sets V_1 and V_2, then there exists a matching which saturates V_1 if and only if, for every subset X of V_1, $|X| \leq |A(X)|$.

Proof. It remains to prove that the given condition is sufficient, so we assume that $|X| \leq |A(X)|$ for all subsets X of V_1.

In particular, this means that every vertex in V_1 is joined to at least one vertex in V_2 and also that $|V_1| \leq |V_2|$.

Assume that there is no matching which saturates all vertices of V_1. We derive a contradiction.

We turn G into a directed network in exactly the same manner as with the job assignment application.

Specifically, we adjoin two vertices s and t to G and draw directed arcs from s to each vertex in V_1 and from each vertex in V_2 to t.

Assign a weight of 1 to each of these new arcs. Orient each edge of G from its vertex in V_1 to its vertex in V_2, and assign a large integer $I > |V_1|$ to each of these edges.

As noted before, there is a one-to-one correspondence between matchings of G and (s, t)-flows in this network, and the value of the flow equals the number of edges in the matching.

Since we are assuming that there is no matching which saturates V_1, it follows that every flow has value less than $|V_1|$ and hence by Max-Flow-Mincut theorem, there exists an (s, t)-cut $\{S, T\}$ $\{s \in S, t \in T)$.

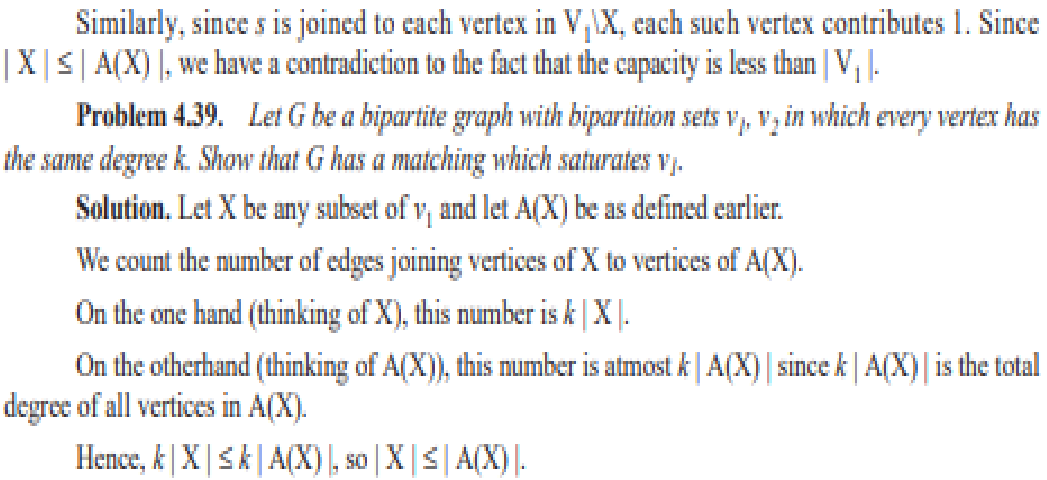
Whose capacity is less than $|V_1|$.

Now every original edge of G has been given a weight larger than $|V_1|$.

Since the capacity of our cut is less than $|V_1|$, no edge of G can join a vertex of S to a vertex of T.

Letting $X = V_1 \cap S$, we have $A(X) \subseteq S$.

Since each vertex in A(X) is joined to $t \in T$, each such vertex contributes 1 to the capacity of the cut.



Similarly, since s is joined to each vertex in $V_1 \backslash X$, each such vertex contributes 1. Since $| X | \leq | A(X) |$, we have a contradiction to the fact that the capacity is less than $| V_1 |$.

Problem 4.39. *Let G be a bipartite graph with bipartition sets v_1, v_2 in which every vertex has the same degree k. Show that G has a matching which saturates v_1.*

Solution. Let X be any subset of v_1 and let A(X) be as defined earlier.

We count the number of edges joining vertices of X to vertices of A(X).

On the one hand (thinking of X), this number is $k | X |$.

On the otherhand (thinking of A(X)), this number is atmost $k | A(X) |$ since $k | A(X) |$ is the total degree of all vertices in A(X).

Hence, $k | X | \leq k | A(X) |$, so $| X | \leq | A(X) |$.

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

SCHOOL OF SCIENCE AND HUMANITIES

DEPARTMENT OF MATHEMATICS

UNIT – V –PLANAR GRAPH – SMT 1505

INTRODUCTION

In this section we will study the question of whether a graph can be drawn in the plane without edges crossing. In particular, we will answer the houses-and-utilities problem. There are always many ways to represent a graph. When is it possible to find atleast one way to represent this graph in a plane without any edges crossing. Consider the problem of joining three houses to each of three separate utilities, as shown in figure below. Is it possible to join these houses and utilities so that none of the connections cross ? This problem can be modeled using the complete bipartite graph $K_{3,\ 3}$. The original question can be rephrased as : can $K_{3,\ 3}$ be drawn in the plane so that no two of its edges cross ?

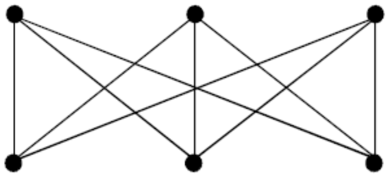


Fig. 2.1. Three houses and three utilities.

2.1 COMBINATORIAL AND GEOMETRIC GRAPHS (REPRESENTATION)

An abstract graph G can be defined as G = (V, E, Ψ) where the set V consists of the five objects named *a*, *b*, *c*, *d* and *e*, that is, V = {*a*, *b*, *c*, *d*, *e*} and the set E consists of seven objects (none of which is in set V) named 1, 2, 3, 4, 5, 6 and 7, that is,

$$E = \{1, 2, 3, 4, 5, 6, 7\}$$

and the relationship between the two sets is defined by the mapping Ψ, which consists of combinatorial representation of the graph.

$$\Psi = \left[\begin{array}{l} 1 \longrightarrow (a, c) \\ 2 \longrightarrow (c, d) \\ 3 \longrightarrow (a, d) \\ 4 \longrightarrow (a, b) \\ 5 \longrightarrow (b, d) \\ 6 \longrightarrow (d, e) \\ 7 \longrightarrow (b, e) \end{array}\right. \qquad \longrightarrow \text{Combinatorial representation of a graph}$$

Here, the symbol $1 \longrightarrow (a, c)$ says that object 1 from set E is mapped onto the (unordered) pair (a, c) of objects from set V.

It can be represented by means of geometric figure as shown below. It is true that graph can be represented by means of such configuration.

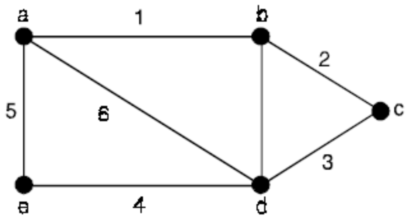


Fig. 2.2. Geometric representation of a graph.

2.2 PLANAR GRAPHS

A graph G is said to be **planar** if there exists some geometric representation of G which can be drawn on a plane such that no two of its edges intersect. The points of intersection are called crossovers.

A graph that cannot be drawn on a plane without a crossover between its edges crossing is called a plane graph. The graphs shown in Figure 2.3(*a*) and are planar graphs.

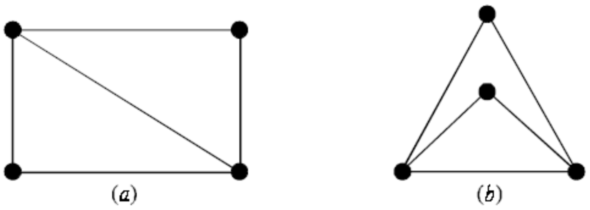


Fig. 2.3.

A drawing of a geometric representation of a graph on any surface such that no edges intersect is called embedding.

Note that if a graph G has been drawn with crossing edges, this does not mean that G is non planar, there may be another way to draw the graph without crossovers. Thus to declare that a graph G is non planar. We have to show that all possible geometric representations of G none can be embedded in a plane.

Equivalently, a graph G is planar is there if there exists a graph isomorphic to G that is embedded in a plane, otherwise G is non planar.

For example, the graph in Figure 2.4(*a*) is apparently non planar. However, the graph can be redrawn as shown in Figure (2.4)(*b*) so that edges don't cross, it is a planar graph, though its appearance is non coplanar.

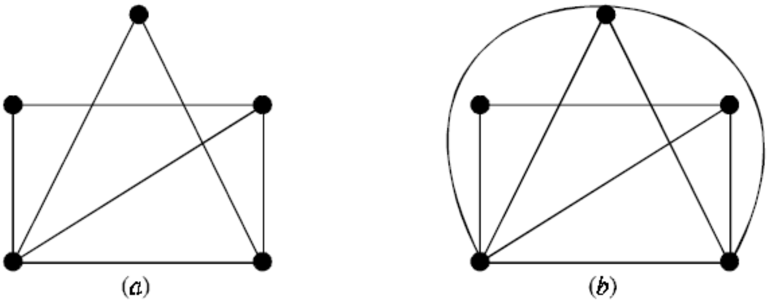


Fig. 2.4.

2.3 KURATOWSKI'S GRAPHS

For this we discuss two specific non-planar graphs, which are of fundamental importance, these are called Kuratowski's graphs. The complete graph with 5 vertices is the first of the two graphs of Kuratowski. The second is a regular, connected graph with 6 vertices and 9 edges.

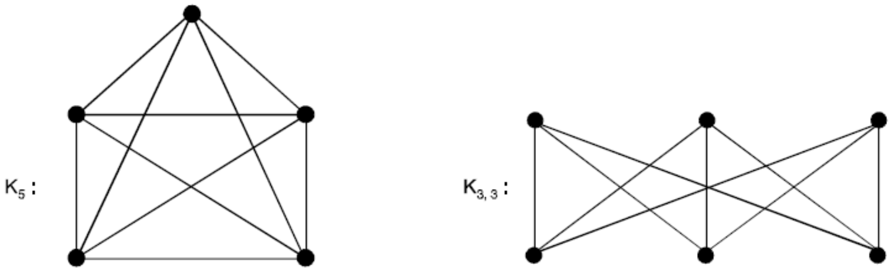


Fig. 2.5.

Observations

(*i*) Both are regular graphs

(*ii*) Both are non-planar graphs

(*iii*) Removal of one vertex or one edge makes the graph planar

(*iv*) (Kuratowski's) first graph is non-planar graph with smallest number of vertices and (Kuratowski's) second graph is non-planar graph with smallest number of edges. Thus both are simplest non-planar graphs.

The first and second graphs of Kuratowski are represented as K_5 and $K_{3,\,3}$. The letter K being for Kuratowski (a polish mathematician).

2.4 HOMEOMORPHIC GRAPHS

Two graphs are said to be homeomorphic if and only if each can be obtained from the same graph by adding vertices (necessarily of degree 2) to edges.

The graphs G_1 and G_2 in Figure (2.6) are homeomorphic since both are obtainable from the graph G in that figure by adding a vertex to one of its edges.

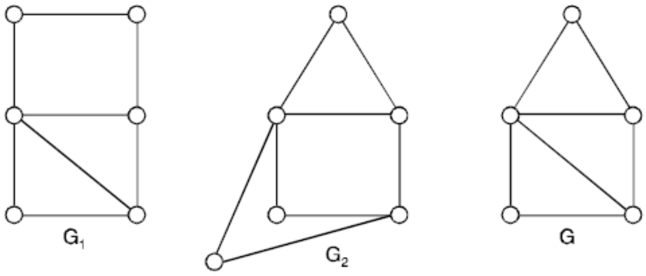


Fig. 2.6. Two homeomorphic graphs obtained from G by adding vertices to edges.

In Figure 2.7, we show two homeomorphic graphs, each obtained from K_5 by adding vertices to edges of K_5 (In each case, the vertices of K_5 are shown with solid dots).

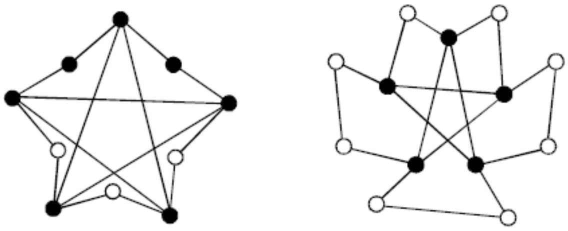


Fig. 2.7. Two homeomorphic graphs obtained from K_5.

2.5 REGION

A plane representation of a graph divides the plane into regions (also called windows, faces, or meshes) as shown in figure below. A region is characterized by the set of edges (or the set of vertices) forming its boundary.

Note that a region is not defined in a non-planar graph or even in a planar graph not embedded in a plane.

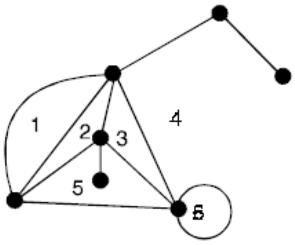


Fig. 2.8. Plane representation (the numbers stand for regions).

For example, the geometric graph in figure below does not have regions.

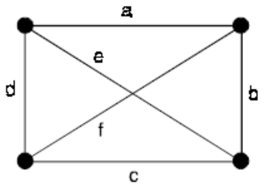


Fig. 2.9.

2.6 MAXIMAL PLANAR GRAPHS

A planar graph is maximal planar if no edge can be added without loosing planarity. Thus in any maximal planar graph with $p \geq 3$ vertices, the boundary of every region of G is a triangle for this maximal planar graphs (or plane graphs) are also refer to as triangulated planar graph (or plane graph).

2.7 SUBDIVISION GRAPHS

A subdivision of a graph is a graph obtained by inserting vertices (of degree 2) into the edges of G. For the graph G of the figure below, the graph H is a subdivision of G, while F is not a subdivision of G.

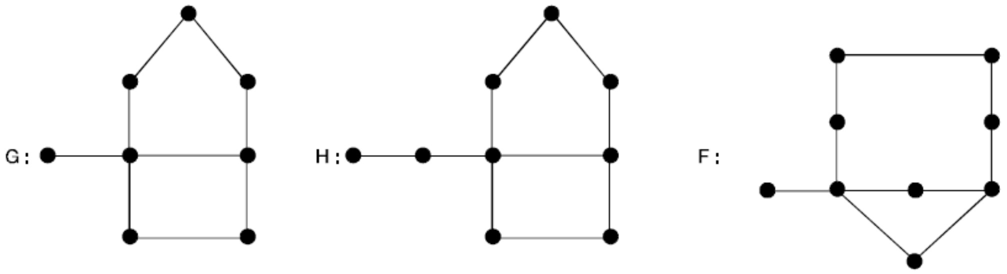


Fig. 2.10.

2.8 INNER VERTEX SET

A set of vertices of a planar graph G is called an inner vertex set I(G) of G. If G can be drawn on the plane in such a way that each vertex of I(G) lies only on the interior region and I(G) contains the minimum possible vertices of G. The number of vertices i(G) of I(G) is said to be the inner vertex number if they lie in interior region of G.

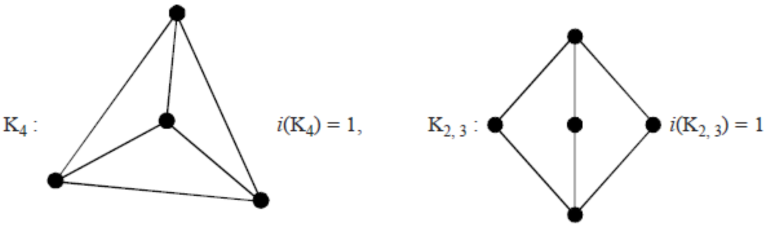


Fig. 2.11.

For any cycle C_p, $i(C_p) = 0$.

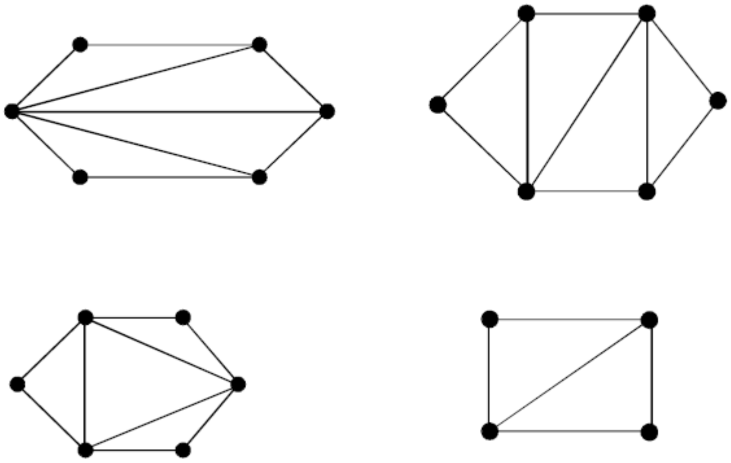
2.9 OUTER PLANAR GRAPHS

A planar graph is said to be outer planar if $i(G) = 0$. For example, cycles, trees, $K_4 - x$.

2.9.1. Maximal outer planar graph

An outer planar graph G is maximal outer planar if no edge can be added without losing outer planarity.

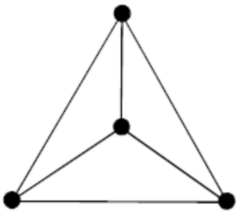
For example,



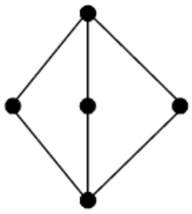
2.9.2. Minimally non-outer planar graphs

A planar graph G is said to be minimally non outer planar if $i(G) = 1$

For example, K_4 :



$K_{2,3}$:

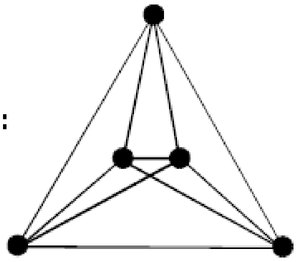


2.10 CROSSING NUMBER

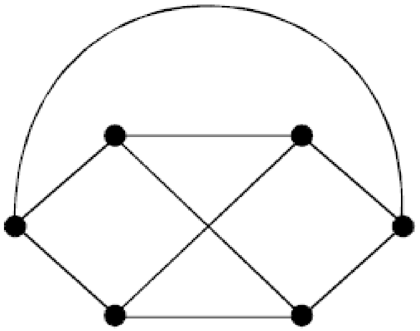
The crossing number C(G) of a graph G is the minimum number of crossing of its edges among all drawings of G in the plane.

A graph is planar if and only if $C(G) = 0$. Since K_4 is planar $C(K_4) = 0$ for $p \leq 4$. On the other hand $C(K_5) = 1$. Also $K_{3,3}$ is non planar and can be drawn with one crossing.

K_5 :



$K_{3,3}$:



2.12 EULER'S FORMULA

The basic results about planar graph known as Euler's formula is the basic computational tools for planar graph.

Theorem 2.1. Euler's Formula

If a connected planar graph G has n vertices, e edges and r region, then $n - e + r = 2$.

Proof. We prove the theorem by induction on e, number of edges of G.

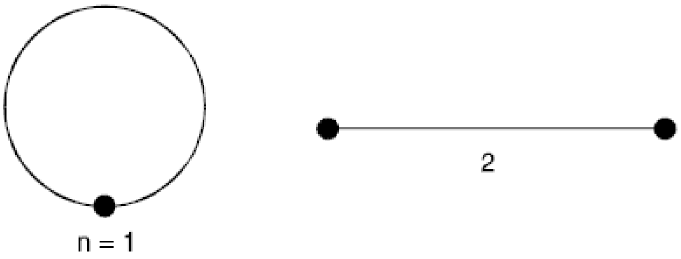
Basis of induction : If $e = 0$ then G must have just one vertex.

i.e., $n = 1$ and one infinite region, *i.e.,* $r = 1$

Then $n - e + r = 1 - 0 + 1 = 2$.

If $e = 1$ (though it is not necessary), then the number of vertices of G is either 1 or 2, the first possibility of occurring when the edge is a loop.

These two possibilities give rise to two regions and one region respectively, as shown in Figure below.



In the case of loop, $n - e + r = 1 - 1 + 2 = 2$ and in case of non-loop, $n - e + r = 2 - 1 + 1 = 2$.

Hence the result is true.

Induction hypothesis :

Now, we suppose that the result is true for any connected plane graph G with $e - 1$ edges.

Induction step :

We add one new edge K to G to form a connected supergraph of G which is denoted by G + K.

There are following three possibilities.

(*i*) K is a loop, in which case a new region bounded by the loop is created but the number of vertices remains unchanged.

(*ii*) K joins two distinct vertices of G, in which case one of the region of G is split into two, so that number of regions is increased by 1, but the number of vertices remains unchanged.

(*iii*) K is incident with only one vertex of G on which case another vertex must be added, increasing the number of vertices by one, but leaving the number of regions unchanged.

If let n', e' and r' denote the number of vertices, edges and regions in G and n, e and r denote the same in G + K. Then

In case (*i*) $n - e + r = n' - (e' + 1) + (r' + 1) = n' - e' + r'$.

In case (*ii*) $n - e + r = n' - (e' + 1) + (r' + 1) = n' - e' + r'$

In case (*iii*) $n - e + r = (n' + 1) - (e' + 1) + r' = n' - e' + r'$.

But by our induction hypothesis, $n' - e' + r' = 2$.

Thus in each case $n - e + r = 2$.

Now any plane connected graph with e edges is of the form G + K, for some connected graph G with $e - 1$ edges and a new edge K.

Hence by mathematical induction the formula is true for all plane graphs.

Corollary (1)

If a plane graph has K components then $n - e + r = K + 1$.

The result follows on applying Euler's formula to each component separately, remembering not to count the infinite region more than once.

Corollary (2)

If G is connected simple planar graph with n (≥ 3) vertices and e edges, then $e \leq 3n - 6$.

Proof. Each region is bounded by atleast three edges (since the graphs discussed here are simple graphs, no multiple edges that could produce regions of degree 2 or loops that could produce regions of degree 1, are permitted) and edges belong to exactly two regions.

$$2e \geq 3r$$

If we combine this with Euler's formula, $n - e + r = 2$, we get $3r = 6 - 3n + 3e \leq 2e$ which is equivalent to $e \leq 3n - 6$.

Corollary (3)

If G is connected simple planar graph with n (≥ 3) vertices and e edges and no circuits of length 3, then $e \leq 2n - 4$.

Proof. If the graph is planar, then the degree of each region is atleast 4.

Hence the total number of edges around all the regions is atleast $4r$.

Since every edge borders two regions, the total number of edges around all the regions is $2e$, so we established that $2e \geq 4r$, which is equivalent to $2r \leq e$.

If we combine this with Euler's formula $n - e + r = 2$, we get

$$2r = 4 - 2n + 2e \leq e$$

which is equivalent to $e \leq 2n - 4$.

Problem

Show that K_n is a planar graph for $n \leq 4$ and non-planar for $n \geq 5$.

Solution. A K_4 graph can be drawn in the way as shown in the Figure (2.18). This does not contain any false crossing of edges.

Thus, it is a planar graph.

Graphs K_1, K_2 and K_3 are by construction a planar graph, since they do not contain a false crossing of edges.

K_5 is shown in the Figure (2.19)

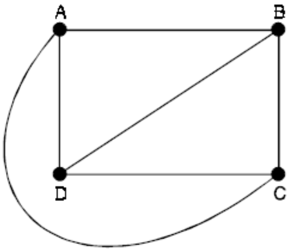


Fig. 2.18.

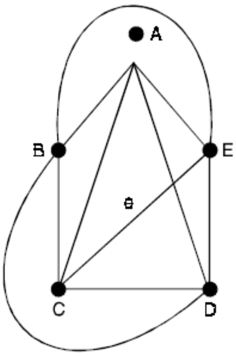


Fig. 2.19.

It is not possible to draw this graph on a 2-dimentional plane without false crossing of edges. Whatever way we adopt, at least one of the edges, say e, must cross the other for graph to be completed.

Hence K_5 is not a planar graph.

For any $n > 5$, K_n must contain a subgraph isomorphic to K_5.

Since K_5 is not planar, any graph containing K_5 as its one of the subgraph cannot be planar.

Problem

Show that $K_{3,3}$ is a non-planar graph.

Solution. Graph $K_{3,3}$ is shown in the Figure (2.20) below.

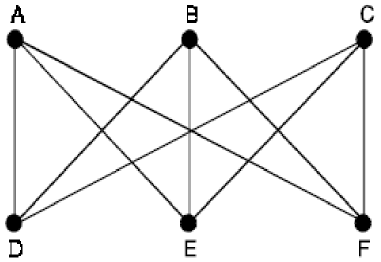


Fig. 2.20.

It is not possible to draw this graph such that there is no false crossing of edges. This is classic problem of designing direct lanes without intersection between any two houses, for three houses on each side of a road.

In this graph there exists an edge, say *e*, that cannot be drawn without crossing another edge.

Hence $K_{3,3}$ is a non-planar graph.

It is easy to determine that the chromatic number of this graph is 2.

Theorem 2.2. *Sum of the degrees of all regions in a map is equal to twice the number of edges in the corresponding graph.*

Proof. As discussed earlier, a map can be drawn as a graph, where regions of the map is denoted by vertices in the graph and adjoining regions are connected by edges.

Degree of a region in a map is defined as the number of adjoining region.

Thus, degree of a region in a map is equal to the degree of the corresponding vertices in the graph.

We know that the sum of the degrees of all vertices in a graph is equal to the twice the number of edges in the graph.

Therefore, we have $2e = \Sigma\deg(R_i)$.

Problem 2.9. *Prove that K_4 and $K_{2,2}$ are planar.*

Solution. In K_4, we have $v = 4$ and $e = 6$

Obviously, $6 \leq 3 * 4 - 6 = 6$

Thus this relation is satisfied for K_4.

For $K_{2,2}$, we have $v = 4$ and $e = 4$.

Again in this case, the relation $e \leq 3v - 6$

i.e., $4 \leq 3 * 4 - 6 = 6$ is satisfied.

Hence both K_4 and $K_{2,2}$ are planar.

Theorem 2.3. KURATOWSKI'S

$K_{3,3}$ and K_5 are non-planar.

Proof. Suppose first that $K_{3,3}$ is planar.

Since $K_{3,3}$ has a cycle $u \to v \to w \to x \to y \to z \to u$ of length 6, any plane drawing must contain this cycle drawn in the form of hexagon, as in Figure (2.23).

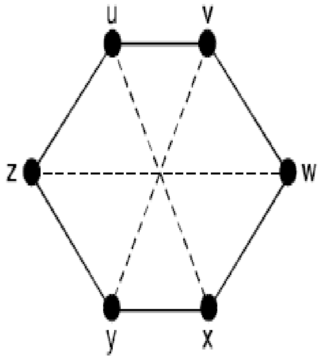


Fig. 2.23.

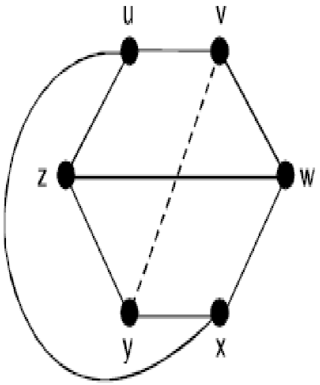


Fig. 2.24.

Now the edge wz must lie either wholly inside the hexagon or wholly outside it. We deal with the case in which wz lies inside the hexagon, the other case is similar.

Since the edge ux must not cross the edge wz, it must lie outside the hexagon ; the situation is now as in Figure (2.24).

It is then impossible to draw the edge vy, as it would cross either ux or wz.

This gives the required contradiction.

Now suppose that K_5 is planar.

Since K_5 has a cycle $v \to w \to x \to y \to z \to v$ of length 5, any plane drawing must contain this cycle drawn in the form of a pentagon as in Figure (2.25).

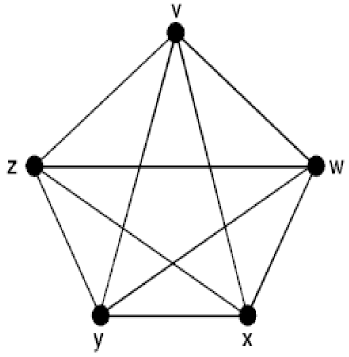


Fig. 2.25.

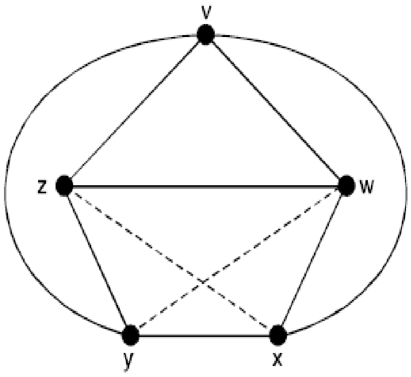


Fig. 2.26.

Now the edge wz must lie either wholly inside the pentagon or wholly outside it.

We deal with the case in which wz lies inside the pentagon, the other case is similar.

Since the edges vx and vy do not cross the edge wz, they must both lie outside the pentagon, the situation is now as in Figure (2.26)

But the edge xz cannot cross the edge vy and so must lie inside the pentagon.

Similarly the edge wy must lie inside the pentagon, and the edges wy and xz must then cross.

This gives the required contradiction.

Theorem 2.4. *Let G be a simple connected planar (p, q)-graph having at least K edges in a boundary of each region. Then (k – 2)q ≤ k(p – 2).*

Proof : Every edge on the boundary of G, lies in the boundaries of exactly two regions of G.

Further G may have some pendent edges which do not lie in a boundary of any region of G.

Thus, sum of lengths of all boundaries of G is less than twice the number of edges of G.

i.e., $kr \leq 2q$ ...(1)

But, G is a connected graph, therefore by Euler's formula

We have $r = 2 + q - p$ ...(2)

Substituting (2) in (1), we get

$$k(2 + q - p) \leq 2q$$

$$\Rightarrow \quad (k - 2)q \leq k(p - 2).$$

Problem 2.13. *Suppose G is a graph with 1000 vertices and 3000 edges. Is G planar ?*

Solution. A graph G is said to be planar if it satisfies the inequality. *i.e.,* $q \leq 3p - 6$

Here P = 1000, q = 3000 then

$$3000 \leq 3p - 6$$

i.e., $3000 \leq 3000 - 6$

or $3000 \leq 2994$ which is impossible.

Hence the given graph is not a planar.

Problem 2.14. *A connected graph has nine vertices having degrees 2, 2, 2, 3, 3, 3, 4, 4 and 5. How many edges are there ? How many faces are there ?*

Solution. By Handshaking lemma,

$$\sum_{i=1}^{n} \deg v_i = 2q$$

i.e., $2q = 2 + 2 + 2 + 3 + 3 + 3 + 4 + 4 + 5 = 28$

$\Rightarrow \quad q = 24$

Now by Euler's formula $p - q + r = 2$ or $9 - 14 + r = 2 \quad \Rightarrow \quad r = 7$

Hence there are 14 edges and 7 regions in the graph.

2.12.2. Kuratowski's Theorem

A graph is planar if and only if it has no subgraph homeomorphic to K_5 or $K_{3,3}$.

Proof. Let H be the inner piece guaranteeed by lemma (2) which is both $u_0 - v_0$ separating and $u_1 - v_1$ separating. In addition, let w_0, w_0', w_1 and w_1' be vertices at which H meets $Z(u_0, v_0)$, $Z(v_0, u_0)$, $Z(u_1, v_1)$ and $Z(v_1, u_1)$ respectively.

There are now four cases to consider, depending on the relative position on Z of these four vertices.

Case 1. One of the vertices w_1 and w_1' is on $Z(u_0, v_0)$ and the other is on $Z(v_0, u_0)$.

We can then take, say, $w_0 = w_1$ and $w_0' = w_1'$, in which case G contains a subgraph homeomorphic to $K_{3,3}$ as indicated in Figure (2.44)(*a*) in which the two sets of vertices are indicated by open and closed dots.

Case 2. Both vertices w_1 and w_1' are on either $Z(u_0, v_0)$ or $Z(v_0, u_0)$.

Without loss of generality we assume the first situation. There are two possibilities : either $v_1 \neq w_0'$ or $v_1 = w_0'$.

If $v_1 \neq w_0'$, then G contains a subgraph homeomorphic to $K_{3,3}$ as shown in Figure (2.44)(*b* or *c*), dependending on whether w_0' lies on $Z(u_1, v_1)$ or $Z(v_1, u_1)$ respectively.

If $v_1 = w_0'$ (see Figure 2.44), then H contains a vertex r from which there exist disjoint paths to w_1, w_1' and v_1, all of whose vertices (except w_1, w_1' and v_1) belong to H.

In this case also, G contains a subgraph homeomorphic to $K_{3,3}$.

Case 3. $w_1 = v_0$ and $w_1' \neq u_0$.

Without loss of generality, let w_1' be on $Z(u_0, v_0)$. Once again G contains a subgraph homeomorphic to $K_{3,3}$.

If w_0' is on (v_0, v_1), then G has a subgraph $K_{3,3}$ as shown in Figure 2.44(*e*).

If, on the other hand, w_0' is on $Z(v_1, u_0)$, there is a $K_{3,3}$ as indicated in Figure 2.44(*f*).

This Figure is easily modified to show G contains $K_{3,3}$ if $w_0' = v_1$.

Case 4. $w_1 = v_0$ and $w_1' = u_0$.

Here we assume $w_0 = u_1$ and $w_0' = v_1$, for otherwise we are in a situation covered by one of the first 3 cases.

We distinguish between two subcases.

Let P_0 be a shortest path in H from u_0 to v_0, and let P_1 be such a path from u_1 to v_1,

The paths P_0 and P_1 must intersect.

If P_0 and P_1 have more than one vertex in common, then G contains a subgraph homeomorphic to $K_{3,3}$ as shown in Figure 2.44(*g*).

Otherwise, G contains a subgraph homeomorphic to K_5 as in Figure 2.44(*h*).

Since these are all possible cases, the theorem has been proved.

2.13 DETECTION OF PLANARITY OF A GRAPH :

If a given graph G is planar or non planar is an important problem. We must have some simple and efficient criterion. We take the following simplifying steps :

Elementary Reduction :

Step 1 : Since a disconnected graph is planar if and only if each of its components is planar, we need consider only one component at a time. Also, a separable graph is planar if and only if each of its blocks is planar. Therefore, for the given arbitrary graph G, determine the set.

$$G = \{G_1, G_2, \ldots\ldots G_k\}$$

where each G_i is a non separable block of G.

Then we have to test each G_i for planarity.

Step 2 : Since addition or removal of self-loops does not affect planarity, remove all self-loops.

Step 3 : Since parallel edges also do not affect planarity, eliminate edges in parallel by removing all but one edge between every pair of vertices.

Step 4 : Elimination of a vertex of degree two by merging two edges in series does not affect planarity. Therefore, eliminate all edges in series.

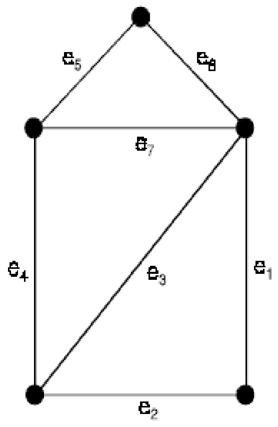
Repeated application of step 3 and 4 will usually reduce a graph drastically.

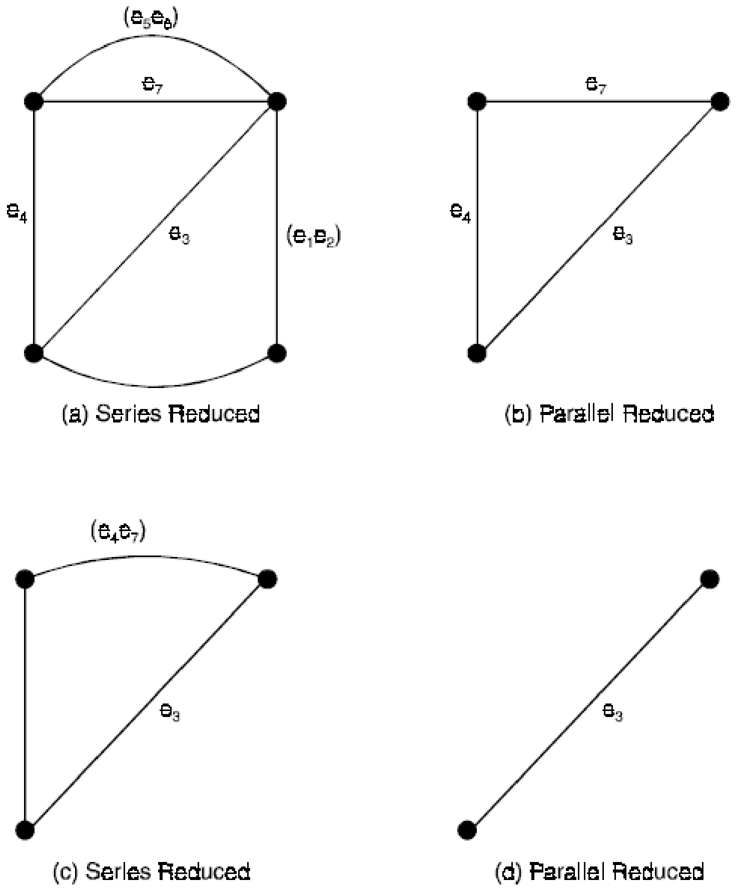
For example, Figure (2.46) illustrates the series-parallel reduction of the graph of Figure (2.45).

Let the non separable connected graph G_i be reduced to a new graph H_i after the repeated application of step 3 and 4. What will graph H_i look like ?

Graph H_i is

1. A single edge, or
2. A complete graph of four vertices, or
3. A non separable, simple graph with $n \geq 5$ and $e \geq 7$.





(a) Series Reduced

(b) Parallel Reduced

(c) Series Reduced

(d) Parallel Reduced

Problem

Carryout the elementary reduction process for the following graph :

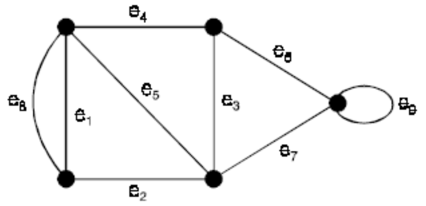
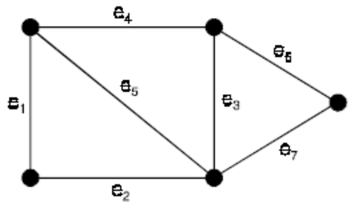


Fig. 2.55.

Solution. The given graph G is a single non separable block. Therefore, the set A of step 1 contains only G. As per step 2, we have to remove the self loops. In the graph, there is one self-loop consisting of the edge e_9. Let us remove it.

As per step 3, we have to remove one of the two parallel edges from each vertex pair having such edges. In the given graph, e_1, e_8 are parallel edges. Let us remove e_8 from the graph.

The graph left-out after the first three steps is as shown below :



As per step 4, we have to eliminate the vertices of degree 2 by merging the edges incident on these vertices.

Thus, we merge (*i*) the edges e_1 and e_2 into an edge e_{10} (say) and (*ii*) the edges e_6 and e_7 into an edge e_{11} (say).

The resulting graph will be as shown below :

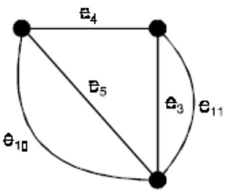
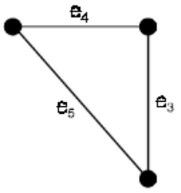


Fig. 2.57.

As per step 3, let us remove one of the parallel edges e_5 and e_{10} and one of the parallel edges e_3 and e_{11}. The graph got by removing e_{10} and e_{11} will be as shown below :



As per step 4, we merge the edges e_3 and e_4 into an edge e_{12} (say) to get the following graph.

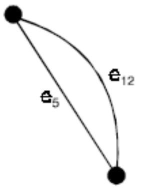
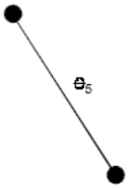


Fig. 2.59.

As per step 3, we remove one of the two parallel edges, say e_{12}. Thus, we get the following graph :



This graph is the final graph obtained by the process of elementary reduction applied to the graph in Figure (1). This final graph which is a single edge is evidently a planar graph.

Therefore, the graph in Figure (1) is also planar.

2.14.1. Uniqueness of the dual

Given a planar graph G, we can construct more than one geometric dual of G. All the duals so constructed have one important property. This property is stated in the following result :

All geometric duals of a planar graph G are 2-isomorphic, and every graph 2-isomorphic to a geometric dual of G is also a geometric dual of G.

2.14.2. Double dual

Given a planar graph G, suppose we construct its geometric dual G* and the geometric dual G** of G*.

Then G** is called a double geometric dual of G.

If G is a planar graph, then G** and G are 2-isomorphic.

2.14.3. Self-dual graphs

A planar graph G is said to be self-dual if G is isomorphic to its geometric dual G*, *i.e.,* if $G \approx G^*$.

Consider the complete graph K_4 of four vertices show in Figure (2.61)(*a*). Its geometric dual K_4^* can be constructed. This is shown in Figure (2.61)(*b*).

2.14.4. Dual of a subgraph

Let G be a planar graph and G* be its geometric dual. Let *e* be an edge in G and *e** be its dual in G*. Consider the subgraph $G - e$ got by deleting *e* from G. Then, the geometric dual of $G - e$ can be constructed as explained in the two possible cases.

Case (1) :

Suppose *e* is on a boundary common to two regions in G.

Then the removal of *e* from G will merge these two regions into one.

Then the two corresponding vertices in G* get merged into one, and the edge *e** gets deleted from G*.

Thus, in this case, the dual of $G - e$ can be obtained from G* by deleting the edge *e** and then fusing the two end vertices of *e** in $G^* - e^*$.

Case (2) :

Suppose *e* is not on a boundary common to two regions in G.

Then *e* is a pendant edge and *e** is a self-loop.

The dual of $G - e$ is now the same as $G^* - e^*$.

Thus, the geometric dual of $G - e$ can be constructed for all choices of the edge *e* of G.

Since every subgraph H of a graph is of the form $G - s$ where *s* is a set edges of G.

2.14.5. Dual of a homeomorphic graph

Let G be a planar graph and G* be its geometric dual.

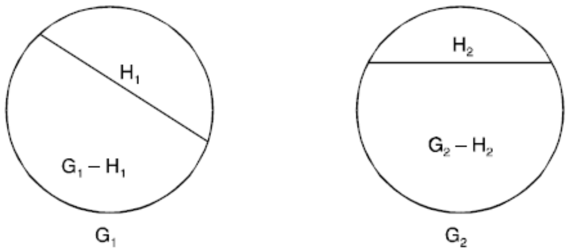
Let *e* be an edge in G and *e** be its dual in G*.

Suppose we create an additional vertex in G by introducing a vertex of degree 2 in the edge *e*. This will simply add an edge parallel to *e** in G*. If we merge two edges in series in G then one of the corresponding parallel edges in G* will be eliminated. The dual of any graph homeomorphic to G can be obtained from G*.

2.14.7. Combinatorial dual

Given two planar graphs G_1 and G_2, we say that they are combinatorial duals of each other if there is a one-to-one correspondence between the edges of G_1 and G_2 such that if H_1 is any subgraph of G_1 and H_2 is the corresponding subgraph of G_2, then

Rank of $(G_2 - H_2)$ = Rank of G_2 – Nullity of H_1



Consider the graph G_1 and G_2 shown in Figure (2.62) above, and their subgraphs H_1 and H_2 shown in Figure (2.64)(*a*, *b*).

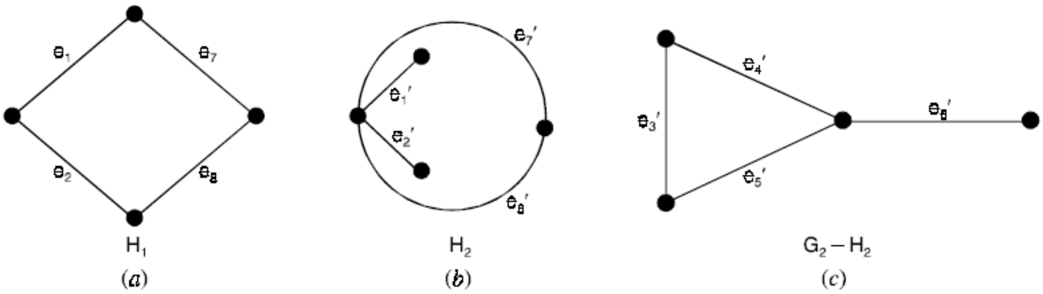


Fig. 2.64.

Note that there is one-to-one correspondence between the edges of G_1 and G_2 and that the subgraphs H_1 and H_2 correspond to each other.

The graph of $G_2 - H_2$ is shown in Figure (2.64)(*c*).

This graph is disconnected and has two components.

Rank of $G_2 = 5 - 1 = 4$, Rank of $H_1 = 4 - 1 = 3$

Nullity of $H_1 = 4 - 3 = 1$

Rank of $(G_2 - H_2) = 5 - 2 = 3$.

$\Rightarrow$ Rank of $(G_2 - H_2) = 3$ = Rank of G_2 – Nullity of H_1.

Hence, G_1 and G_2 are combinatorial duals of each other.

Theorem 2.23. *A graph has a dual if and only if it is planar.*

Proof. Suppose that a graph G is planar.

Then G has a geometric dual in G*.

Since G* is a geometric dual, it is a dual.

Thus G has a dual.

Conversely, suppose G has a dual.

Assume that G is non planar. Then by Kuratowski's theorem, G contains K_5 and $K_{3, 3}$ or a graph homeomorphic to either of these as a subgraph.

But K_5 and $K_{3, 3}$ have no duals and therefore a graph homeomorphic to either of these also has no dual.

Thus, G contains a subgraph which has no dual.

Hence G has no dual. This is a contradiction.

Hence G is planar if it has a dual.

2.15 GRAPH COLORING

Coloring problem

Suppose that you are given a graph G with n vertices and are asked to paint its vertices such that no two adjacent vertices have the same color. What is the minimum number of colors that you would require. This constitutes a coloring problem.

2.15.1. Partitioning problem

Having painted the vertices, you can group them into different sets—one set consisting of all red vertices, another of blue, and so forth. This is a partitioning problem.

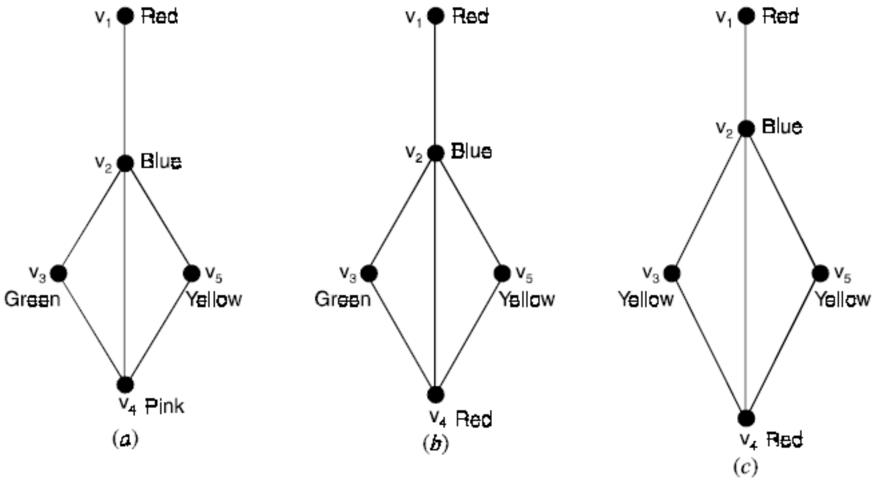
For example, finding a spanning tree in a connected graph is equivalent to partitioning the edges into two sets—one set consisting of the edges included in the spanning tree, and the other consisting of the remaining edges. Identification of a Hamiltonian circuit (if it exists) is another partitioning of set of edges in a given graph.

2.15.2. Properly coloring of a graph

Painting all the vertices of a graph with colours such that no two adjacent vertices have the same colour is called the proper colouring (or simply colouring) of a graph.

A graph in which every vertex has been assigned a colour according to a proper colouring is called a properly coloured graph.

Usually a given graph can be properly coloured in many different ways. Figure (2.69)(a) shows three different proper colouring of a graph.



(a) (b) (c)

The K-colourings of the graph G is a colouring of graph G using K-colours. If the graph G has colouring, then the graph G is said to be K-colourable.

2.15.3. Chromatic number

A graph G is said to be K-colourable if we can properly colour it with K (number of) colours.

A graph G which is n-colourable but not (K – 1)-colourable is called a K-chromatic graph.

In other words, a K-chromatic graph is a graph that can be properly coloured with K-colours but not with less than K colours.

If a graph G is K-chromatic, then K is called chromatic number of the graph G. Thus the chromatic number of a graph is the smallest number of colours with which the graph can be properly coloured.The chromatic number of a graph G is usually denoted by $\chi(G)$.

We list a few rules that may be helpful :

1. $\chi(G) \leq |V|$, where $|V|$ is the number of vertices of G.
2. A triangle always requires three colours, that is $\chi(K_3) = 3$; more generally, $\chi(K_n) = n$, where K_n is the complete graph on n vertices.
3. If some subgraph of G requires K colors then $\chi(G) \geq K$.
4. If degree $(v) = d$, then atmost d colours are required to colour the vertices adjacent to v.

5. $\chi(G)$ = maximum $\{\chi(C)/C$ is a connected component of G$\}$
6. Every K-chromatic graph has at least K vertices v such that degree $(v) \geq k - 1$.
7. For any graph G, $\chi(G) \leq 1 + \Delta(G)$, where $\Delta(G)$ is the largest degree of any vertex of G.
8. When building a K-colouring of a graph G, we may delete all vertices of degree less than K (along with their incident edges).

   In general, when attempting to build a K-colouring of a graph, it is desirable to start by K-colouring a complete subgraph of K vertices and then successively finding vertics adjacent to K – 1 different colours, thereby forcing the color choice of such vertices.
9. These are equivalent :

   (*i*) A graph G is 2-colourable (*ii*) G is bipartite

   (*iii*) Every cycle of G has even length.
10. If $\delta(G)$ is the minimum degree of any vertex of G, then $\chi(G) \geq \frac{|V|}{|V|} - \delta(G)$ where $|V|$ is the number of vertices of G.

2.16 CHROMATIC POLYNOMIAL

A given graph G of n vertices can be properly coloured in many different ways using a sufficiently large number of colours. This property of a graph is expressed elegantly by means of a polynomial. This polynomial is called the chromatic polynomial of G.

The value of the chromatic polynomial $P_n(\lambda)$ of a graph with n vertices gives the number of ways of properly colouring the graph, using λ of fewer colours. Let C_i be the different ways of properly colouring G using exactly i different colours. Since i colours can be chosen out of λ colours in $\binom{\lambda}{i}$ different ways, there are $c_i \binom{\lambda}{i}$ different ways of properly colouring G using exactly i colours out of λ colours.

Since i can be any positive integer from 1 to n (it is not possible to use more than n colours on n vertices), the chromatic polynomial is a sum of these terms, that is,

$$P_n(\lambda) = \sum_{i=1}^{n} C_i \binom{\lambda}{i}$$

$$= C_1 \frac{\lambda}{1!} + C_2 \frac{\lambda(\lambda-1)}{2!} + C_3 \frac{\lambda(\lambda-1)(\lambda-2)}{3!} + \ldots\ldots$$

$$\ldots + C_n \frac{\lambda(\lambda-1)(\lambda-2)\ldots\ldots(\lambda-n+1)}{n!}$$

Each C_i has to be evaluated individually for the given graph.

For example, any graph with even one edge requires at least two colours for proper colouring, and therefore $C_1 = 0$.

A graph with n vertices and using n different colours can be properly coloured in n ! ways.

that is, $C_n = n$!.

Problem 2.44. *Find the chromatic polynomial of the graph given in Figure (2.70).*

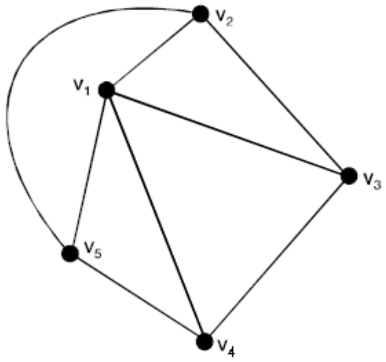


Fig. 2.70. A 3-chromatic graph.

Solution. $P_5(\lambda) = C_1\lambda + C_2 \frac{\lambda(\lambda-1)}{2!} + C_3 \frac{\lambda(\lambda-1)(\lambda-2)}{3!}$

$$+ C_4 \frac{\lambda(\lambda-1)(\lambda-2)(\lambda-3)}{4!} + C_5 \frac{\lambda(\lambda-1)(\lambda-2)(\lambda-3)(\lambda-4)}{5!}$$

Since the graph in Figure 2.70 has a triangle, it will require at least three different colours for proper colourings.

Therefore, $C_1 = C_2 = 0$ and $C_5 = 5$!

Moreover, to evaluate C_3, suppose that we have three colours x, y and z.

These three colours can be assigned properly to vertices v_1, v_2 amd v_3 in 3 ! = 6 different ways.

Having done that, we have no more choices left, because vertex v_5 must have the same colour as v_3 and v_4 must have the same colour as v_2.

Therefore, $C_3 = 6$.

Similarly, with four colours, v_1, v_2 and v_3 can be properly coloured in 4 • 6 = 24 different ways.

The fourth colour can be assigned to v_4 or v_5, thus providing two choices.

The fifth vertex provides no additional choice.

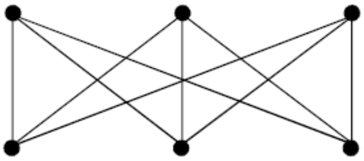
Therefore, $C_4 = 24 \bullet 2 = 48$.

Substituting these coefficients in $P_5(\lambda)$, we get, for the graph in Figure (2.70).

$$\begin{aligned} P_5(\lambda) &= \lambda(\lambda - 1)(\lambda - 2) + 2\lambda(\lambda - 1)(\lambda - 2)(\lambda - 3) + \lambda(\lambda - 1)(\lambda - 2)(\lambda - 3)(\lambda - 4) \\ &= \lambda(\lambda - 1)(\lambda - 2)(\lambda^2 - 5\lambda + 7) \end{aligned}$$

The presence of factors $\lambda - 1$ and $\lambda - 2$ indicates that G is at least 3-chromatic.

Problem 2.45. *Find the chromatic polynomial and chromatic number for the graph $K_{3,\,3}$.*



Solution. Chromatic polynomial for $K_{3,\,3}$ is given by $\lambda(\lambda - 1)^5$.

Thus chromatic number of this graph is 2.

Since $\lambda(\lambda - 1)^5 > 0$ first when $\lambda = 2$.

Here, only two distinct colours are required to colour $K_{3,\,3}$.

The vertices *a*, *b* and *c* may have one colours, as they are not adjacent.

Similarly, vertices *d*, *e* and *f* can be coloured in proper way using one colour.

But a vertex from $\{a, b, c\}$ and a vertex from $\{d, e, f\}$ both cannot have the same colour.

In fact every chromatic number of any bipartite graph is always 2.